



FACULTY OF APPLIED SCIENCES
UNIVERSITY
OF WEST BOHEMIA

DEPARTMENT OF
COMPUTER SCIENCE
AND ENGINEERING



Doctoral Dissertation

Computerised muscle modelling

Martin Červenka



PILSEN, CZECH REPUBLIC

2024



FAKULTA APLIKOVANÝCH VĚD
ZÁPADOČESKÉ UNIVERZITY
V PLZNI

KATEDRA INFORMATIKY
A VÝPOČETNÍ TECHNIKY

Disertační práce

Počítačové modelování svalů

Ing. Martin Červenka

Školitel

Doc. Ing. Josef Kohout, Ph.D.



**FACULTY OF APPLIED SCIENCES
UNIVERSITY
OF WEST BOHEMIA**

**DEPARTMENT OF
COMPUTER SCIENCE
AND ENGINEERING**

Doctoral Dissertation

Computerised muscle modelling

Ing. Martin Červenka

Supervisor

Doc. Ing. Josef Kohout, Ph.D.

© 2024 Martin Červenka.

All rights reserved. No part of this document may be reproduced or transmitted in any form by any means, electronic or mechanical including photocopying, recording or by any information storage and retrieval system, without permission from the copyright holder(s) in writing.

Citation in the bibliography/reference list:

ČERVENKA, Martin. *Computerised muscle modelling*. Pilsen, Czech Republic, 2024. Doctoral Dissertation. University of West Bohemia, Faculty of Applied Sciences, Department of Computer Science and Engineering. Supervisor Doc. Ing. Josef Kohout, Ph.D.

Declaration

I hereby declare that this Doctoral Dissertation is completely my own work and that I used only the cited sources, literature, and other resources. This thesis has not been used to obtain another or the same academic degree.

I acknowledge using ChatGPT 4 (<https://chat.openai.com>) to enhance the English level of the dissertation. The prompts used include "Rewrite the following statement:" and "Reformulate:" with the corresponding text to enhance. The enhancement affected about 10% of the text.

I acknowledge that my thesis is subject to the rights and obligations arising from Act No. 121/2000 Coll., the Copyright Act as amended, in particular the fact that the University of West Bohemia has the right to conclude a licence agreement for the use of this thesis as a school work pursuant to Section 60(1) of the Copyright Act.

In Pilsen, on 26 August 2024

.....
Martin Červenka

The names of products, technologies, services, applications, companies, etc. used in the text may be trademarks or registered trademarks of their respective owners.

Abstrakt

Muskuloskeletální modelování se ukazuje jako silný nástroj pro simulaci komplexních lidských pohybů. Tato disertace má za cíl použít aproximaci pomocí radiálních bázových funkcí (RBF) k zlepšení přesností těchto modelů. Cílem je demonstrovat vhodnost RBF k zachycení tvaru a pohybu modelu svalu s uvážením širšího spektra vlastností vstupního svalu. RBF může být také použito pro rekonstrukci povrchu svalových úponů z množiny bodů v kontextu odhadu tvaru svalového úponu na kosti.

Disertace si klade za cíl popsat nejmodernější metody modelování svalů, popisáním všech nyní známých a relevantních přístupů. Dále text přechází do návrhu RBF matematického modelu k popisu tvaru svalu za použití množiny RBF, zahrnující techniky hledání optimálních pozic středů dle několika vlastností, například kraje svalu, lokální extrémy, body inflexe, strategicky zvolené pseudonáhodné pozice, tzv. žravé umístování RBF a dále hledání vhodných tvarových parametrů. Poslední částí je popis matematického modelu pohybu geometrie svalu, což je hlavní část této disertace.

Klíčová slova

Modelování svalů, aproximace, Radiální Bázové Funkce, RBF, umístování středů, křivost, Pozičně orientovaná dynamika, PBD, Technika co největší tuhosti, ARAP, interpolace, přes body, Systém pružin a hmotných bodů, Metoda konečných prvků.

Abstract

Musculoskeletal modelling has emerged as a powerful tool for simulating complex human movements. This doctoral dissertation focuses on using the Radial Basis Function (RBF) approximation technique to enhance the precision of such models. The aim is to demonstrate the efficacy of RBF in capturing the shape and motion of a muscle model by considering a wider spectrum of features in the input function. RBF can also be used to reconstruct surfaces from sets of attachment points in the context of muscle attachment estimation.

The doctoral dissertation aims to describe the current state-of-the-art muscle modelling field, describing all of the currently known approaches. The dissertation then transitions to propose a novel RBF mathematical model to describe a muscle using a set of RBF, which introduces a technique of finding an optimal centre point using multiple groups, such as vertices at borders, local extrema, points of inflexion, strategic pseudorandom positions, greedy MSE placement, and more, finding suitable shape parameters, and, ultimately, describing the mathematical model for the movement of the geometry. Therefore, the outcome of the dissertation is a mathematical model for the dynamical muscle model

Keywords

Muscle modelling • approximation • Radial Basis Functions • RBF • centre placement • curvature • Position-Based Dynamics • PBD • As-rigid-as-possible • ARAP • interpolation • Via-points • Mass-spring systems • Finite Element Method

Acknowledgement

I would like to thank Doc. Ing. Josef Kohout, PhD, for supervising not only this particular work but also during the whole of my study. I am also grateful to my colleagues at the University of West Bohemia. My deepest thanks go to Vaclav Skala for valuable discussions, personal support, and hints during my research. However, my most profound thankfulness goes to my partner because this work would not be possible without her support.

This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic, project no. SGS-2022-015 and SGS-2019-016; and the Czech Science Foundation, project no. 23-04622L and 17-05534S.

To Péťa, the tiny heartbeat that inspires every word.



Contents

1	Introduction	7
2	Problem specification	9
2.1	Shape of the muscle	11
2.2	Volume change	11
2.3	Collisions	11
3	Acquiring data	13
3.1	Muscle-tendon units	14
3.2	Muscle attachments	14
3.3	Non-Invasive techniques	15
3.4	Invasive techniques	16
3.5	Physiological signals	17
4	Estimation approaches	19
4.1	Constant and piecewise linear estimation	20
4.2	Bézier curves	21
4.3	Catmull-Rom spline	21
4.4	Discrete Fourier transform	22
5	Related work	25
5.1	Hill-Type Model	26
5.2	Via-points	27
5.3	Wrapping Obstacles	28
5.4	Finite Element Method	29
5.4.1	FEM in muscle modelling	30
5.5	Other optimization problems	31
5.5.1	Mass-Spring System (MSS)	32
5.5.2	Optimization	33
5.5.3	ARAP - As-Rigid-As-Possible Deformation	34
5.5.4	Position-Based Dynamics	38

6	Radial basis functions	49
6.1	Available functions	50
6.1.1	Local functions	51
6.1.2	Global functions	51
6.2	Polynomial extension	52
6.3	Centre point distribution	53
6.3.1	Iterative greedy search	53
6.3.2	Grid distribution	54
6.3.3	Halton distribution	54
6.3.4	Distribution concerning the original function	54
6.4	An approximation example	55
6.5	RBFs for muscle modelling	58
7	Novel approach	59
7.1	Sample placement	60
7.2	Centre placement and shape selection	61
7.3	Dynamics	62
7.4	Mathematical model	63
7.4.1	Mean curvature	63
7.5	Regularisation	65
8	Model verification	67
8.1	Datasets	67
8.2	Static surface generation	68
8.2.1	Gluteus maximus	68
8.2.2	Gluteus medius	71
8.2.3	Iliacus	71
8.2.4	Optimal and sub-optimal shapes	72
8.2.5	Limitations	73
8.3	Dynamics	74
8.3.1	Random surface movement	75
8.3.2	Femur movement	76
9	Author's contribution	79
9.1	A New Strategy for Scattered Data Approx. Using RBF Respecting Points of Inflection	82
9.2	Novel RBF Approx. Method Based on Geom. Prop. with a New RBF Function	98
9.3	Modified Radial Basis Functions Approximation Respecting Data Local Features	105

9.4	Fast and Realistic Approach to Virtual Muscle Deformation	112
9.5	Behavioral Study of Various RBFs for Approximation and Interpolation Purposes	124
9.6	Finding Points of Importance for RBF Approximation of Large Scattered Data	131
9.7	Conditionality Analysis of the Radial Basis Function Matrix	144
9.8	Muscle Deformation Using Position Based Dynamics	159
9.9	Geometry Algebra and GEM for solving a linear system of equations without division	184
9.10	Collision detection and response approaches for computer muscle modelling	190
9.11	Non-planar Surf. Shape Rec. from a Point Cloud in the Context of Muscles Attach. Est.	197
9.12	Computerised muscle modelling and simulation for interactive applications	206
9.13	A mathematical model for smooth RBF implicit surface model for muscle modelling	215
10 Conclusion, Summary & Future work		251
Bibliography		253
About the author		265
Project activites		266
Lectures		267
Lecture’s feedback		268

Preface

In biomechanics and musculoskeletal research, the search for accurate and efficient modelling techniques has been a continuous journey driven by the desire to understand the complexities of human and animal movement and function. This dissertation delves into musculoskeletal modelling, focusing specifically on using Radial Basis Function (RBF) approximation techniques as a powerful tool for enhancing the accuracy and smoothness of such models.

In this dissertation, I have adopted an integrative writing approach that seamlessly weaves existing research into the narrative of my argument. Rather than separating the literature review, analysis, and findings into distinct sections, I have blended these elements throughout the text. This method allows for a more coherent and fluid presentation of ideas, where the research directly informs and supports the ongoing discussion. By doing so, I aim to maintain a strong connection between the theoretical framework and the empirical evidence, ensuring that the reader can easily follow the progression of my argument. Proper attribution is given through meticulous citation, ensuring the sources are acknowledged while allowing my analysis and insights to remain at the forefront of the discussion.

It begins with a review of the existing literature, a survey of the landscape of musculoskeletal modelling methodologies, and a light on the challenges researchers face in this domain. The core idea of the study is then described. The dissertation then transitions to a detailed exposition of the author's work.

Much of this dissertation is dedicated to the description and insight into the RBF approximation technique in musculoskeletal modelling. The study aims to demonstrate RBF's efficacy in capturing a muscle model's shape and motion.

This dissertation contributes to the ongoing discourse in musculoskeletal modelling. Using the power of RBF approximation, one can aspire to unlock new dimensions of understanding in human biomechanics, potentially paving the way for innovative applications in fields ranging from rehabilitation engineering to sports science.

Martin Červenka,
author

Introduction

1

The complex quest for accurate and realistic modelling of human muscles remains a cornerstone in biomechanics and musculoskeletal research. This doctoral dissertation hypothesises that another representation of a muscle can be used, focusing on integrating Radial Basis Function approximation techniques. The research stands at the intersection of computer science, computational mathematics, and biomechanics, aiming to describe some of the most challenging aspects of muscle modelling.

Musculoskeletal models are indispensable in various fields, from medical diagnostics to the design of advanced prosthetics, sports science, and the development of human-like animations in computer graphics. However, the complexity of human anatomy, combined with the dynamic nature of muscle movement, presents unique challenges. Existing general modelling techniques often fail to capture the intricate details of muscle shapes and their movements, leading to a gap between simulation and reality, since they represent a muscle by a set of lines, e.g. [1], ignore intermuscular behaviour, e.g., [2].

The research delves into the intricacies of muscle modelling, scrutinizing the limitations of current modelling methodologies. This work proposes a novel use of RBF approximation, a mathematical technique known for its flexibility and precision in representing complex multidimensional shapes [3, 4]. The dissertation introduces innovative ways to accurately represent muscle geometry and dynamics by applying RBF to musculoskeletal modelling.

Furthermore, this work provides a comprehensive review of the existing literature in muscle modelling, establishing a solid foundation for the proposed methodologies. It not only debates the current state-of-the-art but also identifies potential areas for improvement. The dissertation then transitions into a detailed exposition of the proposed RBF-based models, their theoretical underpinnings, and practical applications. It includes extensive computational simulations, offering insights into their potential impact on various applications.

The following described work employs various scientific methodologies. It incorporates empirical techniques such as visualizing muscle movement and correlating it with ongoing results. Additionally, general methods like generalization and

simplification convert real-world muscle structures into models. The analytical scientific approach is a primary focus, particularly in the chapter on the novel muscle model. Furthermore, the synthesis method is extensively applied, especially when working with RBFs.

This dissertation is poised to contribute significantly to the musculoskeletal modelling field. Addressing the critical need for more smooth and realistic muscle models paves the way for advancements in medical research, ergonomic design, sports science, and beyond. The dissertation pushes the boundaries of current modelling techniques and opens new avenues for interdisciplinary research, blending biomechanics with cutting-edge computational methods.

The content of this dissertation is described not only via the table of contents but also using a radial plot in Fig. 1.1. The image describes better which parts of the work are more significant than others; the less important parts are mentioned but not described thoroughly due to their less importance to the overall work. Also, in some chapters, more detailed radial plots are placed (see Fig. 3.1, 4.1, 6.1, 5.1, 8.1 and 9.1) to introduce the given chapter more rigorously.



Figure 1.1: The graphical description of the content of the dissertation. Each bar shows the depth of exploration. Created with the www.flourish.studio

Problem specification

2

Muscle modelling is pivotal in advancing our understanding of human physiology and biomechanics. It bridges the gap between theoretical knowledge and practical medical, sports science, and rehabilitation applications. It offers insights into how muscles generate force, interact with skeletal structures, and how these processes contribute to human movement and function. The importance of proper muscle modelling cannot be overstated, as it underpins the development of personalized medical treatments and the enhancement of athletic performance [5].

In the medical field, muscle models are crucial for simulating surgeries, predicting outcomes of rehabilitation protocols, and understanding musculoskeletal disorders at a fundamental level [6]. Furthermore, in sports science, detailed muscle modelling can aid in designing training regimes that optimize performance while minimizing the risk of injury, offering athletes tailored strategies that consider their unique physiological characteristics [7].

Accurately modelling the human musculoskeletal system presents significant challenges due to its complexity. The human body consists of an intricate system of bones and soft tissues (e.g., muscles and tendons), each contributing to our overall movement and functionality. However, the practical application of such models often requires a focused approach. For instance, when considering clinical decisions, such as determining the necessary lengthening of a femur to correct a patient's gait, the model does not need to simulate every molecular interaction within the muscles and bones. Instead, it is crucial to model the key parameters that directly influence the outcome of interest, such as bone length adjustments and their impact on joint alignment and muscle function. This targeted approach allows for adequate decision support in medical interventions without overcomplicating the model beyond what is necessary. Defining specific parameters like muscle activation patterns and their relationship to normal gait can provide practical tools for healthcare professionals to optimize treatment strategies.

The development of novel approaches to muscle modelling, such as those proposed in this dissertation, represents a significant step forward in our ability to

simulate and understand human movement.

Muscle modelling encompasses a broad spectrum of applications, each demanding a precise specification of the challenges involved. Given the intricate nature of the human body, a degree of simplification is essential to feasibly represent its mechanics without compromising on accuracy beyond acceptable limits. This chapter outlines the considerations in approaching muscle modelling, focusing on the methodological decisions underpinning our research.

A pivotal aspect of this dissertation is the decision between employing direct or inverse kinematics. Direct kinematics seeks to mirror the physiological processes, from the excitation of muscle fascicles by electrical impulses to the resultant movements and deformations of bones. In contrast, inverse kinematics works backwards from observed movements to deduce muscle shapes and the forces they exert, circumventing the direct estimation of electrical signal excitations. This method is particularly advantageous for applications in the medical field, where the primary interest often lies in understanding the forces involved.

In our modelling efforts, bones are considered rigid structures to simplify the complex interactions within the musculoskeletal system. Muscles and tendons are modelled based on their real-world properties, with common simplifications to facilitate specific applications. These include the assumption of constant muscle volume and the consideration of muscle tissue's internal structure and anisotropy, which are pivotal depending on the application's requirements.

Moreover, the issue of structural collisions within the musculoskeletal system presents a significant challenge. Our approach has been to review and apply the most effective techniques from extensive research.

For this dissertation, we have adopted specific simplifications to develop a novel model:

1. Focus on inverse kinematics for modelling
2. Treat bones as entirely rigid entities
3. Assume no significant volume change in any structure (bone/muscle)
4. Maintain the initial muscle shape throughout the modelling process

These simplifications are vital to crafting a model that, despite its abstractions, effectively meets the research or application goals. Below, we delve into additional aspects that require clarification.

2.1 Shape of the muscle

The preservation of muscle shape in models can vary significantly, depending on the methodological approach. The commonly adopted model employs a triangular mesh, reducing energy through a balance of global and local terms, as suggested by Sorkine et al. [8] and their followers. This method efficiently manages the transformations and non-rigid deformations within the mesh.

This dissertation aims to refine existing models and introduce a novel approach utilizing a continuous RBF implicit function distinct from the traditional triangular mesh structure. To date, there has been no proposal for preserving the surface of a muscle via a smooth and continuous 3D model. Therefore, this work proposes a method that maintains the original spatial curvature of the model while penalizing any deviations, detailed further in Section 7.4.

2.2 Volume change

The constant volume assumption is crucial in simplifying the mathematical representation of muscle behaviour in muscle modelling. Despite this, muscle activity can involve subtle volume changes, challenging this assumption. This dissertation addresses the challenge of modelling volume changes without significantly complicating the model.

2.3 Collisions

Addressing collisions within the musculoskeletal system is critical for realistic modelling. This dissertation explores the integration of collision detection and response mechanisms into our model, ensuring that interactions between muscles, tendons, and bones are accurately represented. We evaluate various collision handling techniques, ultimately incorporating a method that balances computational efficiency with the need for accuracy. This approach allows us to simulate the complex interplay of musculoskeletal components in a dynamic environment, contributing to the overall realism and utility of the model.

Acquiring data

3

The primary foundation for achieving quality computer muscle modelling success is obtaining relevant data. This section explores methods and opportunities for data acquisition.

Data can be categorised into two types: general and personalised data. Personalised data pertains to measurements taken from a specific subject. Those data are rooted in measurements of a patient's illness or when optimising an athlete's performance. On the contrary, general data is collected from other individuals and may serve as a template but may not be suitable for a particular patient due to the substantial variability in muscle attachment sites between subjects. Further details on this issue can be found in [9], [10], and [11]. Because the musculoskeletal system is too complex to model, few words address some simplifications of the muscle-tendon units.

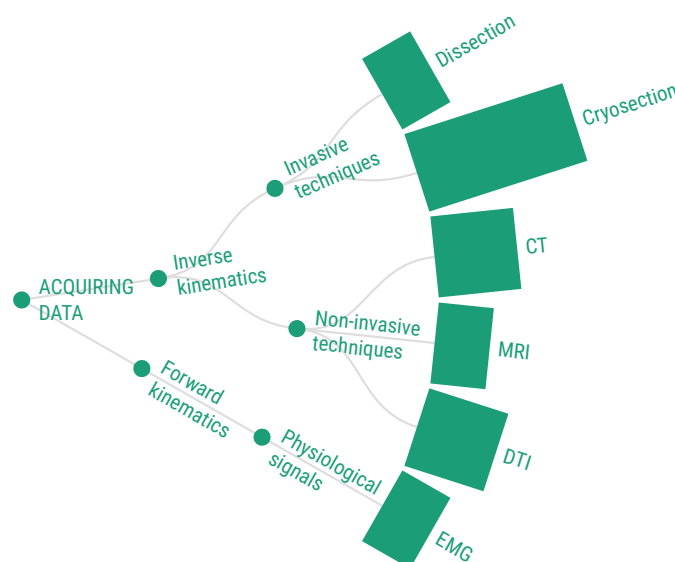


Figure 3.1: The graphical description of the topics on data acquisition discussed in the dissertation (described by the depth of the discussion). Created with the www.flourish.studio

Together with the rest of the text, this section covers the central data acquisition topics. The depth of coverage is visualised in Fig. 3.1. The main focus is on data acquisition for inverse kinematic approaches. However, the acquisition techniques required for direct kinematics are also mentioned briefly.

3.1 Muscle-tendon units

The human body is intricate, and creating a completely accurate model is impractical. Therefore, state-of-the-art musculoskeletal models need to simplify the problem, omitting various phenomena due to the complexity of the human body. The musculoskeletal models described in this text consist mainly of bones and muscles. Some methods also require models of the attachment areas or information about the direction of muscle fibres. All models described in the following text are geometrically represented. It is important to note that throughout this text, muscle models implicitly approximate muscle-tendon units (MTUs), which comprise muscles, tendons, cartilages, aponeurosis, fats, blood vessels, etc. Muscle (MTU) and bone models are often approximated using triangular meshes. The challenge in model acquisition is related to the resolution of acquisition methods, as is discussed further. However, to work with a muscle model, knowing to which bone it is attached is crucial.

3.2 Muscle attachments

Determining which part of a muscle connects to a specific area of a bone requires defining a muscle attachment area. This attachment area can be determined automatically or manually, sometimes requiring additional data. When such data are not available in advance, two options are considered:

1. Fix the set of points of the muscle model that are "close enough" to the bone surface or intersect with the bone before movement.
2. Obtain a dedicated set of points that define the attachment area (e.g., boundary points [12], scattered points over the area [13], or points from a boundary curve) from a user.

In the first case, implementation is relatively straightforward. However, there is a risk of incorrectly fixing some muscle parts, mainly when a muscle part is adjacent (but not attached) to a joint, which can lead to further complications, such as forcing the muscle part into the joint. The second approach is more robust but necessitates obtaining the entire area from the provided boundary points. This issue was addressed in our article [13], which is effective for simple cases but may encounter challenges with the shapes of the curved and complex muscle attached

area, especially those with multiple bends. In the study, we experimented with 15 different algorithms for curve reconstruction, but even the best algorithm achieved a maximum accuracy of 78.74%, which may be insufficient for specific applications.

Furthermore, in the paper, we explored surface plate reconstruction from the set of points using Radial Basis Functions (RBF), as described in more detail in Chapter 6, achieving acceptable results even for more intricate boundaries.

3.3 Non-Invasive techniques

Non-invasive data acquisition methods allow the extraction of personalised data, which is crucial for muscle modelling. Commonly used non-invasive methods include CT, MRI, and PET-based approaches.

CT, a well-established method invented by A. M. Cormack in the early 1970s, involves the creation of 3D models from a combination of X-ray images using the Radon transformation. CT is adept at distinguishing between bones and soft tissues due to substantial differences in Hounsfield units (HU). Although bones typically have HU values in the range of 200-600, soft tissues exhibit more minor HU values, falling within the 40-100 HU range. Figure 3.2 shows an example of CT scan results.

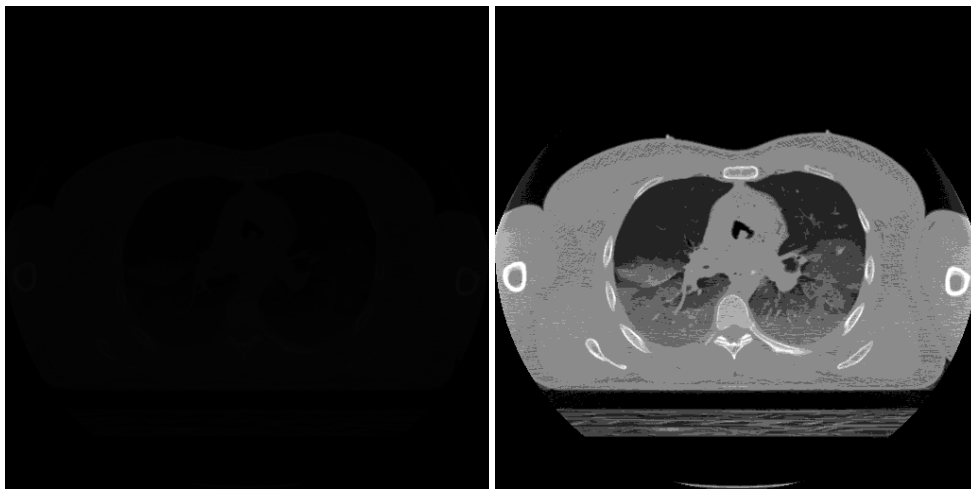


Figure 3.2: CT result. Original data [14] on the left, same data adjusted using the linear transfer function on the right. On the original data, there is hardly something visible (even worse if you read a physical, printed version. On the electronic version, there are at least some features visible). The issue is that the human perception of luminosity is not linear.

Magnetic resonance imaging (MRI), introduced in the early 1980s, relies on measuring the spin echo after applying a strong magnetic pulse. MRI offers several advantages, including the absence of ionising radiation and improved visibility of the soft tissues. However, MRI is often criticised for its longer acquisition times,

which can be inconvenient for physicians and patients. Figure 3.3 illustrates an MRI result.

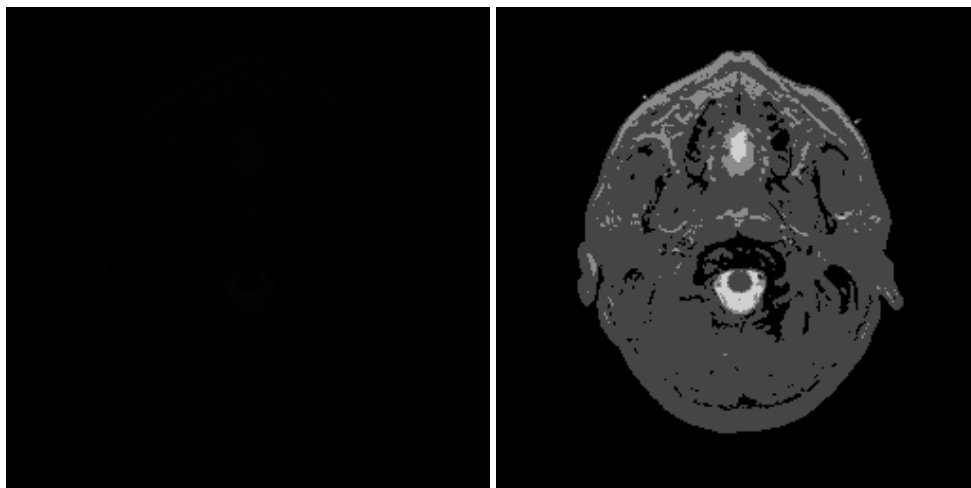


Figure 3.3: MRI result. Original data [14] on the left, the same data adjusted using the linear transfer function on the right. Same phenomena as in Fig. 3.2 occurs here.

A more recent innovation in noninvasive imaging is diffusion-weighted imaging (DWI), introduced in the late 1980s. DWI leverages tissue water diffusion rates for imaging. Diffusion-tensor imaging (DTI), a variation of DWI, employs tensor mathematics to define diffusion. DTI has been used to determine the pennate angle of various human muscles, a critical parameter in musculoskeletal modelling that affects the magnitude of force [15]. However, none of these non-invasive methods can accurately determine muscle attachment areas, as these areas are often invisible or poorly visible on imaging. An alternative approach involves creating personalised bone and muscle models and estimating attachment areas by experts or probabilistic models [16]. In the case of a general model, invasive methods provide more detailed information.

3.4 Invasive techniques

Invasive methods, such as dissection and other surgical procedures, cannot be applied directly to living patients for ethical reasons. Therefore, cadaver experiments are essential to acquire more detailed and accurate data for muscle modelling. These experiments enable the collection of precise measurements and the acquisition of musculoskeletal features that are difficult to discern using non-invasive methods, including muscle tendon separation and attachment areas.

Figure 3.4 provides an example of data obtained through invasive methods, specifically cryosection images of the Visible Human Male, part of the Visible Human Project [14]. In this project, male and female subjects were frozen and cut to

create detailed MRI and CT images. These images serve as invaluable resources for musculoskeletal research. Although the process may seem inhuman and unethical, preserving such data requires a long and complex process of giving authorisation.

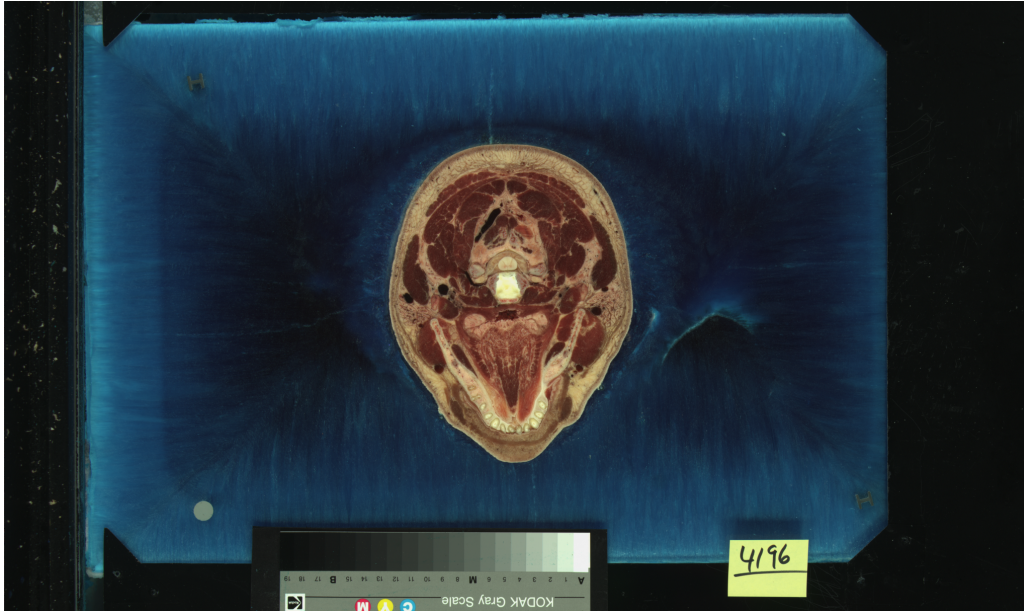


Figure 3.4: The detailed cryosection image of the Visible Human Male [17].

Another invasive approach is described by Fukuda et al. [16], where the hip region was dissected to isolate individual muscles, and the attachment areas were tracked using an optical tracker. A probabilistic model was developed based on data from eight cadaver specimens, although some outliers required manual removal.

Carbone et al. [18] conducted a cadaver study to produce the TLEM 2.0 - Twente Lower Extremity Model. This model included 166 muscle-tendon elements for each leg and was generated by cadaver dissection.

Invasive techniques are also used to measure anatomic fibres and tendons, as these finer details of muscle structure are challenging to capture with non-invasive methods. Lee et al. [15] used cadaveric data to estimate the pennate angle of the muscles, addressing the limitations of non-invasive techniques.

3.5 Physiological signals

Another crucial data source is physiological signals, with electromyography (EMG) as a prominent example. EMG is a diagnostic procedure to measure motor neurone activation signals that control muscle movement. This information is valuable in creating accurate muscle movement models. However, the complexity and variability of muscles between individuals make direct personalised modelling through invasive EMG procedures less preferable. Instead, non-invasive EMG measurements are

commonly used on the skin surface, using sensors placed on the skin, typically with adhesive patches or elastic bands. Although surface EMG measurements are less invasive, they may be less precise due to their inability to access the muscle directly. Because the scope of this dissertation is already broad enough, I recommend that the dear reader find more details about the topic in, e.g., Kalc et al. [19] if interested.

In addition to EMG, movement data is also valuable for muscle modelling. These data are obtained by placing location sensors on the patient and recording their movements during walking, running, or jumping. These movement data can be used for direct kinematics approaches rather than inverse kinematics, which is the primary focus of this dissertation.

The diversity of these data sources may pose challenges, as they may not be perfectly aligned or correspond to each other. Data registration becomes necessary to map data acquired from different modalities or under varying conditions. There are two main classes of data registration: rigid and non-rigid registration. Rigid registration preserves the shape and scale of the transformed object, while non-rigid registration does not. The registration of musculoskeletal models is generally non-rigid (due to the subject displacement between measurements) and has been explored with promising results by Zhao et al. [20]. However, non-rigid registration may introduce self-intersections in the resulting data, showing an attempt to register the 3D muscle surface and surface muscle fibres measured with different modalities using the elastic registration algorithm of Li et al. [21]. The described issue is a complex problem, and while it is briefly introduced here, it deserves further exploration in a separate article or study.

This work focuses primarily on deformation techniques, so a deeper dive into data registration is beyond its scope. For more information on data registration, see references such as [22] and Chapter 4 of the same book.

Estimation approaches

4

Estimation techniques, including both approximation and interpolation¹ play an essential role in creating a smooth model. Even with limited data, these techniques must effectively approximate various model components, such as muscles, bones, muscle fibres, attachment areas, etc. Let us also operate under the assumption that accurate and well-defined knowledge of the model's borders eliminates the need for extrapolation. The estimation techniques used in this chapter are also described graphically in Fig. 4.1.

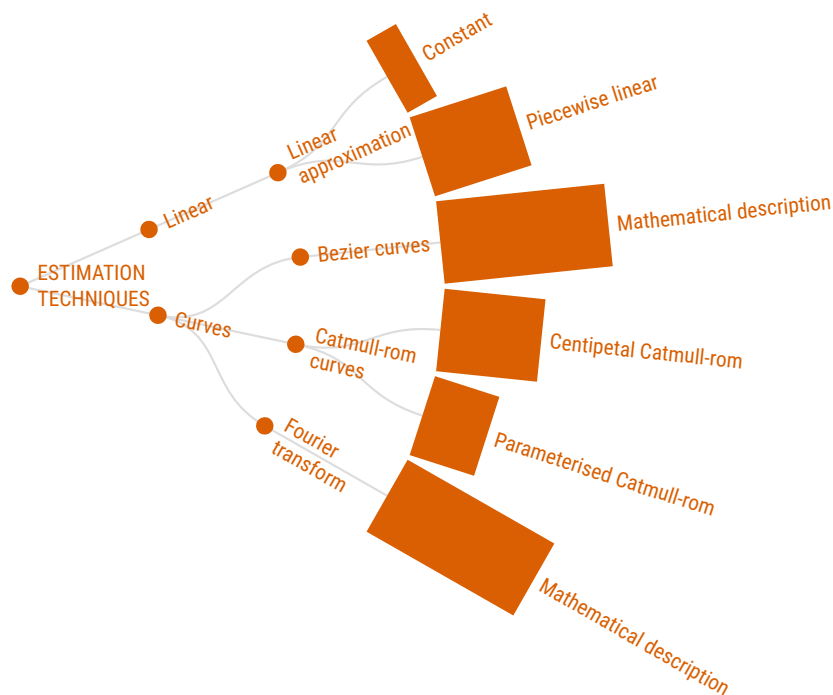


Figure 4.1: The graphical description of the content of this chapter and the depth of description on each topic. Created with the www.flourish.studio

¹Throughout the text, the term "estimation" will be used interchangeably to refer to both approximation and interpolation for simplicity.

An approach to muscle modelling is using Bézier curves, as initially explored by Delp et al. [23] and later by Kohout and Kukacka [24]. However, it has been observed [25] that the muscle fibres modelled can intersect when using Bézier curves. To solve this problem, Kohout and Cholt [25], who introduced the Catmull-Rom spline approximation for muscle fibre modelling to achieve the smoothness of Bézier curves without self-intersections, have proposed Catmull-Rom splines. Although this approach appears successful, estimation using higher smoothness curves could yield better results.

Another significant approach uses the radial basis function (RBF) estimation. RBFs have been designed to address challenges in scattered data estimation and find applications in various fields, including image reconstruction [26], neural networks [27], and surface reconstruction [28], among others. Hardy initially proposed this approach [29], its key advantage being the potential to achieve infinite smoothness (C^∞) when Gaussian RBFs are used as basis functions, contrasting with Bézier or Catmull-Rom, which use polynomial basis functions. RBFs have already been used for muscle modelling, particularly in our research, to estimate muscle attachment areas in [13] and finally to represent the whole muscle shape, as described in this dissertation.

This chapter provides a detailed exploration of these estimation techniques suitable for muscle modelling problems. Each method mentioned will be discussed within a specific dimension (ideal for the best possible explanation). However, it is essential to note that extending these ideas to higher or lower dimensions is generally feasible.

4.1 Constant and piecewise linear estimation

The simplest estimation method is constant estimation, where we assign the value of the nearest independent variable to the unknown independent value. Piecewise linear estimation in one dimension considers the closest values and calculates intermediate values along the straight line connecting these two points. Figure 4.2 illustrates constant (red) and piecewise linear (green) estimations.

However, these estimations have a significant drawback; they are at most C^0 smooth (linear estimation) and can even be discontinuous (constant estimation). Because of that, these estimations do not accurately represent the behaviour of muscles, which are continuous, smooth structures. Such a model would not accurately simulate actual muscles and would not be visually appealing to users.

Higher-dimensional polynomials can be employed to address the issue of low smoothness. An option for using a higher-dimensional polynomial for the approximation is a set of Bézier curves.

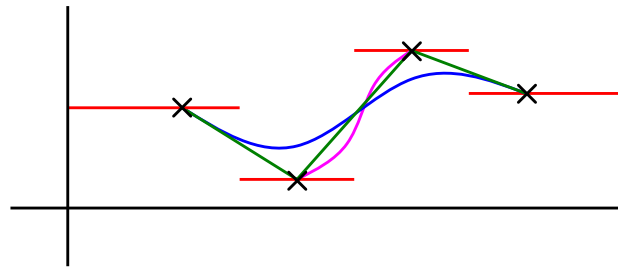


Figure 4.2: Constant (red), piecewise linear (green), Bézier (blue) and Catmull-Rom (pink) interpolation of a set of points.

4.2 Bézier curves

A Bézier curve is defined by two boundary points and a set of control points. A Bézier curve with no control points degenerates into a linear curve, whereas a curve with one control point is termed a quadratic Bézier curve. The cubic Bézier curve has two control points. The first boundary point and the first control point determine the derivative of the curve's start, whereas the last control point and the second boundary point determine the derivative of the curve's end.

The more general recursive definition states that a Bézier curve of degree n is a linear combination of Bézier curves of degree $n - 1$, where each curve omits one of the boundary vertices, and the control points are adjusted accordingly. This recursion ends with Bézier curves of degree 1, which are single points.

The definition can also be expressed as a binomial distribution of all Bézier vertices (boundary and control points), with the probability of success determining the shape of the Bézier curve.

$$P(t) = \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 3 & -3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix} \quad (4.1)$$

The cubic Bézier curve in blue is shown in Figure 4.2. It is important to note that this curve does not pass through the control point, but these control points describe the direction instead.

4.3 Catmull-Rom spline

A Catmull-Rom spline [30][31] is a cubic that interpolates all four given points. It calculates the tangent for each internal vertex based on the previous and next control vertices. However, the tangent for the boundary vertices is not as clearly defined,

but the vertex itself can serve as a substitute. The centripetal variant is expressed in matrix form as follows:

$$P(t) = \frac{1}{2} \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \begin{bmatrix} 0 & 2 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 2 & -5 & 4 & -1 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix} \quad (4.2)$$

When t is within the interval $\langle 0, 1 \rangle$, this formula produces the Catmull-Rom spline between the two middle vertices, P_1 and P_2 . However, the general Catmull-Rom spline introduces a parameter τ :

$$P(t) = \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\tau & 0 & \tau & 0 \\ 2\tau & \tau - 3 & 3 - 2\tau & -\tau \\ -\tau & 2 - \tau & \tau - 2 & \tau \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix} \quad (4.3)$$

The Catmull-Rom spline offers advantages over cubic Bézier curves in three main aspects: it passes through all control vertices, it does not require specifying derivatives at any vertices, and the centripetal parameterisation ($\tau = 0.5$, between uniform $\tau = 0$ and chordal $\tau = 1$) avoids artefacts such as cusps and self-intersections. The Catmull-Rom spline is in Figure 4.2, indicated in pink.

4.4 Discrete Fourier transform

The discrete Fourier transform (DFT) is beneficial when data are equidistantly sampled. In such cases, the DFT decomposes the curve into the sum of individual sinusoidal functions, and if interpolation is needed, these functions are combined. One significant advantage of this approach is that the extrapolation does not diverge, as is common in polynomial estimation. However, a drawback is that the data must be uniformly distant. The formula for DFT is derived from Euler's formula:

$$F_k = \sum_{n=0}^{N-1} P_n e^{-2\pi i k \frac{n}{N}} \quad (4.4)$$

The inverse operation is similar but involves traversing the unit circle in reverse and normalising the result by the number of functions:

$$P_k = \frac{1}{N} \sum_{n=0}^{N-1} F_n e^{2\pi i k \frac{n}{N}} \quad (4.5)$$

This approach produces a C^∞ smooth curve. It is beneficial for equidistantly sampled data with periodic components, such as many medical signals such as EKG,

EEG, and EMG. Extending the DFT to higher dimensions is straightforward: treat each vector component separately, calculate the DFT, and combine these components. Alternative estimation methods should be considered for shapes (of the muscles, bones, etc.) that do not follow equidistant distribution or periodicity, e.g. radial basis function approximation and interpolation (see Chapter 6) showing some promises.

This chapter showed techniques for estimating some values according to others. As stated in the following chapter, these techniques can be used for muscle modelling.

Related work

5

This dissertation describes the fundamental concepts of muscle modelling and some of the most relevant methods. The primary distinction in muscle modelling lies between forward and inverse kinematics approaches. Forward kinematics simulates the electrical excitation of muscle cells, cellular responses, fibre responses, and general muscle responses. In contrast, inverse kinematics affects muscle motion in the opposite direction, requiring knowledge of the desired outcome (e.g., bone motions) to determine how the muscle must change to produce the motion. The ultimate goal, which is not addressed here, involves determining the electrical excitation of all muscle units and, thus, the internal muscle forces, allowing for a direct kinematic simulation of the muscle. However, since this dissertation is already widely focused, the description will stay only on inverse kinematics methods; therefore, further details regarding direct kinematics are omitted. The overview of the topics is (as before) shown in Fig. 5.1.

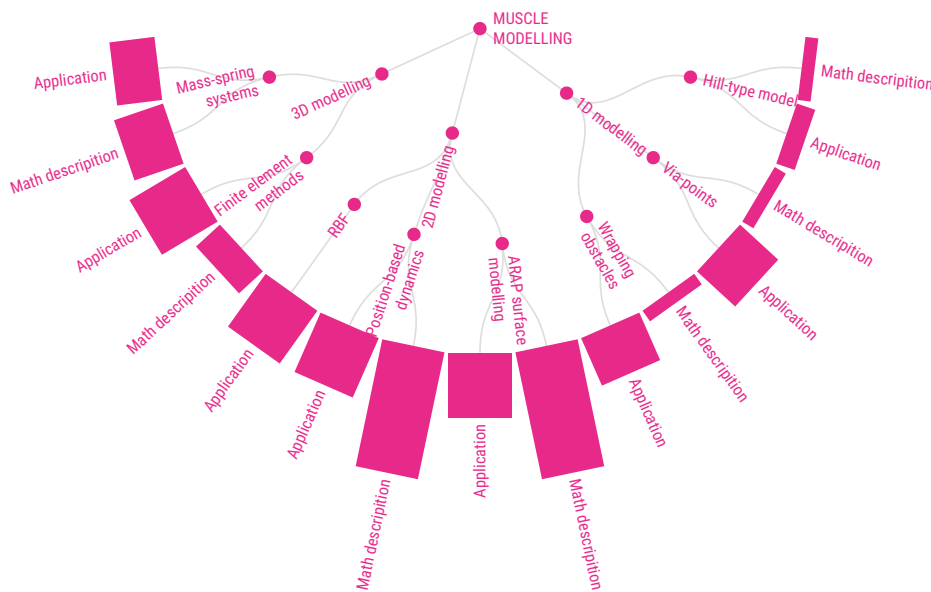


Figure 5.1: The graphical description of the content of this chapter and the depth of description on each topic. Created with the www.flourish.studio.

An extensive review of state-of-the-art methods has recently been conducted by Dereshgi et al. [32], with a previous review by Lee et al. [33]. Although they concisely summarise many existing methods, this report delves deeper into fewer methods. Furthermore, the assertion made by Dereshgi et al. that "Although mechanical properties of muscles such as force, power, and work are well known" [32] appears to overestimate the current state of the field, given the extensive list of recently published papers on the topic.

There are numerous methods available to model human movement and muscle behaviour. These methods can be categorised based on the dimensionality of their operation. Let us first focus on one-dimensional modelling, representing muscles as lines, curves, or sets of lines/curves. The muscle model can be approximated using a limited set of lines or curves that match the orientation of natural muscle fibres due to the anisotropic nature of muscles. Let me start with the simplest and one of the oldest muscle models – the Hill-type muscle model.

5.1 Hill-Type Model

The Hill-type models are based on approximating muscle fibres with a triplet of parallel, serial, and contractile elements. This model represents one of the most basic and possibly the oldest mathematical models of muscle, initially presented by Hill in 1938 [34]. Hill conducted experiments on frog muscles and described muscle contraction dynamics, supported by experiments, with the equation:

$$(F + a)v = b(F_0 - F) \quad (5.1)$$

In this equation, F represents the current muscle force, F_0 is the maximum muscle strength, a (in force units) depends primarily on the maximum strength (and hence indirectly on muscle size) and b (in velocity units) is a constant (assuming constant temperature). At first glance, the seemingly steady variable b increases at higher temperatures, allowing faster muscle contractions, and is also proportional to muscle length l . The variable v denotes the velocity of shortening. Equation (5.1) can also be expressed as follows, implying an inverse relationship between load F and velocity v ; a higher load results in slower movement:

$$(F + a)(v + b) = \text{const.} \quad (5.2)$$

One of the primary challenges with this model is obtaining accurate values for the variables a and b for each muscle in the human body [34]. Additionally, muscle shapes vary between individuals, and muscle cross-sectional areas and lengths change during muscle activity, further affecting the values of a and b .

Hill originally proposed modelling muscle fibres with three elementary units: energy, heat shortening, and external mechanical work, arranged in a series-parallel configuration. The model is depicted in Figure 5.2. The Hill model has been improved over time, with significant additions such as the parallel element [35] in 1989, leading to what are currently known as "Hill-type" models. Another addition is the inclusion of a viscous damping element [36].

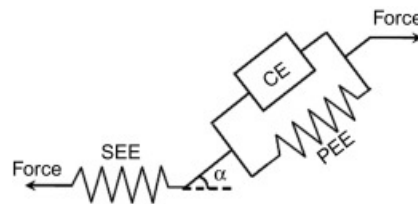


Figure 5.2: Hill-type model of a muscle fiber [37]. PEE = parallel element, SEE = serial element, CE = contractile element, α - pennate angle.

However, in some cases, the pure Hill-type model cannot calculate internal muscle forces. As Modenese et al. pointed out, this model provides a valid representation of a three-dimensional muscle only when the line segments pass through the centroids of the force distribution within the muscle sections [38]. This condition does not apply to many muscles. Furthermore, Martins et al. showed that medical applications involve muscle structures that cannot be reduced to one dimension, such as the pelvic floor and diaphragm [39]. Valente et al. [40] also found that modelling muscles using a single line segment could result in errors of up to 75% in estimating muscle forces (e.g. when modelling the gluteus minimus muscle).

As a result, contemporary methods typically do not use Hill-type fibre models to construct musculoskeletal models. Instead, they incorporate the principles of Hill-type fibre behaviour into different approaches (e.g., position-based dynamics and finite element methods), aiming to either determine variable values or mimic Hill-type model behaviour using fewer parameters.

5.2 Via-points

A via-points approach works with a predefined set of points defining the muscle fibre model. There are many options for defining these points. The most common ones are points directly fixed to a bone, so whenever the bone moves, the point moves accordingly. The second option is that the point is present only if a condition is met (for example, if the joint flexion angle is more significant than x), so the natural shape of the fibre model is partially restored. The third option is a point that may move depending on some state (for example, depending on some angle, typically between two bones), following a predefined curve (see Fig. 5.3).

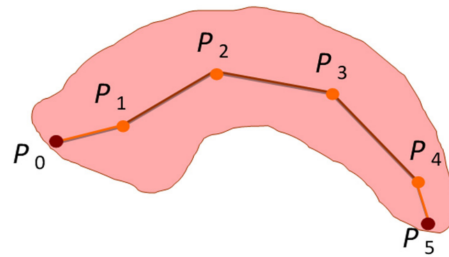


Figure 5.3: Via-points control curve inside a muscle[1]

There are, however, some catches. The first obvious one lies in the definition of via-points. There must be an approach to define these points because user-defined points will be time-consuming, costly, and subjective for the physician. The second main problem is the intersections of muscle models with the closest adjacent bone model when performing an inverse kinematic movement. Similarly, self-intersections should be handled correctly. Furthermore, the resulting muscle fibre model will sometimes not be smooth. Modenese stated that the "straight line representation of the muscles surrounding the hip joint was limiting the accuracy of hip contact force predictions" [38]. The better approach might not go through a set of points but "wrap around" a predefined set of geometric objects [41].

5.3 Wrapping Obstacles

The wrapping obstacles approach has been developed to address some problems of the via-points approach.

The improvement lies in the smoothness of the curve. Unlike the via-points method, a curve that wraps around a sphere can be formed smoothly. Figure 5.4 illustrates wrapping around a cylinder.

The main challenge is that infinitely many curves can wrap around a volumetric object. From these curves, one can select the shortest, the least average curvature, the one with maximum curvature, and so on. Selection depends primarily on the specific application, the required precision, the computing power, and other factors. This issue becomes even more complex with extreme bone arrangements.

The wrapping obstacles approach has been used in muscle modelling by researchers such as Lloyd et al. [43] and Kohout et al. [42]. Lloyd et al. wrapped around an arbitrary geometrical model, dividing the curve into segments with knots connected by elastic forces to prevent them from penetrating obstacles. The latter approach, which is significantly older, is limited to spheres, single cylinders, and sphere-capped cylinders. The issue is that some of the previously described problems persist:

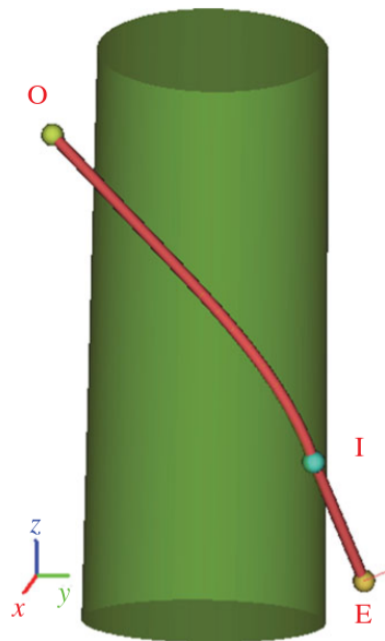


Figure 5.4: Wrapping obstacles approach. A line of action is wrapped around a single cylinder [42].

- The geometric objects must be specified in the same fashion as the via-points.
- In the case of incorrect curve selections, the intersection problem remains.

5.4 Finite Element Method

The Finite Element Method, abbreviated as *FEM*, offers a means to achieve precise physical modelling. It has proven effective in tackling various challenges, such as solving problems related to heat transfer, fluid dynamics, and more. An example of its application is in the work of George-Ghiocel et al. [44], who used the stress-and-strain approximation approach in Storz coupling within fire hose coupling.

In the case of 1D functions, a triangular basis function can be created, whose weighted sum can form an approximative function using a polyline. Those basis functions can be seen in Fig. 5.5.

If we consider 2D triangular bases, solving problems in any dimension requires some form of tessellation of the input space. Although the most common approach involves triangles through Delaunay triangulation, a general division into triangles (or simplices in higher dimensions) is also possible.

The elements are more complex in two dimensions than in the one-dimensional case. By induction, the one-dimensional approach will be extended using the same

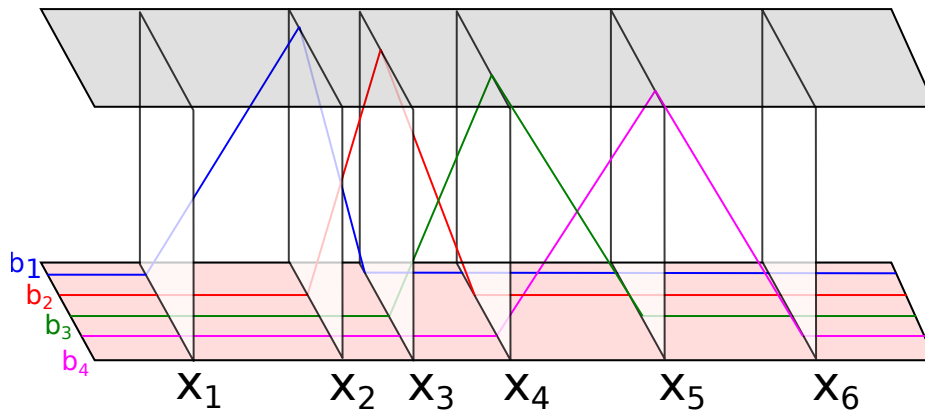


Figure 5.5: Triangular basis functions b_1, \dots, b_4 with values $f(x_1)$ and $f(x_6)$ are fixed (by *Dirichlet* boundary conditions) at zero. Idea adapted from [45].

idea to two dimensions. Each basis function b_i was piecewise linear, with a value of 1 at a specific point, between 0 and 1 (distributed linearly) if the vertex lies in the neighbouring triangles and x_i and zero elsewhere in each sample. See the dissertation thesis [46] for more insight into math properties. The FEM also allows for the use of shapes different from those of simplexes, often employing rectangles, cuboids, and so on. These shapes require other elements (single functions).

5.4.1 FEM in muscle modelling

The finite element method has also been applicable in muscle modelling. This section provides an overview of the relevant approaches in chronological order.

Although multiple FE methods have been published for soft tissue modelling, one of the first papers on musculoskeletal modelling using the FE method is likely from Martins et al. [47], published in 1998. They incorporated the Hill-type model into their FE model, although a coauthor later demonstrated that the Hill-type model may be insufficient in some cases [39]. Their model divided the brachialis muscle into 4050 tetrahedra and assumed constant material properties without considering muscle anisotropy. External forces were applied to an arbitrary part of the muscle ("right end").

Delp and Blemker [23], in 2005, employed a template that was projected onto the target mesh. The projected template was deformed using the finite element method. They used magnetic resonance imaging (MRI) resolution to create the template. They identified the tendon region from MR images to determine boundary conditions, although the complete process was not described in detail.

Boubacker et al. [48] conducted significant work to publish a survey on this topic, providing valuable insights into the problem. However, it should be noted that the study was published in 2006, so some of the information therein may be outdated.

Oberhofer et al. [49], in 2009, used cubic Hermite interpolation functions in their FE approach to ensure C^1 smoothness of the model. They also used the via-point approach (see Section 5.2) for boundary conditions to prevent muscles from penetrating the bones. These via-points were integrated into the FE method, and the objective function included a distance term between landmarks and targets and a Sobolev smoothing constraint.

Kaze et al. [50], in 2017, utilised FE to partition the model using tetrahedra. They derived boundary conditions from anatomical muscle attachment areas and used a mass-spring-like system to simulate tendons (for further details, see Section 5.5.1). Their primary focus was on estimating the maximal strain.

Wei et al. [51], in 2019, used FE to model a human hand, employing the Nolan hyperelastic soft tissue model [52] to accurately represent human skin. Their study mainly focused on analysing the pressure generated by different hand grips.

Currently, several methods use the pure finite element method for various applications. For example, Fougerson et al. [53] applied FE for the analysis of the load of the upper knee socket. In contrast, using FE, Vila Pouca et al. [54] studied muscle fatigue in the pelvic floor. Sun et al. [55] modelled spine movement using FE.

Despite the capability of the described FE methods to produce high-quality results, they present challenges in setup due to the numerous parameters required [56]. Additionally, FE methods are often computationally demanding. For example, Fougerson et al. [53] reported that their approach ran for 40 minutes on a 2-CPU machine, far from real-time simulation. Consequently, this work only briefly introduces FE methods, which deserve more comprehensive coverage. The primary goal of this research is to explore a faster process, ideally capable of real-time simulation, while maintaining comparable results.

Many other methods for muscle modelling, such as position-based dynamics (PBD), mass-spring system (MSS), and as-rigid-as-possible (ARAP), are currently available and promising alternatives that require fewer computational resources while delivering comparable outcomes.

5.5 Other optimization problems

The following methods work on the function minimisation principle. The optimisation is performed to find a geometric model (of muscle, muscle fibre) respecting some restrictions (e.g. volume preservation, shape preservation, fibre length, etc.)

Some of the solutions go against each other. For example, if a muscle is stretched, the fibre length restriction wants to shorten the fibres, but volume preservation wants to extend them to fill more space. Thus, these restrictions are often weighted according to the modelling requirements. There are two common restrictions for all of the techniques. The muscle is attached to a set of bones, so adjacent parts

must be fixed to the corresponding bone surface. In addition, it is not appropriate for the muscle to penetrate the bone. A collision detection and response form has to be implemented to avoid this issue. The following methods will describe what is optimised and how, but these two restrictions may be omitted because they are shared. Also, these restrictions are often threatened as strict (the second one may be a bit relaxed), meaning no attachment displacement or collision should occur.

5.5.1 Mass-Spring System (MSS)

The Mass-Spring System (MSS) functions as its name implies, simulating complex motion by connecting masses (individual points of mass) with virtual springs. Motion propagation occurs through force transfer across a network of these springs. A relatively straightforward equation describes the spring's behaviour:

$$\mathbf{F}_{ij} = -k\mathbf{d}_{ij} \quad (5.3)$$

Here, \mathbf{F}_{ij} represents the force generated by the spring between the i -th and j -th particles, where the variable k denotes the spring constant (the force required to restore the spring to its original shape per unit of spring extension). The displacement of the spring is represented by the variable \mathbf{d}_{ij} .

Although the fundamental concept is not overly complicated, various implementations of this approach exist. A notable implementation comes from Georgii and Westermann [57], who effectively applied this technique to GPU hardware.

Another approach, as presented by Aubel and Thalmann [58], utilises a 1D mass-spring system model for each muscle fibre independently, although with some problems, particularly regarding gaps between muscles. They also introduced angular springs to maintain the angle between the two spring segments. A significant drawback of their work is their somewhat lenient approach to collision detection: "Attempting to handle all muscle-muscle and muscle-bone collisions is unreasonable. In our framework, preventing muscle-bone collisions is easy and fast using repulsive force fields" [58].

A noteworthy contribution in this field is Janak's work [2], which combines the mass-spring system with muscle modelling. Unlike Aubel and Thalmann, Janak's approach operates on muscle fibre models in continuous space. He decomposes the triangular mesh muscle model into a muscle fibre model, uniformly sampling these fibres to obtain nodes (masses), which are then connected by edges (springs). Janak's results show some promise, but a notable challenge lies in his collision detection approach, which approximates the surface mesh with spheres, leading to an "imprecise collision response and partial surface intersection between objects" [2].

Uniform sampling and connection of mass points are depicted in Figure 5.6. It is worth mentioning that the author also considers randomisation and its implications.

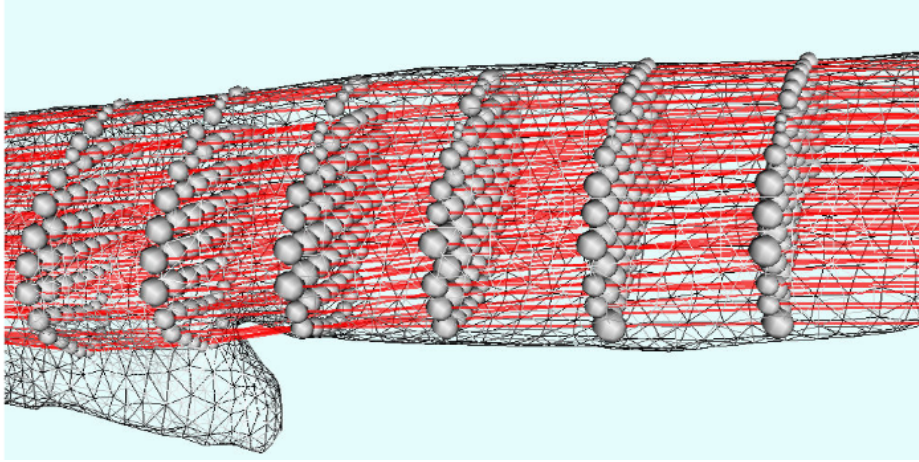


Figure 5.6: Mass-spring system simulating an unspecified human muscle in close detail. An issue arises with the lower portion, which lacks proper approximation. Image source: [59].

5.5.2 Optimization

When using the mass-spring system (MSS), optimisation primarily focuses on achieving the desired muscle geometry while adhering to certain constraints, such as volume and shape preservation. However, some of these constraints may contradict each other, requiring careful consideration and weighting based on modelling requirements.

In all of these MSS techniques, two typical constraints are consistently applied. First, since the muscle is attached to a set of bones, adjacent parts must be fixed to their corresponding bone surfaces. Second, it is essential to prevent the muscle from penetrating the bones. Collision detection and response mechanisms are typically implemented to address this.

5.5.2.1 Optimization for spring length

In the context of MSS, optimisation typically involves finding the optimal spring lengths while satisfying the constraints mentioned earlier. The equations governing the motion of the MSS can be described as follows:

$$m_i \mathbf{x}_i'' + c \mathbf{x}_i' + \sum_{\forall j \in \mathbf{N}_i} \mathbf{F}_{ij} = \mathbf{F}_i^e \quad (5.4)$$

Here, m_i represents the mass of the i -th particle i , \mathbf{x}_i is its position, c is a damping coefficient, \mathbf{N}_i denotes the neighbourhood of the i -th node, \mathbf{F}_{ij} represents the spring force between nodes i and j , and \mathbf{F}_i^e is an external force vector. The damping coefficient c is determined based on the system's properties using the formula [59]:

$$c = 2\sqrt{2m\left(2 + \frac{4}{k_a}\right)} \quad (5.5)$$

Where k_a is the average stiffness. The exact solution to these equations can be obtained by time integration. However, for computational efficiency, an approximate solution using finite differences and a short time step dt is often employed:

$$\mathbf{x}_i(t + dt) = \frac{\mathbf{F}_i^T(t)}{m_i} dt^2 + 2\mathbf{x}_i(t) - \mathbf{x}_i(t - dt) \quad (5.6)$$

$$\mathbf{F}_i^T(t) = \mathbf{F}_i^e - \sum \forall j \in \mathbf{N}_i \mathbf{F}_{ij} - c \frac{\mathbf{x}_i(t) - \mathbf{x}_i(t - dt)}{dt} \quad (5.7)$$

Although this approach can effectively simulate complex motion, it has some limitations. One major challenge is to set up the system due to variations in spring constants k between different springs. Furthermore, this method does not inherently preserve the original volume of the model, which is often a desired outcome in muscle modelling. Addressing this volume preservation issue could significantly increase computational time, as demonstrated by Hong et al. [60].

Despite its potential to simulate complex motion, the Mass-Spring System (MSS) approach requires careful setup. It may not inherently preserve volume, making it a practical but challenging technique for muscle modelling.

5.5.3 ARAP - As-Rigid-As-Possible Deformation

The acronym ARAP, which stands for As-Rigid-As-Possible, represents a technique for determining minimal non-rigid transformations within a surface mesh. Its primary innovation lies in its ability to operate without requiring an internal structure, distinguishing it from methods like Kelnhofer & Kohout [61] that incorporate volume constraints but require the definition of an internal "skeleton."

ARAP has found application in the medical field, as demonstrated by Fasser et al. [62]. They used ARAP to morph a pelvic bone template into subject-specific landmarks. However, their approach overlooks crucial bone properties, such as volume preservation, which the original ARAP method does not inherently ensure.

Wang et al. [63] also explored ARAP but abandoned it due to its inability to produce satisfactory results. They noted issues like non-smooth shapes with spikes resulting from ARAP, among others. Furthermore, their proposed approach did not address the volume preservation requirement.

The essence of As-Rigid-As-Possible deformation is to minimise the model deformation as much as possible. Any non-rigid transformation is penalised through a cost function. This problem is mathematically formulated as solving a system of

linear equations, where the matrix is a discrete Laplace operator of the mesh (after applying boundary conditions, particularly fixed points), and the right-side vector contains second differences of each vertex concerning its local neighbourhood.

5.5.3.1 Laplace Operator

The primary motivation behind the Laplace operator, Δ in ARAP, is its ability to capture significant local shape changes. When applied to a triangular mesh, the continuous space definition is adapted to discrete space, typically called the Laplace-Beltrami operator.

5.5.3.2 Volume Preservation

The original ARAP method, while fast and precise, does not inherently address the preservation of the initial model's volume. Volume preservation is crucial in muscle modelling as muscles do not significantly change their volume while contracting.

To achieve proper volume preservation, scaling the entire model by a scalar value to restore the original volume is straightforward. This scaling factor can be calculated using Equation 5.8:

$$\sqrt[3]{\frac{V_0}{V}} \quad (5.8)$$

Here, V_0 represents the initial volume, and V represents the current volume. The cube root is applied to account for the three-dimensional nature of the scaling.

However, this method may not work when the muscle is attached to multiple bones at specific fixed coordinates, as these points cannot be moved, leading to volume discrepancies. Alternative methods have been explored, such as the one proposed by Aubel and Thalmann [64], which describes the scale factor as the square root of muscle elongation. However, practical experiments have shown that this approach may not yield accurate results in all cases, especially compared to more straightforward methods such as Position Based Dynamics (PBD) [65].

Seylan et al. [66] have employed ARAP for shape deformation with volume preservation. They addressed volume preservation by adding more edges to the mesh, improving the results. However, they acknowledge that some volume loss still occurs. However, Dvorak et al. [67] demonstrated promising results with ARAP and volume tracking for surfaces that vary over time, significantly exceeding previous experiments. While their approach is practical for noiseless data and general meshes, it may require further adaptation for muscle modelling, particularly in addressing specific challenges such as muscle anisotropy, collision detection and response, and execution speed.

5.5.3.3 Optimization

In the ARAP approach, optimisation involves finding the optimal translation and rotation for a point within a given neighbourhood. The primary objective of ARAP is to minimise the energy defined in Equation (5.9)[8]. Any variable marked with an apostrophe (') belongs to the deformed mesh, while variables without it belong to the original mesh.

$$E(S') = \sum_{i=1}^n w_i \sum_{j \in \mathcal{N}(i)} E \left\| \left(\mathbf{p}'_i - \mathbf{p}'_j \right) - \mathbf{R}_i \left(\mathbf{p}_i - \mathbf{p}_j \right) \right\| \quad (5.9)$$

In this equation, E represents the resulting energy, S is the triangular mesh on which the penalisation is calculated, and w_i denotes the weight associated with the connection between each pair of vertices. These weights can be calculated using various methods, such as the cotangent formula (as described in Equation 5.12), the Tutte formula, the Kirchhoff formula, or any other suitable method for the given problem. The variable n represents the total number of vertices in the mesh, and $\mathcal{N}(i)$ denotes the neighbourhood of vertex i (comprising all vertices connected by edges). The position of the i -th vertex is marked as \mathbf{p}_i , and the variable \mathbf{R}_i represents the rigid rotation in the i -th cell.

The optimal translation and rotation for each vertex in the surface mesh within ARAP can be described as follows. Starting with the optimal translation:

- Calculate the centroid $\bar{\mathbf{c}}$ of the original vertices.
- Compute the centroid $\bar{\mathbf{c}}'$ of the transformed vertices.
- The optimal translation is given by $\bar{\mathbf{c}}' - \bar{\mathbf{c}}$.

The centroid is determined as a weighted average coordinate of neighbouring vertices (i.e., vertices connected by edges), with weights specified in Equation (5.12). The optimal rotation can be calculated as follows:

- Change the origins to centroids: $\mathbf{o}'_i = \mathbf{p}'_i - \bar{\mathbf{c}}'$ and $\mathbf{o}_i = \mathbf{p}_i - \bar{\mathbf{c}}$.
- Arrange \mathbf{o}_i in matrix \mathbf{P} ($3 \times N$) and \mathbf{o}'_i into matrix \mathbf{P}' ($3 \times N$).
- Compute the covariance matrix $\mathbf{C} = \mathbf{P}\mathbf{P}'^T$ (3×3).
- Perform Singular Value Decomposition (SVD) on \mathbf{C} : $\mathbf{C} = \mathbf{P}\mathbf{P}'^T = \mathbf{U}\mathbf{\Sigma}\mathbf{V}$.
- The optimal rotation is given by $\mathbf{R}_i = (\mathbf{U}\mathbf{V})^T$.

Sorkine et al. [8] show that alternating between these transformations is sufficient until the error reaches a reasonably small value.

5.5.3.4 Linear equation system

After determining the local translation vectors and rotation matrices, addressing the interdependence of optimal local rotations and translations is essential. While optimal local rotations are independent of each other (each cell's rotation can be calculated independently using the SVD method described above), the same cannot be said for translations. When a vertex moves, it affects all adjacent cells. Sorkine[8] proposed the use of the Laplace-Beltrami operator on the already deformed positions \mathbf{p} (utilising local rotation \mathbf{R}_i) to resolve this issue and minimise the energy (for more details, refer to [8]). This system is denoted as $\mathbf{L}\mathbf{p}' = \mathbf{p}$, and its components are explained in more detail below.

An illustration of the construction of the linear equation system matrix \mathbf{L} is shown in Fig. 5.7. All edge weights are set to "1" for simplicity in this example. The degree of the negative vertex is stored on the main diagonal (representing the sum of all weights in general). For other cases, if an edge exists between the given pair of vertices, the value is "1" (representing the edge weight in general). This definition ensures that the sum in each row and column remains zero. The nonzero elements become more intricate when different weights are introduced (e.g., cotangent weights as shown in Equation 5.12).

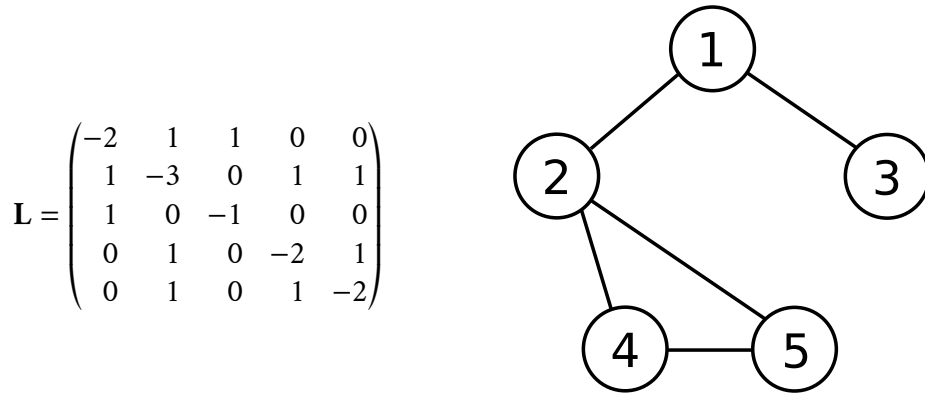


Figure 5.7: An example of the ARAP linear equation system. The Laplace-Beltrami operator \mathbf{L} on the right is derived from the mesh connectivity on the left.

The right-hand side of the equation, denoted as \mathbf{b} , is more complex to express than the matrix \mathbf{L} . Each element of the vector is calculated as follows:

$$\mathbf{b}_i = \sum_{j \in \mathcal{N}(i)} \frac{w_{ij}}{2} (\mathbf{R}_i + \mathbf{R}_j) (\mathbf{p}_i - \mathbf{p}_j) \quad (5.10)$$

If certain vertices are fixed, the number of dependent variables is reduced, leading to a rectangular matrix. To solve the system in this scenario, ignoring or removing the rows corresponding to the fixed points is applicable, or the Ordinary Least Squares approach can be applied to solve the rectangular matrix system as follows:

$$\begin{aligned}
 \mathbf{Ax} &= \mathbf{b} \\
 \mathbf{A}^T \mathbf{Ax} &= \mathbf{A}^T \mathbf{b} \\
 (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{Ax} &= (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{b} \\
 \mathbf{x} &= (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{b}
 \end{aligned} \tag{5.11}$$

5.5.3.5 Edge weights

Edge weights can vary depending on computational demands, accuracy requirements, or specific use cases. The simplest scenario is where $w_{ij} = 1$ (referred to as Kirchhoff), which means that all edges are treated equally. This approach is suitable for equilateral polygons and has minimal computational requirements. However, it can pose challenges when solving the resulting diagonally dominant matrix due to numerical precision issues.

To address the issue, normalising each row can be done by dividing it by the value on the diagonal. This normalisation step results in diagonal elements having a "-1." This normalisation approach may be more suitable for specific numerical methods and helps with compression (as the diagonal values no longer need storage). This method is known as Tutte.

Kirchhoff and Tutte's approaches belong to the combinatorial Laplacian category, which solely considers mesh connectivity. Another approach, the cotangent formula, considers the mesh's geometry. Edge weights are calculated as the average of the cotangents of the angles opposite the given edge, ensuring that longer edges (with wider angles) receive lower weights than shorter ones. Mathematically, it is expressed as follows:

$$w_{ij} = \frac{1}{2} (\cot \alpha_{ij} + \cot \beta_{ij}) \tag{5.12}$$

Variables α_{ij} and β_{ij} represent the angles opposite the edge between the vertices i and j . The ARAP approach has some limitations for muscle modelling, mainly the already discussed volume preservation, collision handling and more. Let us discuss the following approach, called Position-Based Dynamics, which better suits the purposes.

5.5.4 Position-Based Dynamics

Position-based dynamics, often abbreviated as PBD [68], is a fast and widely used approach, primarily in the animation industry, for modelling elastic deformations of objects, including cloth. It has also gained popularity in the realm of physical

simulations. The original PBD algorithm is designed for general objects and does not inherently account for object anisotropy. It takes a manifold surface mesh as input and produces its deformed variant.

An extended version of PBD, known as xPBD, incorporates the concept of elastic potential energy and eliminates the need to specify the time step and iteration count. For a more detailed understanding, the readers can refer to Macklin et al. work [69].

Romeo et al. [56] made the initial significant contribution by applying the PBD algorithm to muscle modelling problems. They recognised the limitations of the traditional finite element method (FEM) and finite volume method (FVM), which, although providing excellent results, lacked qualities such as fast simulation convergence, ease of setup, intuitive controls, and artistic control [56]. Their fundamental idea involved creating an internal structure above the surface mesh to account for muscle anisotropy, aligning with the general direction of muscle fibres. Through an intelligent edge-creation process, they could construct a volumetric model better suited for the PBD algorithm. It is important to note that they used XPBD (eXtended PBD), which incorporates the concept of elastic potential energy.

In 2019, Angles et al. [70] developed a PBD-based approach for muscle modelling. Their method virtually decomposes the muscle into "rods" that approximate muscle fibres. These rods are allowed to change their diameter to preserve volume. The essential contribution of their work was to achieve real-time simulation, a capability that Romeo's approach lacked due to its substantial processing time of approximately 40 seconds per frame [70].

The journey of utilising PBD for muscle modelling continued with the author's master's thesis [12], which expanded the basic PBD approach to include anisotropy considerations. This work coincided with Romeo's article [56], published in the same year. Following the thesis, the article "Fast and Realistic Approach to Virtual Muscle Deformation" [65] extensively tested and integrated the approach into an existing framework. Subsequently, the article "Muscle Deformation using Position-Based Dynamics" [71] further assessed the approach and compared its results with a current FEM approach. A notable advantage of the proposed approach is that it does not require an interior representation, and anisotropy is computed exclusively on the mesh surface using muscle fibres defined on the mesh surface, indicating the fibre direction. During the author's PhD studies, the implementation was integrated into OpenSim, a well-established platform for modelling various physical phenomena. Additionally, the publication [72] extended the previous work, improving collision detection and response methods, based on the bachelor's thesis of a colleague [73].

The output of the PBD algorithm is a deformed triangular mesh suitable for visualisation. However, the malformed muscle must be transformed into a set of fibres to calculate properties like muscle force. This process is called muscle decomposition, and detailed information can be found in Kohout and Kukacka [24] or

Kohout and Cholt [25] articles.

The pseudocode for the PBD algorithm is presented in Listing 1. In this pseudocode, x_i represents the position of each vertex, v_i represents its velocity, Δt denotes the discretization step (smaller values lead to better accuracy), w_i is the inverse of the weight associated with each vertex, and p_i is a "working" position for each vertex. The variable C_i represents a constraint; more details will be provided in the following text.

Algorithm 1 PBD algorithm [12].

```

1: for all vertices  $i$  do
2:   initialise  $\mathbf{x}_i = \mathbf{x}_i^0$ ,  $\mathbf{v}_i = \mathbf{v}_i^0$ ,  $w_i = \frac{1}{m_i}$ .
3: end for
4: loop
5:   for all vertices  $i$  do
6:      $\mathbf{v}_i \leftarrow \mathbf{v}_i + \Delta t w_i \mathbf{f}_{ext}(\mathbf{x}_i)$ 
7:   end for
8:   dampVelocities( $\mathbf{v}_1, \dots, \mathbf{v}_N$ )
9:   for all vertices  $i$  do
10:     $\mathbf{p}_i \leftarrow \mathbf{x}_i + \Delta t \mathbf{v}_i$ 
11:   end for
12:   for all vertices do
13:    generateCollisionConstraints( $\mathbf{x}_i \rightarrow \mathbf{p}_i$ )
14:   end for
15:   loop solverIterations times
16:     projectConstraints( $C_1, \dots, C_{M+M_{coll}}, \mathbf{p}_1, \dots, \mathbf{p}_N$ )
17:   end loop
18:   for all vertices  $i$  do
19:      $\mathbf{v}_i \leftarrow \frac{\mathbf{p}_i - \mathbf{x}_i}{\Delta t}$ 
20:      $\mathbf{x}_i \leftarrow \mathbf{p}_i$ 
21:   end for
22:   velocityUpdate( $\mathbf{v}_1, \dots, \mathbf{v}_N$ )
23: end loop

```

5.5.4.1 Mathematical background

The fundamental mathematical concept behind the PBD algorithm involves iteratively minimising a cost function denoted as C . The specific definitions for each C will be elaborated on later in the text. The cost function is a scalar function with n variables, and the objective is to minimise its value using gradient descent. To do this, we need to calculate the gradient of C . If we ignore the variable mass (a valid assumption for homogeneous muscles), the change in point position $\Delta \mathbf{p}_i$ is determined by Equation (5.13).

$$\Delta \mathbf{p}_i = -\frac{\nabla_{\mathbf{p}_i} C(\mathbf{p}_1, \dots, \mathbf{p}_n)}{\sum_j |\nabla_{\mathbf{p}_j} C(\mathbf{p}_1, \dots, \mathbf{p}_n)|^2} \cdot C(\mathbf{p}_1, \dots, \mathbf{p}_n) \quad (5.13)$$

In this equation, $\Delta \mathbf{p}_i$ represents the movement of a point, which is the opposite direction of the gradient (in the numerator) scaled by the magnitude of the cost function value (in the denominator). The normalization by the sum of gradients ensures that the movement direction falls within the range $\langle 0; 1 \rangle$, as the sum of normalized vector magnitudes should equal one.

When considering point masses, the simple sum of $\Delta \mathbf{p}_i$ becomes a weighted sum, with the weights being the inverses of the masses. The movement is in the opposite direction of the gradient, consistent with the gradient descent approach.

Since multiple cost functions are involved, the problem transforms into a system of nonlinear equations. This system is solved using the Gauss-Seidel method, which treats each equation independently and propagates results between equations, leading to significant speed improvements at the cost of some acceptable errors.

5.5.4.2 Point distance cost function

The primary goal is to preserve the initial distances between each pair of points in the mesh.

This preservation is achieved similarly to a mass-spring system [59]. The cost function is zero if the distance between a pair of points matches the initial distance at the beginning of the simulation. If the distance becomes longer or shorter, the cost increases.

Equation (5.14) defines the distance constraint [68]. The function C is a scalar cost function with two variables (points), and its value quantifies the deviation from the initial distance. The vector variables \mathbf{p}_1 and \mathbf{p}_2 represent temporary point positions (as seen in Listings 1), and d represents the original distance between the given pair of points.

$$C(\mathbf{p}_1, \mathbf{p}_2) = |\mathbf{p}_1 - \mathbf{p}_2|_2 - d \quad (5.14)$$

To proceed with further calculations, determining the gradient of the cost function C w.r.t. both \mathbf{p}_1 and \mathbf{p}_2 is necessary.

Gradient of the Norm of the Vector Difference First, let's determine the gradient of the L_2 norm from Equation (5.14). The gradient for point \mathbf{p}_1 is calculated as shown in Equation (5.15), and a similar derivation can be performed for \mathbf{p}_2 .

$$\begin{aligned}
 \nabla_{\mathbf{p}_1} |\mathbf{p}_1 - \mathbf{p}_2|_2 &= \nabla_{\mathbf{p}_1} \sqrt{(p_{1x} - p_{2x})^2 + (p_{1y} - p_{2y})^2 + (p_{1z} - p_{2z})^2} \\
 &= \frac{1}{2 |\mathbf{p}_1 - \mathbf{p}_2|_2} \nabla_{\mathbf{p}_1} \left((p_{1x} - p_{2x})^2 + (p_{1y} - p_{2y})^2 + (p_{1z} - p_{2z})^2 \right) \\
 &= \frac{\left[\frac{\partial(p_{1x}-p_{2x})^2}{\partial p_{1x}} \quad \frac{\partial(p_{1y}-p_{2y})^2}{\partial p_{1y}} \quad \frac{\partial(p_{1z}-p_{2z})^2}{\partial p_{1z}} \right]}{2 |\mathbf{p}_1 - \mathbf{p}_2|_2} \\
 &= \frac{2 \left[\frac{\partial(p_{1x}-p_{2x})}{\partial p_{1x}} \quad \frac{\partial(p_{1y}-p_{2y})}{\partial p_{1y}} \quad \frac{\partial(p_{1z}-p_{2z})}{\partial p_{1z}} \right]}{2 |\mathbf{p}_1 - \mathbf{p}_2|_2} = \frac{\mathbf{p}_1 - \mathbf{p}_2}{|\mathbf{p}_1 - \mathbf{p}_2|_2}
 \end{aligned} \tag{5.15}$$

Resulting gradient of the cost function C is (using previous derivation 5.15) shown in 5.16.

$$\begin{aligned}
 \nabla_{\mathbf{p}_1} C(\mathbf{p}_1, \mathbf{p}_2) &= \nabla_{\mathbf{p}_1} (|\mathbf{p}_1 - \mathbf{p}_2|_2 - d) = \frac{\mathbf{p}_1 - \mathbf{p}_2}{|\mathbf{p}_1 - \mathbf{p}_2|_2} = \mathbf{u} \\
 \nabla_{\mathbf{p}_2} C(\mathbf{p}_1, \mathbf{p}_2) &= \nabla_{\mathbf{p}_2} (|\mathbf{p}_1 - \mathbf{p}_2|_2 - d) = \frac{\mathbf{p}_2 - \mathbf{p}_1}{|\mathbf{p}_1 - \mathbf{p}_2|_2} = -\mathbf{u}
 \end{aligned} \tag{5.16}$$

These cost function derivations represent the direction vectors of the edges formed by the involved points. Intuitively, this is the correct way to adjust the distance between points, as moving along the directional vector is logical. The result of this concept is further illustrated in Figure 5.8.

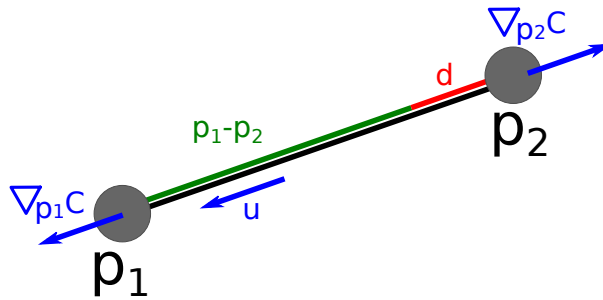


Figure 5.8: Visualization of the cost associated with distance change. The gradient direction is highlighted in blue.

5.5.4.3 Volume preservation

Another essential feature of muscles is the preservation of their initial volume. In the context of the musculoskeletal system, especially muscles, this requirement is reasonable, as muscles typically do not change their volume due to a constant density.

We must first establish a method to calculate the volume of a triangular, closed, and manifold mesh. While the mesh may have various shapes, including more complex ones for higher genera, solving this problem is elegant and straightforward. Initially, we select an arbitrary point in space, often chosen as $(0, 0, 0)$ for simplicity. Alternatively, the centre of gravity may be selected for numerical stability. Then, for each triangle in the mesh, we calculate the volume of the tetrahedron formed by the triangle and the arbitrary point. The volume of the tetrahedron is one-sixth of the parallelepiped volume formed by three vectors originating from the arbitrary centre point and ending at each vertex of the triangle. This calculation is expressed in Equation (5.17) with the arbitrary point at the origin of the coordinate system.

Sometimes, the volume may become negative, which is also a desired phenomenon because it allows for the subtraction of concave parts or parts that form a higher genus in the mesh.

$$V_{tetra} = \frac{1}{6} \begin{vmatrix} a_x & b_x & c_x \\ a_y & b_y & c_y \\ a_z & b_z & c_z \end{vmatrix} = \mathbf{a} \cdot (\mathbf{b} \times \mathbf{c}) \quad (5.17)$$

Vectors \mathbf{a} , \mathbf{b} , and \mathbf{c} represent the edges of the tetrahedron, all sharing a common vertex. In this context, the origin is used as the arbitrary point, simplifying the calculation and allowing us to utilize the vertexes \mathbf{p} of the triangular mesh instead.

An illustrative example is provided in Figure 5.9. By taking the cross product of vectors \mathbf{a} and \mathbf{b} , we obtain vector \mathbf{d} (depicted in pink), which has a length equivalent to the area of the yellow region. Subsequently, by taking the dot product of this vector with vector \mathbf{c} , we can determine the volume of the entire parallelepiped. This procedure is also known as the "triple product." One-sixth of the resulting parallelepiped volume corresponds to the desired volume of the tetrahedron.

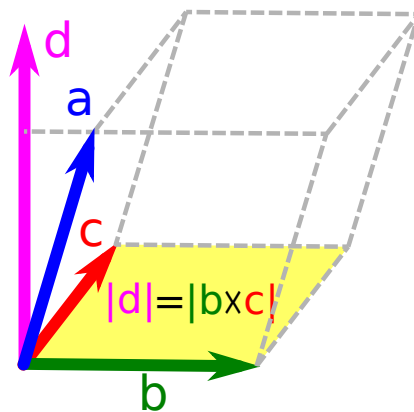


Figure 5.9: Volume of the tetrahedra formed by vectors \mathbf{a} , \mathbf{b} and \mathbf{c} .

We can now also determine the cost function of the volume change. The equation is shown on 5.18.

$$C(\mathbf{p}_1, \dots, \mathbf{p}_n) = \sum_{i=1}^m \left(\mathbf{p}_{t_1}^i \cdot \left(\mathbf{p}_{t_2}^i \times \mathbf{p}_{t_3}^i \right) \right) - kV_0 \quad (5.18)$$

In the equation, the variable m represents the total triangle count, and $\mathbf{p}_{t_1}^i$ denotes the first vertex of the i -th triangle in the mesh. The variable V_0 signifies the initial volume of the mesh, which ensures that the cost function value will be zero if it matches this initial volume. An optional parameter k is included, allowing us to modulate the volume over time, although it is currently set to 1.

The derivative of the cost function can be derived either from Equation (5.18) or from the determinant sum in Equation (5.17). For simplicity, let's select a mesh vertex to calculate the derivative. We will focus on a single triangle for this derivation. The total gradient is obtained by applying the sum rule of derivatives, as shown in Equation (5.19).

$$\nabla_{\mathbf{p}_i} C(\mathbf{p}_1, \dots, \mathbf{p}_n) = \begin{vmatrix} p_{ix} & p_{iy} & p_{iz} \\ p_{jx} & p_{jy} & p_{jz} \\ p_{kx} & p_{ky} & p_{kz} \end{vmatrix} + \dots = \mathbf{p}_j \times \mathbf{p}_k + \dots \quad (5.19)$$

The equation in (5.20) implies equality between the derivative in a vertex triangle and the cross product of the other two points. If we put all triangles together, the derivative of a vertex equals the cross product of each pair of vertices sharing the triangle with the selected vertex. Mathematically speaking, as shown in (5.20).

$$\nabla_{\mathbf{p}_i} C(\mathbf{p}_1, \dots, \mathbf{p}_n) = \sum_{h=1}^t \mathbf{p}_j \times \mathbf{p}_k; \quad i \neq j \neq k \quad (5.20)$$

Variable t represents the number of triangles that share the vertex with index i . Vertices with indices i , j , and k are all part of the same triangle. The control variable h is not required for this specific case.

In his research, Janak [59] did not consider volume preservation, which could potentially enhance his findings. However, the mathematics behind volume preservation in the mass-spring system is exceedingly complex [60].

5.5.4.4 Shape preservation

The final constraint is to maintain the initial shape of the muscle. One approach is to preserve the dihedral angle between each pair of adjacent triangles as much as possible.

It is also essential to describe the calculation of the dihedral angle between two adjacent triangles. Given vertices as \mathbf{p}_1 , \mathbf{p}_2 , \mathbf{p}_3 , and \mathbf{p}_4 , vertices with indices one and two are shared, while index three belongs to the first triangle only. Additionally,

index four belongs to the second triangle. Using the cross product, we obtain triangle normals as shown in Equation 5.21.

$$\begin{aligned}\mathbf{u} &= (\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_1) \\ \mathbf{n} &= \frac{\mathbf{u}}{|\mathbf{u}|_2}\end{aligned}\quad (5.21)$$

The vector \mathbf{u} is introduced to simplify the subsequent derivation. This vector shares the same direction as the normal but may not necessarily have a unit length.

Vertices \mathbf{p}_1 , \mathbf{p}_2 , and \mathbf{p}_3 belong to the first triangle, for which the normal is being calculated. A similar calculation will be performed for the second triangle.

Consequently, the angle between two adjacent triangles can be determined. Initially, both normals (\mathbf{n}_1 and \mathbf{n}_2) are required. Subsequently, the angle between these two normals is obtained using the definition of the dot product. Equation 5.22 presents the definition and derivation.

$$\begin{aligned}\cos \varphi &= \frac{\mathbf{n}_1 \cdot \mathbf{n}_2}{|\mathbf{n}_1|_2 |\mathbf{n}_2|_2} = \mathbf{n}_1 \cdot \mathbf{n}_2 \\ \varphi &= \arccos(\mathbf{n}_1 \cdot \mathbf{n}_2)\end{aligned}\quad (5.22)$$

The next step involves defining a cost function. The same methodology will be applied to create volume and distance cost functions, where the initial angle values are subtracted from the current ones. Equation 5.23 provides the formula for this cost function.

$$\begin{aligned}C(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4) &= \varphi - \varphi_0 \\ &= \arccos(\mathbf{n}_1 \cdot \mathbf{n}_2) - \varphi_0 \\ &= \arccos\left(\frac{(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_1)}{|(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_1)|_2} \cdot \frac{(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_4 - \mathbf{p}_1)}{|(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_4 - \mathbf{p}_1)|_2}\right) - \varphi_0\end{aligned}\quad (5.23)$$

In the equation, φ_0 represents the initial angle, calculated according to Equation 5.22.

Deriving this equation is a bit more complex. Initially, we leverage the property that the cost function value is translation-invariant, allowing us to move the pair of triangles so that one of the vertices lies on the origin of the coordinate system.

The first vertex is positioned at the origin by translating each point using the vector $-\mathbf{p}_1$, significantly simplifying the orthonormal vector calculation for both triangles. When the entire system is translated, the dihedral angle remains unchanged.

This transformation is also represented in Equation 5.24, where vectors with apostrophes are those translated by $-\mathbf{p}_1$.

$$\begin{aligned}
 \mathbf{n}_1 &= \frac{(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_1)}{|(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_1)|_2} = \frac{\mathbf{p}'_2 \times \mathbf{p}'_3}{|\mathbf{p}'_2 \times \mathbf{p}'_3|_2} \Leftrightarrow \mathbf{p}'_1 = 0 \\
 \mathbf{n}_2 &= \frac{(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_4 - \mathbf{p}_1)}{|(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_4 - \mathbf{p}_1)|_2} = \frac{\mathbf{p}'_2 \times \mathbf{p}'_4}{|\mathbf{p}'_2 \times \mathbf{p}'_4|_2} \Leftrightarrow \mathbf{p}'_1 = 0
 \end{aligned} \tag{5.24}$$

Finally, utilising the derivative of the function arccos (defined as $\frac{d}{dx} \arccos x = -\frac{1}{\sqrt{1-x^2}}$), we can compute the gradients for each of the cost functions using Equation 5.25. Since we have chosen \mathbf{p}_1 to be at the origin, it will be defined differently than the second shared vertex, \mathbf{p}_2 . However, \mathbf{p}_3 and \mathbf{p}_4 will exhibit similarities due to the system's symmetry. Notably, the gradient is independent of the absolute location of the object, negating the need for any reverse translation.

$$\begin{aligned}
 d &= \mathbf{n}_1 \cdot \mathbf{n}_2 \\
 \nabla_{\mathbf{p}'_4} C &= -\frac{1}{\sqrt{1-d^2}} (\nabla_{\mathbf{p}'_4} (\mathbf{n}_2) \cdot \mathbf{n}_1) \\
 \nabla_{\mathbf{p}'_3} C &= -\frac{1}{\sqrt{1-d^2}} (\nabla_{\mathbf{p}'_3} (\mathbf{n}_1) \cdot \mathbf{n}_2) \\
 \nabla_{\mathbf{p}'_2} C &= -\frac{1}{\sqrt{1-d^2}} (\nabla_{\mathbf{p}'_2} (\mathbf{n}_1) \cdot \mathbf{n}_2 + \nabla_{\mathbf{p}'_2} (\mathbf{n}_2) \cdot \mathbf{n}_1) \\
 \nabla_{\mathbf{p}'_1} C &= -\sum_{i=2}^4 \nabla_{\mathbf{p}'_i} C
 \end{aligned} \tag{5.25}$$

The fundamental concept behind calculating the gradients of the orthonormal of the triangles involves breaking down the normals into cross-products, which can also be expressed using matrix calculus. Each element of the matrix can then be solved independently. This concept is elaborated on in the case of \mathbf{n}_2 orthonormal at the vertex \mathbf{p}'_4 .

The subsequent step is to determine the partial derivative of the normalised cross-product.

Normalized cross-product partial derivative. First, I will replace the cross product with the vector \mathbf{r} (as demonstrated in Equation 5.26) to simplify the subsequent derivation process.

$$\mathbf{r} = \mathbf{p}'_2 \times \mathbf{p}'_4 \tag{5.26}$$

The subsequent step involves calculating solely the partial derivative of the cross product without normalisation, as illustrated in Equation 5.27. The result of this

calculation is a matrix denoted as \mathbf{A} . When multiplied by a vector, this matrix yields the cross product of the given vector and the vertex \mathbf{p}'_2 .

$$\begin{aligned} \nabla_{\mathbf{p}'_4} \mathbf{r} &= \begin{bmatrix} \frac{\partial p'_{2y} p'_{4z} - p'_{4y} p'_{2z}}{\partial p'_{4x}} & \frac{\partial p'_{2y} p'_{4z} - p'_{4y} p'_{2z}}{\partial p'_{4y}} & \frac{\partial p'_{2y} p'_{4z} - p'_{4y} p'_{2z}}{\partial p'_{4z}} \\ \frac{\partial p'_{4x} p'_{2z} - p'_{2x} p'_{4z}}{\partial p'_{4x}} & \frac{\partial p'_{4x} p'_{2z} - p'_{2x} p'_{4z}}{\partial p'_{4y}} & \frac{\partial p'_{4x} p'_{2z} - p'_{2x} p'_{4z}}{\partial p'_{4z}} \\ \frac{\partial p'_{2x} p'_{4y} - p'_{4x} p'_{2y}}{\partial p'_{4x}} & \frac{\partial p'_{2x} p'_{4y} - p'_{4x} p'_{2y}}{\partial p'_{4y}} & \frac{\partial p'_{2x} p'_{4y} - p'_{4x} p'_{2y}}{\partial p'_{4z}} \end{bmatrix} \\ &= \begin{bmatrix} 0 & -p'_{2z} & p'_{2y} \\ p'_{2z} & 0 & -p'_{2x} \\ -p'_{2y} & p'_{2x} & 0 \end{bmatrix} = \mathbf{A} \end{aligned} \quad (5.27)$$

We then consider the norm, as indicated in Equation 5.28. The outcome remains the cross product, but now it is associated with the orthonormal vector \mathbf{n}_2 .

$$\begin{aligned} \nabla_{\mathbf{p}'_4} |\mathbf{r}|_2 &= \nabla_{\mathbf{p}'_4} \sqrt{r_x^2 + r_y^2 + r_z^2} = \frac{1}{2|\mathbf{r}|_2} \nabla_{\mathbf{p}'_4} (r_x^2 + r_y^2 + r_z^2) \\ &= \frac{1}{2|\mathbf{r}|_2} \begin{bmatrix} \frac{\partial r_x^2 + r_y^2 + r_z^2}{\partial p'_{4x}} & \frac{\partial r_x^2 + r_y^2 + r_z^2}{\partial p'_{4y}} & \frac{\partial r_x^2 + r_y^2 + r_z^2}{\partial p'_{4z}} \end{bmatrix} \\ &= \frac{1}{|\mathbf{r}|_2} \begin{bmatrix} r_y p'_{2z} - r_z p'_{2y} & -r_x p'_{2z} + r_z p'_{2x} & r_x p'_{2y} - r_y p'_{2x} \end{bmatrix} \\ &= \frac{\mathbf{r} \times \mathbf{p}'_2}{|\mathbf{r}|_2} = \frac{|\mathbf{r}|_2 \mathbf{n}_2 \times \mathbf{p}'_2}{|\mathbf{r}|_2} = \mathbf{n}_2 \times \mathbf{p}'_2 = \mathbf{b} \end{aligned} \quad (5.28)$$

Last but not least, combining both partial derivatives of Equation 5.27 and Equation 5.28 is essential. To achieve this, we employ the derivative quotient rule, which results in the outcome presented in Equation 5.29.

$$\begin{aligned} \nabla_{\mathbf{p}'_4} \frac{\mathbf{r}}{|\mathbf{r}|_2} &= \frac{\mathbf{A} |\mathbf{r}|_2 - \mathbf{r} \cdot \mathbf{b}}{|\mathbf{r}|_2^2} = \frac{1}{|\mathbf{r}|_2} (\mathbf{A} - \mathbf{n}_2 \cdot \mathbf{b}) \\ &= \frac{1}{|\mathbf{p}'_2 \times \mathbf{p}'_4|_2} \left(\begin{bmatrix} 0 & -p'_{2z} & p'_{2y} \\ p'_{2z} & 0 & -p'_{2x} \\ -p'_{2y} & p'_{2x} & 0 \end{bmatrix} - \mathbf{n}_2 \cdot (\mathbf{n}_2 \times \mathbf{p}'_2) \right) \end{aligned} \quad (5.29)$$

Similarly, the other partial derivative in Equation 5.25 can be obtained, differing only in indices and signs. The result of this partial derivative is then incorporated into Equation 5.25, determining the final partial derivative of the cost function.

Following the same procedure as other constraints, the adjustments will follow Equation 5.13, as all necessary information is already available.

Figure 5.10 also visually represents the entire concept. For simplicity, only two triangles are depicted. The normal vectors of both triangles are indicated in green

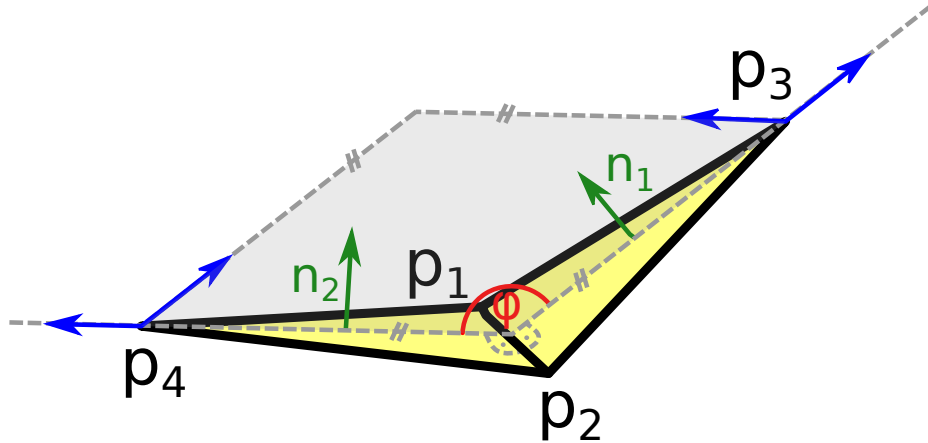


Figure 5.10: Angle φ preservation between two triangles (dihedral angle).

(\mathbf{n}_1 and \mathbf{n}_2). The gradients at vertices \mathbf{p}_3 and \mathbf{p}_4 are shown for both vertices using two blue vectors, forming a surface (in grey) where these vertices are allowed to move. It is important to note that movement in other directions would not alter the angle between the triangles. The gradients at the midpoint were not illustrated due to the complexity of the \mathbf{p}_1 and \mathbf{p}_2 gradients.

In this chapter, we have described seven significant approaches to muscle modelling: the Hill-type model, Via-points, Wrapping obstacles, Finite element method, Mass-spring system, ARAP, and PBD. The following table 5.1 summarises these methods, including a (partially subjective) comparison of speed and precision.

Algorithm	Muscle model form	Speed	Precision	Reference
Hill-type	Single line	+++	---	[34][35][74]
Via-points	Polyline	++	--	[1][41]
Wrap. Obst.	Complex curve	+	-	[42][43]
FEM	Volumetric model	---	+++	[47][48][23]
MSS	Volumetric model	++	-	[57][58][2]
ARAP	Surface model	++	?	[8][64][67]
PBD	Surface model ¹	++	?	[68][56][71]

Table 5.1: Comparison of the muscle modelling approaches.

¹With the exception in the paper from Romeo et al. [56], where they introduced volumetric data to improve their results.

Radial basis functions

6

This chapter mainly focuses on describing the radial basis functions in the muscle modelling domain and, more generally. The main outline of this chapter is visualised in Fig. 6.1, so the main parts resemble an exploration of the existing RB functions, centre point placement and shape parameter selection.

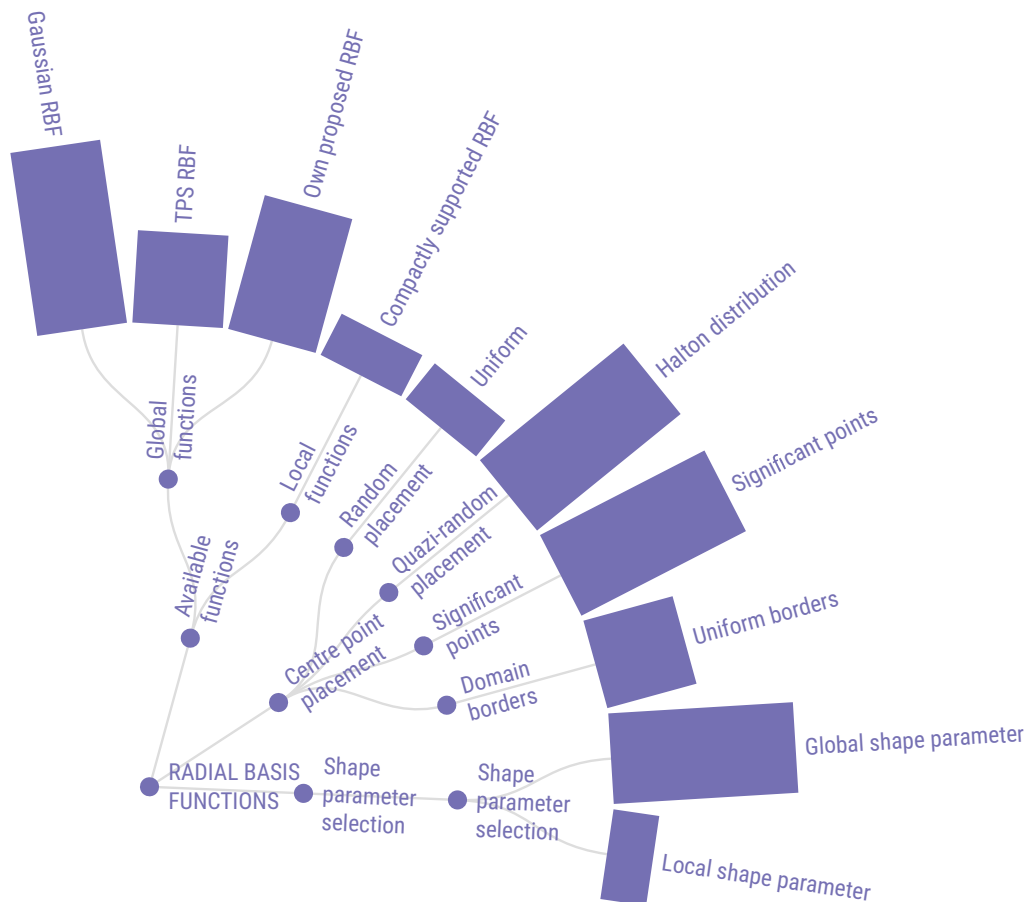


Figure 6.1: The graphical description of the content of this chapter and the depth of description on each topic. Created with the www.flourish.studio.

The challenge of interpolating and approximating scattered data is prevalent in various engineering and research domains. The challenge is exemplified by the work of Oliver et al. [75], who applied the kriging interpolation method to geographical data. Similarly, Kaymaz [76] demonstrated the utility of this approach in addressing structural reliability issues. The technique is also used in modelling, as shown by Sakata et al. [77] in their work on wing structures and in the creation of metamodels by Joseph et al. [78]. Furthermore, Radial Basis Function (RBF) methods are applicable in solving partial differential equations, particularly in engineering-related problems.

The RBF methodology, introduced initially by Hardy [79] and later refined [80], has seen continuous development and modification over time. Majdisova et al. [81] have contributed by proposing various placement methods within this framework. There has also been significant research into the behaviour of *shape parameters* (described further in the text) in RBF methods, including efforts by Wang et al. [82] to find optimal parameters, explorations by Afiatdoust et al. [83] into this area. The use of different local shape parameters as investigated by Cohen et al. [84], Sarra et al. [85], and Skala et al. [86]. The mathematical formulation of the RBF approach is presented as follows:

$$f(\mathbf{x}) = \sum_{i=1}^N \lambda_i \varphi(\|\mathbf{x} - \mathbf{P}_i\|) \quad (6.1)$$

The RBF approach is a linear combination of a set of RBFs (Radial Basis Functions) denoted as φ , with centres at points \mathbf{P}_i . These RBFs are adjusted to interpolate the vertices \mathbf{P}_i using appropriate weights λ_i . This equation can be viewed as a linear system equation $\mathbf{A}\mathbf{x} = \mathbf{b}$, where the matrix \mathbf{A} is composed of values of the RBF function, λ_i represents an unknown vector, and $f(\mathbf{x})$ will be populated with known values at each of the input points.

A reduced number of RBFs can be used to achieve an approximation, making the problem overdetermined and less straightforward. It can then be solved, for instance, using the ordinary least squares (OLS) approach while also considering some associated drawbacks, as mentioned in works such as Skala and Kansa [87].

$$f(\mathbf{x}) = \sum_{i=1}^M \lambda_i \varphi(\|\mathbf{x} - \xi_i\|) \quad (6.2)$$

6.1 Available functions

A radial basis function is defined by its dependency on the distance from a central point. This function can be applied in spaces of any dimension, but it is commonly

expressed in terms of the distance (denoted as r) rather than as a function of a spatial coordinate.

A broad range of RBF functions are generally categorized into local and global. Local RBFs have a characteristic feature where their value becomes zero beyond a certain distance, effectively limiting their influence. In contrast, global RBFs lack this restriction and maintain influence regardless of distance. This discussion will initially focus more on the characteristics and applications of local RBFs.

6.1.1 Local functions

The local RBF functions are limited by a sphere of influence, and beyond this sphere, it evaluates to zero. Some RBFs with those properties are listed in Table 6.1.

ID	Function	ID	Function
1	$(1 - r)_+$	2	$(1 - r)_+^3(3r + 1)$
3	$(1 - r)_+^5(8r^2 + 5r + 1)$	4	$(1 - r)_+^2$
5	$(1 - r)_+^4(4r + 1)$	6	$(1 - r)_+^6(35r^2 + 18r + 3)$
7	$(1 - r)_+^8(32r^3 + 25r^2 + 8r + 3)$	8	$(1 - r)_+^3$
9	$(1 - r)_+^3(5r + 1)$	10	$(1 - r)_+^7(16r^2 + 7r + 1)$

Table 6.1: Typical examples of local RBF functions - compactly-supported RBF. The "+" sign means that every nonpositive value is set to zero instead. [88].

Their main advantage is computational. Due to their small influence, they produce a better conditioned linear equation system (LES). Suppose the order of vertices reflects the spatial position of the original vertices. In that case, the LES becomes diagonally dominant, which is particularly useful for most solvers (Gaussian elimination, Jacobi method, LU decomposition method and more) to obtain more accurate results.

6.1.2 Global functions

The scope of the global RBFs is not limited, so their influence can be infinite. The infinitely broad scope is particularly advantageous in situations where, for example, an RBF centre point change should influence the whole space. Table 6.2 lists some of the most notable ones.

Name	Function
Gaussian RBF	$e^{-\alpha r}$
Thin-plate spline	$r^2 \log r$
An RBF proposed in [89]	$r^2 (r^\alpha - 1)$

Table 6.2: Some of the most notable global RBFs.

The advantage of the global function over the local one is that the single change of a single parameter (one centre point, for example) would influence the whole space. The disadvantage of some is the presence of a shape parameter α . The parameters define the variance (indirectly) of the function. The higher the value, the higher the variance and the further the influence. However, there is also an inverse proportion between the shape parameter and the computational stability of the LES. If we take it to the extreme, the tiny shape parameter will lead to nonzero values only for a vertex with itself, producing a very much solvable LES. On the other hand, a nearly infinite shape parameter would lead to a situation where no matter how far apart two vertices would be, the value would stay the same, leading to the limit to a singular and constant matrix.

6.2 Polynomial extension

Consider the scenario where we must approximate a nonzero constant function using the RBF approach. Without a polynomial extension, accurately approximating such a function would be challenging, as the centre points would need to be distributed uniformly throughout the entire function domain. Therefore, a polynomial extension is introduced to handle functions that exhibit constant- and polynomial-like behaviour.

The extension involves incorporating a polynomial approximation into the RBF equation, expanding the matrix by the same number of rows and columns as the degree of the polynomial used. In the case of a function with a degree of $2\frac{1}{2}D$, this is achieved as follows:

$$\begin{bmatrix} \varphi_{11} & \dots & \varphi_{1N} & 1 & x_1 & y_1 \\ \vdots & \ddots & \vdots & 1 & \vdots & \vdots \\ \varphi_{N1} & \dots & \varphi_{NN} & 1 & x_N & y_N \\ 1 & 1 & 1 & 0 & 0 & 0 \\ x_1 & \dots & x_N & 0 & 0 & 0 \\ y_1 & \dots & y_N & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_N \\ a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} f_1 \\ \vdots \\ f_N \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (6.3)$$

Here, φ_{ij} represents the value of the Radial Basis Function (RBF) based on the distance between the vertex i and the vertex j , while x_i and y_i denote the coordinates of the centre points. The coefficients include a_0 as the constant coefficient and a_1, a_2 as the linear coefficients of the linear expression $a_0 + a_1x + a_2y$. Variables f_i represent the function values of the given vertices, and λ_i means the unknown weight of the RBF to be determined.

This matrix can be expressed more succinctly as follows:

$$\begin{bmatrix} \mathbf{A} & \mathbf{P} \\ \mathbf{P}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \lambda \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \mathbf{0} \end{bmatrix} \quad (6.4)$$

In this formulation, \mathbf{A} represents an RBF submatrix, \mathbf{P} stands for a polynomial submatrix, \mathbf{a} is a subvector containing polynomial coefficients and \mathbf{f} is a subvector containing the values of the interpolated or approximated function.

Usually, in practice, filling this equation yields satisfactory results. However, the underlying theoretical issue arises from using different units within the matrix. φ_{ij} represents some form of distance that has passed through the RBF function (with units somewhat unknown in the case of the Gaussian RBF, akin to "exponential metres" (e^m) when all constants are excluded). On the other hand, x_i and y_i are typically measured in units related to the function domain, commonly in metres (m). This amalgamation of distinct units raises theoretical challenges rarely addressed in the literature [90].

6.3 Centre point distribution

The fundamental issue at hand revolves around determining the optimal placement of the centre points ξ_i . There are several commonly employed options, each with its own set of advantages and drawbacks.

6.3.1 Iterative greedy search

In many mathematical scenarios, iterative methods often compete with direct methods. The search for centre points is no exception. The most straightforward iterative approach involves initially searching for a single centre point, which reduces the discrepancy between the approximated function and the approximated one. Subsequently, more centres are sought within the space defined by the difference between these two functions to minimise this difference as much as possible. This iterative process continues until the overall error reaches an acceptable level or the available number of centre points is exhausted.

The primary advantage of this method is that the approximant preserves the essential features of the original function. Furthermore, the shape parameter can be adjusted at each step of the greedy search to achieve a better fit. However, the disadvantage lies in its computational complexity, as each "guess" entails solving a potentially significant linear equation and the summation of Radial Basis Functions (RBFs). Some RBFs can be computationally expensive to evaluate.

6.3.2 Grid distribution

Grid centre point distribution entails arranging all centre points in a grid format, which can take various forms, such as regular, cartesian, rectilinear, or curvilinear grids. Although the grid setup is straightforward, it presents significant challenges. The most prominent challenge is the potential for poor conditioning of the RBF matrix [91] if the shape parameter is not carefully selected. Another drawback is that it does not capture the intrinsic features of the interpolated or approximated function.

6.3.3 Halton distribution

The Halton distribution [92] represents a quasi-random point distribution. Its original version is one-dimensional and operates within the interval $(0, 1)$ (scalable by a constant). The sequence's elements are defined as:

$$\text{Halton}_k(p) = \sum_{i=0}^{\lfloor \log_p k \rfloor} \frac{1}{p^{i+1}} \left(\left\lfloor \frac{k}{p^i} \right\rfloor \bmod p \right) \quad (6.5)$$

Here, p is an arbitrary prime number, and k represents the index of the sequence element. One can achieve this in the case of multidimensional sequences by selecting multiple prime numbers, resulting in a vector of Halton sequences with different prime values for each dimension. For example, the Halton sequence $[2, 3]$ begins with $\left[\left[\frac{1}{2}, \frac{1}{3} \right], \left[\frac{1}{4}, \frac{2}{3} \right], \left[\frac{3}{4}, \frac{1}{9} \right] \right]$. Essentially, it partitions the $(0, 1)$ interval into p subintervals of equal size and outputs the boundary points. Subsequently, each subspace is subdivided, yielding additional boundary points. A breadth-first approach is employed for traversal, resulting in a more evenly spread distribution than a depth-first (recursive) approach.

The same sequence can be generated by expressing k in base p , inverting it, and placing it after the decimal point. For example, when $p = 2$:

$$0.1_2 = \frac{1}{2}, 0.01_2 = \frac{1}{4}, 0.11_2 = \frac{3}{4}, 0.001_2 = \frac{1}{8}, 0.101_2, 0.011_2, 0.111_2 \dots \quad (6.6)$$

The Halton sequence offers a significant advantage by exhibiting fewer regularities, which helps avoid ill-conditioned matrices and ensures excellent interval coverage. However, a notable drawback is that the endpoints (i.e., 0 and 1) must be explicitly included in most cases, as they are not part of the sequence by default.

6.3.4 Distribution concerning the original function

An entirely justifiable approach is to select the centre points based on specific features of the original function. The likely candidates are the minima and maxima

since RBFs often exhibit extrema at the centre points. Additionally, inflexion points can be included to cover a broader range of features.

We have previously addressed this issue in a different context in [3], which resulted in the "sophisticated placement of radial basis functions significantly improving the quality of the RBF approximation" [3]. The primary takeaway from this research was to minimise the use of grid or equidistant centre point distributions whenever possible.

6.4 An approximation example

The Radial Basis Function (RBF) concept can be effectively illustrated through a simple example. This example involves approximating four signal values using merely two global RBFs, though this concept is capable of broader application. We'll begin by considering our dataset, which might comprise, for instance, five evenly spaced values within the range from 0 to 1:

x	0	0.25	0.5	0.75	1
$f(x)$	-1	-2	0	2	1

The initial step in this process is to select an appropriate Radial Basis Function (RBF). The choice of RBF should ideally be based on the origin of the data. In this case, for the sake of illustration, we will opt for the Gaussian RBF. Next, we need to determine the positions for the two RBF centres, which can be done by identifying local extrema, such as at $x = 0.25$ and $x = 0.75$, or by referring to locations mentioned earlier in the text. The final requirement for constructing the linear equation system is to establish the shape parameters for each RBF. These parameters can be experimentally determined; a single global parameter is often chosen. Fortunately, we can use insights from the immediate vicinity of the centre point to make an effective approximation. Note that this approach initially disregards the influence of a given RBF on vertices other than its immediate neighbours. At the same time, this is a valuable starting point, and subsequent adjustments are necessary for improved accuracy. Both functions should accommodate the values at the boundaries, influenced by the values at the centre point. Therefore, we aim to find a shape parameter that would cause the value of the Gaussian RBF to decrease by half when the domain shifts by 0.25. This calculation is relatively straightforward.

$$\varphi(r) = e^{-\alpha r^2} = d, \quad \alpha = -\frac{\log d}{r^2} \quad r = 0.25, d = 0.5 \quad \alpha = -16 \log 0.5 \approx 11.09 \quad (6.7)$$

The three components, the choice of radial basis function, the determination of RBF centres and the establishment of shape parameters, are essential to construct the linear equation system:

$$\mathbf{A} = \begin{bmatrix} \varphi(\|0 - 0.25\|_2^2) & \varphi(\|0 - 0.75\|_2^2) \\ \varphi(\|0.25 - 0.25\|_2^2) & \varphi(\|0.25 - 0.75\|_2^2) \\ \varphi(\|0.5 - 0.25\|_2^2) & \varphi(\|0.5 - 0.75\|_2^2) \\ \varphi(\|0.75 - 0.25\|_2^2) & \varphi(\|0.75 - 0.75\|_2^2) \\ \varphi(\|1 - 0.25\|_2^2) & \varphi(\|1 - 0.75\|_2^2) \end{bmatrix} = \begin{bmatrix} \varphi(0.25^2) & \varphi(0.75^2) \\ \varphi(0) & \varphi(0.5^2) \\ \varphi(0.25^2) & \varphi(0.25^2) \\ \varphi(0.5^2) & \varphi(0) \\ \varphi(0.75^2) & \varphi(0.25^2) \end{bmatrix} = \quad (6.8)$$

$$= \frac{1}{512} \begin{bmatrix} 256 & 512 & 256 & 32 & 1 \\ 1 & 23 & 256 & 512 & 256 \end{bmatrix}^T$$

When dealing with an overdetermined system of equations, as in the case of approximating data points, Ordinary Least Squares (OLS) is a common method used to find the best possible solution. Here's how OLS is applied in this context:

$$\mathbf{Q} = \mathbf{A}^T \mathbf{A} = 2^{-18} \begin{bmatrix} 256 & 512 & 256 & 32 & 1 \\ 1 & 23 & 256 & 512 & 256 \end{bmatrix} \begin{bmatrix} 256 & 1 \\ 512 & 32 \\ 256 & 256 \\ 32 & 512 \\ 1 & 256 \end{bmatrix} = 2^{-18} \begin{bmatrix} 394241 & 98816 \\ 98816 & 394241 \end{bmatrix} \quad (6.9)$$

$$\mathbf{c} = \mathbf{A}^T \mathbf{b} = \frac{1}{512} \begin{bmatrix} 256 & 512 & 256 & 32 & 1 \\ 1 & 23 & 256 & 512 & 256 \end{bmatrix} \begin{bmatrix} -1 \\ -2 \\ 0 \\ 2 \\ 1 \end{bmatrix} = \frac{1}{512} \begin{bmatrix} -1215 \\ 1215 \end{bmatrix} \quad (6.10)$$

The weights of both RBFs can be found as a solution of the matrix \mathbf{Q} on the left side and the vector \mathbf{c} on the right side. The resulting vector $[\lambda_1, \lambda_2]$ provides the weights for each RBF, which, when applied, should best fit the data in a least-squares sense:

$$2^{-18} \begin{bmatrix} 394241 & 98816 \\ 98816 & 394241 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = \frac{1}{512} \begin{bmatrix} -1215 \\ 1215 \end{bmatrix} \quad (6.11)$$

$$\begin{bmatrix} 394241 & 98816 \\ 98816 & 394241 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = \begin{bmatrix} -622080 \\ 622080 \end{bmatrix} \quad (6.12)$$

When employing a linear equation system solver, such as the Gaussian Elimination method, the solution obtained is $[\lambda_1, \lambda_2] = [-13824/6565, 13824/6565]$.

The final step involves reconstructing the original function by integrating these parameters.

$$\begin{bmatrix} \varphi(0.25^2) & \varphi(0.75^2) \\ \varphi(0) & \varphi(0.5^2) \\ \varphi(0.25^2) & \varphi(0.25^2) \\ \varphi(0.5^2) & \varphi(0) \\ \varphi(0.75^2) & \varphi(0.25^2) \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = \frac{1}{512} \begin{bmatrix} 256 & 1 \\ 512 & 32 \\ 256 & 256 \\ 32 & 512 \\ 1 & 256 \end{bmatrix} \frac{13824}{6565} \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \frac{27}{6565} \begin{bmatrix} -255 \\ -480 \\ 0 \\ 480 \\ 255 \end{bmatrix} = \tag{6.13}$$

$$= \begin{bmatrix} \frac{-1377}{1313} \\ \frac{1313}{-2592} \\ \frac{-2592}{1313} \\ 0 \\ \frac{1377}{1313} \\ \frac{1313}{2592} \\ \frac{2592}{1313} \end{bmatrix} \approx \begin{bmatrix} -1.04874 \\ -1.97411 \\ 0 \\ 1.97411 \\ 1.04874 \end{bmatrix} \tag{6.14}$$

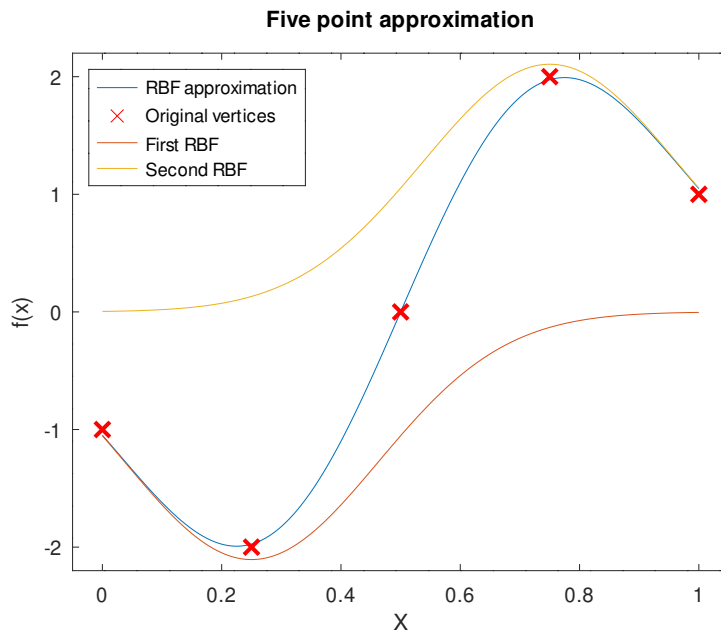


Figure 6.2: Illustration of an RBF approximation (depicted in blue) closely matching five original data points, represented as red crosses. The image also displays individual RBFs, coloured in red and yellow, whose collective sum creates the blue curve.

The approximation closely mirrors the original data points $[-1, -2, 0, 2, 1]$. The approximation result is visually represented in blue in Fig. 6.2. The approximation's mean square error is around 10^{-3} , indicating a highly satisfactory level of accuracy. Further improvements in approximation could be achieved by adjusting the shape parameter and allowing more flexibility in the placement of the centre points

within the domain (in this case, the centres were positioned at 0.25 and 0.75 for computational simplicity).

6.5 RBFs for muscle modelling

Our in-depth studies of RBF [89, 3, 91, 93, 94, 88] have revealed its potential for approximation purposes. To my knowledge, RBF approximation methods have not been widely applied in muscle modelling. We have contributed to the use of the RBF approach in muscle modelling in the article titled "Non-planar Surface Shape Reconstruction from a Point Cloud in the Context of Muscle Attachment Estimation" [13], where RBF is used to reconstruct surfaces from sets of attachment points.

This chapter has explored various approximation and interpolation methods that may find applications in data preprocessing and as integral components of muscle modelling techniques. The first method discussed is the finite element method (FEM), which is used primarily to model various physical phenomena, including muscle modelling. It should be noted that even the FEM relies on an underlying approximation method to operate effectively.

Novel approach

7

The research of existing methods described in chapter 5 shows some potential to improve the existing processes. The study of RBF outlined in the previous chapter also indicates its potential for muscle modelling. Therefore, the possibility of using RBF in muscle modelling became increasingly apparent and led to multiple publications from us, mainly the last paper described in chapter 9.13. This chapter will extend and generalise the ideas from that paper so the technique can be used elsewhere.

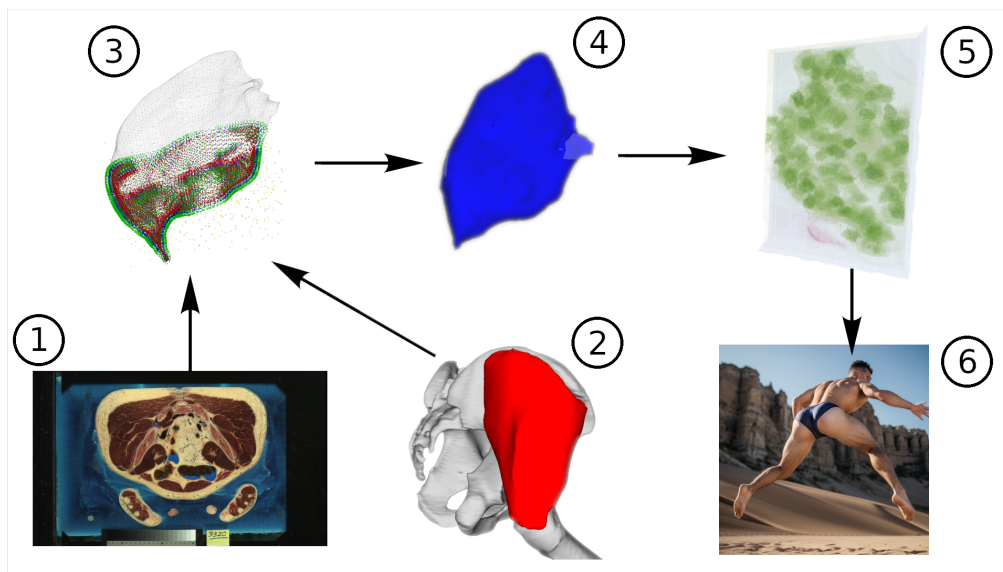


Figure 7.1: The pipeline of the proposed approach. The volumetric (1) [14] or already triangulated (2) data are sampled (3) and then used for the creation of an implicit RBF shape (4). On this shape, initial and dynamic volumetric curvature is estimated (5) to produce a dynamic shape preserving its original curvature .

The creation of the static model, represented in steps 3 and 4 in Figure 7.1, is elaborated upon in Algorithm 2. It's important to emphasize that the pseudocode provided is intended for a clearer understanding of the pipeline and not for direct implementation. The metric in line 15 could include mean squared error, Jaccard

index, or another relevant measure. A weighted mix of MSE and JI emerged as the most influential metric throughout the research and testing phases, further detailed in Section 9.13.

Algorithm 2 The pseudocode of the static RBF surface creation.

```

1:  $S \leftarrow loadSurface(file)$ 
2:  $\vec{b} \leftarrow AABBbox(S)$  ▷ Read bounding box of surface  $S$ 
3:  $\vec{x}, \vec{y}, \vec{z} \leftarrow \vec{s}_i \in \vec{S}_v$  ▷ Set all vertices forming the surface  $S$ 
4: append  $Halton(\vec{b}, 2, 3, 5)$  to  $\vec{x}, \vec{y}, \vec{z}$  ▷ Add Halton points in bounding box  $\vec{b}$ 
5: append  $S_f + d\vec{n}_f$  to  $\vec{x}, \vec{y}, \vec{z}$  ▷ Add points  $d$  away (out) from the  $S$ 
6: append  $S_f - d\vec{n}_f$  to  $\vec{x}, \vec{y}, \vec{z}$  ▷ Add points  $d$  away (in) from the  $S$ 
7:  $f = discregrid(\vec{x}, \vec{y}, \vec{z})$  ▷ Create SDF on top of the samples
8: for  $i \in 1 \dots \text{number of centres}$  do ▷ RBF centre selection
9:    $C \leftarrow []$  ▷ Array of possible RBF evaluations inside  $\vec{b}$ 
10:  for  $j \in 1 \dots \text{number of samples}$  do ▷ Sample selection
11:    for  $\alpha \in [0.001, 0.01, 1, 10, 100 \dots]$  do ▷ Shape parameter selection
12:      append  $RBF(\vec{x}_j, \vec{y}_j, \vec{z}_j, \alpha)$  to  $C$  ▷ Evaluation of the RBF inside  $\vec{b}$ 
13:    end for
14:  end for
15:   $f \leftarrow \min(|f - C_i|)$  ▷ Subtracting the best RBF from SDF
16: end for

```

7.1 Sample placement

Since many samples would lead to an extensive RBF equation system to solve, samples have to be chosen with patience. The method of sample placement to generate a static RBF surface arises from extensive research on RBF behaviour, further elaborated in Chapter 9. The samples are strategically positioned at critical vertices, including the surface itself and those adjacent to it, both internally and externally. These points of importance have already been proposed by Carr et al. [4] and work well for this scenario.

Additionally, the research identified the necessity to cover empty spaces partially, leading to the adoption of the sparse Halton sequence for its superior performance in space coverage compared to alternatives like random or equidistant methods. It improves the RBF approximation because it allows, e.g. a wide RBF at the centre of the shape or a wide RBF outside to cut out a big part of a space.

Figure 7.2 illustrates the placement of an RBF on the gluteus maximus, with critical vertices highlighted in blue, red, and green and the Halton sequence in yellow. These points are intersected by a plane to display both the vertices and the original triangular mesh, ensuring comprehensive coverage of the entire shape.

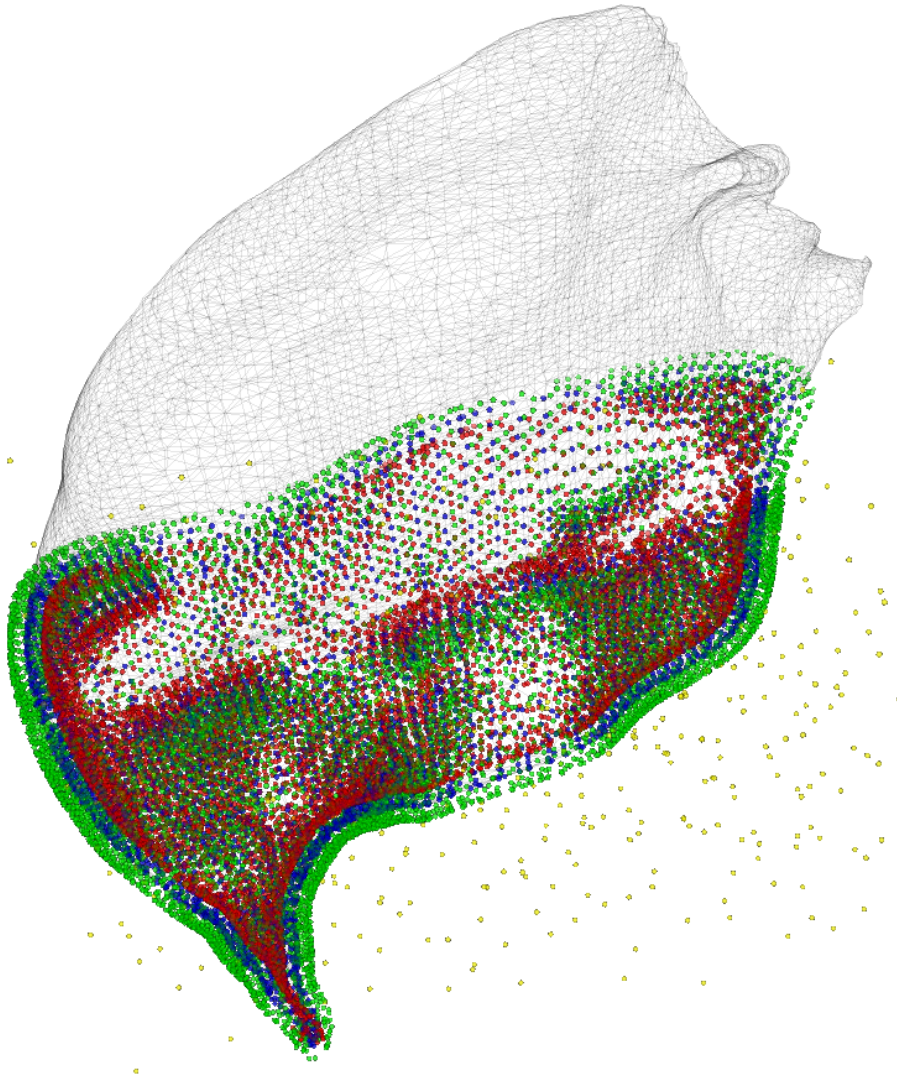


Figure 7.2: The RBF centre placement possibilities. The yellow points are distributed using Halton (2,3,5) distribution, blue points are the vertices lying on the surface, and the red and green vertices lie a fixed distance from the surface inwards/outwards.

7.2 Centre placement and shape selection

The centres are strategically positioned within a select group of samples; generally, while they could be located anywhere, confining them to a specific subset accelerates the process considerably. The method employed for placing these centres adopts a greedy strategy, which builds upon the approach suggested by Carr et al. [4]. However, it has been enhanced to achieve substantially better results.

During each phase of the iterative process, a single Radial Basis Function (RBF) is chosen to minimise the approximation error (as suggested by the metric out-

lined in chapter 9.13). Minimising is accomplished by thoroughly evaluating every conceivable location within the sample space and all the shape parameters from a pre-established collection to ensure the most accurate error reduction.

7.3 Dynamics

After constructing a static RBF model, discussions about its dynamics are possible. Specifically, in inverse kinematics muscle modelling, parts of the muscle attach to adjacent bones, while the majority of the muscle, which is mobile, needs its position estimated. As before, in the case of the static model creation, the dynamics are described using pseudocode in 3.

Algorithm 3 The pseudocode of the dynamic RBF surface movement.

```

1:  $S \leftarrow \text{loadSurface}(\text{file})$  ▷ Load already created static surface
2:  $R_{\text{init}} \leftarrow \text{RBFsurface}(S)$  ▷ Create static RBF surface
3:  $\vec{b} \leftarrow \text{AABBbox}(S)$  ▷ Read bounding box of surface  $S$ 
4:  $f_{\text{init}} \leftarrow \text{curvature}(R_{\text{init}}, \vec{b})$  ▷ Estimate initial curvature of  $R_{\text{init}}$  in the region
5: while  $S$  deforms do
6:    $R \leftarrow \text{RBFsurface}(S)$  ▷ Create deformed static RBF surface
7:   repeat ▷ Perform gradient descent
8:      $f \leftarrow \text{curvature}(R, \vec{b})$  ▷ Estimate current curvature in the region
9:      $\epsilon = |f - f_{\text{init}}|$  ▷ Calculate current curvature error
10:     $R \leftarrow \text{mathModel}(R, f_{\text{init}}, f)$  ▷ Fix the shape by the math model
11:   until  $\epsilon \leq \text{threshold}$ 
12: end while

```

Determining the appropriate criteria for these mobile parts is crucial to ensure they deform realistically. For triangular mesh models, most methods, such as Position-Based Dynamics (PBD), Finite Element Method (FEM), and Mass-Spring System (MSS), rely on mesh connectivity to maintain structural features like dihedral angles (in PBD) and edge lengths (in PBD, FEM, MSS). Real-world criteria are also simulated by delving into detail, such as muscle volume preservation in PBD, which leverages tessellation insights to manipulate mesh vertices. For the RBF model, the primary aim is to maintain a smooth surface. Therefore, the optimal feature to preserve is the model's initial curvature, which ensures that the free parts do not excessively bend, wobble, or produce any other unwanted local artefacts. Also, by preserving the curvature, the initial volume of the muscle should not change much because the curvature field created at the start forces the shape to hold the volume inside itself.

7.4 Mathematical model

The derivation of the mathematical model is straightforward. It involves describing the surface as a sum of RBF implicit functions and the curvature preservation condition. At first, the innovative mathematical model starts by declaring the general notation of the RBF approximation:

$$f(\mathbf{x}) = \sum_{i=1}^N \lambda_i \varphi(\|\mathbf{x} - \xi_i\|) = \sum_{i=1}^N \lambda_i \varphi(r_i) \quad (7.1)$$

Because the curvature calculation involves estimating the Hessian matrix, the first step is to find the gradient of the RBF approximation, which will be needed afterwards. It can be described as follows:

$$\nabla f(\mathbf{x}) = \sum_{i=1}^N \lambda_i \nabla \varphi(r_i) = \sum_{i=1}^N \lambda_i \frac{\partial \varphi}{\partial r_i} \frac{\partial r_i}{\partial x_j} = \sum_{i=1}^N \lambda_i \frac{r_{ij}}{r_i} \frac{\partial \varphi}{\partial r_i}, \quad r_{ij} = x_j - \xi_{ij} \quad (7.2)$$

The gradient evaluation was the first step in estimating the Hessian matrix of the approximator. Second partial derivatives declare the Hessian of a function, generally as:

$$\mathbf{H}(f(\mathbf{x})) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \frac{\partial^2 f}{\partial x_1 \partial x_3} & \cdots \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \frac{\partial^2 f}{\partial x_2 \partial x_3} & \cdots \\ \frac{\partial^2 f}{\partial x_3 \partial x_1} & \frac{\partial^2 f}{\partial x_3 \partial x_2} & \frac{\partial^2 f}{\partial x_3^2} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \quad (7.3)$$

In our case of RBF approximation, where each second partial derivative can be evaluated from the first ones in (7.2), a single Hessian matrix element can be declared as:

$$\frac{\partial^2 f}{\partial x_j \partial x_k} = \sum_{i=1}^N \lambda_i \left(\frac{\partial^2 \varphi}{\partial r_i^2} \frac{r_{ij}^2}{r_i^2} + \frac{\partial \varphi}{\partial r_i} \left(\frac{\delta_{ij}}{r_i} - \frac{r_{ij}^2}{r_i^3} \right) \right) \quad (7.4)$$

The Kronecker delta function δ_{ij} indicates that it behaves differently on and off the diagonal. Luckily, for further evaluation, the derivation depends solely on the elements on the main diagonal.

7.4.1 Mean curvature

The mean curvature of a model describes the shape of the model, so it should be beneficial to preserve it. The mean curvature is defined as a mean eigenvalue of the Hessian or as the mean of the trace of the Hessian:

$$\kappa_\mu(\mathbf{H}(f(\mathbf{x}))) = \kappa_{\mu f} = \bar{\lambda}_{\mathbf{H}} = \frac{\text{Tr}(\mathbf{H})}{D} = \frac{1}{D} \sum_{i=1}^N \lambda_i \left(\frac{\partial^2 \varphi}{\partial r_i^2} + \frac{\partial \varphi}{\partial r_i} \frac{D-1}{r_i} \right) \quad (7.5)$$

The subsequent step involves determining the cost function, which is essentially the squared L2 norm between the new and original curvatures across the entire space:

$$C_f = \int \dots \int \|\kappa_{\mu_f} - \kappa_{\mu_{f_i}}\|_2^2 dx_1 \dots dx_d = \int_{\mathbb{R}^d} \|\kappa_{\mu_f} - \kappa_{\mu_{f_i}}\|_2^2 d\mathbf{x} \quad (7.6)$$

One may employ the gradient descent method to discover the optimal values of ξ_i . Initially, we must compute the gradient of the curvature w.r.t. ξ_i :

$$\nabla \kappa_{\mu_f} = \left[\frac{\partial \kappa_{\mu_f}}{\partial \xi_{i1}} \quad \frac{\partial \kappa_{\mu_f}}{\partial \xi_{i2}} \quad \frac{\partial \kappa_{\mu_f}}{\partial \xi_{i3}} \quad \dots \right] \quad (7.7)$$

After the detailed following derivation, the gradient ultimately used for the gradient descent method to evaluate the new shape would take the form:

$$\begin{aligned} \frac{\partial \kappa_{\mu_f}}{\partial \xi_{kj}} &= \frac{\partial}{\partial \xi_{kj}} \frac{1}{D} \sum_{i=1}^N \lambda_i \left(\frac{\partial^2 \varphi}{\partial r_i^2} + \frac{\partial \varphi}{\partial r_i} \frac{D-1}{r_i} \right) = \frac{1}{D} \sum_{i=1}^N \lambda_i \left(\frac{\partial g(\xi_i)}{\partial \xi_{kj}} \right) = \\ &= \frac{\lambda_k}{D} \frac{\partial g(\xi_k)}{\partial \xi_{kj}} = \frac{\lambda_k}{D} \frac{\partial}{\partial \xi_{kj}} \left(\frac{\partial^2 \varphi}{\partial r_k^2} + \frac{\partial \varphi}{\partial r_k} \frac{D-1}{r_k} \right) = \\ &= -\frac{\lambda_k r_{kj}}{D r_k^3} \left(r_k^2 \frac{\partial^3 \varphi}{\partial r_k^3} + r_k (D-1) \frac{\partial^2 \varphi}{\partial r_k^2} - (D-1) \frac{\partial \varphi}{\partial r_k} \right) \end{aligned} \quad (7.8)$$

Now, we need to compute the gradient of the cost function C , which is expressed as follows:

$$\nabla C_f = \nabla_{\xi} \left(\int_{\mathbb{R}^d} \|\kappa_{\mu_f} - \kappa_{\mu_{f_i}}\|_2^2 d\mathbf{x} \right) = 2 \int_{\mathbb{R}^d} (\kappa_{\mu_f} - \kappa_{\mu_{f_i}}) \nabla_{\xi} \kappa_{\mu_f} d\mathbf{x} \quad (7.9)$$

Given the definition of the partial derivatives of κ , the complete cost function gradient can be expressed as:

$$\nabla C_{f_{kj}} = 2 \int_{\mathbb{R}^d} (\kappa_{\mu_f} - \kappa_{\mu_{f_i}}) \left(-\frac{\lambda_k r_{kj}}{D r_k^3} \left(r_k^2 \frac{\partial^3 \varphi}{\partial r_k^3} + r_k (D-1) \frac{\partial^2 \varphi}{\partial r_k^2} - (D-1) \frac{\partial \varphi}{\partial r_k} \right) \right) d\mathbf{x} \quad (7.10)$$

If we, for example, consider gaussian RBF $\varphi(r) = e^{-ar^2}$ in threedimensional space $D = 3$, then the resulting gradient would take the form:

$$\begin{aligned} \frac{\partial \varphi}{\partial r} &= -2ar\varphi(r), \quad \frac{\partial^2 \varphi}{\partial r^2} = (4a^2r^2 - 2a)\varphi(r), \quad \frac{\partial^3 \varphi}{\partial r^3} = (12a^2r - 8a^3r^3)\varphi(r) \\ \nabla C_{f_{kj}} &= 2 \int_{\mathbb{R}^d} (\kappa_{\mu_f} - \kappa_{\mu_{f_i}}) \left(-\frac{\lambda_k r_{kj}}{D r_k^3} \left(r_k^2 \frac{\partial^3 \varphi}{\partial r_k^3} + r_k (D-1) \frac{\partial^2 \varphi}{\partial r_k^2} - (D-1) \frac{\partial \varphi}{\partial r_k} \right) \right) d\mathbf{x} = \\ &= -\frac{2}{3} \int_{\mathbb{R}^3} (\kappa_{\mu_f} - \kappa_{\mu_{f_i}}) e^{-ar^2} \frac{\lambda_k r_{kj}}{r^3} \left(r^2 (12a^2r - 8a^3r^3) + 2r (4a^2r^2 - 2a) + 4ar \right) = \\ &= -\frac{2}{3} \int_{\mathbb{R}^3} (\kappa_{\mu_f} - \kappa_{\mu_{f_i}}) e^{-ar^2} \lambda_k r_{kj} \left(12a^2 - 8a^3r^2 + 2 \left(4a^2 - \frac{2a}{r^2} \right) + \frac{4a}{r^2} \right) = \\ &= -\frac{2}{3} \int_{\mathbb{R}^3} (\kappa_{\mu_f} - \kappa_{\mu_{f_i}}) e^{-ar^2} \lambda_k r_{kj} \alpha^2 (20 - 8ar^2) = \\ &= \frac{8}{3} \int_{\mathbb{R}^3} (\kappa_{\mu_f} - \kappa_{\mu_{f_i}}) e^{-ar^2} \alpha^2 \lambda_k r_{kj} (2ar^2 - 5) \end{aligned}$$

The equation follows our latest paper, further described in Chapter 9.13. The gradient from the paper was described as follows:

$$\nabla C_{fkj} = \frac{8}{d} \int_{\mathbb{R}^d} (\kappa_{\mu_f} - \kappa_{\mu_i}) \alpha_k^2 g_k(\mathbf{x}) (x_j - \xi_{kj}) (2\alpha_k \|\mathbf{x} - \xi_{\mathbf{k}}\|_2^2 - 2 - d) d\mathbf{x} \quad (7.11)$$

In this section and the paper, slightly different notations and substitutions were made. However, both results correspond to each other. Both models were calculated by hand and then checked by symbolic mathematics using Python's SymPy library. The source code is shown in Listings 7.1. Both generic and Gaussian RBF are shown, uncommenting either row 14 or 15.

Source code 7.1: The SymPy code to verify the created mathematical model

```

1 from sympy import *
2 d = 3 # Number of dimensions
3 N = 10 # Number of RBF centres
4 # Vertices where the function would be evaluated
5 x = Matrix(d, 1, lambda j, _: symbols(f'x_{j+1}'))
6 # RBF centre positions
7 xi = Matrix(N, d, lambda i, j: symbols(f'xi_{i+1}_{j+1}'))
8 i = Idx('i', (1, N)) # Index for the number of RBF centres
9 j = Idx('j', (1, d)) # Index for the dimension number
10 lam = IndexedBase('lambda', i) # RBF weights
11 alpha = IndexedBase('alpha', i) # Shape parameter
12 # L2 norm of a difference of a position "x" and a RBF centre
13 norm_x_xi = sqrt(Sum((x[j-1,0] - xi[i-1, j-1])**2, (j, 1, d)))
14 phi = Function('phi')(norm_x_xi) # Generic function
15 #phi = exp(-alpha[i] * norm_x_xi**2) # Gaussian function
16 f = Sum(lam[i] * phi, (i, 1, N)).doit() # The RBF approximation
17 grad_f = Matrix(d, 1, lambda j, _: diff(f, x[j,0])).doit()
18 # Hessian matrix H
19 hessian_f = Matrix(d, d, lambda i, j: diff(grad_f[i], x[j,0])).\
    doit()
20 # Mean curvature estimation
21 mean_curv = (hessian_f.trace()/d).doit()
22 # Gradient of the curvature
23 grad_curv = Matrix(d, 1, lambda i, _: diff(mean_curv, xi[0,i])).\
    doit()
24 pprint(latex(grad_curv)) # Print the result

```

7.5 Regularisation

The RBF centre placement in the later stages creates numerous RBFs with very local impact. This technique does not cause any problems with static surface creation. However, it often creates a very cluttered and unstable curvature gradient field. The issue was first found in the paper described in Chapter 9.13 and addressed by forcing the shape parameters to be as large as possible. Also, the RBFs were forced to be inside (with positive weight) rather than outside (with negative weights). The penalisation would be in (7.12).

$$C_r = \gamma C_m + (1 - \gamma) \frac{d_{v_i}}{d_{\max}} \frac{\alpha_i}{\alpha_{\max}} \quad (7.12)$$

In the equation, C_r is the cost function, C_m is the original cost function, γ is the regularisation factor (between 0 and 1), d_{v_i} denotes the distance to the surface, which is normalised by the maximum possible value d_{\max} . Lastly, α_i is the shape parameter used, and it is also normalised by the maximum possible shape parameter α_{\max} . The complete in-depth description can be found in [95].

This chapter introduces an innovative approach to muscle modelling using Radial Basis Functions (RBF), building on the foundational studies and techniques detailed in earlier text sections. It extends and generalizes previously discussed methods for broader applicability in different modelling scenarios.

The integration of RBF in muscle modelling is explored with an emphasis on transforming volumetric or triangulated data into a dynamic shape-preserving model. This technique is crucial for maintaining the original curvature of the muscle, which is essential for realistic animations and simulations. Creating such a model is visualized through figures and pseudocode, outlining the steps in developing static RBF surfaces. Strategically placing sample points using the Halton sequence is included, as creating a Signed Distance Function (SDF) based on these samples and carefully adjusting these elements to model muscle structure accurately.

Further discussion in the chapter focuses on the dynamics of the model, particularly on how changes in muscle attachment and position can be simulated realistically. The dynamics are elaborated by adjusting the RBF surface in response to movement, ensuring that the muscle's original curvature and volume are preserved. These adjustments are vital for the model's fidelity in practical applications.

In terms of mathematical modelling, the derivation of the mathematical model is provided in extensive detail. The RBF function and its gradient are defined, followed by the computation of the Hessian matrix to estimate curvature. This section culminates in developing a cost function, which aids in minimizing the difference between the new and original curvatures, an essential aspect of realistic modelling.

The chapter also delves into effective centre placement and sample strategy, which are critical to minimizing computational load and improving model accuracy. The refined method for selecting strategic sample locations and centre placements enhances the overall efficiency and accuracy of the muscle modelling process.

The following chapter shows the potential of the proposed approach on a set of artificial scenarios and muscles.

Model verification

8

In the preceding chapter, we developed a mathematical model for static and dynamic muscle representations. In this chapter, we will thoroughly test this model.

The testing scenarios are divided into two main categories: artificial data and realistic data. Each category is further subdivided into specific tests. The detailed structure of this chapter is illustrated in Fig. 8.1.

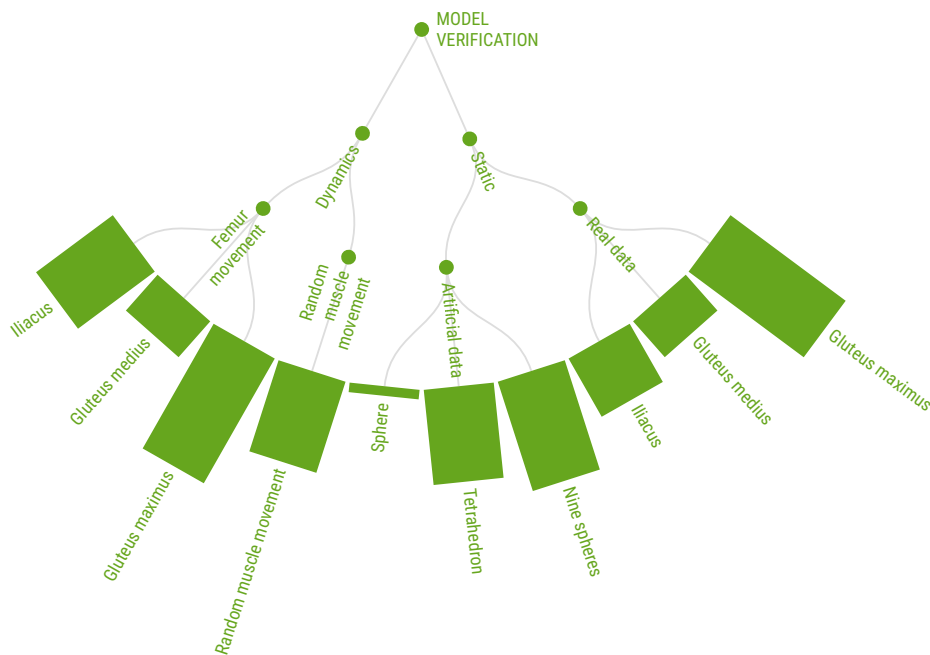


Figure 8.1: The graphical description of the content of this chapter and the depth of description on each topic. Created with the www.flourish.studio.

8.1 Datasets

The proposed approach will be tested on three muscles and three artificial datasets in Table 8.1. The three real datasets consist of triangular meshes of three hip muscles: gluteus maximus, medius, and iliacus.

Shape	Vertices	Volume [l]	Surface area [dm ²]	S/V ratio
Gluteus maximus	9878	0.75	6.36	8.52
Gluteus medius	5313	0.27	3.36	12.41
Iliacus	6931	0.1	2.45	23.91
Nine spheres	2338	5.34	16.73	3.13
Tetrahedron	4	0.51	4.62	9
Sphere	$+\infty$	4.19	12.57	3

Table 8.1: Numerical description of all datasets used in this section. The table is separated horizontally into real and artificial datasets. The last row shows an ideal example of a sphere for comparison.

Although the approach was tested on additional muscles, these three were selected to represent the primary categories based on their surface-to-volume ratios: high, medium, and low. These ratios are also detailed in Table 8.1.

The three artificial datasets also fall into three categories: one represents the optimal scenario for RBF approximation (a sphere), the second is suboptimal (nine spheres), and the third represents one of the most challenging shapes to approximate.

This chapter begins with generating static surfaces and then transitions into the dynamic movements of these shapes.

8.2 Static surface generation

Before testing the dynamic model, the static surface generation needs to be verified. The gluteus maximus will be the first muscle examined, with each aspect thoroughly tested. The results will be presented in the following sections. The gluteus maximus is the most optimal muscle for RBF approximation among the tested muscles, with subsequent tests demonstrating less optimal cases.

8.2.1 Gluteus maximus

The first test will focus on the gluteus maximus muscle, whose shape is well-suited for using the RBF implicit function to describe its form. The accuracy of static surface generation primarily depends on the number of RBFs used for the approximation and the selection of shape parameters. We will first examine the impact of the number of RBFs.

8.2.1.1 Number of RBFs

The number of RBFs is directly correlated with the resulting approximation precision¹. Generally, a higher number of RBFs leads to greater precision because it allows for the coverage of more intricate shapes. In this experiment, we use the gluteus maximus due to its favourable properties: it lacks sharp corners and has a shape and surface/volume ratio that resembles a sphere.

The first experiment, using insufficient RBFs, is shown in Fig. 8.2. This figure demonstrates that while the general outline of the muscle is approximated well, the intricate parts of the muscle are not preserved. An insufficient number of RBFs is particularly noticeable at both ends of the muscle, where it typically attaches to adjacent bones.

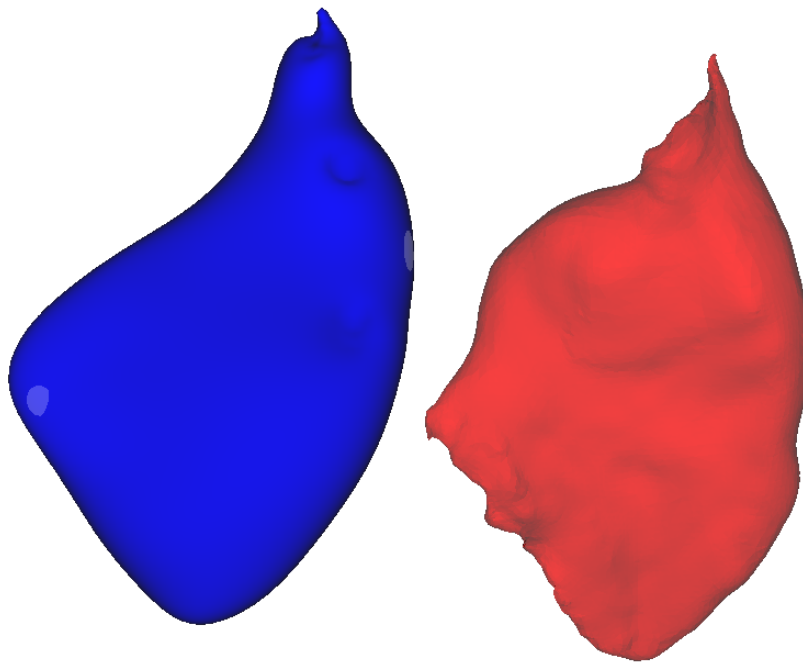


Figure 8.2: The low number of 50 RBFs to approximate a gluteus maximus muscle. Only rough shape is preserved, but there are no details.

In the second experiment, the number of RBFs is significantly increased (see Fig. 8.3). The experiment results in more apparent muscle details; however, some artefacts and outliers are still present, along with some inaccuracies, particularly in the same areas. These inaccuracies occur because these parts are the sharpest on the entire surface.

¹Excluding certain edge cases such as the approximation of a sphere, etc.

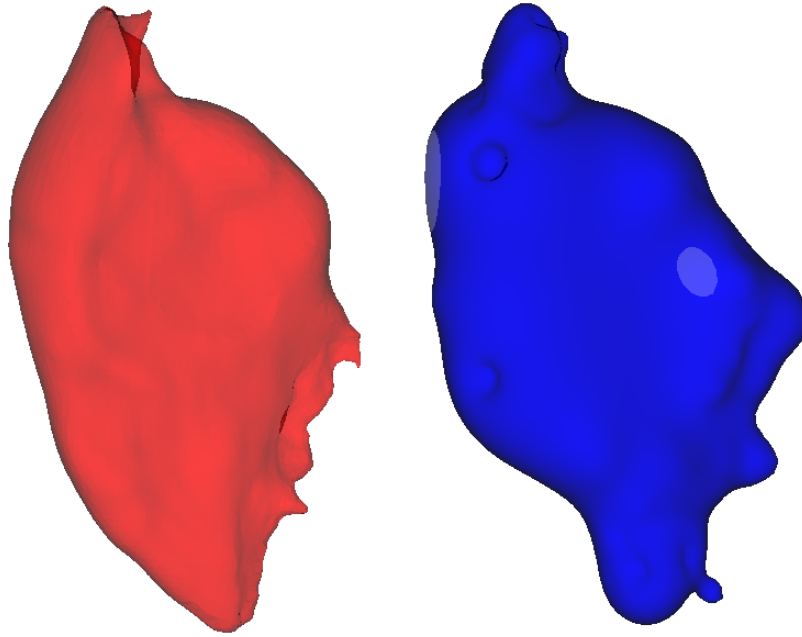


Figure 8.3: The 1,000 RBFs approximation of a gluteus maximus muscle. The details are visible, but some artefacts are present.

In the third experiment, the number of RBFs is high enough to capture even the fine details (as shown in Fig. 8.4), demonstrating the significance of the number of RBFs and confirming that a higher number generally leads to a better approximation.

8.2.1.2 Global or local shape parameters

The selection of the shape parameter is another crucial aspect to verify. The proposed method of choosing the best shape parameter from a predefined set will be compared to using a single global shape parameter. While the single-parameter approach offers less flexibility, it only requires storing a single value. To ensure a fair comparison, increasing the number of RBFs is performed in the case of the single global shape parameter so the amount of data stored remains the same. The comparison will be made against the results shown in Fig. 8.4.

During testing, the best global parameter for the gluteus maximus muscle was $\alpha = 0.35263[\text{mm}^{-2}]$, estimated using a simple binary search algorithm to five decimal places. Despite using one-third more RBFs, the approximation is unsatisfactory, as shown in Fig. 8.5. This parameter preserved only 78.6% of the volume, and adjusting the parameter either way by more than 10^{-5} further decreased this percentage.

The shape parameter significantly impacts the approximation for Gaussian RBFs ($e^{-\alpha r^2}$). A higher shape parameter results in a narrower RBF, focusing more on local details. Conversely, choosing a lower shape parameter would result in losing finer

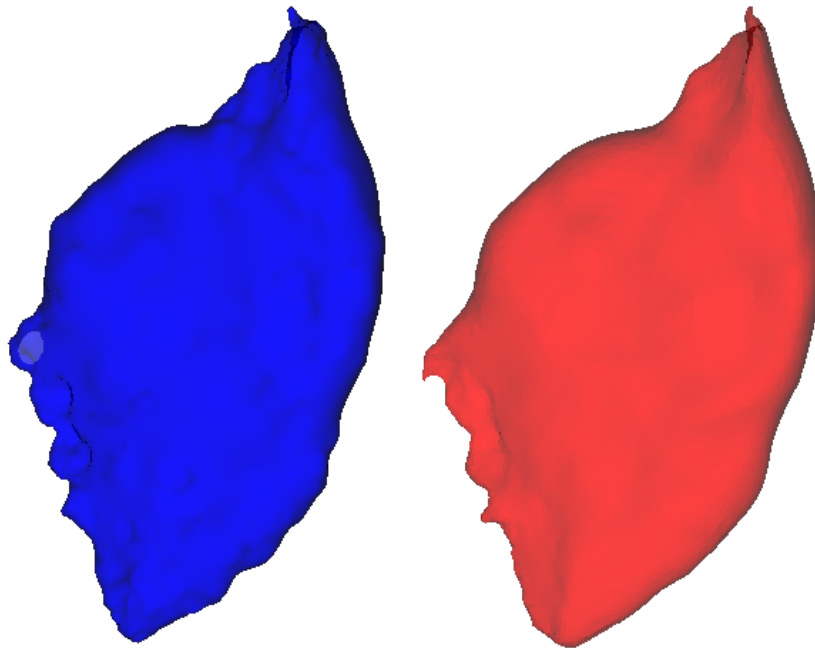


Figure 8.4: The 10,000 RBFs approximation of a gluteus maximus muscle. The details are visible, and only small imperfections are present.

details. If the global shape parameter is set too high, the surface would appear as a collection of discrete spheres rather than a continuous surface.

8.2.2 **Gluteus medius**

In the case of gluteus medius, the 1000 RBF surface approximation would take the form as in Fig. 8.6.

The surface-to-volume ratio is about 50% higher than in the gluteus maximus case, meaning that this muscle is not as suitable for the approximation as the gluteus maximus. However, the approximation is still satisfactory, including only minor inaccuracies.

8.2.3 **Iliacus**

The following static RBF surface creation test focuses on the iliacus muscle. This muscle is characterized by its narrow shape, resembling a thin rod rather than a sphere. Approximating the shape using 6000 RBFs, the resulting surface is successfully generated by the approach, as shown in Fig. 8.7.

The thin surface of the iliacus muscle requires more RBFs with smaller radii, which can result in a rougher surface. Although increasing the number of RBFs can improve the approximation, doing so would be counterproductive as the number

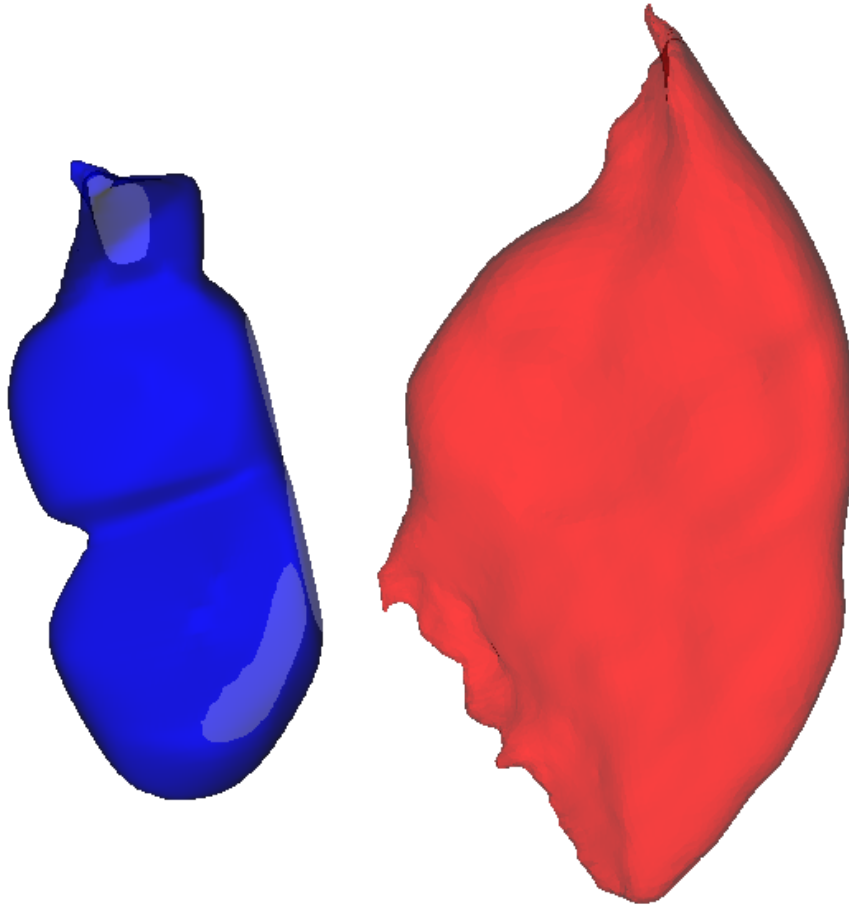


Figure 8.5: The single global shape parameter of $\alpha = 0.35263 [mm^{-2}]$. Even though the parameter was chosen carefully, the approximation is far from satisfactory.

of parameters would exceed those of the triangular mesh, negating the primary advantage of using RBF approximation.

8.2.4 Optimal and sub-optimal shapes

As previously mentioned, the RBF implicit function is well-suited for approximating sphere-like shapes, which typically have a high internal volume-to-surface area ratio. A sphere is so simple that a single RBF can be used for its approximation, so this result will not be presented. Instead, we will present a more complex shape consisting of a union of nine spheres, which remains highly suitable for the RBF approach. The original shape is shown in red in Fig. 8.8, with the approximation in blue. As you can see, the approximation is nearly perfect. However, the outer spheres are slightly dislocated, as visible the best on the bottom left sphere.

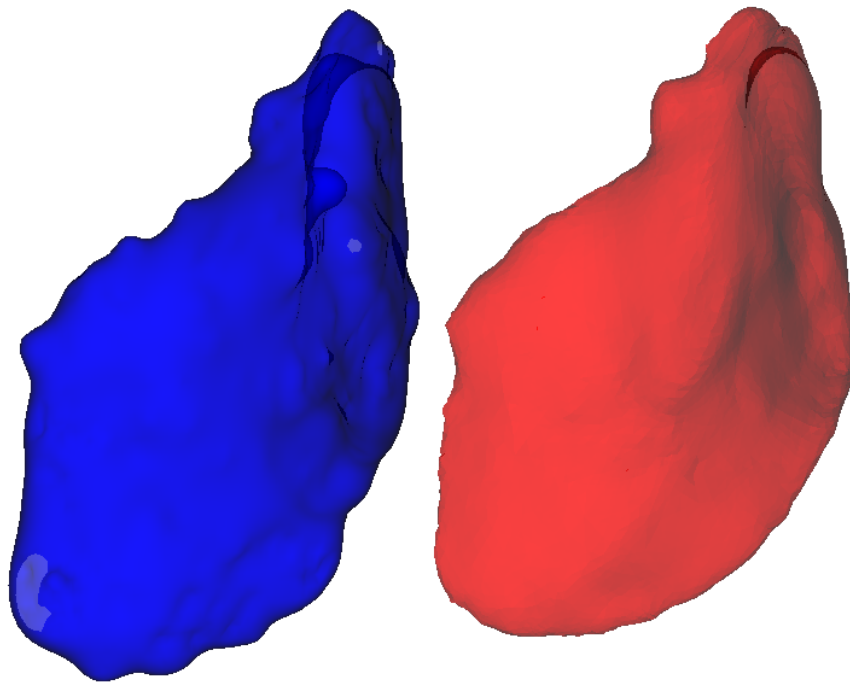


Figure 8.6: The gluteus medius muscle. The original shape is red, and the RBF implicit surface form is blue.

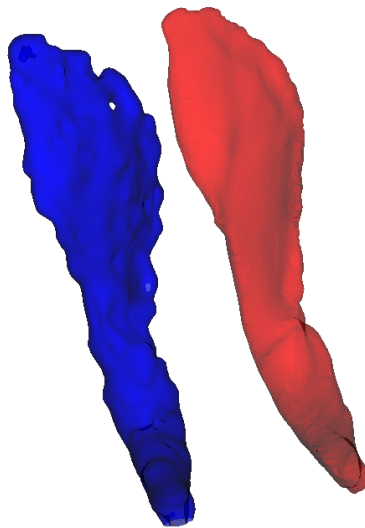


Figure 8.7: The iliacus muscle. The original shape is red, and the RBF implicit surface form is blue. The hole in the top part confirms that this is one of the problematic muscles to approximate.

8.2.5 Limitations

A well-known issue with the RBF approximation is its unsuitability for approximating sharp edges. This limitation is less significant in muscle modelling, as most

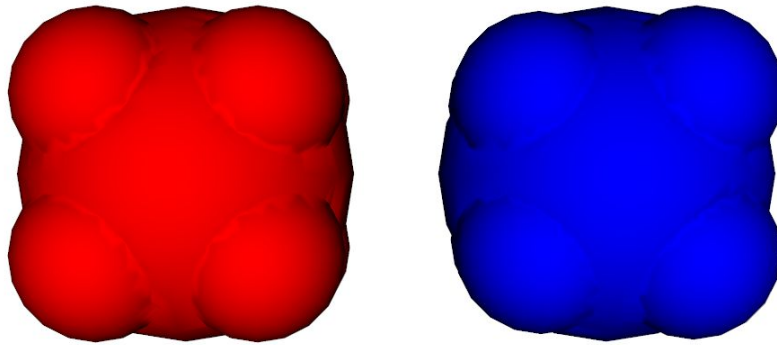


Figure 8.8: A surface of a union of nine spheres, with a high volume-to-surface ratio. The original shape is red, and the RBF implicit surface form is blue.

soft tissues do not have sharp edges and are relatively smooth, often resembling spherical shapes. However, we can illustrate this issue with a simple 3D volumetric shape like a tetrahedron. A tetrahedron has six sharp edges (with a sharp dihedral angle of more than 70°) and four corners, which pose significant challenges for RBF approximation.

The root of the problem lies in RBFs, where a single RBF forms a sphere as an isosurface at a given value. Creating a sharp corner is theoretically possible only with infinite infinitesimal RBFs. An attempt at RBF static surface creation for a tetrahedron can be seen in Fig. 8.9.

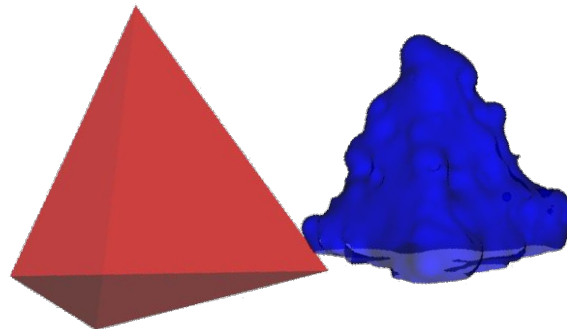


Figure 8.9: An attempt to create an RBF surface of a tetrahedron. RBF is not suitable for approximating sharp edges, such as a tetrahedron.

8.3 Dynamics

The explored static surface creation and its positives and weak points allow for dynamic model exploration. At first, random movements will be introduced, and

then, more realistic movements of the hip muscles will be discussed, producing a femur bone movement.

8.3.1 Random surface movement

The first test is created to test the dynamic model's ability to restore its shape in the case of a random movement. At first, the static model is created by the surface (e.g. gluteus maximus), after which all vertices are moved in random directions. The movement magnitude in the test was set to 5% of the length of the shape AABB diagonal (which is used for normalisation purposes). The test is discussed thoroughly in the publication [95] introduced in Section 7.4. That is why the images presented here will be smaller, and the details will be visible in that paper. The initial stage, the stage during gradient descent and the final stage of the muscle are shown in Fig. 8.10. In the paper [95], a regularisation schema was introduced to enhance the results, so results with regularisation are presented in Fig. 8.11.

As shown from Fig. 8.10 and 8.11, the regularisation is crucial to obtain a satisfactory deformation, where only minimal inaccuracies occur. During this test, ten gradient descent deformation iterations were performed, after which the result would not change only slightly; the average centre point movement is about 0.01% of the AABB diagonal length, which is practically not visible.

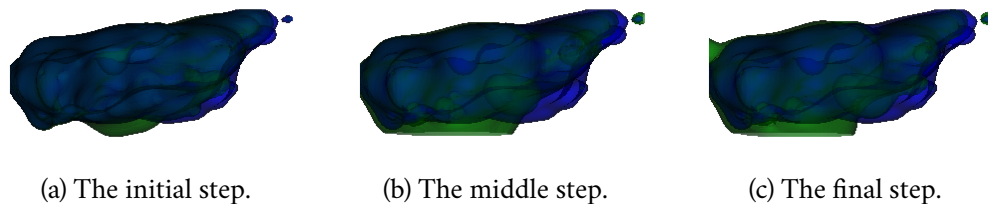


Figure 8.10: The gluteus muscle shape restoration after a random vertex movement. The blue is the original shape, and the green is the approaching one.

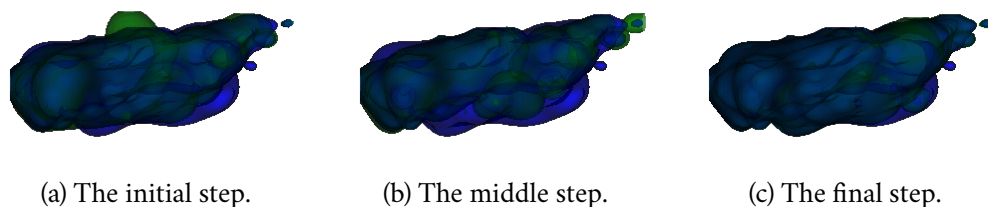


Figure 8.11: The gluteus muscle shape restoration after a random vertex movement and incorporating the regularisation schema. The blue is the original shape, and the green is the approaching one.

8.3.2 Femur movement

The testing on actual measured data was performed to make the test more accurate and realistic. These include the tests on hip muscles, namely gluteus maximus, medius and iliacus, to be consistent with the previous chapter about static surface generation.

The actual movement of the femur bone is the flexion of the bone from the zero-angle (in this position is the femur when the body is lying) to the eighty-degree angle (representing sitting) [96]. The actual bones are not visualised for the test to show the see muscle volume during the deformation. The first muscle to test is the gluteus maximus.

8.3.2.1 Gluteus maximus muscle

The deformed gluteus maximus muscle is visualised in Fig. 8.12. Although the muscle needs to stay as a topological sphere, its surface-to-volume ratio increases to 9.36 dm^{-1} due to the bend. Luckily, this fact does not affect the result much; only the downside is that the surface is rougher than before.

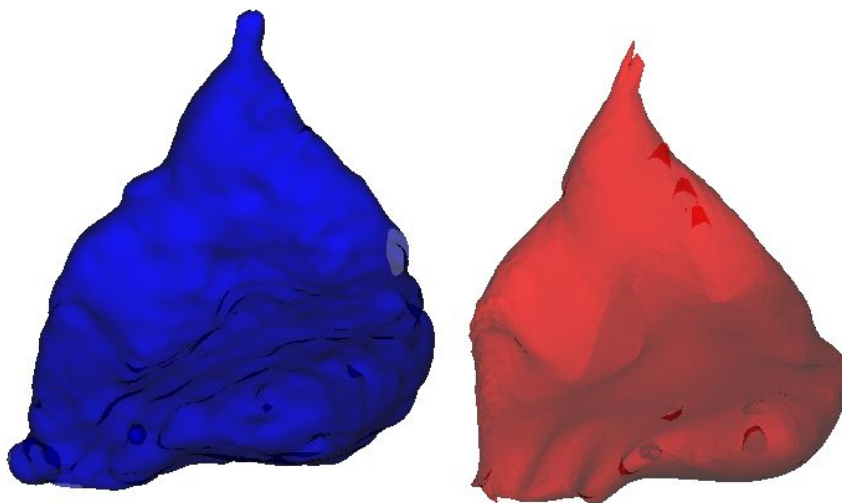


Figure 8.12: RBF reconstruction of the gluteus maximus muscle after the 80° flexion.

8.3.2.2 Gluteus medius muscle

The deformation of the gluteus medius muscle is less apparent than the gluteus maximus muscle because its centre is close to the centre of rotation. After the flexion, the surface-to-volume ratio increases slightly, from 12.41 to 12.90. The result can

be seen in Fig. 8.13. Because the deformation is less apparent, the deformation of the red muscle was adjusted to be faster and less precise to produce some spikes at the end (not to let the simulation have time to reconstruct fully, but only partially). As we can see, the RBF can smooth those spikes well.

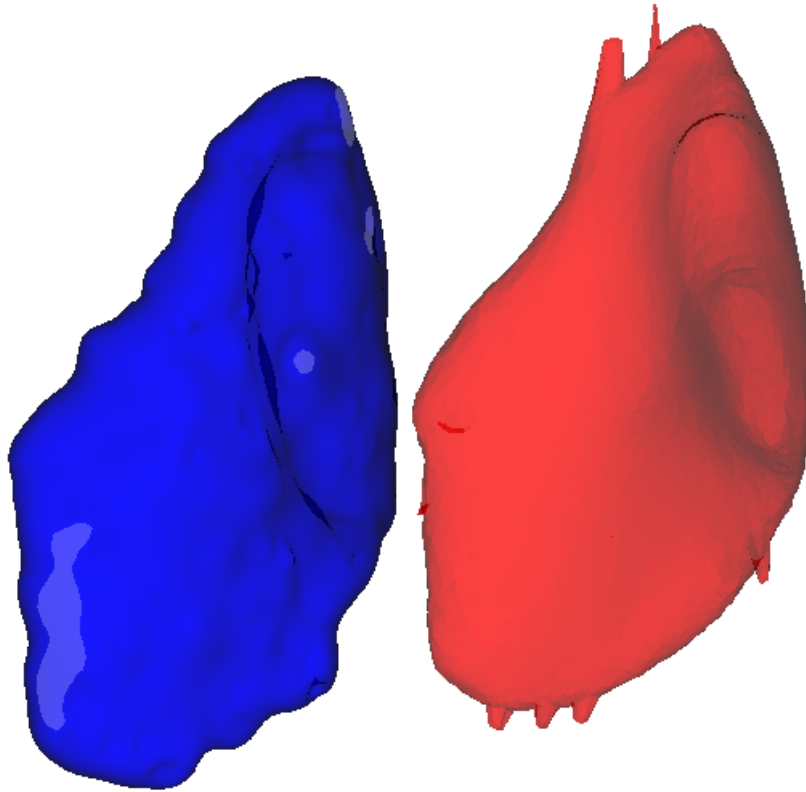


Figure 8.13: RBF reconstruction of the gluteus medius muscle after the 80° flexion.

8.3.2.3 **Iliacus muscle**

The last muscle to test is the problematic iliacus due to its flat superior part. As shown in Fig. 8.14, the deformed muscle is approximated well considering the circumstances, containing no holes and approximating the overall shape nicely. The roughness of the flat part is apparent, but this is expected because the algorithm is trying to "fill a narrow gap with spheres". The approximation seems even better than in the resting pose, confirmed by a slightly reduced surface-to-volume ratio of 23.61 dm^{-1} . The ratio increased because the inferior part contracted.

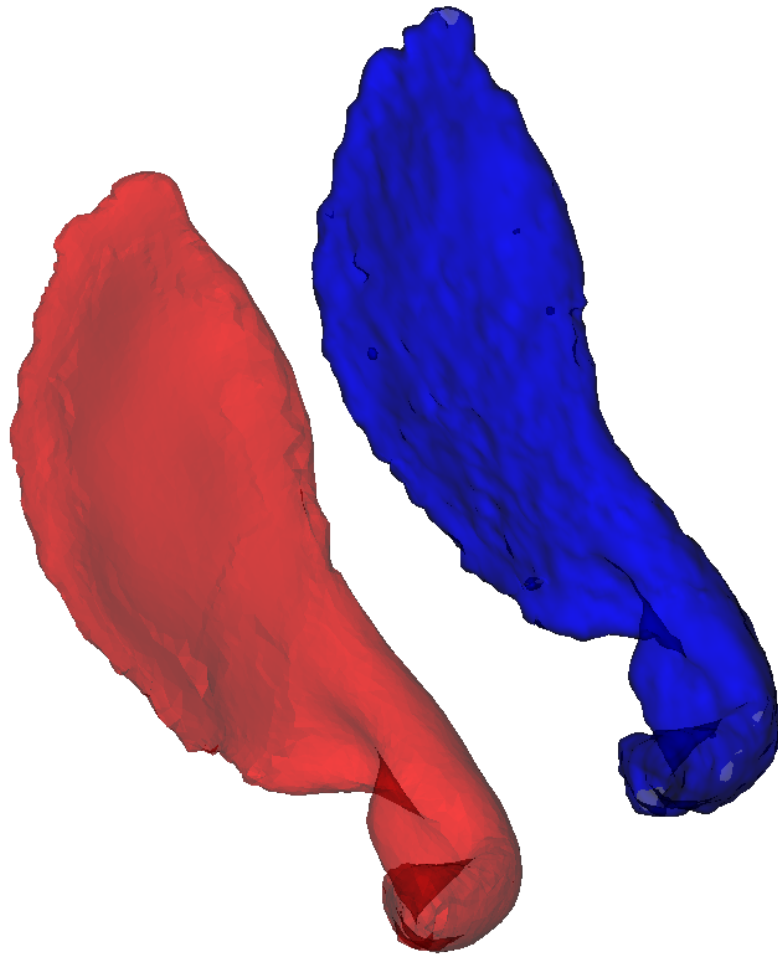


Figure 8.14: RBF reconstruction of the iliacus muscle after the 80° flexion.

All of these experiments prove that there is a space for RBF to be used in muscle modelling. There is plenty of space to improve in the future, mainly to reduce the roughness of the surface and maintain the number of RBFs used, but the idea and the math are correct, and the basic algorithm also works by the outputs provided.

The previous text summarises my research during the years of study; the next chapter of this dissertation focuses on our research papers during that time, explaining the evolution of the research.

Author's contribution

9

The main overview of this chapter is shown in Fig. 9.1, which focuses mainly on paper topic separation and also in Fig. 9.2, focusing on the timeline of the contributions.

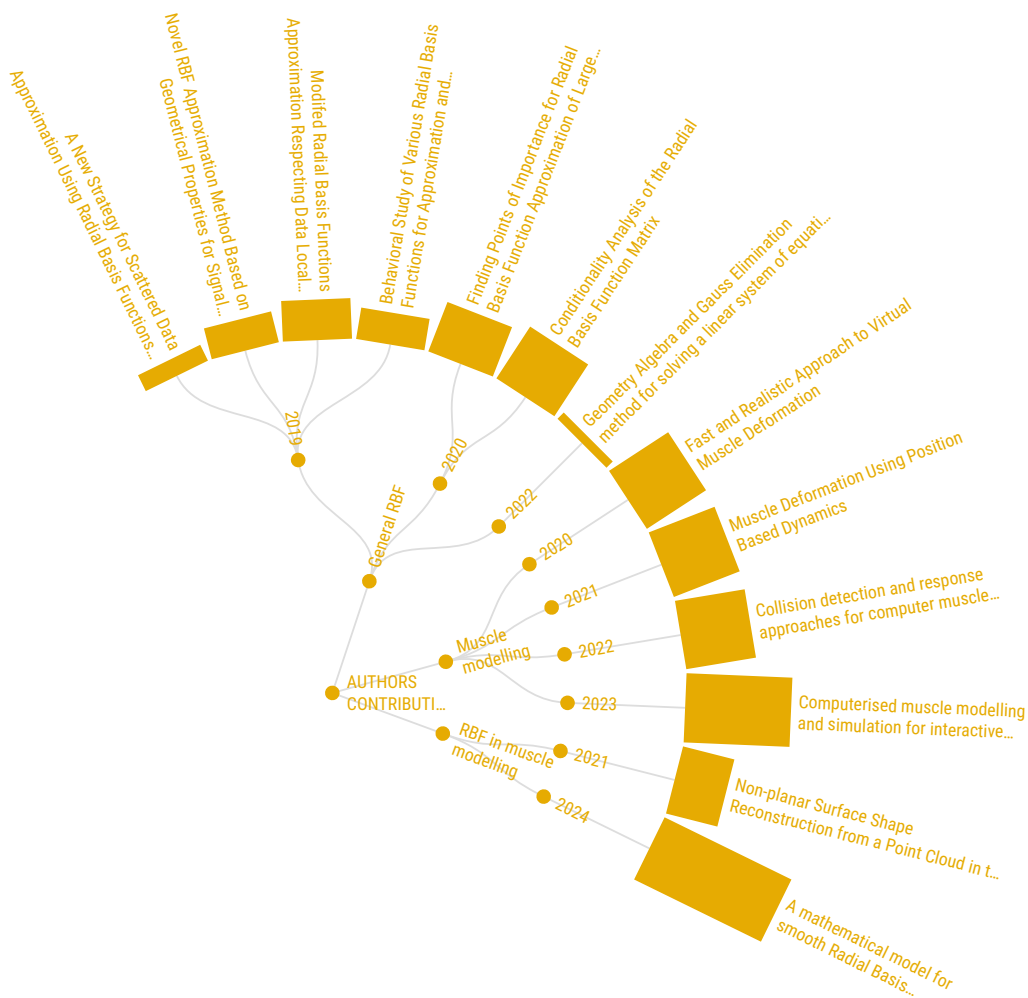


Figure 9.1: The graphical description of the content of this chapter and the depth of description on each topic. Created with the www.flourish.studio.

My contribution counts up to thirteen papers throughout my doctoral research, culminating in a pivotal publication central to this dissertation. A detailed timeline of this contribution is illustrated in Fig. 9.2, displayed on a dedicated page for a full-detail view. The first page of each section provides a concise summary of these papers. Additionally, a radar chart at the beginning of the document visually represents my contribution to each paper in percentages.

Since I can't entirely agree with using only a single percentage of work on a publication, a nuanced view of my involvement is provided, and the work is categorized into five distinct areas:

- Experimenting - Conducting exhaustive experiments with the established approach, choosing appropriate parameters, and generating results (images, concrete values) for the papers.
- Novelty - Developing new ideas based on existing methods, requiring refinement and precise articulation.
- Implementation - Writing the source code, which is then evaluated in the experimenting phase.
- Review - Thoroughly revising the paper, ensuring clarity, filling informational gaps, correcting spelling errors, and improving the English quality.
- State-of-the-art - Undertaking comprehensive literature reviews and integrating these findings effectively into the papers.

An additional category, "overall", averages these five aspects to present an overall percentage of my contribution. These charts are my honest estimations of how much I contributed in each category. Furthermore, each summary includes the complete citation of the paper, allowing readers to access further details about the work quickly. An understandable phenomenon is visible following all of the publications. The later the publication was produced, the more work shifted from the implementation and experimenting section into the novelty, state-of-the-art and review sections, which is reasonable due to the absorbance of new knowledge during the study.

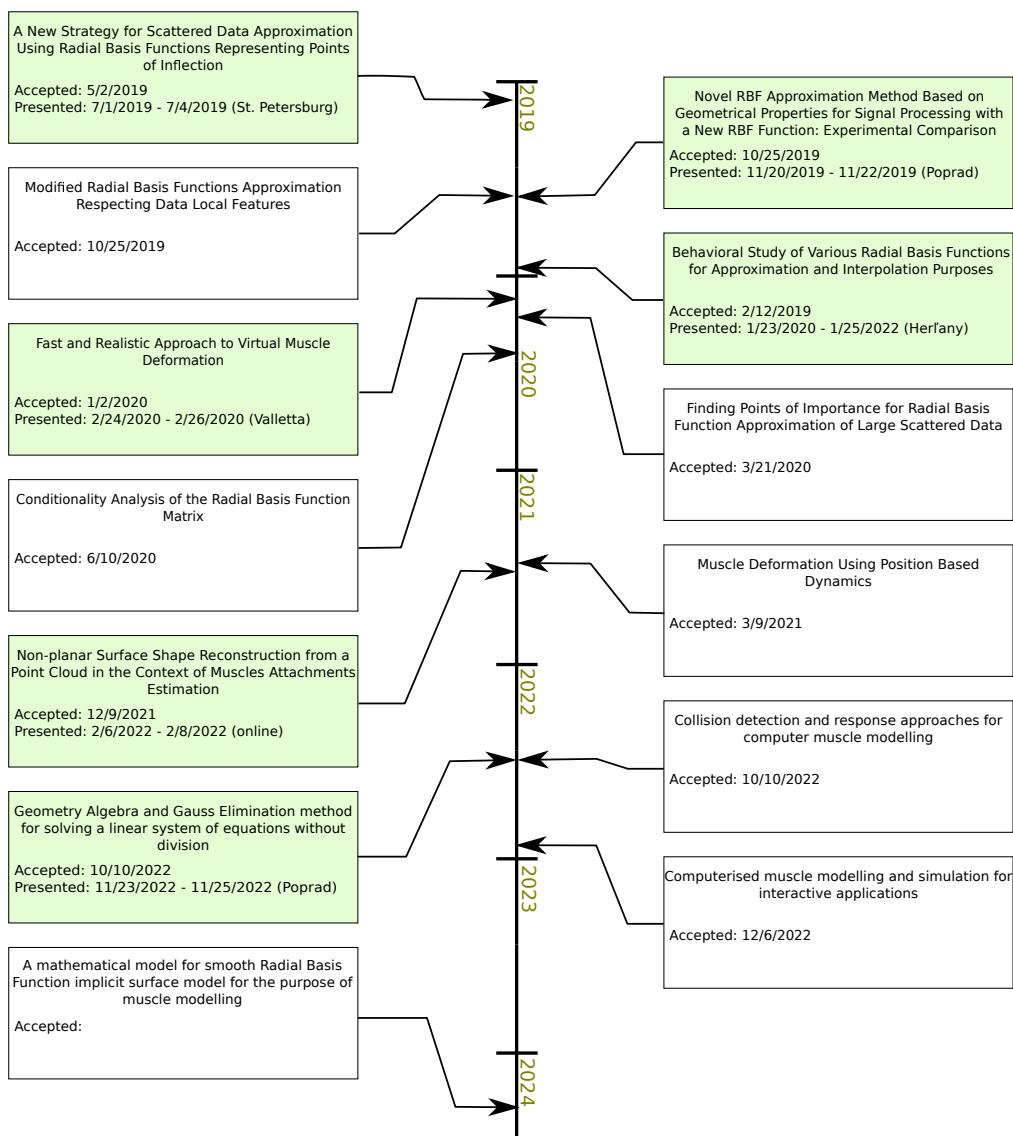


Figure 9.2: Timeline of the author's contribution. The author of the dissertation has already presented the green-marked publications abroad; the others were either presented by a colleague or online due to epidemiologic restrictions.

9.1 A New Strategy for Scattered Data Approximation Using Radial Basis Functions Respecting Points of Inflection

The author's initial contribution, documented in [3], was undertaken during their master's studies, with subsequent utilization of its results. The paper used radial basis functions (RBFs) to approximate 2D functions, highlighting identifying features in the original function.

The paper focused on finding an optimal centre point using four groups: vertices at the borders, local extrema, points of inflexion, and pseudo-random positions. Borders prevented extrapolation, and RBFs at the extrema and inflexion points captured the function behaviour, minimising errors. The pseudo-random placement, using the Halton distribution, filled potential gaps. Also, some challenges were noted, especially with sharp edges, where the RBF technique was unsuitable.

The publication is a solid foundation for the RBF approximation of general functions. However, there is a lack of discussion on simplifications specific to the nature of the data. Subsequent papers by the authors modified the RBF placement a bit. It also becomes clear that the musculoskeletal data differ from a general 2D function. Due to this fact, inflexion points were replaced with locations exhibiting the highest Mean Squared Error (MSE), driven mainly by higher data dimensionality and the different nature of the musculoskeletal geometry. This adjustment was applied to approximate musculoskeletal models. Maintaining RBFs at the function boundaries proved crucial for a successful final approximation.



Publication [3]:

CERVENKA, M.; SMOLIK, M.; SKALA, V. A New Strategy for Scattered Data Approximation Using Radial Basis Functions Representing Points of Inflection. *Computational Science and Its Application, ICSSA 2019 proceedings, Part I, LNCS 11619*. 2019, pp. 322–226. ISBN 978-3-030-24288-6. ISSN 0302-9743. Available from DOI: https://doi.org/10.1007/978-3-030-24289-3_24. UT WoS: 000661318700024, EID: 2-s2.0-85069157052, OBD: 43926678



A New Strategy for Scattered Data Approximation Using Radial Basis Functions Respecting Points of Inflection

Martin Cervenka, Michal Smolik^(*), and Vaclav Skala

Faculty of Applied Sciences, University of West Bohemia, Plzen, Czech Republic
{cervemar,smolik,skala}@kiv.zcu.cz

Abstract. The approximation of scattered data is known technique in computer science. We propose a new strategy for the placement of radial basis functions respecting points of inflection. The placement of radial basis functions has a great impact on the approximation quality. Due to this fact we propose a new strategy for the placement of radial basis functions with respect to the properties of approximated function, including the extreme and the inflection points. Our experimental results proved high quality of the proposed approach and high quality of the final approximation.

Keywords: Radial basis functions · Approximation · Stationary points

1 Introduction

The Radial basis functions (RBF) are well known technique for scattered data approximation in d -dimensional space in general. A significant advantage of the RBF application is its computational complexity, which is nearly independent of the problem dimensionality. The formulation is leading to a solution of a linear system of equations $\mathbf{Ax} = \mathbf{b}$. There exists several modifications and specifications of the RBF use for approximation. The method of RBF was originally introduced by Hardy in a highly influential paper in 1971 [8,9]. The paper [8] presented an analytical method for representation of scattered data surfaces. The method computes the sum of quadric surfaces. The paper also stated the importance of the location of radial basis functions. This issue is solved by several papers. Some solutions are proposed by the papers [3,15,16], which use the regularization in the forward selection of radial basis function centers. The paper [31] presents an improvement for the problem with the behavior of RBF interpolants near boundaries. The paper [13] compares RBF approximations with different radial basis functions and different placement of those radial basis functions. However, all the radial basis functions are placed with some random or uniform distribution. A bit more sophisticated placement is presented in [14].

The research was supported by projects Czech Science Foundation (GACR) No. 17-05534S and partially by SGS 2019-016.

© Springer Nature Switzerland AG 2019
S. Misra et al. (Eds.): ICCSA 2019, LNCS 11619, pp. 322–336, 2019.
https://doi.org/10.1007/978-3-030-24289-3_24

The selection of a shape parameter is another problem. Wrong selection of this parameter can lead to an ill-conditioned problem or to an inaccurate approximation. The selection of the best shape parameter is thus very important. Fornberg and Wright [5] presents an algorithm which avoids this difficulty, and which allows numerically stable computations of Multi-Quadric RBF interpolants for all shape parameter values. The paper [29] derives a range of suitable shape parameters using the analysis of the condition number of the system matrix, error of energy and irregularity of node distribution. A lot of approaches for selection of a good value of the shape parameter use some kind of random generator. Examples of this approaches are [2, 20]. The paper [1] proposes a genetic algorithm to determine a good variable shape parameter, however the algorithm is very slow.

2 Radial Basis Functions

The Radial basis function (RBF) is a technique for scattered data interpolation [17] and approximation [4, 27]. The RBF interpolation and approximation is computationally more expensive compared to interpolation and approximation methods that use an information about mesh connectivity, because input data are not ordered and there is no known relation between them, i.e. tessellation is not made. Although RBF has a higher computational cost, it can be used for d -dimensional problem solution in many applications, e.g. solution of partial differential equations [11, 33], image reconstruction [28], neural networks [7, 10, 32], vector fields [24, 26], GIS systems [12, 18], optics [19] etc. It should be noted that it does not require any triangulation or tessellation mesh in general. There is no need to know any connectivity of interpolation points, all points are tied up only with distances of each other. Using all these distances we can form the interpolation or approximation matrix, which will be shown later.

The RBF is a function whose value depends only on the distance from its center point. Due to the use of distance functions, the RBFs can be easily implemented to reconstruct the surface using scattered data in 2D, 3D or higher dimensional spaces. It should be noted that the RBF interpolation and approximation is not separable by dimension. For the readers reference a compressed description of the RBF is given in the following paragraphs, for details consider [25, 26].

Radial function interpolants have a helpful property of being invariant under all Euclidean transformations, i.e. translations, rotations and reflections. It does not matter whether we first compute the RBF interpolation function and then apply a Euclidean transformation, or if we first transform all the data and then compute the radial function interpolants. This is a result of the fact that Euclidean transformations are characterized by orthonormal transformation matrices and are therefore two-norm invariant. Radial basis functions can be divided into two groups according to their influence. The first group are “global” RBFs [21]. Application of global RBFs usually leads to ill-conditioned system, especially in the case of large data sets with a large span [13, 23]. An example of “global” RBF is the Gauss radial basis function.

324 M. Cervenka et al.

$$\varphi(r) = e^{-\epsilon r^2}, \quad (1)$$

where r is the distance of two points and ϵ is a shape parameter.

The “local” RBFs were introduced in [30] as compactly supported RBF (CSRBF) and satisfy the following condition:

$$\begin{aligned} \varphi(r) &= (1-r)_+^q P(r) \\ &= \begin{cases} (1-r)^q P(r) & 0 \leq r \leq 1 \\ 0 & r > 1 \end{cases} \end{aligned} \quad (2)$$

where $P(r)$ is a polynomial function, r is the distance of two points and q is a parameter.

2.1 Radial Basis Function Approximation

RBF interpolation was originally introduced by [8] and is based on computing the distance of two points in any k -dimensional space. The interpolated value, and approximated value as well, is determined as (see [22]):

$$h(\mathbf{x}) = \sum_{j=1}^M \lambda_j \varphi(\|\mathbf{x} - \boldsymbol{\xi}_j\|) \quad (3)$$

where λ_j are weights of the RBFs, M is the number of the radial basis functions, φ is the radial basis function and $\boldsymbol{\xi}_j$ are centers of radial basis functions. For a given dataset of points with associated values, i.e. in the case of scalar values $\{\mathbf{x}_i, h_i\}_1^N$, where $N \gg M$, the following overdetermined linear system of equations is obtained:

$$\begin{aligned} h_i = h(\mathbf{x}_i) &= \sum_{j=1}^M \lambda_j \varphi(\|\mathbf{x}_i - \boldsymbol{\xi}_j\|) \\ &\text{for } \forall i \in \{1, \dots, N\} \end{aligned} \quad (4)$$

where λ_j are weights to be computed; see Fig. 1 for a visual interpretation of (3) or (4) for a $2\frac{1}{2}D$ function. Point in $2\frac{1}{2}D$ is a $2D$ point associated with a scalar value.

Equation (4) can be rewritten in a matrix form as

$$\mathbf{A}\boldsymbol{\lambda} = \mathbf{h}, \quad (5)$$

where $A_{ij} = \varphi(\|\mathbf{x}_i - \boldsymbol{\xi}_j\|)$ is the entry of the matrix in the i -th row and j -th column, the number of rows $N \gg M$, M is the number of unknown weights $\boldsymbol{\lambda} = [\lambda_1, \dots, \lambda_M]^T$, i.e. a number of reference points, and $\mathbf{h} = [h_1, \dots, h_N]^T$ is a vector of values in the given points. The presented system is overdetermined, i.e. the number of equations N is higher than the number of variables M . This linear system of equations can be solved by the least squares method (LSE) as

$$\mathbf{A}^T \mathbf{A} \boldsymbol{\lambda} = \mathbf{A}^T \mathbf{h}, \quad (6)$$

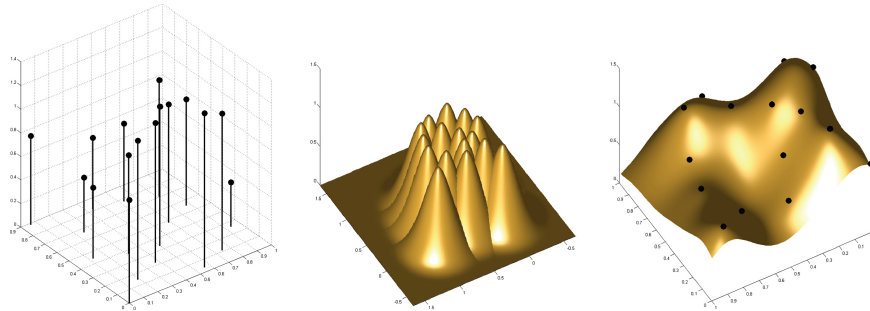


Fig. 1. Data values, the RBF collocation functions, the resulting interpolant (from [26]).

where the matrix $\mathbf{A}^T \mathbf{A}$ is symmetrical. Another possibility to solve the overdetermined system of linear equations $\mathbf{A}\boldsymbol{\lambda} = \mathbf{h}$ is using the QR decomposition.

The RBF approximation can be done using “global” or “local” functions. When using “global” radial basis functions, the matrix \mathbf{A} will be full and ill conditioned in general. When using “local” radial basis functions, the matrix \mathbf{A} might be sparse, which can be beneficial when solving the overdetermined system of linear equations $\mathbf{A}\boldsymbol{\lambda} = \mathbf{h}$.

3 Proposed Approach

We propose a new approach for scattered data approximation of $2\frac{1}{2}D$ functions using Radial basis functions with respecting inflection points of the function. Inflection points are computed from the discrete mesh and from curves given by implicit points. For a simplicity, the proposed approach is demonstrated on sampled regular grid.

The input $2\frac{1}{2}D$ function $f(x, y)$ is for the sake of simplicity of evaluation sampled on a regular grid. For a general case neighbours have to be determined, e.g. by using a kd-tree. One important feature when computing the RBF approximation is the location of radial basis functions. We will show two main groups of locations, where the radial basis functions should be placed.

The first group are extreme points of the input data set. Most of the radial basis functions have the property of having its maximum at its center (we will use only those) and thus it is very suitable to place the radial basis functions at the locations of extreme points.

The second group are inflection points of the input data set. The inflection points are important as the surface crosses its tangent plane, i.e. the surface changes from being concave to being convex, or vice versa. The surface at those locations should be approximated as accurately as possible in order to maintain the main features of the surface.

326 M. Cervenka et al.

3.1 Determination of Extreme Points

The local extreme points of the function $f(x, y)$ can be either minimum or maximum, i.e.

$$\frac{\partial f}{\partial x} = 0 \quad \& \quad \frac{\partial f}{\partial y} = 0. \quad (7)$$

The decision if a point is a local extreme point can be done using only surrounding points. In our case, i.e. regular grid, we use four surrounding points, i.e. point on the right, left, up and down. In general case, neighbor points need to be determined, e.g. using a kd-tree. If a point is a local maximum, then all four surrounding points must be lower. The same also applies to a local minimum, i.e all four surrounding points must be higher (Fig. 2).

4	6	8	6	5	4	3	4	5
5	7	6	5	5	2	1	3	5
6	8	7	7	4	3	2	2	4
9	4	4	5	5	4	4	3	5

Fig. 2. Location of local extreme points. The values 1 and 8 (green) are local extremes, i.e. local minimum and local maximum. The value 4 (red) is not a local extreme as the four surrounding values are higher and smaller as well. (Color figure online)

The situation on the border of the input data set is a little bit different as we cannot use all four surrounding points, there will always be at least one missing. One solution is to skip the border of the input data set, however in this way we could omit some important extremes. Therefore, we determine the extremes from only three or two surrounding points.

3.2 Determination of Inflection Points

The second group of important locations for radial basis functions placement are inflection points, which forms actually curves of implicit points. The inflection points are located where the Gaussian curvature is equal zero. The Gaussian curvature for $2\frac{1}{2}D$ function $f(x, y)$ is computed as

$$k_{gauss} = \frac{\frac{\partial^2 f}{\partial x^2} \frac{\partial^2 f}{\partial y^2} - \left(\frac{\partial^2 f}{\partial x \partial y}\right)^2}{\left(\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2 + 1\right)^2}. \quad (8)$$

The Gaussian curvature is equal zero when

$$\frac{\partial^2 f}{\partial x^2} \frac{\partial^2 f}{\partial y^2} - \left(\frac{\partial^2 f}{\partial x \partial y}\right)^2 = 0. \quad (9)$$

This formula is equivalent to the calculation using the Hessian matrix, i.e.

$$\begin{vmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial y \partial x} & \frac{\partial^2 f}{\partial y^2} \end{vmatrix} = 0. \quad (10)$$

To find out the locations, where the Gaussian curvature, i.e. the determinant of Hessian matrix, is equal zero, we can sample the following function from the input discrete data set.

$$I(x, y) = \frac{\partial^2 f}{\partial x^2} \frac{\partial^2 f}{\partial y^2} - \left(\frac{\partial^2 f}{\partial x \partial y} \right)^2. \quad (11)$$

An application example of (11) can be seen in Fig. 3.

Now, we need to find out the locations, where this sampled function is equal zero. We again use the four surrounding points. If at least one is positive and at least one is negative, then there must be a zero value in between them. For the simplicity and to speed-up the calculation we consider the center point as the inflection point (see Figs. 3 and 4 for illustration).

4	6	8	6	5	4	-2	-3	-4
5	2	6	5	5	2	1	-1	-5
2	1	3	7	4	3	2	2	4
-2	-1	0	5	5	4	4	3	5

Fig. 3. Location of inflection points. The green positions with values 1 are considered as inflection points. The red position with value 4 is not an inflection point as all four surrounding values from (11) are positive. (Color figure online)

The resulting inflection points form a curve of implicit points. It is quite densely sampled. However, for the purpose of the RBF approximation, we can reduce the inflection points to obtain a specific number of inflection points or reduce them as the distance between the closest two is larger than some threshold value.

3.3 RBF Approximation with Respecting Inflection Points

In the previous chapters, we presented the location of radial basis functions for $2\frac{1}{2}D$ function approximation. These locations are well placed to capture the main shape of the function $f(x, y)$. However we should add some more additional points to cover the whole approximation space. One set of additional radial basis functions is placed on the border and in the corners. The last additional radial basis functions are placed at locations with Halton distribution [4]

$$Halton(p)_k = \sum_{i=0}^{\lfloor \log_p k \rfloor} \frac{1}{p^{i+1}} \left(\left\lfloor \frac{k}{p^i} \right\rfloor \bmod p \right), \quad (12)$$

328 M. Cervenka et al.

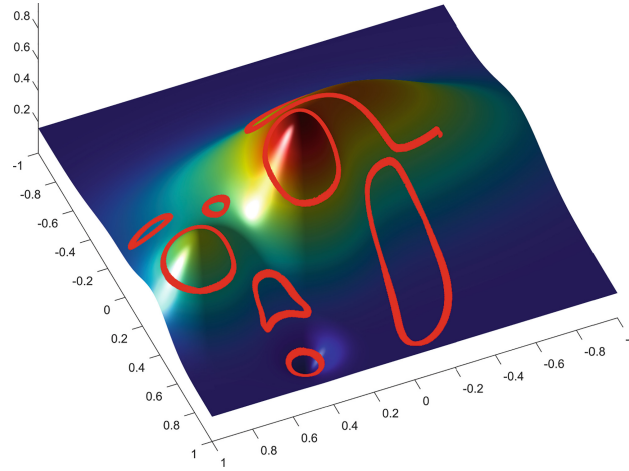


Fig. 4. An example of $2\frac{1}{2}D$ function with the curves of inflection points. The red curve represents the location of inflection points calculated using (11). (Color figure online)

where p is the prime number and k is the index of the calculated element, see Fig. 5 for an example of generated points distribution. It is recommended to use different primes for x and y coordinate.

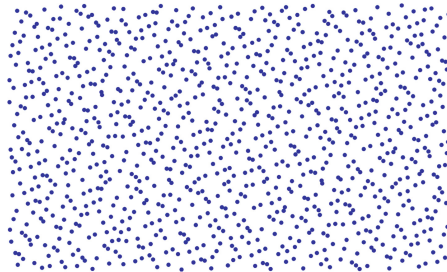


Fig. 5. The example of 10^3 Halton points. The Halton sequence was generated using two prime numbers [2, 3], i.e. for x coordinates a Halton sequence with the prime number 2 and for y coordinates a Halton sequence with the prime number 3.

Knowing all the positions of radial basis functions, we can compute the RBF approximation of the $2\frac{1}{2}D$ function. For the calculation we use the approach described in Sect. 2.1.

4 Experimental Results

In this section, we test the proposed approach for Radial basis function approximation. We tested the approach on all standard testing functions from [6]. In this

paper we present the experimental results on only three testing functions, while the results for other testing functions are similar. The selected testing functions are the following

$$f_1(x, y) = \frac{2}{11} \left(\sin(4x^2 + 4y^2) - (x + y) + \frac{5}{2} \right) \quad (13)$$

$$f_2(x, y) = \frac{3}{4} e^{-\frac{1}{4}((9x-2)^2 + (9y-2)^2)} + \frac{3}{4} e^{-\frac{1}{49}(9x+1)^2 - \frac{1}{10}(9y+1)^2} \\ + \frac{1}{2} e^{-\frac{1}{4}(9x-7)^2 - \frac{1}{4}(9y-3)^2} - \frac{1}{5} e^{-(9x-4)^2 - (9y-7)^2} \quad (14)$$

$$f_3(x, y) = \frac{1}{9} \tanh(9y - 9x) + 1 \quad (15)$$

All testing functions $z = f(x, y)$ were “normalized” to the interval $x, y \in \langle -1, 1 \rangle$ and the “height” z to $\langle 0, 1 \rangle$ in order to easily compare the proposed approximation properties and approximation error for all testing functions and we used Gaussian radial basis function in all experiments.

$$\varphi(r) = e^{-\epsilon r^2}. \quad (16)$$

Only some representative results are presented in this chapter. The visualization of (13) is in Fig. 6, the visualization of (14) is in Fig. 9 and the visualization of (15) is in Fig. 12.

The first function (13) is an inclined sine wave. This function contains inflection points formed in the elliptical shapes as can be seen in Fig. 8b. In the experiments we used the Gaussian radial basis function with a shape parameter $\epsilon = 1$. The visualization of original function together with the RBF approximation is in Fig. 6. The approximation consists of 246 radial basis functions (78 are at locations of inflection points and extremes, 24 are at the borders and 144 are Halton points). It can be seen that the RBF approximation is visually identical to the original one. Also precision of approximation is very high, see Fig. 8a

To have a more closer look at the quality of RBF approximation, we computed the isocontours of the both original and approximated functions, see Fig. 7. Those isocontours are again visually identical and cannot be seen any difference.

To compare the original and RBF approximated functions, we can compute the approximation error using the following formula for each point of evaluation.

$$Err = |f(x, y) - f_{RBF}(x, y)|, \quad (17)$$

where $f(x, y)$ is the value from input data set and $f_{RBF}(x, y)$ is the approximated value. Absolute error is used for evaluations data are normalized to $\langle -1, 1 \rangle \times \langle -1, 1 \rangle \times \langle 0, 1 \rangle$ as described recently. If we compute the approximation error for all input sample points, then we can calculate the average approximation error, which is $2.37 \cdot 10^{-4}$, and also the histogram of approximation error, see Fig. 8a. It can be seen that the most common approximation errors are quite low values below 0.4%. The higher approximation errors appear only few times. This proves a good properties of the approximation method.

330 M. Cervenka et al.

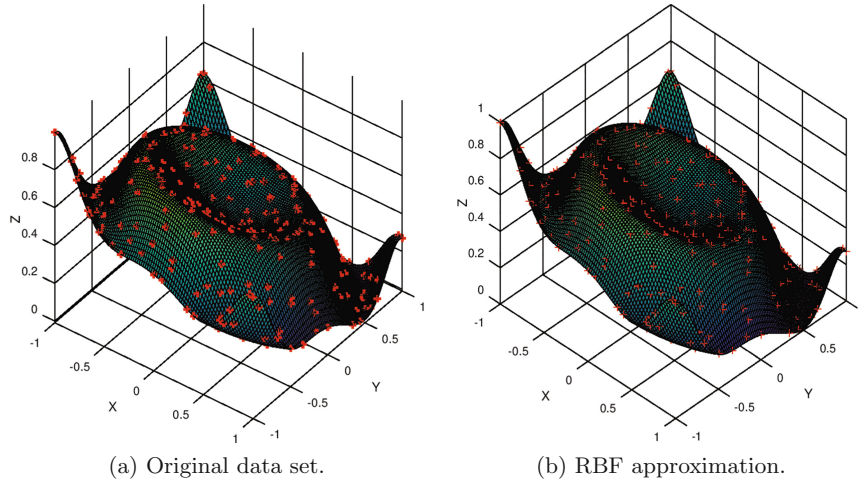


Fig. 6. The RBF approximation of $2\frac{1}{2}D$ function (13). The total number of RBF centers is 246 (red marks). (Color figure online)

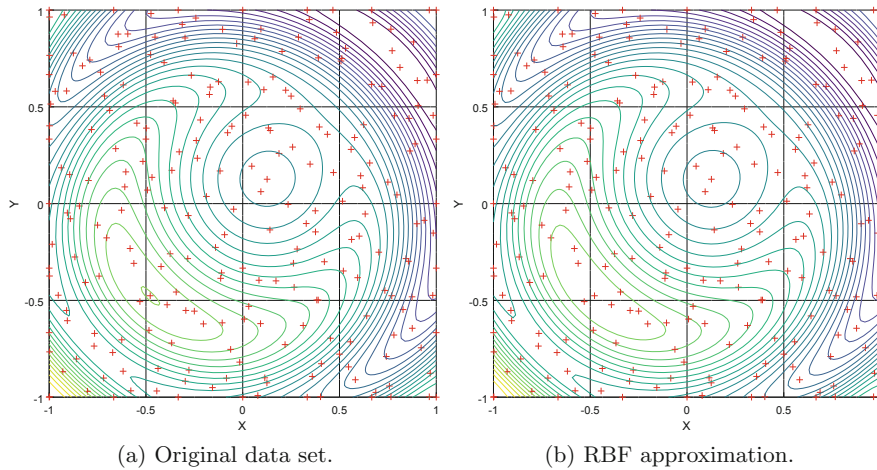


Fig. 7. The comparison of function isocontours.

The next testing function (14) is visualized together with the RBF approximation in Fig. 9. This function consists of four hills and the RBF approximation preserves the main shape. The only small difference is at the borders, which can be seen in more details in Fig. 10. The Gauss function with the shape parameter $\epsilon = 26$ was used as radial basis function.

The average approximation error is $2.75 \cdot 10^{-3}$ and the distribution of the approximation error can be seen in the histogram in Fig. 11.

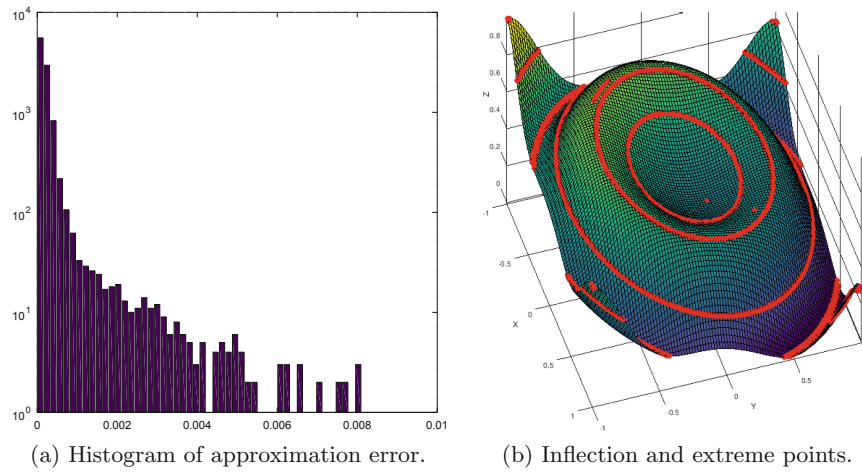


Fig. 8. The histogram (a) of approximation error for the function (13). The horizontal axis represents the absolute approximation error computed as (17). It should be noted, that the vertical axis is in logarithmic scale. The visualization (b) of all located inflection and extreme points.

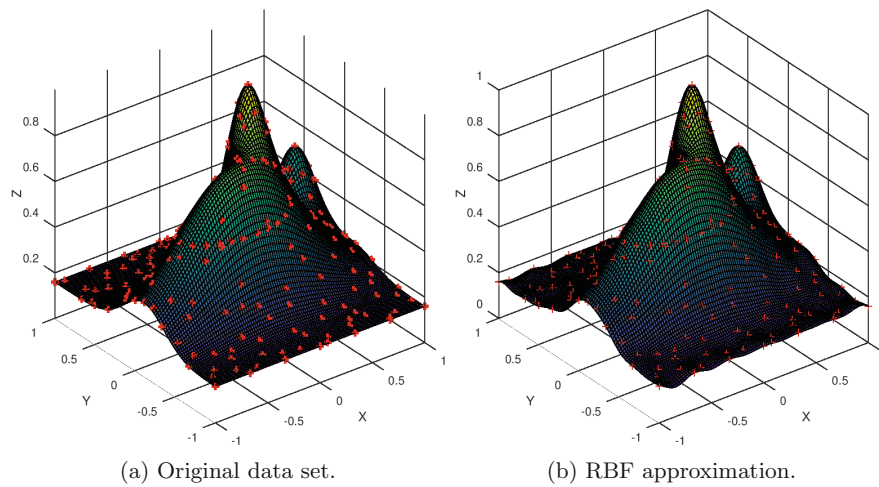


Fig. 9. The RBF approximation of $2\frac{1}{2}D$ function (14). The total number of RBF centers is 244 (red marks). (Color figure online)

The last function (15) for testing the proposed RBF approximation is visualized in Fig. 12a. This function is quite exceptional, because it has a sharp cliff on $x = y$. Such sharp cliffs are always very hard to approximate using the RBF. However using the proposed distribution of the radial basis functions, we are

332 M. Cervenka et al.

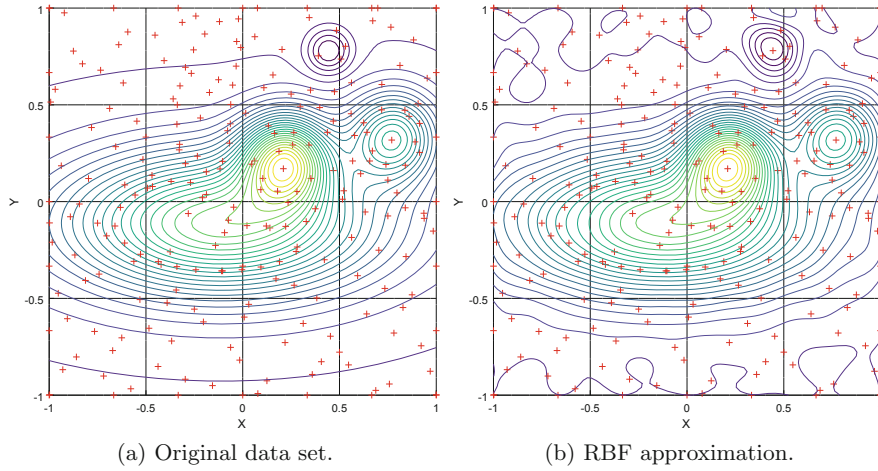


Fig. 10. The comparison of function isocontours.

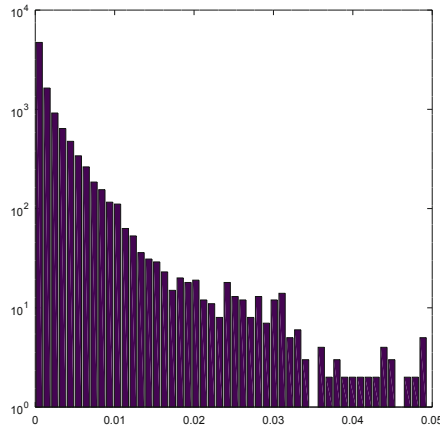


Fig. 11. The histogram of approximation error for the function (14). The horizontal axis represents the absolute approximation error computed as (17). It should be noted, that the vertical axis is in logarithmic scale.

able to approximate the sharp cliff quite well, see Fig. 12b. The problem, that comes up, is the wavy surface for $y > x$, see more details in Fig. 13. The Gauss function with the shape parameter $\epsilon = 25$ was used as the radial basis function.

The average approximation error for this specific function is $1.22 \cdot 10^{-2}$. This result is quite positive as the function is very hard to approximate using the RBF approximation technique. The histogram of the approximation error is visualized in Fig. 14.

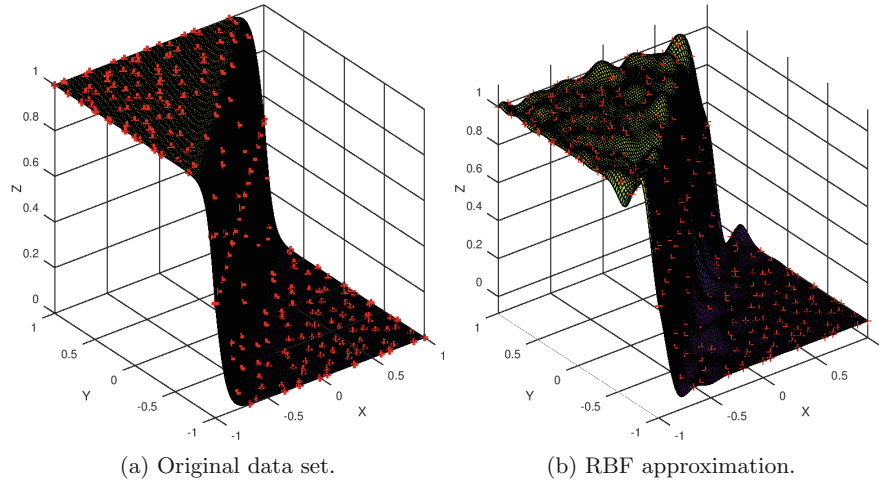


Fig. 12. The RBF approximation of $2\frac{1}{2}D$ function (15). The total number of RBF centers is 244 (red marks). (Color figure online)

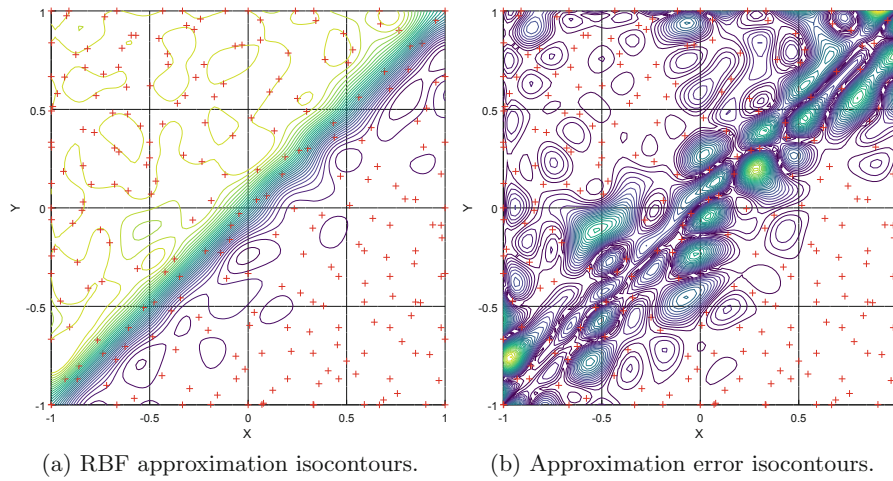


Fig. 13. The visualization of approximated function isocontours (a). The visualization of approximation error as isocontours plot (b), please note that the average approximation error is $1.22 \cdot 10^{-2}$.

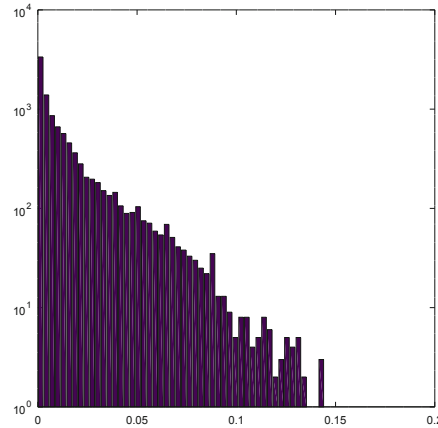


Fig. 14. The histogram of approximation error for the function (15). The horizontal axis represents the absolute approximation error computed as (17). It should be noted, that the vertical axis is in logarithmic scale.

5 Conclusion

We presented a new approach for approximation of $2\frac{1}{2}D$ scattered data using Radial basis functions respecting inflection points in the given data set. The RBF approximation uses the properties of the input data set, namely the extreme and the inflection points to determine the location of radial basis functions. This sophisticated placement of radial basis functions significantly improves the quality of the RBF approximation. It reduces the needed number of radial basis functions and thus creates even more compressed RBF approximation, too.

In future, the proposed approach is to be extended to approximate $3\frac{1}{2}D$ scattered data, while utilizing the properties of the input data set for the optimal placement of radial basis functions. Also efficient finding a shape parameters is to be explored.

Acknowledgments. The authors would like to thank their colleagues at the University of West Bohemia, Plzen, for their discussions and suggestions, and anonymous reviewers for their valuable comments and hints provided. The research was supported by projects Czech Science Foundation (GACR) No. 17-05534S and partially by SGS 2019-016.

References

1. Afatdoust, F., Esmailbeigi, M.: Optimal variable shape parameters using genetic algorithm for radial basis function approximation. *Ain Shams Engineering J.* **6**(2), 639–647 (2015)

2. Biazar, J., Hosami, M.: Selection of an interval for variable shape parameter in approximation by radial basis functions. In: *Advances in Numerical Analysis 2016* (2016)
3. Chen, S., Chng, E., Alkadhimi, K.: Regularized orthogonal least squares algorithm for constructing radial basis function networks. *Int. J. Control* **64**(5), 829–837 (1996)
4. Fasshauer, G.E.: *Meshfree Approximation Methods with MATLAB*, vol. 6. World Scientific (2007)
5. Fornberg, B., Wright, G.: Stable computation of multiquadric interpolants for all values of the shape parameter. *Comput. Math. Appl.* **48**(5–6), 853–867 (2004)
6. Franke, R.: A critical comparison of some methods for interpolation of scattered data. Technical report, Naval Postgraduate School Monterey CA (1979)
7. Ghosh-Dastidar, S., Adeli, H., Dadmehr, N.: Principal component analysis-enhanced cosine radial basis function neural network for robust epilepsy and seizure detection. *IEEE Trans. Biomed. Eng.* **55**(2), 512–518 (2008)
8. Hardy, R.L.: Multiquadric equations of topography and other irregular surfaces. *J. Geophys. Res.* **76**(8), 1905–1915 (1971)
9. Hardy, R.L.: Theory and applications of the multiquadric-biharmonic method 20 years of discovery 1968–1988. *Comput. Mathe. Appl.* **19**(8–9), 163–208 (1990)
10. Karim, A., Adeli, H.: Radial basis function neural network for work zone capacity and queue estimation. *J. Transp. Eng.* **129**(5), 494–503 (2003)
11. Larsson, E., Fornberg, B.: A numerical study of some radial basis function based solution methods for elliptic PDEs. *Comput. Math. Appl.* **46**(5), 891–902 (2003)
12. Majdisova, Z., Skala, V.: Big geo data surface approximation using radial basis functions: a comparative study. *Comput. Geosci.* **109**, 51–58 (2017)
13. Majdisova, Z., Skala, V.: Radial basis function approximations: comparison and applications. *Appl. Math. Model.* **51**, 728–743 (2017)
14. Majdisova, Z., Skala, V., Smolik, M.: Determination of stationary points and their bindings in dataset using RBF methods. In: Silhavy, R., Silhavy, P., Prokopova, Z. (eds.) *CoMeSySo 2018. AISC*, vol. 859, pp. 213–224. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-00211-4_20
15. Orr, M.J.: *Regularised centre recruitment in radial basis function networks*. Centre for Cognitive Science, Edinburgh University. Citeseer (1993)
16. Orr, M.J.: Regularization in the selection of radial basis function centers. *Neural Comput.* **7**(3), 606–623 (1995)
17. Pan, R., Skala, V.: A two-level approach to implicit surface modeling with compactly supported radial basis functions. *Eng. Comput.* **27**(3), 299–307 (2011)
18. Pan, R., Skala, V.: Surface reconstruction with higher-order smoothness. *Vis. Comput.* **28**(2), 155–162 (2012)
19. Prakash, G., Kulkarni, M., Sripathi, U.: Using RBF neural networks and Kullback-Leibler distance to classify channel models in free space optics. In: *2012 International Conference on Optical Engineering (ICOE)*, pp. 1–6. IEEE (2012)
20. Sarra, S.A., Sturgill, D.: A random variable shape parameter strategy for radial basis function approximation methods. *Eng. Anal. Bound. Elem.* **33**(11), 1239–1245 (2009)
21. Schagen, I.P.: Interpolation in two dimensions - a new technique. *IMA J. Appl. Math.* **23**(1), 53–59 (1979)
22. Skala, V.: Fast interpolation and approximation of scattered multidimensional and dynamic data using radial basis functions. *WSEAS Trans. Math.* **12**(5), 501–511 (2013)

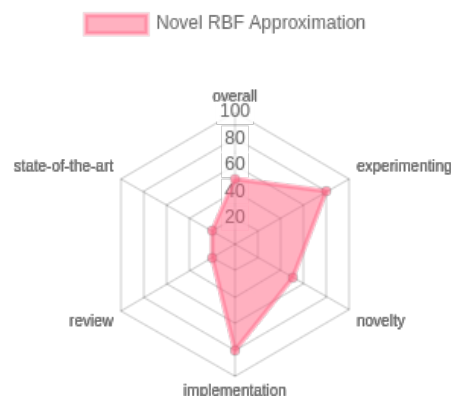
23. Skala, V.: RBF interpolation with CSRBF of large data sets. *Procedia Comput. Sci.* **108**, 2433–2437 (2017)
24. Smolik, M., Skala, V.: Spherical RBF vector field interpolation: experimental study. In: 2017 IEEE 15th International Symposium on Applied Machine Intelligence and Informatics (SAMII), pp. 000431–000434. IEEE (2017)
25. Smolik, M., Skala, V.: Large scattered data interpolation with radial basis functions and space subdivision. *Integr. Comput.-Aided Eng.* **25**(1), 49–62 (2018)
26. Smolik, M., Skala, V., Majdisova, Z.: Vector field radial basis function approximation. *Adv. Eng. Softw.* **123**(1), 117–129 (2018)
27. Smolik, M., Skala, V., Nedved, O.: A comparative study of LOWESS and RBF approximations for visualization. In: Gervasi, O., et al. (eds.) ICCSA 2016. LNCS, vol. 9787, pp. 405–419. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-42108-7_31
28. Uhlir, K., Skala, V.: Reconstruction of damaged images using radial basis functions. In: 2005 13th European Signal Processing Conference, pp. 1–4. IEEE (2005)
29. Wang, J., Liu, G.: On the optimal shape parameters of radial basis functions used for 2-D meshless methods. *Comput. Methods Appl. Mech. Eng.* **191**(23–24), 2611–2630 (2002)
30. Wendland, H.: Computational aspects of radial basis function approximation. *Stud. Comput. Math.* **12**, 231–256 (2006)
31. Wright, G.B.: Radial basis function interpolation: numerical and analytical developments (2003)
32. Yingwei, L., Sundararajan, N., Saratchandran, P.: Performance evaluation of a sequential minimal radial basis function (RBF) neural network learning algorithm. *IEEE Trans. Neural Netw.* **9**(2), 308–318 (1998)
33. Zhang, X., Song, K.Z., Lu, M.W., Liu, X.: Meshless methods based on collocation with radial basis functions. *Comput. Mech.* **26**(4), 333–343 (2000)

9.2 Novel RBF Approximation Method Based on Geometrical Properties for Signal Processing with a New RBF Function: Experimental Comparison

The primary objective of the second contribution of the authors, as described in [93], was to exhaustively investigate the behaviour of a carefully chosen subset of RBFs. The study involved testing the performance of CSRBF, Gaussian RBF, and TPS RBF, together with a newly proposed radial basis function was performed, to choose the suitable RBFs. Sixteen distinct 1D signals, strategically designed to expose potential weaknesses in each type of RBF, were used for the testing. The methodology used in the previous article [3] was adopted as a comparable centre placement approach.

The experiment results revealed that accurate approximations could be achieved, with a mean square error consistently below 1% for all global RBFs. In particular, the proposed RBF, denoted $\varphi(r) = r^2(r^\alpha - 1)$, outperformed all other RBFs in these cases. However, more extensive testing is required to ensure its relevance to real-world data, especially when dealing with higher-dimensional datasets. The Gaussian RBF showed suboptimal performance in the testing scenarios, although it was stable in terms of the conditionality of the equation system and increased predictability.

The exploration of individual RBFs in this study laid the foundation for a more informed and efficient selection of RBFs for muscle modelling. The novel and the Gaussian RBFs were tested as potential candidates to approximate a muscle model, emerging as the best options among those investigated in this study. Further research and testing are recommended to refine the proposed RBF's understanding and suitability for practical applications involving higher-dimensional datasets.



Publication [93]:

SKALA, V.; CERVENKA, M. Novel RBF Approximation Method Based on Geometrical Properties for Signal Processing with a New RBF Function: Experimental Comparison. *Informatics 2019, IEEE proceedings*. 2019, pp. 357–362. ISBN 978-1-7281-3178-8. Available from DOI: <https://doi.org/10.1109/Informatics47936.2019.9119276>. UT WoS: 000610452900074, EID: 2-s2.0-85087090327, OBD: 43929007

Novel RBF Approximation Method Based on Geometrical Properties for Signal Processing with a New RBF Function: Experimental Comparison

Vaclav Skala
University of West Bohemia
Pilsen, Czech Republic
<http://www.VaclavSkala.eu>

Martin Cervenka
University of West Bohemia
Pilsen, Czech Republic
cervemar@kiv.zcu.cz¹

Abstract— Interpolation and approximation methods are widely used in many areas. They can be divided to methods based on meshing (tessellation) of the data domain and to meshless (meshfree) methods, which do not require the domain tessellation of scattered data. Scattered n -dimensional data radial basis function (RBF) interpolation and approximation leads to a solution of linear system of equations.

This contribution presents a new approach to the RBF approximation based on analysis of geometrical properties of signals, i.e. sampled curves. Also a newly developed radial basis function was used and proved better precision of approximation.

Experimental comparison of several RBF functions (Gauss, Thin-Plate Spline, CS-RBF and a new proposed RBF) is described with analysis of their properties. Special attention was taken to the precision of approximation and conditionality issues. The proposed approach can be extended to a higher dimensional case and for vector data, e.g. fluid flow, too.

Index Terms—Approximation, Radial basis functions, RBF, Signal processing, Computer graphics, Meshless methods.

I. INTRODUCTION

Interpolation and approximation of scattered data is required in many areas. As there is no ordering defined for d -dimensional case, if $d \geq 1$, usually two approaches are taken:

- Tessellation of the data domain, e.g. using Delaunay triangulation and application of a selected interpolation or approximation method. However, the Delaunay tessellation has a computational complexity $O(n^{\lceil d^2/2 \rceil})$. This leads high computational complexity and to implementation problems in the case that $d > 2$. Another computational problems can be expected, if smoothness of the interpolation or approximation is required.
- Use of meshless methods based on Radial Basis Functions (RBF) use leads to a solution of a linear system of equations $\mathbf{Ax} = \mathbf{b}$, in general, and the computational complexity is nearly independent from the dimensionality of the data domain, see Hardy [1]. Even more, if relevant RBF function is selected, higher degree of smoothness is obtained. On the other hand, interpolation

and approximation methods based on meshless approach usually have a problem with a precision on borders or on discontinuities, in general. The meshless methods can be also used for approximation of vector data, i.e. fluid flow etc. However, some RBFs applications might lead to numerical problems due to ill-conditioned matrix of the linear system, especially for large data sets.

Usually, the approximation methods use a general method, which is not taking directly geometrical properties of the signal into account, e.g. Singh [2], Skala & Smolik & Nedved [3]. However, if some information on signal geometry can be extracted from data and used, the approximation should be more precise and simpler. Such approach has been used by Majdisova & Skala & Smolik [4]. This approach seems to be quite complicated, as it is based on properties of cubic curve in floating data window.

This contribution is focused on the following main aspects:

- how geometrical properties of a signal can be efficiently used for good and robust approximation,
- how to approximate a signal, i.e. a sampled curves, with a good precision with a minimal number of radial basis functions (RBFs),
- what kind of RBFs probably gives better results,
- what are properties of a newly developed RBF in terms of precision and numerical precision.

As some RBF functions have a parameter, called a shape parameter, some problems can be expected with an optimal shape parameter selection or estimation. Some proposal how to select suitable shape parameters were introduced by Karageorghis [5], Wang & Liu [6] for range of shape parameters generation, Afiatdoust & Esmailbeigi [7] presented use of genetic algorithm. Moreover, Sarra & Sturgill [8] propose non-deterministic approach based on random shape parameter generation. However, some experiments recently made by Skala & Karim & Zabran [9] proved, that there is probably no optimal constant shape parameter nor one vector of optimal shape parameters for each RBF.

Another problem is shape parameter selection Karageorghis [5]. There are two particular cases which may occur in

The research was supported by Czech Science Foundation (GACR) project No.GA 17-05534S and partially by SGS 2019-016 project

¹ Corresponding author

general: approximation will be inaccurate or the problem may become ill-conditioned. That is why correct shape parameter selection is needed. Some approaches to select suitable shape parameters already exists. Approach introduced by Wang & Liu [6] for example generates range of shape parameters, Afiatdoust & Esmailbeigi [7] presented their approach using genetic algorithm. Moreover, Sarra & Sturgill [8] propose non-deterministic approach based on random generator. Some recent research have been devoted to variable shape parameters, i.e., each RBF function has a different shape parameter, e.g. Majdisova [4], Skala [9].

Application of the RBF interpolation and approximation in engineering practice can be found in Biancolini [10], Fasshauer [11], Menandro [12]. Also RBFs are used for vector field interpolation and approximation, e.g. Smolik [13], [14], [15], [16], Skala [17], solution of partial differential equations (PDE) Zhang [18], Neural Networks RBF Yinwey [19] and reconstruction of implicit curves Cuomo [20]. Comparison of selected RBFs can be found in Majdisova [21].

II. RBF INTERPOLATION

According to Hardy [22], RBF interpolation is based on determining the distance of two point (in the d -dimensional space in general). The interpolation is given in the form:

$$h(\mathbf{x}) = \sum_{j=1}^N \lambda_j \varphi(\|\mathbf{x} - \mathbf{x}_j\|) = \sum_{j=1}^N \lambda_j \varphi(r_j) \quad (1)$$

where r_j is the distance from a point x to the point x_j . As the parameter of the function φ is a distance of two points in the d -dimensional space, the interpolation is non-separable by a dimension. The RFBs will be described in detailed latter on.

For each point x_i the interpolating function has to have value h_i . Therefore, we are getting a system of linear equations:

$$h(\mathbf{x}_i) = \sum_{j=1}^N \lambda_j \varphi(\|\mathbf{x}_i - \mathbf{x}_j\|) = \sum_{j=1}^N \lambda_j \varphi(r_{ij}) \quad (2)$$

where λ_j are unknown weights for each radial basis function, N is the number of given points and φ is the radial basis function itself. It can be rewritten if the matrix form as:

$$\mathbf{A}\lambda = \mathbf{h} \quad (3)$$

or in a detailed form as (4):

$$\begin{bmatrix} \varphi_{11} & \cdots & \varphi_{1j} & \cdots & \varphi_{1N} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \varphi_{i1} & \cdots & \varphi_{ij} & \cdots & \varphi_{iN} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \varphi_{N1} & \cdots & \varphi_{Nj} & \cdots & \varphi_{NN} \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_i \\ \vdots \\ \lambda_N \end{bmatrix} = \begin{bmatrix} h_1 \\ \vdots \\ h_i \\ \vdots \\ h_N \end{bmatrix} \quad (4)$$

After solving the system of linear equations, interpolated value at the point \mathbf{x} is computed using (1). However, due to numerical robustness and stability, additional polynomial

conditions are usually added Skala [23] [24]. In the case of an additional polynomial we obtain:

$$h(\mathbf{x}_i) = \sum_{j=1}^N \lambda_j \varphi(\|\mathbf{x}_i - \mathbf{x}_j\|) + P_k(\mathbf{x}_i) \quad (5)$$

In the case of a linear polynomial (in $2\frac{1}{2}D$ case):

$$P_k(x, y) = a_0 + a_1x + a_2y \quad (6)$$

This additional conditions can be rewritten as:

$$\sum_{j=1}^N \lambda_j = 0 \quad \sum_{j=1}^N \lambda_j \mathbf{x}_j = 0 \quad (7)$$

$$\begin{bmatrix} \varphi_{11} & \cdots & \varphi_{1N} & 1 & x_1 & y_1 \\ \vdots & \ddots & \vdots & 1 & \vdots & \vdots \\ \varphi_{N1} & \cdots & \varphi_{NN} & 1 & x_N & y_N \\ 1 & 1 & 1 & 0 & 0 & 0 \\ x_1 & \cdots & x_N & 0 & 0 & 0 \\ y_1 & \cdots & y_N & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_N \\ a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_N \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (8)$$

This matrix can be further rewritten in more compact way:

$$\begin{bmatrix} \mathbf{A} & \mathbf{P} \\ \mathbf{P}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \lambda \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} \mathbf{h} \\ \mathbf{0} \end{bmatrix} \quad (9)$$

The matrix \mathbf{P} represents polynomial additional conditions, λ is vector of RBF weights, vector \mathbf{a} contains resulting polynomial coefficients and \mathbf{h} are given values at the given points.

It should be noted that in some cases that it can be counterproductive especially for large scope of domain data Jäger [25], Skala [23] [24].

III. RBF APPROXIMATION

Approximation methods are slightly different from interpolation as the final approximated curve does not need to "pass" all the given points. If the matrix \mathbf{A} is a square matrix (RBF count M is equal to size of the \mathbf{x} vector), this is an interpolation problem. On the other hand, when M is smaller, it becomes approximation problem, because equation system became over-determined. Let us ξ_j are significant points in the given signal, then the approximation can be formulated as:

$$h(\mathbf{x}_i) = \sum_{j=1}^M \lambda_j \varphi(\|\mathbf{x}_i - \xi_j\|) \quad (10)$$

where λ_j are weights of each radial basis function, M is count of RBF being used and $M \ll N$, the φ is the RBF, ξ_j are center (important) points. Then the approximation can be expressed by equation (11):

$$\begin{bmatrix} \varphi_{11} & \cdots & \varphi_{1M} \\ \vdots & \ddots & \vdots \\ \varphi_{i1} & \cdots & \varphi_{iM} \\ \vdots & \ddots & \vdots \\ \varphi_{N1} & \cdots & \varphi_{NM} \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_M \end{bmatrix} = \begin{bmatrix} h_1 \\ \vdots \\ h_i \\ \vdots \\ h_N \end{bmatrix} \quad (11)$$

Overdetermined linear equation system is to be solved, e.g. by the Least Square Error method (12).

$$\mathbf{x} = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{h} \quad (12)$$

However, in the approximation case, the additional polynomial conditions cannot be used Majdisova [26], [27].

IV. RBF FUNCTIONS

There are several radial basis functions Fasshauer [11], Majdisova [28] [29], Buhmann [30]. They can be divided into two major groups (Smolik [13]):

- "global" RBFs having global influence, e.g. $r^2 \lg(r)$, $\exp(-\alpha r^2)$, $\sqrt{\alpha + r^2}$, $1/\sqrt{\alpha + r^2}$, $1/(\alpha + r^2)$, etc. where α is a shape parameter. The RBF matrix is usually full and ill conditioned.
 - "local" RBFs - Compactly Supported RBF (CS-RBF) have a non-zero value for the interval $\langle 0, 1 \rangle$ only. The RBF matrix is usually sparse as it depends on the scaling of the interval $\langle 0, 1 \rangle$ to the required one. Some examples are listed in Tab.I and they are shown on Fig.1. The article Menandro [12] describes this class of RBFs.
- Some examples of RBFs are listed in Tab.I and at Fig.1.

ID	Function
1	$(1-r)_+$
2	$(1-r)_+^3(3r+1)$
3	$(1-r)_+^5(8r^2+5r+1)$
4	$(1-r)_+^2$
5	$(1-r)_+^4(4r+1)$
6	$(1-r)_+^6(35r^2+18r+3)$
7	$(1-r)_+^8(32r^3+25r^2+8r+3)$
8	$(1-r)_+^3$
9	$(1-r)_+^5(5r+1)$
10	$(1-r)_+^7(16r^2+7r+1)$

TABLE I: List of well-known CS-RBF.

V. PROPOSED RBF

In this paper a new global radial basis function is proposed. It has one shape parameter and is defined as is in (13).

$$\varphi(r) = r^2(r^\alpha - 1) \quad (13)$$

where α is a shape parameter (we use $\alpha = 1.8$ globally). The function is shown at Fig.2.

VI. DESCRIPTION OF EXPERIMENTS

For sake of simplicity, all signal values has been normalized to interval $h(x_i) \in \langle 0, 1 \rangle$, i. e. $y_i = h(x_i)$. Signal domain has been set to the $x_i \in \langle 0, 1 \rangle$ as well for the same reason.

As already mentioned, four RBF has been used for testing purposes. Gaussian RBF (in equation (14)) is global radial basis function with one shape parameter α defining its dispersion.

$$\varphi(r) = e^{-\alpha r^2} \quad (14)$$

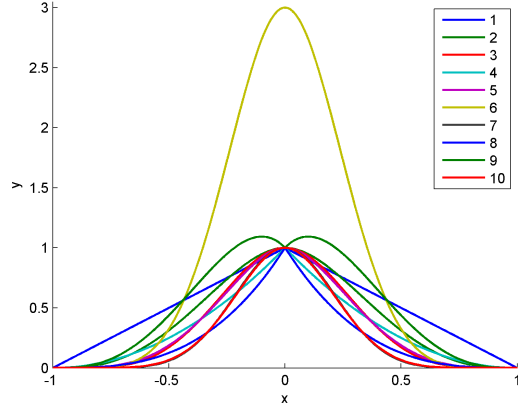


Fig. 1: Plotted CSRBFs taken from [13].

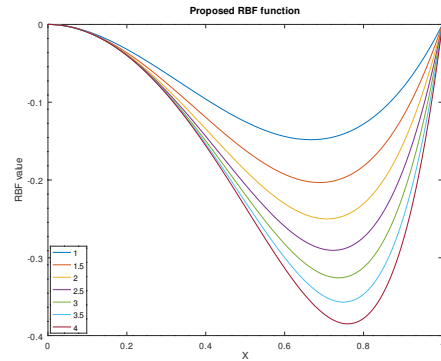


Fig. 2: Proposed RBF function with various shape parameters.

Thin plate spline (TPS) function is the next one which has been tested. It is global RBF as well, and it is defined in (15).

$$\varphi(r) = r^2 \log r \quad (15)$$

Next class of function on the list is CS-RBF. In particular, (16) function has been selected from the Tab.I. It is worth noting that this function (like all CS-RBF) is local.

$$\varphi(r) = (1-r)_+^7(16r^2+7r+1) \quad (16)$$

Last but not least we propose another RBF. It is global RBF with a shape parameter α and it is described by equation (13).

Described radial basis functions approximation has been tested against multiple signals, however, there are listed only some of tested signals in this paper. This signal subset contains following functions:

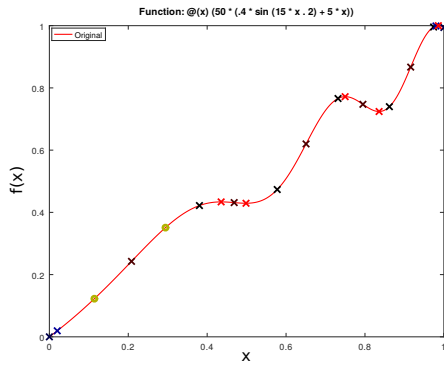


Fig. 3: Function 1.

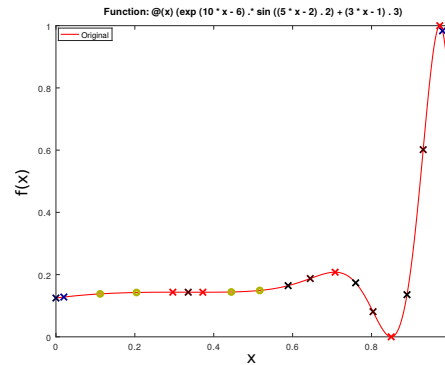


Fig. 5: Function 3.

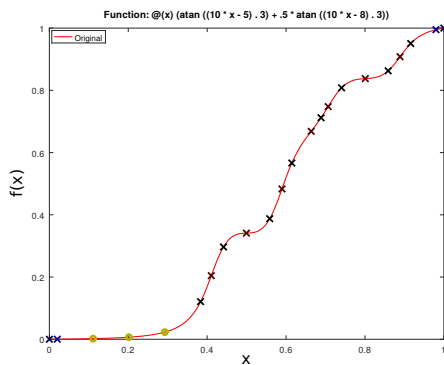


Fig. 4: Function 2.

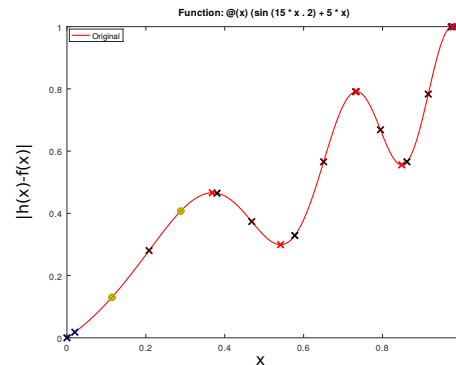


Fig. 6: Function 4.

- 1) $f_1(x) = 50(0.4\sin(15x^2) + 5x)$
- 2) $f_2(x) = \operatorname{atan}((10x - 5)^3) + 0.5\operatorname{atan}((10x - 8)^3)$
- 3) $f_3(x) = e^{10x-6}\sin((5x - 2)^2) + (3x - 1)^3$
- 4) $f_4(x) = \sin(15x^2) + 5x$

Selected signals (sampled curves) are shown at Fig.3 – 6.

The RBF center points ξ_i are shown as various marks on plotted signal curve on each plot. Four already mentioned RBFs (Gauss, TPS, CS-RBF and the proposed one) were selected to approximate these signals (among others).

VII. EXPERIMENTAL RESULTS

The proposed approach was tested on several testing functions, see Tab.II. It should be noted explicitly, that all function were normalized for the interval $x \in (0, 1), y \in (0, 1)$. In order to easily compare errors of the proposed RBF approximation methods.

As the proposed RBF approximation is based on finding significant geometric properties, such as maxima, minima, inflection points, etc., the conditionality of the RBF metrics and

mutual comparison of errors were analyzed. Error behaviour is considered as the critical issue in approximation in general.

In this contribution only couple of function used are presented, see Fig.3 – 6. The relevant approximation error behaviour is presented at Fig.7 – 10.

Tab.III – V present the error behaviour numerically. Exact numerical experimental results are presented in following Tab.III (mean square error), Tab.IV (maximum absolute error) and Tab.V (condition numbers of equation system matrix \mathbf{A} defined in equation 3) respectively. It can be seen that the high error is caused by significant under-sampling. Inclusion of additional point(s) leads to significant decrease of the approximation error. It should be noted, that this contribution is analyzing the approximation behaviour at the lowest border of the sampling frequency.

The experiments proved that the sampled curves can be efficiently approximated by the few "important" points, i.e. extrema, inflections etc., with acceptably low error. The proposed method also leads to good data compression.

1	$\sin(15x^2) + 5x$
2	$0.5 \cos(20x) + 5x$
3	$50(0.4 \sin(15x^2) + 5x)$
4	$\sin(8\pi x)$
5	$\sin(6\pi x^2)$
6	$\sin(25x + 0.1)/(25x + 0.1)$
7	$2 \sin(2\pi x) + \sin(4\pi x)$
8	$2 \sin(2\pi x) + \sin(4\pi x) + \sin(8\pi x)$
9	$-2 \sin(2\pi x) + \cos(6\pi x)$
10	$2 \sin(2\pi x) + \cos(6\pi x)$
11	$-2 \sin(2\pi x) + \cos(6\pi x) - x$
12	$-2 \cos(2\pi x) - \cos(4\pi x)$
13	$\operatorname{atan}\left(\frac{(10x-5)^3}{(10x-8)^3}\right) + 0.5 \operatorname{atan}\left(\frac{(10x-8)^3}{(10x-5)^3}\right)$
14	$(4.48x - 1.88) \sin\left(\frac{(4.88x - 1.88)^2}{(10x-8)^3}\right) + 1$
15	$e^{10x-6} \sin\left(\frac{(5x-2)^2}{(10x-8)^3}\right) + (3x-1)^3$
16	$(1/9) \operatorname{tanh}(9x + 0.5)$

TABLE II: Tested artificial signals.

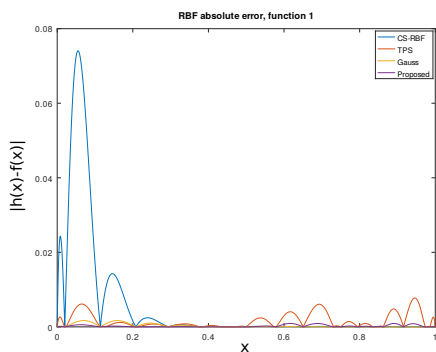


Fig. 7: Differences for $y = f_1(x)$.

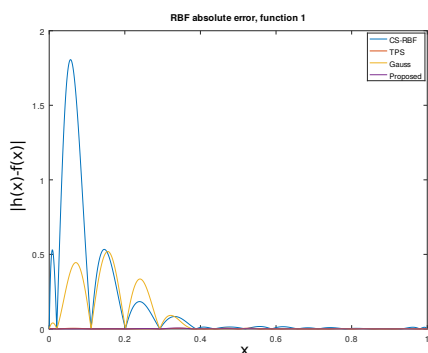


Fig. 8: Differences for $y = f_2(x)$.

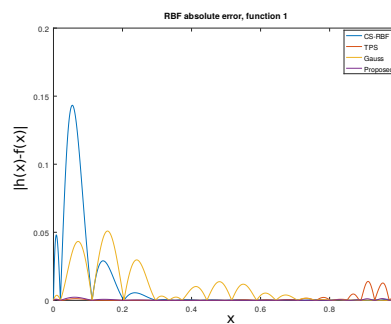


Fig. 9: Differences for $y = f_3(x)$.

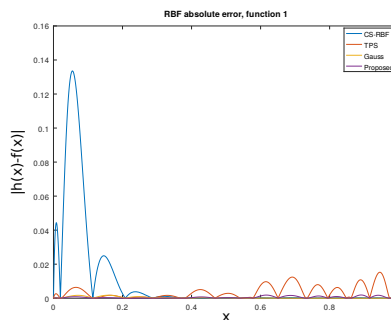


Fig. 10: Differences for $y = f_4(x)$.

Experiments also proved that application of the CS-RBFs with a constant shape parameter is not convenient, unless additional points are not included in the approximation. (maximum error for the function 2, if CS-RBF is used).

The proposed RBF approximation method was also tested for a newly developed RBF. The experiments proved significant precision improvement of the final approximation over the TPS function.

Function number	Radial basis function			
	CS-RBF	TPS	Gauss	Proposed
1	$2.41 \cdot 10^{-4}$	$6.40 \cdot 10^{-6}$	$3.13 \cdot 10^{-7}$	$1.21 \cdot 10^{-7}$
2	$1.54 \cdot 10^{-1}$	$3.95 \cdot 10^{-6}$	$2.56 \cdot 10^{-2}$	$2.34 \cdot 10^{-6}$
3	$9.21 \cdot 10^{-4}$	$8.67 \cdot 10^{-6}$	$2.51 \cdot 10^{-4}$	$3.30 \cdot 10^{-7}$
4	$7.92 \cdot 10^{-4}$	$2.56 \cdot 10^{-5}$	$3.12 \cdot 10^{-7}$	$5.23 \cdot 10^{-7}$

TABLE III: Mean square error.

VIII. CONCLUSION

In this contribution a novel approach for RBF approximation based on geometrical properties of a sampled curve (signal) is presented. Experiments proved advantages of the global functions over CS-RBFs are sensitive to the shape parameter

Function number	Radial basis function			
	CS-RBF	TPS	Gauss	Proposed
1	$7.33 \cdot 10^{-2}$	$7.76 \cdot 10^{-3}$	$1.70 \cdot 10^{-3}$	$9.21 \cdot 10^{-4}$
2	$1.81 \cdot 10^0$	$5.24 \cdot 10^{-3}$	$5.19 \cdot 10^{-1}$	$6.29 \cdot 10^{-3}$
3	$1.44 \cdot 10^{-1}$	$1.39 \cdot 10^{-2}$	$5.09 \cdot 10^{-2}$	$2.30 \cdot 10^{-3}$
4	$1.33 \cdot 10^{-1}$	$1.52 \cdot 10^{-2}$	$1.76 \cdot 10^{-3}$	$1.89 \cdot 10^{-3}$

TABLE IV: Maximum absolute error.

Function number	Radial basis function			
	CS-RBF	TPS	Gauss	Proposed
1	$6.60 \cdot 10^{-19}$	$8.80 \cdot 10^{-6}$	$1.44 \cdot 10^{-12}$	$2.49 \cdot 10^{-9}$
2	$2.68 \cdot 10^{-18}$	$8.47 \cdot 10^{-5}$	$2.43 \cdot 10^{-12}$	$1.05 \cdot 10^{-7}$
3	$4.05 \cdot 10^{-18}$	$4.58 \cdot 10^{-6}$	$2.35 \cdot 10^{-12}$	$2.25 \cdot 10^{-9}$
4	$2.78 \cdot 10^{-18}$	$2.44 \cdot 10^{-6}$	$9.17 \cdot 10^{-14}$	$3.20 \cdot 10^{-10}$

TABLE V: Condition numbers.

selection and require more points for acceptable approximation in general.

The newly developed RBF function is better in the precision terms over the TPS function, however, the TPS function has a little bit worse conditionality which can be improved by additional polynomial.

The proposed RBF seems to be an alternative to the TPS function offering better error, however the influence of the shape parameter α is under investigation. The optimal choice of the shape parameter α is an open question.

The experiments also proved that the CS-RBFs require variable shape parameter which is significant result as CS-RBFs are used in many areas, e.g. solution of partial differential equations, etc. The adaptive shape parameter for CS-RBFs is to be explored in future.

IX. ACKNOWLEDGMENT

The authors would like to thank their colleagues and students at the University of West Bohemia for their discussions and suggestions, and especially to Zuzana Majdisova for making some testing functions, Michal Smolik, Marek Zabran and Maria Martynova at the University of West Bohemia for the help with the MATLAB programming issues. Thanks belong also to anonymous reviewers for their valuable comments and hints provided.

REFERENCES

[1] R. L. Hardy, "Theory and applications of the multiquadric-biharmonic method 20 years of discovery 1968-1988," *Computers & Mathematics with Applications*, vol. 19, pp. 163–208, 1990.

[2] B. Singh and D. Toshniwal, "MOWM: Multiple Overlapping Window Method for RBF based missing value prediction on big data," *Expert Systems with Applications*, vol. 122, pp. 303 – 318, 2019.

[3] M. Smolik, V. Skala, and O. Nedved, "A comparative study of LOWESS and RBF approximations for visualization," in *International Conference on Computational Science and Its Applications*, 2016, pp. 405–419.

[4] Z. Majdisova, V. Skala, and M. Smolik, "Algorithm for placement of reference points and choice of an appropriate variable shape parameter for the RBF approximation (accepted for publication)," *Integrated Computer Aided Engineering*, IOS Press, 2019.

[5] A. Karageorghis and P. Tryfonos, "Shape parameter estimation in RBF function approximation," *International Journal of Computational Methods and Experimental Measurements*, vol. 7, pp. 246–259, 2019.

[6] J. Wang and G. Liu, "On the optimal shape parameters of radial basis functions used for 2-d meshless methods," *Computer methods in applied mechanics and engineering*, vol. 191, pp. 2611–2630, 2002.

[7] F. Afshardoust and M. Esmailbeigi, "Optimal variable shape parameters using genetic algorithm for radial basis function approximation," *Ain Shams Engineering Journal*, vol. 6, pp. 639–647, 2015.

[8] S. A. Sarra and D. Sturgill, "A random variable shape parameter strategy for radial basis function approximation methods," *Engineering Analysis with Boundary Elements*, vol. 33, pp. 1239–1245, 2009.

[9] V. Skala, S. Karim, and M. Zabran, "Radial basis function approximation optimal shape parameters estimation: Preliminary experimental results (accepted for publication)," in *SKSM 27 Conference, Selangor, Malaysia, Symposium Kebangsaan Sains ke 27*. AIP Press, 2019.

[10] M. E. Biancolini, *Fast Radial Basis Functions for Engineering Applications*, 1st ed. Springer International Publishing, 2017.

[11] G. Fasshauer, *Meshfree Approximation Methods with Matlab*, 1st ed. World Scientific, 2007.

[12] F. Menandro, "Two new classes of compactly supported radial basis functions for approximation of discrete and continuous data," *Engineering Reports*, vol. 2019;1:e12028, pp. 1–30, 2019.

[13] M. Smolik and V. Skala, "Large scattered data interpolation with radial basis functions and space subdivision," *Integrated Computer Aided Engineering*, vol. 25, pp. 49–62, 2018.

[14] —, "Efficient simple large scattered 3D vector fields radial basis function approximation using space subdivision," in *Computational Science and Its Application, ICCSA 2019 proceedings*, 2019, pp. 337–350.

[15] —, "Classification of critical points using a second order derivative," in *ICCS 2017, Procedia Computer Science*, vol. 108. Elsevier, 2017, pp. 2373–2377.

[16] —, "Vector field second order derivative approximation and geometrical characteristics," in *ICCSA 2017 Conf.*, vol. 10404. Springer, 2017, pp. 148–158.

[17] V. Skala, R. Pan, and O. Nedved, "Making 3D replicas using a flatbed scanner and a 3d printer," in *ICCSA 2014*. Springer, 2014, pp. 76–86.

[18] X. Zhang, K. Z. Song, M. W. Lu, and X. Liu, "Meshless methods based on collocation with radial basis functions," *Computational mechanics*, vol. 26, pp. 333–343, 2000.

[19] L. Yingwei, N. Sundararajan, and P. Saratchandran, "Performance evaluation of a sequential minimal radial basis function (RBF) neural network learning algorithm," *IEEE Transactions on neural networks*, vol. 9, pp. 308–318, 1998.

[20] S. Cuomo, A. Galletti, G. Giunta, and L. Marcellino, "Reconstruction of implicit curves and surfaces via RBF interpolation," *Applied Numerical Mathematics*, vol. 116, pp. 157 – 171, 2017.

[21] Z. Majdisova and V. Skala, "Radial basis function approximations: comparison and applications," *Applied Mathematical Modelling*, vol. 51, pp. 728–743, 2017.

[22] R. L. Hardy, "Multiquadric equations of topography and other irregular surfaces," *Journal of geophysical research*, vol. 76, pp. 1905–1915, 1971.

[23] V. Skala, "RBF interpolation and approximation of large span data sets," in *MCSI 2017 Corfu*. IEEE, 2018, pp. 212–218.

[24] —, "RBF interpolation with CSRBF of large data sets," in *ICCS 2017, Procedia Computer Science*, vol. 108. Elsevier, 2017, pp. 2433–2437.

[25] J. Jäger, "Advances in radial and spherical basis function interpolation," Ph.D. dissertation, Justus-Liebig-Universität, Otto-Behaghel-Str. 8, 35394 Giessen, 2018.

[26] Z. Majdisova and V. Skala, "A radial basis function approximation for large datasets," in *SIGRAD 2016*, 2016, pp. 9–14.

[27] —, "A new radial basis function approximation with reproduction," in *CGVCVIP 2016*, 2016, pp. 215–222.

[28] —, "Radial basis function approximations: Comparison and applications," *Applied Mathematical Modelling*, vol. 51, pp. 728–743, 2017.

[29] —, "Big geo data surface approximation using radial basis functions: A comparative study," *Computers and Geosciences*, vol. 109, pp. 51–58, 2017.

[30] M. Buhmann, "On quasi-interpolation with radial basis functions," *Journal of Approximation Theory*, vol. 72, no. 1, pp. 103 – 130, 1993.

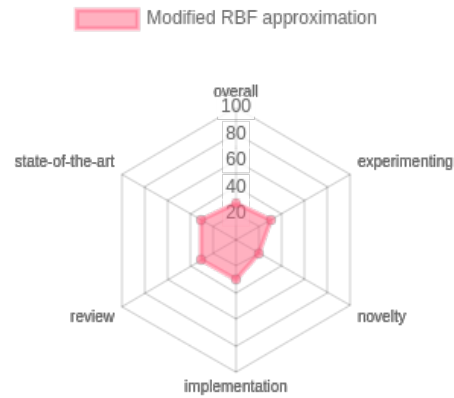
9.3 Modified Radial Basis Functions Approximation Respecting Data Local Features

The subsequent paper [94] builds on the foundations established in the initial paper [3]. This specific contribution exploits diverse features of the input function as optimal locations for RBF centres. These features contain edges, stationary curvature points, pseudorandom positions, and centres at the function's borders. It is necessary to include the latter two features according to Section 9.1. The centres situated at edges (identified, e.g. through the Canny edge detector on the height map image of the function) and stationary curvature points yield an even more enhanced interpolation result than inflexion points and local extrema.

The exploration of curvature concepts was further developed and ultimately integrated into the author's most recent publication, as elucidated in Section 9.13.

Furthermore, the discussion in Section 9.1 emphasised the importance of including pseudorandom positions and centres at the border. This deliberate integration stems from the overarching belief that centres located at edges and stationary curvature points possess the potential to enhance interpolation results beyond what can be achieved with inflexion points and local extrema alone.

The research presented in [94] not only builds on the concepts introduced in [3] but also advances the understanding of optimal RBF centre placement by considering a broader spectrum of features in the input function. The continued evolution and practical applicability are underscored by the subsequent developments, as outlined in Section 9.13, showing its relevance in subsequent research.



Publication [94]:

VASTA, J.; SKALA, V.; SMOLIK, M.; CERVENKA, M. Modified Radial Basis Functions Approximation Respecting Data Local Features. *Informatics 2019, IEEE proceedings*. 2019, pp. 445–449. ISBN 978-1-7281-3178-8. Available from DOI: <https://doi.org/10.1109/Informatics47936.2019.9119330>. UT WoS: 000610452900015, EID: 2-s2.0-8508762067, OBD: 43928987

Modified Radial Basis Functions Approximation Respecting Data Local Features

Jakub Vasta

*Department of Computer Science and Engineering
Faculty of Applied Sciences, University of West Bohemia
Plzen, Czech Republic
vastja@students.zcu.cz*

Vaclav Skala

*Department of Computer Science and Engineering
Faculty of Applied Sciences, University of West Bohemia
Plzen, Czech Republic
skala@kiv.zcu.cz*

Michal Smolik

*Department of Computer Science and Engineering
Faculty of Applied Sciences, University of West Bohemia
Plzen, Czech Republic
smolik@kiv.zcu.cz*

Martin Cervenka

*Department of Computer Science and Engineering
Faculty of Applied Sciences, University of West Bohemia
Plzen, Czech Republic
cervemar@kiv.zcu.cz*

Abstract—This paper presents new approaches for Radial basis function (RBF) approximation of 2D height data. The proposed approaches respect local properties of the input data, i.e. stationary points, inflection points, the curvature and other important features of the data. Positions of radial basis functions for RBF approximation are selected according to these features, as the placement of radial basis functions has significant impacts on the final approximation error. The proposed approaches were tested on several data sets. The tests proved significantly better approximation results than the standard RBF approximation with the random distribution of placements of radial basis functions.

Index Terms—Radial basis function, approximation, inflection points, stationary points, Canny edge detector, curvature

I. INTRODUCTION

The approximation is commonly used and well known technique in many computer science disciplines. This technique can be divided into two groups. The first one is the approximation that use the mesh and its connectivity. Some well known approaches that use the triangulation are [1]–[4]. However, all those approaches need the mesh connectivity, i.e. triangulation, which can be time consuming and difficult to compute for higher dimensions. On the opposite site, the second group are approximation techniques that does not require any mesh, i.e. they are called meshless methods. This paper focuses on this kind of approximation.

In the book [5] is provided an introduction for each of the most important and classic meshless methods along with the complete mathematical formulations. In total, it presents 19 meshless methods in detail with full mathematical formulations showing numerical properties such as convergence, consistency and stability. One example of approximation technique is Kriging [6]. It depends on expressing spatial variation

The authors would like to thank their colleagues at the University of West Bohemia, Plzen, for their discussions and suggestions. The research was supported by the projects: Czech Science Foundation (GACR) No. GA17-05534S and partially by SGS 2019-016.

of the property in terms of the variogram, and it minimizes the prediction errors which are themselves estimated. Extension and variations of this method are available in [7]–[9]. Another approximation technique is weighted least square method [10], [11] it is simple because it is based on the well-known standard least squares theory. It is attractive because it allows one to directly use the existing body of knowledge of the least squares theory and it is flexible because it can be used to a broad field of applications in the error-invariable models. Very similar approach is the LOWESS method [12] which is used for meshless smoothing and approximation of noisy data.

The Radial basis function (RBF) methods have been widely used for approximation of scattered data, recently. A brief introduction to this method is in [13]. Comparison of different radial basis functions is in [14], [15]. The paper [16] presents an approach for large scattered data interpolation. It uses the space subdivision to reduce the computation time and more importantly to reduce the needed memory for RBF approximation. A modification of this algorithm for 3D vector field data approximation is presented in [17]. Very important for the final RBF approximation quality is the distribution of radial basis functions. This problem is described and suggested solution in [18], [19]. Many papers also propose a solution to the selection of the best shape parameters of radial basis functions [20]–[24].

II. RADIAL BASIS FUNCTIONS

The Radial basis functions (RBF) are commonly used for n-dimensional scattered data approximation and interpolation. This approach is used in many areas, e.g. image reconstruction [25], [26], neural networks [27] and surface reconstruction [28], [29]. The task can be stated as follow. Find analytic function for given pairs of values (x_i, h_i) , where x_i is a point position in n-dimensional space and h_i is value in this point. For such data it is not possible to use standard approximation and interpolation techniques because lack of

knowledge about data connectivity and ordering. Therefore, the RBF approximation has the following attributes:

- Designed for scattered data approximation/interpolation
- Independent of the data dimension
- Not separable, i.e. it is not valid to approximate/interpolate data "dimension by dimension"
- Invariant with respect to euclidean transformations

Hardy [30] proposed RBF interpolation based on interpolation equation:

$$f(\vec{x}) = \sum_{i=1}^M \lambda_i \theta(\|\vec{x} - \vec{x}_i\|), \quad (1)$$

where \vec{x}_i is data point and λ_i is point weight. $\theta(r_{ij})$ is radial basis function, where $r_{ij} = \|\vec{x} - \vec{x}_i\|$. Radial basis function can differ, but they can be divided into two main groups by their range of influence, i.e. global and local functions.

RBF approximation/interpolation leads to the linear equation system $A\vec{x} = \vec{b}$, where approximation differ from interpolation only with form of matrix A . Solvability and stability problems were solved for example in [31] [32]. Wright [32] extend original RBF interpolation with polynomial and added more conditions.

A. Radial Basis Functions approximation

RBF approximation is based on point distance in n-dimensional space and is derived from the same equation (2) as interpolation is.

$$f(\vec{x}) = \sum_{i=1}^M \lambda_i \theta(\|\vec{x} - \epsilon_i\|), \quad (2)$$

where M is number of radial basis functions, λ_i is weight of radial basis function, θ is radial basis function and ϵ_i is placement of radial basis function.

Given set of value pairs $\{\vec{x}_i, h_i\}_1^N$, where \vec{x}_i is point position in n-dimensional space, h_i is value in this point, N is number of given points. $N \ll M$ therefore we obtain over-determined system of linear equations.

$$h_i = f(\vec{x}_i) = \sum_{i=1}^M \lambda_i \theta(\|\vec{x}_i - \epsilon_j\|) \quad i = \{1, \dots, N\}, M \ll N \quad (3)$$

It can be rewritten in matrix form

$$A\vec{\lambda} = \vec{h} \quad (4)$$

This over-determined system of linear equations can be solved by LSE or QR decomposition.

III. PROPOSED APPROACH

Radial basis function placement is important factor for approximation error. In this contribution, property of these good points are proposed with the way to find them. First group are extreme points i.e. local/global minimum or maximum. Next group are points of inflection. These points represents changes in data. Another proposed group of points are stationary points of curvature. These points represents extreme curvature values and in some case they are similar to points of inflection. Last group are edge points as known from image processing, because we can look at data as on image depending on data structure or sampling. Search for important points is amended with Halton sequence [13] sampling with special stress on border and corner sampling for covering whole data set. The last step is reduction of points number with nearest neighbour condition.

The Halton sequence is computed using the following formula:

$$Halton(p)_k = \sum_{i=0}^{\lfloor \log_p k \rfloor} \frac{1}{p^{i+1}} \left(\left\lfloor \frac{k}{p^i} \right\rfloor \bmod p \right), \quad (5)$$

where p is a prime number, k is the order of the element of the Halton sequence, i.e. $k \in \{1, \dots, n\}$. For generation of random points with Halton distribution in higher dimension, different values of p are used for each dimension.

A. RBF Approximation with Stationary Points

It is known that stationary points are such points that hold equation

$$\frac{\delta f}{\delta x} = 0 \wedge \frac{\delta f}{\delta y} = 0 \quad (6)$$

This condition is not enough to determine whether the point is global/local extreme or just a saddle point. It is necessary to examine Hessian to determine point property. On the other hand it can be seen that saddle points are as important as points of extreme.

Evaluation of partial derivatives and comparing with zero is not optimal way to find stationary points. Better way is to compare given point with its surrounding i.e. masks for minimum, maximum and saddle point can be created.

B. RBF Approximation with Inflection Points

Points of inflection are such points where surface change from convex to concave or the other way round. For points of inflection in continuous space hold that Gauss curvature is equal to zero

$$\kappa_{gauss} = \frac{\frac{\delta^2 f}{\delta x^2} \frac{\delta^2 f}{\delta y^2} - \left(\frac{\delta^2 f}{\delta x \delta y} \right)^2}{\left(\left(\frac{\delta f}{\delta x} \right)^2 + \left(\frac{\delta f}{\delta y} \right)^2 + 1 \right)^2} \quad (7)$$

It can be seen, from equations above, that Gauss curvature is zero only when numerator is equal zero, i.e. Hessian matrix determinant is equal to zero.

It is not necessary to compute exact curvature value to find point of inflection. We need to find just points where curvature sign change from negative to positive or vice versa. It can be seen that sign of Gauss curvature only depends on numerator sign because denominator is always positive.

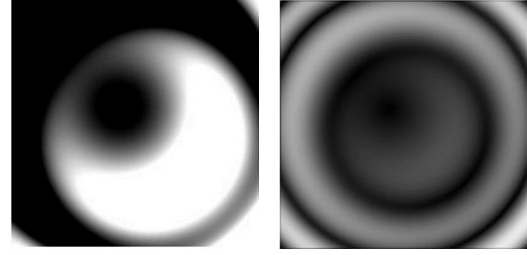
C. RBF Approximation with Stationary Points of Curvature

From (7) we can compute curvature of given surface in every data point. Then it is possible to find stationary points in curvature similar to process described in the section III-A

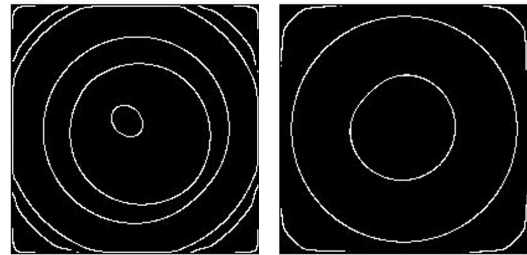
D. RBF Approximation with Edge Detection

In the simplified scenario, where the points are sampled in the grid pattern, we can look at the data as image i.e. value h on position $[x, y]$ can be considered to be brightness intensity I in pixel (i, j) . In case of scatter data there is need to use some special data structure to obtain points adjacency information e.g. kd-tree or adjust used algorithms.

We proposed to detect edges in image i.e. transitions between low and high values. Another suggested approach is to compute data gradient magnitude in each data point and then run edge detector over such field of gradient magnitudes. This approach will find transitions between low and high gradient magnitude areas. To detect edges we can use existing detectors from image processing e.g. Canny, Sobel, Prewitt etc.



(a) Height map converted to image (b) Gradient magnitudes converted to image



(c) Edge detector over height map (d) Edge detector over gradient magnitudes map

Fig. 2: The edge detection from height map (a), (c). The edge detection from gradient magnitudes map (b), (d).

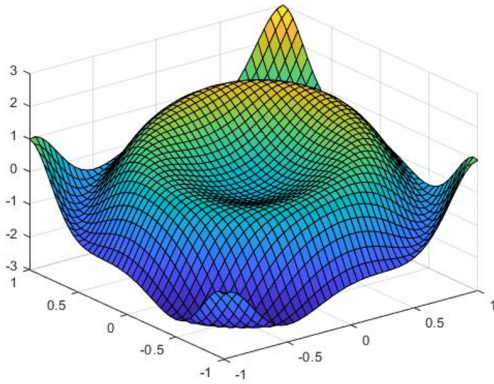


Fig. 1: Test function

IV. EXPERIMENTAL RESULTS

Proposed methods were tested on several test functions which were designed to represent special data set behaviour. Sampling step is 0.01 and functions are normalized to interval $(x, y, z) \in (-1, 1) \times (-1, 1) \times (0, 1)$ for comparison purposes. For all functions Gauss radial basis function $\phi(r) = e^{-\epsilon r^2}$ was used.

$$\begin{aligned}
 f_1(x, y) &= \frac{2}{11} \left(\sin(4x^2 + 4y^2) - x + y - \frac{5}{2} \right) \\
 f_2(x, y) &= \frac{3}{4} e^{-\frac{1}{4}((9x-2)^2 + (9y-2)^2)} \\
 &\quad + \frac{3}{4} e^{-\frac{1}{49}((9x+1)^2 - \frac{1}{16}(9y+1)^2)} \\
 &\quad + \frac{1}{2} e^{-\frac{1}{4}((9x-7)^2 + (9y-3)^2)} - \frac{1}{5} e^{-(9x-4)^2 - (9y-7)^2} \\
 f_3(x, y) &= \frac{1}{9} \tanh(9y - 9x) + 1
 \end{aligned}
 \tag{8}$$

For comparison was used square mean error per point as can be seen in Fig. 6, Fig. 7 and Fig. 8. In the 1st test function (8) random distribution of placement with Halton sequence provide good results in comparison with other methods, see Fig. 3. This is caused by function shape which fill whole space.

In the 2nd test function (8) can be seen improvement when proposed methods are used because of its special behaviour only in some areas of its domain, see Fig. 4.

The last test function (8) is design to test RBF approximation in general and even with our improvements lot of methods fails, see Fig. 5. What is even more it was found that with proper placement it has no effect on precision to add more points from Halton sequence.

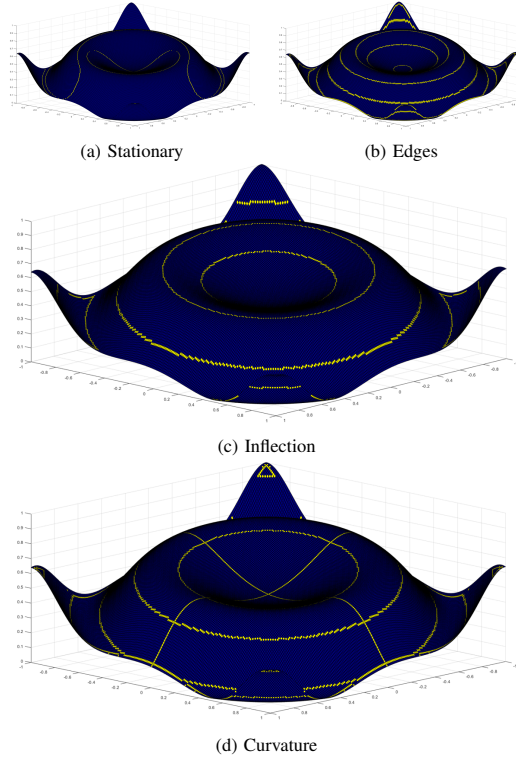


Fig. 3: Methods applied on 1st test function

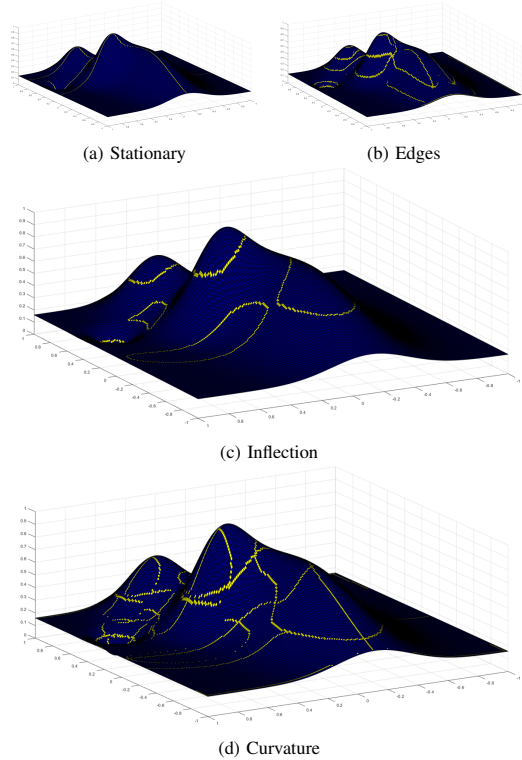


Fig. 4: Methods applied on 2nd test function

V. CONCLUSION

The proposed methods were tested on several standard testing functions, however, only some representative functions are mentioned in this contribution. The above presented methods proved very good results in precision of approximation, even though some special types of functions, e.g. fast changes, are problematic for all approaches. The experiments also proved validity of the proposed methods for the Radial basis function approximation of scattered data, with regard to a low approximation error with high points reduction leading to a high compression ratio.

ACKNOWLEDGMENT

The authors would like to thank their colleagues at the University of West Bohemia, Plzen, especially to colleague Zuzana Majdisova, for their discussions and suggestions. This research was supported by the projects: Czech Science Foundation (GACR) No. GA17-05534S and partially by SGS 2019-016.

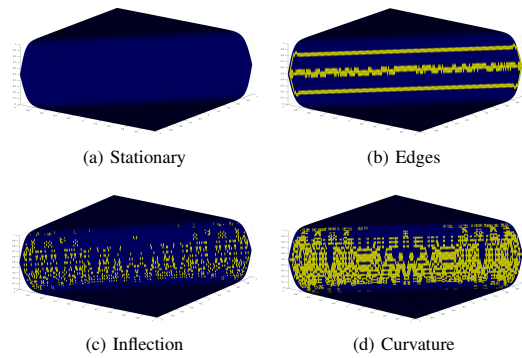
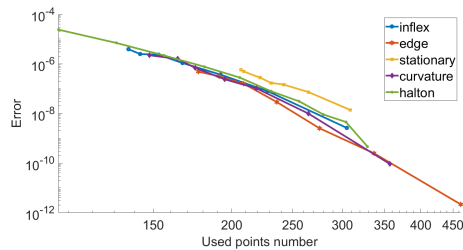
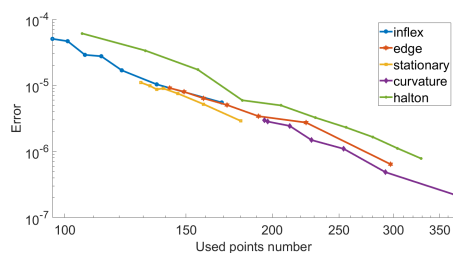
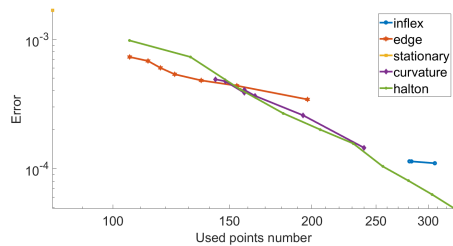


Fig. 5: Methods applied on 3rd test function

Fig. 6: Mean square error on 1st test function (8)Fig. 7: Mean square error on 2nd test function (8)Fig. 8: Mean square error on 3rd test function (8)

REFERENCES

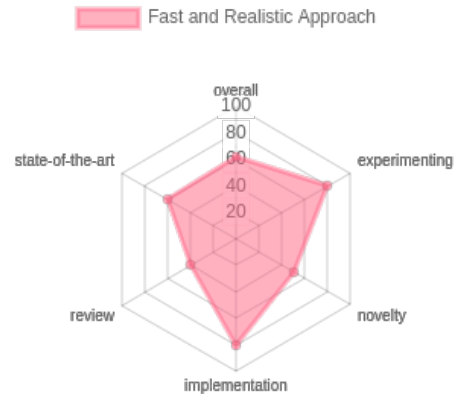
- [1] M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr, "Discrete differential-geometry operators for triangulated 2-manifolds," in *Visualization and mathematics III*. Springer, 2003, pp. 35–57.
- [2] M. Garland and P. S. Heckbert, "Fast polygonal approximation of terrains and height fields," 1995.
- [3] D. Cohen-Steiner, P. Alliez, and M. Desbrun, "Variational shape approximation," in *ACM Transactions on Graphics (ToG)*, vol. 23, no. 3. ACM, 2004, pp. 905–914.
- [4] C. L. Bajaj, F. Bernardini, and G. Xu, "Automatic reconstruction of surfaces and scalar fields from 3d scans," in *Computer graphics and interactive techniques*. ACM, 1995, pp. 109–118.
- [5] H. Li and S. S. Mulay, *Meshless methods and their numerical properties*. CRC press, 2013.
- [6] M. A. Oliver and R. Webster, "Kriging: a method of interpolation for geographical information systems," *International Journal of Geographical Information System*, vol. 4, no. 3, pp. 313–332, 1990.
- [7] I. Kaymaz, "Application of kriging method to structural reliability problems," *Structural Safety*, vol. 27, no. 2, pp. 133–151, 2005.
- [8] S. Sakata, F. Ashida, and M. Zako, "An efficient algorithm for kriging approximation and optimization with large-scale sampling data," *Computer methods in applied mechanics and engineering*, vol. 193, no. 3-5, pp. 385–404, 2004.
- [9] V. R. Joseph, Y. Hung, and A. Sudjianto, "Blind kriging: A new method for developing metamodels," *Journal of mechanical design*, vol. 130, no. 3, p. 031102, 2008.
- [10] A. Amiri-Simkooei and S. Jazaeri, "Weighted total least squares formulated by standard least squares theory," *Journal of geodetic science*, vol. 2, no. 2, pp. 113–124, 2012.
- [11] T. Asparouhov and B. Muthén, "Weighted least squares estimation with missing data," *Mplus Technical Appendix*, vol. 2010, pp. 1–10, 2010.
- [12] M. Smolik, V. Skala, and O. Nedved, "A comparative study of lowess and rbf approximations for visualization," in *ICCSA*. Springer, 2016, pp. 405–419.
- [13] G. E. Fasshauer, *Meshfree approximation methods with MATLAB*. World Scientific, 2007, vol. 6.
- [14] Z. Majdisova and V. Skala, "Radial basis function approximations: comparison and applications," *Applied Mathematical Modelling*, vol. 51, pp. 728–743, 2017.
- [15] —, "Big geo data surface approximation using radial basis functions: A comparative study," *Computers & Geosciences*, vol. 109, pp. 51–58, 2017.
- [16] M. Smolik and V. Skala, "Large scattered data interpolation with radial basis functions and space subdivision," *Integrated Computer-Aided Engineering*, vol. 25, no. 1, pp. 49–62, 2018.
- [17] —, "Efficient simple large scattered 3d vector fields radial basis functions approximation using space subdivision," in *ICCSA*. Springer, 2019, pp. 337–350.
- [18] M. Cervenka, M. Smolik, and V. Skala, "A new strategy for scattered data approximation using radial basis functions respecting points of inflection," in *ICCSA*. Springer, 2019, pp. 322–336.
- [19] V. Skala and S. Karim, "Finding points of importance for radial basis function approximation of large scattered data," in *Symposium Kebangsaan Sains Metemakik ke 27*, 2019.
- [20] J. Wang and G. Liu, "On the optimal shape parameters of radial basis functions used for 2-d meshless methods," *Computer methods in applied mechanics and engineering*, vol. 191, no. 23-24, pp. 2611–2630, 2002.
- [21] J. González, I. Rojas, J. Ortega, H. Pomares, F. J. Fernandez, and A. F. Díaz, "Multiobjective evolutionary optimization of the size, shape, and position parameters of radial basis function networks for function approximation," *IEEE Transactions on Neural Networks*, vol. 14, no. 6, pp. 1478–1495, 2003.
- [22] S. A. Sarra and D. Sturgill, "A random variable shape parameter strategy for radial basis function approximation methods," *Engineering Analysis with Boundary Elements*, vol. 33, no. 11, pp. 1239–1245, 2009.
- [23] Z. Majdisova, V. Skala, and M. Smolik, "Near optimal placement of reference points and choice of an appropriate variable shape parameter for RBF approximation," *Integrated Computer-Aided Engineering*, p. 16, 2019.
- [24] V. Skala, S. Karim, and M. Zabran, "Radial basis function approximation optimal shape parameters estimation: Preliminary experimental results," in *Symposium Kebangsaan Sains Metemakik ke 27*, 2019.
- [25] K. Uhlir and V. Skala, "Reconstruction of damaged images using radial basis functions," in *European Signal Processing Conference*. IEEE, 2005, pp. 1–4.
- [26] J. Zapletal, P. Vaněček, and V. Skala, "Rbf-based image restoration utilising auxiliary points," in *Computer Graphics International Conference*. ACM, 2009, pp. 39–43.
- [27] S. Haykin, *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1994.
- [28] R. Pan and V. Skala, "Continuous global optimization in surface reconstruction from an oriented point cloud," *Computer-Aided Design*, vol. 43, no. 8, pp. 896–901, 2011.
- [29] —, "A two-level approach to implicit surface modeling with compactly supported radial basis functions," *Engineering with Computers*, vol. 27, no. 3, pp. 299–307, 2011.
- [30] R. L. Hardy, "Multiquadric equations of topography and other irregular surfaces," *Journal of geophysical research*, vol. 76, no. 8, pp. 1905–1915, 1971.
- [31] C. A. Micchelli, "Interpolation of scattered data: distance matrices and conditionally positive definite functions," in *Approximation theory and spline functions*. Springer, 1984, pp. 143–145.
- [32] G. B. Wright, "Radial basis function interpolation: numerical and analytical developments," 2003.

9.3. Modified Radial Basis Functions Approximation Respecting Data Local Features

J. Vasta et al. • Modified Radial Basis Functions Approximation Respecting Data Local Features

9.4 Fast and Realistic Approach to Virtual Muscle Deformation

The fourth article [65] provided an overview of the contemporary landscape at its creation. It delved into a muscle modelling approach rooted in position-based dynamics, serving as a fundamental algorithm. Although this approach was operated directly on a triangular mesh representing the muscle surface, it is essential to note that it is still undergoing development, as indicated in sections 9.8 and 9.10. However, the author has since redirected their focus towards muscle modelling methods using Radial Basis Function (RBF) approximation techniques, as detailed throughout the text.



Within the paper, Section 6, titled "Discussion," not only expounded upon the state-of-the-art but also scrutinised various challenges and impediments associated with employing techniques such as Position-Based Dynamics (PBD), Finite Element Method (FEM), Mass-spring systems (MSS), via-points approaches, wrapping obstacles, and more.

The identified issues encompassed the lack of smoothness in the model, the possible difficulties in collision detection and response (exemplified by the hip joint issue discussed in the article, subsequently addressed in further research, as evidenced in Section 9.10), and the excessive amount of data utilised for the model compared to its actual requirements. Faced with these challenges, the notion of adopting an entirely different geometrical description for a muscle emerged at the time of writing. However, this concept has not yet been formally presented in the paper.

Publication [65]:

CERVENKA, M.; KOHOUT, J. Fast and Realistic Approach to Virtual Muscle Deformation. in *Proceedings of the 14th International Joint Conference on Biomedical Engineering Systems and Technologies - Volume 5: HEALTHINF*. 2020, pp. 217–227. ISBN 978-989-758-398-8. Available from DOI: <https://doi.org/10.5220/0009129302170227>. UT WoS: 000571479400020, EID: 2-s2.0-85083710925, OBD: 43929104

Fast and Realistic Approach to Virtual Muscle Deformation*

Martin Cervenka¹ ^a and Josef Kohout² ^b

¹University of West Bohemia, Faculty of Applied Sciences,
Department of Computer Science and Engineering, Czech Republic
²University of West Bohemia, Faculty of Applied Sciences,
NTIS - New Technologies for the Information Society, Czech Republic

Keywords: Muscle Modelling, Muscle Deformation, Muscle Fibres, Position based Dynamics.

Abstract: This paper describes a real-time simulation of muscle movement that is based on inverse kinematics where bone placement is known apriori and muscle shape is calculated by a modified position-based dynamics (PBD) method. The method is comparable and competitive with the others, moreover, it is enhanced with some novel features like approach for respecting muscle anisotropy, really fast simplistic collision detection, etc. This real-time simulation presents visual plausibility of resulting muscle deformation in most cases.

1 INTRODUCTION

One of the diseases with high prevalence is osteoporosis (Wade et al., 2014). This disease causes weakening of bone tissue, which results in weak bones prone to break. This is a valid reason to be concerned about forces employed to softened bones. Bergmann et. al. (Bergmann et al., 2001) show that twice as much force is exerted during walking than during standing. In severe cases, there is a chance that a weak bone is not able to absorb surrounding forces and fractures.

Osteoarthritis is another musculoskeletal disease to consider. In advanced stages, a form of treatment is needed, which is usually done by replacing the bone joint with an artificial one. Knowledge of forces impacting the bone joint is essential for surgeons to choose suitable artificial joint (Oatis, 2013). An inaccurate choice of artificial joint may be painful for the patient or even cause further harm.


A well-built model can be useful to predict the discussed forces. Such a model should be physically correct and should be also patient-specific because body constitution varies in patients.


Modern technologies allow us to use computers to simulate the musculoskeletal system (or its model, respectively) and get some form of approximate values

from this process. There are some models (e.g. (Delp et al., 1990), (Arnold et al., 2009)) measured on single patient; unfortunately, we need patient-specific models. Even though it may seem like a good idea to use model measured on single patient for every other patient (for the sake of simplicity etc.), it neglects significant features of the individual patient. There are statistical models as well (e.g., PCA based statistical model using data from 26 patients by Zhang et al. (Zhang et al., 2016)), which can achieve better results in general, despite these statistical models cannot be as good as patient-specific models. However, getting complete patient-specific data is nearly impossible due to inaccurate measuring, complexity of data processing, time consumption etc.

To give a real-life example where the simulation is needed, calculation of the stresses to which bones are subjected when performing a specific action can provide fracture prediction for people suffering from osteoporosis. This information enables better prognosis and more precise treatment, and a dynamical analysis and visualisation of muscle activity may help to identify issues in the action of a professional athlete that can lead to more effective training procedures and the identification of ways in which performance can be improved. For such applications, a patient-specific or subject-specific musculoskeletal model is essential for the simulation and its visualisation.

We present a novel, real-time approach for muscle modelling, derived from position-based dynamics (well known in modern computer graphics field).

^a  <https://orcid.org/0000-0001-9625-1872>

^b  <https://orcid.org/0000-0002-3231-2573>

*This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic, project SGS-2019-016 and project PUNTIS (LO1506).

PBD was originally proposed by Müller (Müller et al., 2007) and is currently still in development (e.g. cloth and fluid simulation (Shao et al., 2017)). Main contributions are:

- Using computer graphics related PBD (position based dynamic) method for muscle deformation,
- Extending PBD by concerning anisotropy of muscles,
- Fast minimalistic implementation of muscle modelling approach.

2 APPROACH

2.1 Static Model

Construction of a patient-specific model requires having data on the anatomy of the individual patient. Muscle parameters such as geometrical shape, attachment sites, fibre orientation, etc. are essential to get accurate results of simulations. The geometrical shape of a muscle can be extracted from MRI images and this can be done automatically, though with some issues. However, most parameters regarding the muscle architecture cannot be extracted from these images and generally, they are very difficult, if not impossible, to get *in vivo*. Nevertheless, this information can be obtained from cadaver measurements (not patient-specific). It is possible then to combine this general information with specific patient data.

In this paper, a subset of a comprehensive female cadaver anatomical dataset (81 y/o, 167 cm, 63kg) is used. Specifically, pelvic and femur bones together with several muscles from the pelvic region have been selected.

The complete data are publicly available in LHDL dataset (Viceconti et al., 2008) and has been selected because it includes high-quality surface meshes of bones and muscles. Furthermore, the dataset was improved by removing non-manifold edges, duplicated vertices and degenerate triangles followed by surface smoothing in both muscle and bone models using MeshLab (Cignoni et al., 2008). The dataset also contains muscle attachment areas and geometrical paths of superficial fibres obtained from dissection (Van Sint Jan, 2005).

2.2 Dynamic Model

Having a static model, a dynamic one can be obtained quite simply providing that the motion data for that model are available. If bones are assumed

as completely rigid, motion can be simulated via inverse kinematics. Inverse kinematics means that the location and movement of all bones are known, and muscle actual shape has to be determined according to these situations. We note this is exactly the opposite to what can be seen in real situation, where muscles control bone movement.

In this paper, position-based dynamics (PBD), which was introduced in (Müller et al., 2007) as a fast, stable, and controllable solution to various problems in computer graphics, e.g., simulations of cloth or fluids, is used to reposition the vertices of the surface mesh of muscle with the modelling constraints to preserve the shape and volume of the muscle and prevent mutual penetration of the muscle with the bones. Details will be described in Section 3.

The muscle is then decomposed into a set of fibres using either Kukacka (Kohout and Kukacka, 2014) or CHMD (Kohout and Cholt, 2017) method, depending on which one is more appropriate in the concrete case. Note that this set represents the dynamics of muscle architecture. Details regarding muscle decomposition will be described in Section 5.5.

3 PBD

Müller (Müller et al., 2007) in his paper described PBD with four requirements:

1. Preserve local distances,
2. Maintain local shape,
3. Keep the original volume,
4. Not collide with other models.

In the context of muscle modelling, another requirement, which considers muscle anisotropy, must be introduced to model muscle motion accurately. This will be described later.

PBD method is based on solving equation 1 for discretized muscle model over and over again. It describes a movement of a single point (vertex) during simulation ($\Delta \mathbf{p}_i$ denote the difference in position of i th point of the model), maintaining various features. Mentioned features can be for example volume or shape preservation, but it can be anything giving meaning. Some of the preserved features are mentioned and further described later in the text.

$$\Delta \mathbf{p}_i = - \frac{\nabla_{\mathbf{p}_i} C(\mathbf{p}_1, \dots, \mathbf{p}_n)}{\sum_j |\nabla_{\mathbf{p}_j} C(\mathbf{p}_1, \dots, \mathbf{p}_n)|^2} \cdot C(\mathbf{p}_1, \dots, \mathbf{p}_n) \quad (1)$$

In above equation, C is a constraint function (more details below), n is the number of points in the muscle model and j is the number of constraint functions.

Mathematically speaking, constraint function and its gradients must be known to solve the deformation problem.

3.1 Distance Constraint

A rather basic constraint, which we can imagine, is restricting each model point to change the distance from the others in its neighbourhood. This constraint can be formulated as is in (2), where d is the original distance between points \mathbf{p}_1 and \mathbf{p}_2 .

$$C(\mathbf{p}_1, \mathbf{p}_2) = |\mathbf{p}_1 - \mathbf{p}_2|_2 - d \quad (2)$$

3.2 Volume Constraint

Next, we can restrict the muscle model to change its volume during the simulation process. To calculate this constraint, we need to know how to measure the volume of the model. Assuming triangular mesh, the volume can be computed tetrahedron by tetrahedron, which in summation eventually leads to constraint function:

$$C(\mathbf{p}_1, \dots, \mathbf{p}_n) = \sum_{i=1}^m \left(\mathbf{p}_{i_1} \cdot (\mathbf{p}_{i_2} \times \mathbf{p}_{i_3}) \right) - V_0 \quad (3)$$

In this function, m describes number of triangles forming the muscle model, V_0 is muscle original volume and p_{i_j} denotes j th point of the i th triangle.

3.3 Local Shape Constraint

Above described constraints are not enough to prevent the surface from becoming noisy, full of unrealistic spikes. One possible solution to this problem is to use the distance constraint not only to keep the distances between adjacent points but also between the pairs of points lying on the opposite sides of the muscle. This would, however, need to create a 3D mesh first, which would be quite complex to do. Another option is to ensure that the local shape is maintained. To achieve this, the angles between neighbouring triangles should stay the same during deformation. Calculating the angle of two triangles is integrated into equation (4), where two triangles are described by four points, sharing points with index 1 and 2.

$$\begin{aligned} C(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4) &= \arccos(\mathbf{n}_1 \cdot \mathbf{n}_2) - \varphi_0 \\ &= \arccos \left(\frac{(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_1)}{\|(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_1)\|_2} \cdot \right. \\ &\quad \left. \frac{(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_4 - \mathbf{p}_1)}{\|(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_4 - \mathbf{p}_1)\|_2} \right) - \varphi_0 \end{aligned} \quad (4)$$

3.4 Anisotropy

The PBD algorithm has been originally proposed in the computer graphics field to model isotropic materials (e.g., cloths). However, muscles are anisotropic (may behave differently in two distinct directions), so it is appropriate to take anisotropy into account. The main idea is that muscle surface is stiffer in the direction perpendicular to the muscle fibres and more flexible in the direction parallel to these fibres. Mathematically speaking, we multiply the distance constraint in (2) with the result of equation (5).

$$k_i = 1 - \mathbf{u}_i \cdot \mathbf{v}_i \quad (5)$$

The direction of i th edge is described by normalized vector \mathbf{u}_i , \mathbf{v}_i denotes tangential direction normal vector of nearest fibre on the surface. If both vectors are collinear, the result k_i will be zero, meaning no distance is preserved. If these two vectors are perpendicular, then k_i is equal to one and edge length will be well preserved. This behaviour is assured because value k_i multiplies the result of distance constraint function in (2).

4 COLLISION HANDLING

In the simulation, moving muscles and bones should not intersect each other. There are some methods with different advantages and disadvantages, some of them are described below.

4.1 Brute Force

The most simple way to detect a collision is to test each primitive of a surface mesh with all primitives of the other mesh for an intersection. Time complexity in big-O notation is $O(n^2)$, where n is the number of primitives (triangles in our case), which makes the brute force approach suitable only for small models, for which the overhead associated with more sophisticated approaches is not amortized. For bigger (real) models, we need a sort of space division algorithm.

4.2 Bounding Volume Hierarchies

One of the data structure suitable for space division is a bounding volume hierarchy. The basic idea behind the structure is that space is recursively subdivided employing some volumetric (in 3D, planar in 2D) geometrical primitives (box, sphere, etc.). In big-O notation, we achieve $O(n \log n)$ or even $O(\log^2 n)$, assuming the number of vertices of both tested meshes

depends linearly on each other. The main disadvantage is that the structure has to be built before the simulation or even updated every time the muscle shape changes.

4.2.1 octree

A special case of bounding volume hierarchies is an octree. The octree divides the three-dimensional box into eight (therefore *octree*) smaller boxes (box is divided in half in each axis). The division is done recursively and stops when the box contains a sufficiently small number of elements, i.e., vertices or triangles.

4.3 Voxelization

In this approach, the given three-dimensional box is divided into n^3 (n division in each axis) equally sized boxes. If n is selected carefully, every box contains a (sufficiently small) constant number of element, so testing can be theoretically done in $O(1)$ time using big-O notation, however, memory complexity will be $O(n^3)$ as far as every cube content has to be stored in memory.

Our goal is to do a fast deformation, so we decided memory is not an issue, therefore voxelization is used in the proposed approach.

4.4 Collision Response

In our case, two main scenarios have to be distinguished. In the first one, a muscle moves (e.g. because of surrounding forces) and hits a bone. When vertices of the muscle collide, they are pushed back in the direction they enter the bone so they will not penetrate the bone anymore.

In the second scenario, a bone moves into a muscle. In this case, the colliding muscle vertices did not "enter" the bone, so we do not know the entering direction. We propose a solution where the same transformation used on colliding bone is applied to colliding muscle vertices as well.

5 EXPERIMENTAL RESULTS

The real data described in Section 2.1 is used along with an artificial dataset (described below) to test the proposed approach. The results of the experiments are described in detail below.

5.1 Artificial Data

To test collision detection and behaviour in extreme cases, we prepared an artificial dataset, where a box of 5292 triangles on its surface is squished between two plates (12 triangles each). The initial setup is visualized in Fig.1.

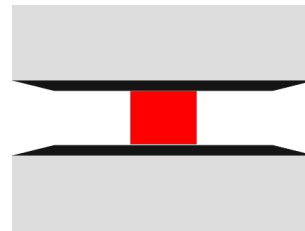


Figure 1: Input artificial data.

At first, in one hundred iterations, the top plate moves to decrease the space between both plates. The distance between plates in 100th iteration is 10% of the original distance in the first iteration. Inverse movement is then performed between 100th and 200th iteration, returning the plates to their initial position. Additional one hundred iterations are used for box stabilization.

As it can be seen in Fig.2, the box is squished quite a lot. Despite the fact, it returns to its original shape in 300th iteration (only with slight rotation caused by asymmetrical triangulation). Even in 200th iteration the results is acceptable, except one corner of the box.

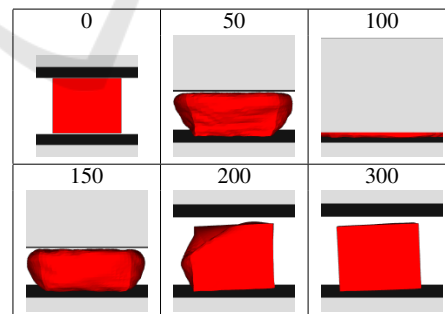


Figure 2: Results in different simulation frames (artificial data).

5.2 Iliacus

We also tested iliacus muscle, which connects the femur and pelvic bones from the front side. The bones and the muscle are visualized in Fig.3. The muscle

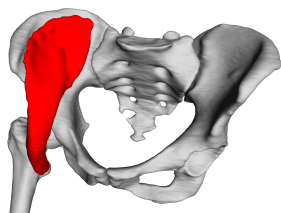


Figure 3: Input iliacus muscle and adjacent bones.

and bones are closed triangular meshes with 13858 and 254442 (42456 for femur) triangles, respectively.

A very similar simulation scenario like in the artificial data case has been applied to iliacus dataset. The first hundred iterations are used for hip flexion. The femur bone rotates one radian during this movement. The second hundred iteration is allocated for the inverse movement. The last hundred (200th-300th) iterations are present to stabilize the muscle.

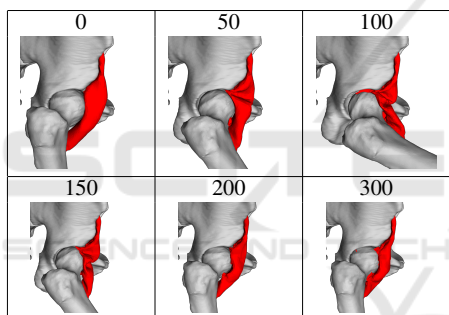


Figure 4: Results in different simulation frames (Iliacus muscle data).

The posterior part of the iliacus muscle is pushed into the joint during the flexion, as it can be seen in Fig.4. The explanation for this behaviour is as follows. This part of the mesh is unrealistically arched towards the joint and, therefore, distance and local shape constraints tend to move the points of this part closer to the joint. As the femur and pelvic bones do not touch, there is a narrow space into which this part of the mesh can squeeze, and from which it is difficult to get out. Even though the result is not visually plausible, the quantitative tests (described later) show that all key factors of the muscle are preserved as much as possible.

5.3 Gluteus Maximus

Gluteus maximus muscle is attached to the same bones as iliacus muscle but from the other side. Tri-



Figure 5: Input gluteus maximus muscle and adjacent bones.

angular mesh of the consists of 19752 triangles. Fig.5 shows how the model looks like.

The muscle undergoes the same movement scenario as iliacus mentioned above. Visualization in 6 important iterations is shown in Fig.6. As we can see, the result in iteration 300 is nearly the same as in the first iteration (the original muscle pose).

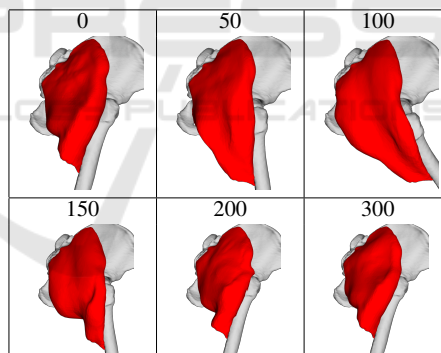


Figure 6: Results in different simulation frames (gluteus maximus muscle data).

5.4 Other Muscles

Within this testing procedure, we have tested gluteus medius and adductor brevis muscles. Both muscles deform quite realistically, as it can be see in Fig.7 and Fig.8, where the situation in the maximal hip flexion (same scenario like in gluteus maximus and iliacus case) is shown.

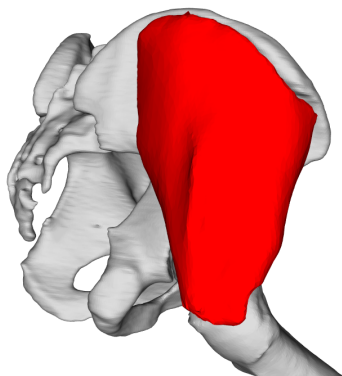


Figure 7: Gluteus medius in maximum flexion position.

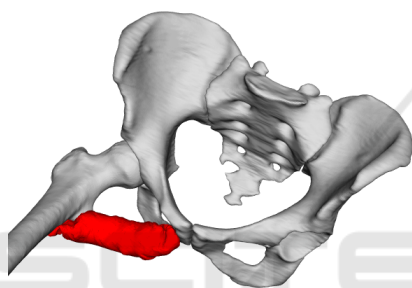


Figure 8: Adductor brevis in maximum flexion position.

5.5 Muscle Decomposition

The approach described so far works primarily with the surface model of a muscle. To calculate the forces (and other physical properties of the muscle), the muscle needs to be decomposed into individual fibres. This can be done using, for example, Kukacka or CHMD muscle decomposition methods, which were proposed in (Kohout and Kukacka, 2014) and (Kohout and Cholt, 2017), respectively. In the following subsections, we briefly describe these methods.

5.5.1 Kukacka Decomposition

The inputs of the Kukacka decomposition method (Kohout and Kukacka, 2014) are:

- Triangular (and manifold) surface model of the muscle to decompose,
- Fibre template, giving information about internal fibre arrangement,
- Attachment areas to adjacent bones (origin and insertion), defined by a set of points lying on the adjacent bone surface models

Decomposition is then done as follows. Attachment areas are projected from the bone surface onto the muscle surface to define the parts of the muscle that are subsequently removed. Isocontours are then computed on the modified muscle model, using a piece-wise linear scalar field. The scalar field has its maximum on the insertion boundary vertices, whereas it has its minimum on the origin boundary vertices. User can specify how many isocontours are generated in this process.

Similarly to (Delp, 2005), muscle fibre architecture (template) is represented by a unit 3D space with arbitrary (user-defined) number of fibres inside the space. The fibres are represented analytically by Bezier spline curves. From multiple templates, one is selected according to the muscle being modelled (depends on if it has parallel or pennate fibres, optionally on a pennate angle, etc.) and it is mapped one-to-one on isocontours calculated in the previous step, forming the fibres going through the muscle model. Finally, noise is eliminated using quadratic smoothing to make the result more realistic and visually plausible.

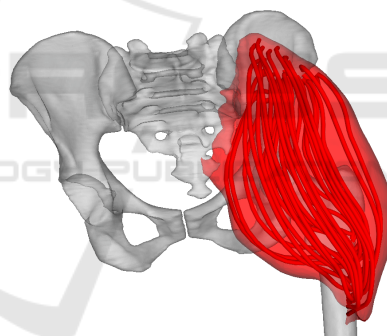


Figure 9: Gluteus maximus decomposed to individual fibres by Kukacka's (Kohout and Kukacka, 2014) algorithm.

5.5.2 CHMD Decomposition

A technique by Kohout & Cholt (Kohout and Cholt, 2017) performs a centripetal Catmull-Rom interpolation of the input fibres lying on the surface model of the muscle, or nearby, to get the fibres inside the muscle. Their approach can even work with multiple headed muscles, distributing the fibres automatically among the heads.

In comparison with Kukacka's proposed method, this method needs specification of fibres on the surface, which typically requires some manual effort because these are not available for the patient but must be adopted from a cadaveric dataset. On the

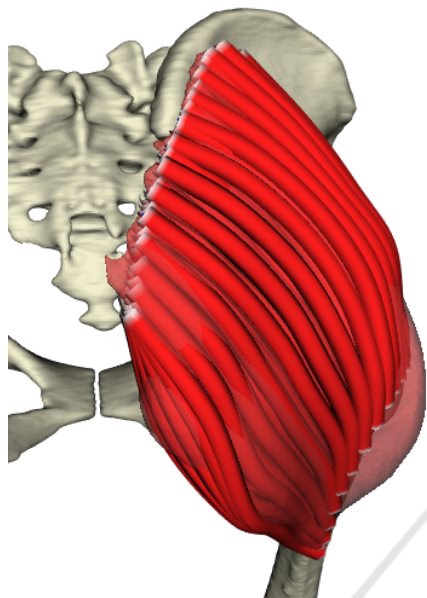


Figure 10: Gluteus maximus decomposed to fibres by Cholt's (Kohout and Cholt, 2017) algorithm.

other hand, it can work with multiple headed muscles, whereas Kukacka's approach can not.

5.6 Quantitative Tests

To make some exact result, we use quantitative tests. These tell us how well are constraints satisfied during simulation.

Volume preservation constraint is tested by determining ratio between both original and actual volumes. Fig.11 for artificial data, Fig.12 and Fig.13 for real data respectively, show us the volume preservation results.

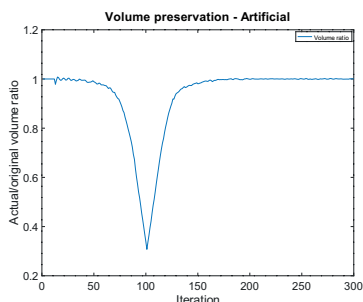


Figure 11: Volume preservation of artificial data.

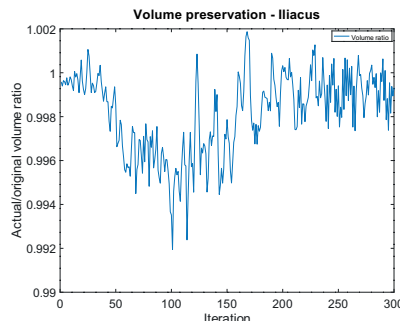


Figure 12: Volume preservation of iliacus muscle data.

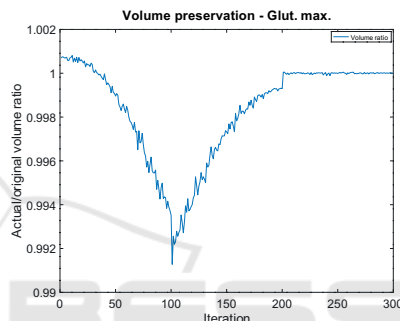


Figure 13: Volume preservation of gluteus maximus muscle data.

As we can see from the results, the volume is well preserved in both real data (the error is less than 1% in both cases), on artificial data, there is a nice curve describing squishing (reducing) box volume during the simulation and then restoring.

Next measurable property is average edge extension. At first, we cannot say much about artificial data (squished box) from the plot on Fig.14. As for the iliacus muscle, some edges remain longer than normal (see Fig.15) because they are stuck in the hip joint. In the case of gluteus maximus dataset (Fig.16), the first 100 iterations show edge extension during hip flexion (it is correct behaviour because muscle extends in this phase) and the second hundred iterations return the average length extension to almost 1 (i.e., the muscle returns to its original pose correctly). We can also see that the last 100 iterations are not crucial in this scenario.

We also tested how well the dihedral angles are preserved during the simulation. In this paper, the dihedral angle is the angle between two adjacent triangles in the muscle triangle mesh. According to plots in Fig.17, Fig.18 and Fig.19, we can conclude that there are some pairs of triangles which do not pre-

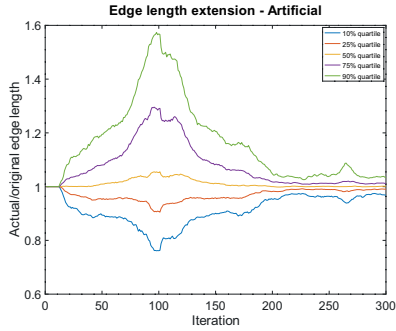


Figure 14: Average edge extension of artificial data.

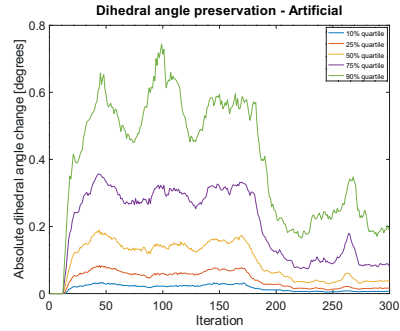


Figure 17: Average absolute dihedral angle change of artificial data.

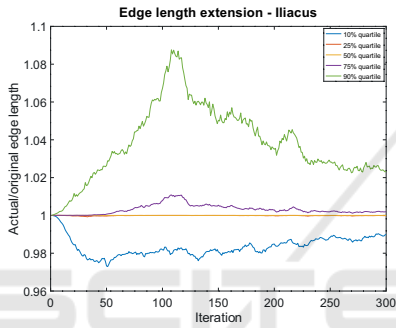


Figure 15: Average edge extension of iliacus muscle data.

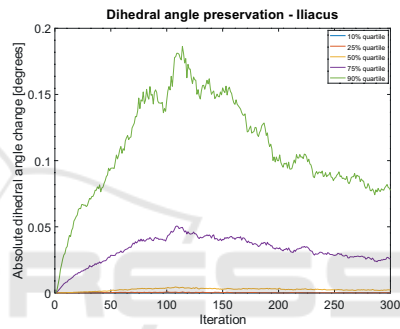


Figure 18: Average absolute dihedral angle change of iliacus muscle data.

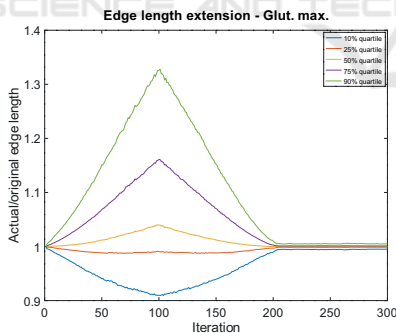


Figure 16: Average edge extension of gluteus maximus muscle data.

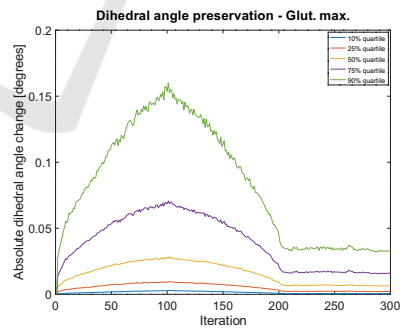


Figure 19: Average absolute dihedral angle change of gluteus maximus muscle data.

serve its original angle, but most of them do.

5.7 Fibre Length

Last but not least, the lengths of fibres were analyzed. In the iliacus muscle case, as we can see in Fig.20, many length curves exhibit two big bumps shortly after 100th iteration. This is caused by the fact that a

part of the muscle is stuck in the hip joint (as discussed previously). Nevertheless, when the bones return to their initial rest-pose (i.e., after 200 iterations), the vast majority of fibres restore their original lengths quite well. Gluteus maximus muscle behaves as expected – see Fig.21. During the flexion, all lengths increase; during the extension, they decrease.

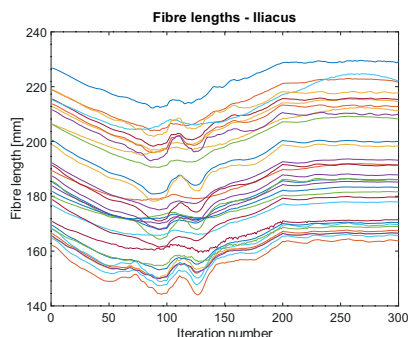


Figure 20: Total length of each individual fibre during simulation in iliacus muscle.

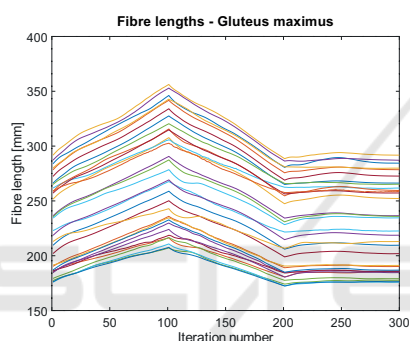


Figure 21: Total length of each individual fibre during simulation in gluteus maximus muscle.

5.8 Speed

The proposed method was designed to be not only precise, but mainly, fast. It was implemented in C++ using VTK toolkit. Its current version is publicly available at <https://github.com/cervenkam/muscle-deformation-PBD>.

All testing scenarios above were measured how fast each one is. FPS (Frames Per Second) is used as speed metric in this case.

All tests were performed on Intel® Core™ i7-4930K 3.40GHz CPU, Radeon HD 8740 GPU and WDC WD40EURX-64WRWY0 4TB HDD. Results are listed on Tab.1.

As it can be seen from the results, FPS strictly depends on number of triangles (Spearman's $\rho = -1$). The more triangles is used, the slower the method is.

Even though the program is mostly unoptimized and runs sequentially at the moment, the FPS is sufficient for considered purposes in general.

Table 1: FPS of each simulation.

Deforming object	Triangle count	FPS
Gluteus maximus	19752	33.85
Abductor brevis	17124	35.89
Iliacus	13858	47.21
Gluteus medius	10622	57.12
Artificial box	5292	153.61

6 DISCUSSION

The most simple approach to muscle deformation problem is probably to use line segments to approximate both muscles and tendons. An example is shown in Fig.22. The coordinates of each end-point of the corresponding line segments are updated when bone moves, causing shortening or extending of the line segments. Various values (e.g., moment arms) can be consequently calculated from the length of each line. These models are popular in practice (they are used, e.g., in AnyBody¹ or OpenSim²) because of their simplicity and speed. However, the accuracy of calculations based on these models is, especially, for complex muscles, e.g., gluteus medius, low (Delp, 2005). A model can be more precise if we assume more lines (Valente et al., 2012) per a muscle or if we use more complex lines wrapping around some kind of parametric surfaces (e.g. spheres, cylinders (Audenaert and Audenaert, 2008), etc.), nonetheless, all of these are difficult to set up. This may be the reason why most studies use gait2392 model shipped with OpenSim software even though there are no more than three lines per muscle and these lines penetrate the bones in some poses. Using our approach, described in this paper, the user can easily generate hundreds of lines (i.e., fibres) automatically and impenetrability of muscles and bones is guaranteed.



Figure 22: Muscle approximation using line segments – yellow lines (taken from (Kohout et al., 2013)).

Position-based dynamics (PBD), which is the core part of our approach, was firstly introduced in (Müller et al., 2007) as a computer graphics algorithm. Since

¹<https://www.anybodytex.com/software/>

²<https://opensim.stanford.edu/>

then, it has been further developed (e.g., (Macklin et al., 2019) proposed recently some speed and accuracy improvements) and has found many (close to) real-time applications, not only in computer graphics but even in other domains. For example, Kotsalos et al. use PBD to model blood cells (Kotsalos et al., 2019). As far as we know, however, there is no PBD-based method for muscle modelling even though one could expect a good compromise between speed and accuracy from such a method.

A mass-spring system (MSS) is another approach to consider. Janak et al. use MSS to approximate muscle (Janak, 2012), showing promising, simple method with visually plausible results. However, there are some issues in the approach they proposed. First, to avoid penetration between muscles and bones, the authors choose a particle-based collision detection method requiring many particles to get reasonable results, which, however, causes high time and memory complexity. Secondly, and more importantly, the main issue is that muscle volume is not preserved during deformation. This could be probably solved using the approach described in (Hong et al., 2006), however, it would increase computational time dramatically. Finally, our experiments show that although this method retains the smooth shape of iliacus muscle during flexion, it twists the part of the muscle close to the insertion. This is because, unlike our method, the particles are in the entire volume of the muscle, which results in a model that is much more rigid, and as anisotropy is not exploited, rigid in all directions. Our method supports anisotropy, preserves the volume and runs in a fraction of time while requiring no extra parameter or input in comparison with this method.

On the contrary to line segment approximation, finite element method (FEM) is the most complex method. Well discretized muscle provides a physically very accurate result (see e.g., (Delp, 2005)). However, computational complexity is high, meaning the FEM-based methods are unsatisfactorily slow. Therefore, it is quite impractical for real-time application or even clinical assessments. Next issue is a difficult set up of FEM methods, making them unsuitable for personalised musculoskeletal method deformation. Despite these facts, these methods can be seen in the movie industry, see e.g. Ziva VFX³ plugin for Maya, and in muscle physiology research, see e.g. (Oberhofer et al., 2009) or (Kojic et al., 1998). In comparison with these methods, our method is quite simple to set up and runs fast providing the promising results in most cases.

³<https://zivadynamics.com/>

7 CONCLUSION & FUTURE WORK

The proposed muscle deformation technique is capable to do fast and relatively accurate simulation. Despite problems with muscle trapped in the hip joint, we believe that a better collision detection can fix the issue.

Moreover, the method is ready to be included in OpenSim (a state-of-the-art simulation software) as a plugin, allowing common users to use the method more intuitively. Its source code is available at <https://github.com/cervenkam/muscle-deformation-PBD>.

In this paper, we verified the method, but to prove correctness, the method needs to be validated in real life. There are some works (e.g. (Modenese et al., 2018)) providing correct momentum values during muscle movement, which can be useful for validation.

ACKNOWLEDGMENT

Authors would like to thank their colleagues and students for valuable discussion, worthwhile suggestions and constructive comments. Authors would like to thank also anonymous reviewers for their hints and notes provided.

REFERENCES

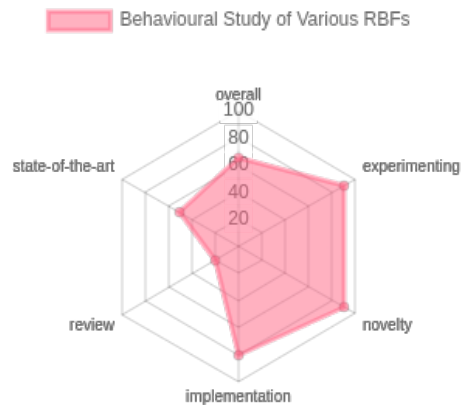
- Arnold, E., Ward, S., Lieber, R., and Delp, S. (2009). A model of the lower limb for analysis of human movement. *Annals of biomedical engineering*, 38:269–79.
- Audenaert, A. and Audenaert, E. (2008). Global optimization method for combined spherical-cylindrical wrapping in musculoskeletal upper limb modelling. *Computer methods and programs in biomedicine*, 92:8–19.
- Bergmann, G., Deuretzbacher, G., Heller, M., Graichen, F., Rohlmann, A., Strauss, J., and Duda, G. (2001). Hip contact forces and gait patterns from routine activities. *Journal of Biomechanics*, 34(7):859 – 871.
- Cignoni, P., Callieri, M., Corsini, M., Dellepiane, M., Ganovelli, F., and Ranzuglia, G. (2008). Meshlab: an open-source mesh processing tool. *Computing*, 1:129–136.
- Delp, S. (2005). Three-dimensional representation of complex muscle architectures and geometries 1. *Annals of Biomedical Engineering - ANN BIOMED ENG*, 33:1134–1134.
- Delp, S. L., Loan, J. P., Hoy, M. G., Zajac, F. E., Topp, E. L., and Rosen, J. M. (1990). An interactive graphics-based model of the lower extremity to study orthopaedic surgical procedures. *IEEE Transactions on Biomedical Engineering*, 37(8):757–767.

- Hong, M., Jung, S., Choi, M.-H., and Welch, S. (2006). Fast volume preservation for a mass-spring system. *IEEE computer graphics and applications*, 26:83–91.
- Janak, T. (2012). Fast soft-body models for musculoskeletal modelling. Technical Report DCSE/TR-2012-5, University of West Bohemia, Faculty of Applied Sciences.
- Kohout, J. and Cholt, D. (2017). Automatic reconstruction of the muscle architecture from the superficial layer fibres data. *Computer Methods and Programs in Biomedicine*, 150.
- Kohout, J., Clapworthy, G., Zhao, Y., Tao, Y., Gonzalez-Garcia, G., Dong, F., Wei, H., and Kohoutová, E. (2013). Patient-specific fibre-based models of muscle wrapping. *Interface focus*, 3:20120062.
- Kohout, J. and Kukacka, M. (2014). Real-time modelling of fibrous muscle. *Computer Graphics Forum*, 33(8):1–15.
- Kojic, M., Mijailovic, S., and Zdravkovic, N. (1998). Modelling of muscle behaviour by the finite element method using hill's three-element model. *International Journal for Numerical Methods in Engineering*, 43(5):941–953.
- Kotsalos, C., Latt, J., and Chopard, B. (2019). Bridging the computational gap between mesoscopic and continuum modeling of red blood cells for fully resolved blood flow. *Journal of Computational Physics*, 398. cited By 0.
- Macklin, M., Storey, K., Lu, M., Terdiman, P., Chentanez, N., Jeschke, S., and Müller, M. (2019). Small steps in physics simulation. In *Proceedings of the 18th Annual ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '19, pages 2:1–2:7, New York, NY, USA. ACM.
- Modenese, L., Montefiori, E., Wang, A., Wesarg, S., Viceconti, M., and mazzà, C. (2018). Investigation of the dependence of joint contact forces on musculotendon parameters using a codified workflow for image-based modelling. *Journal of Biomechanics*.
- Müller, M., Heidelberger, B., Hennix, M., and Ratcliff, J. (2007). Position based dynamics. *Journal of Visual Communication and Image Representation*, 18:109–118.
- Oatis, C. (2013). *Kinesiology: The mechanics and pathomechanics of human movement: Second edition*. Lippincott Williams & Wilkins.
- Oberhofer, K., Mithraratne, K., Stott, N. S., and Anderson, I. A. (2009). Anatomically-based musculoskeletal modeling: prediction and validation of muscle deformation during walking. *Vis. Comput.*, 25(9):843–851.
- Shao, X., Liao, E., and Zhang, F. (2017). Improving sph fluid simulation using position based dynamics. *IEEE Access*, PP:1–1.
- Valente, G., Martelli, S., Taddei, F., Farinella, G., and Viceconti, M. (2012). Muscle discretization affects the loading transferred to bones in lower-limb musculoskeletal models. *Proc. Inst. Mech. Eng. H J. Eng. Med.*, 226(2):161–169.
- Van Sint Jan, S. (2005). Introducing anatomical and physiological accuracy in computerized anthropometry for increasing the clinical usefulness of modeling systems. *Critical Reviews in Physical and Rehabilitation Medicine*, 17:149–174.
- Viceconti, M., Clapworthy, G., and Van Sint Jan, S. (2008). The virtual physiological human — a european initiative for in silico human modelling —. *The journal of physiological sciences : JPS*, 58:441–6.
- Wade, S. W., Strader, C., Fitzpatrick, L. A., Anthony, M. S., and O'Malley, C. D. (2014). Estimating prevalence of osteoporosis: examples from industrialized countries. *Archives of Osteoporosis*, 9(1):182.
- Zhang, J., Fernandez, J., Hislop-Jambrich, J., and Besier, T. F. (2016). Lower limb estimation from sparse landmarks using an articulated shape model. *Journal of Biomechanics*, 49(16):3875 – 3881.

9.5 Behavioral Study of Various Radial Basis Functions for Approximation and Interpolation Purposes

The study [89] further explores various RBFs, where the placement of the centre was mainly adopted from the previous paper [93] described in Section 9.3. The new RBF of that paper has also been tested. These more in-depth tests of the narrowed subset of RBFs showed some weaknesses of the author's RBF, mainly that there is a pattern (whole-numbered shape parameter), while the RBF approximation is ill-conditioned. The Gaussian RBF does not provide that shortcoming in the case of 1D signals; however, it also has its issues with shape parameter selection considering conditionality (see Section 9.7).

Specifically, the results presented in the article show that selecting a whole number as the shape parameter α for the proposed RBF can lead to peaks or singularities in the mean square error and conditionality plots. These singularities indicate instability or significant variations in the approximation error and conditionality, which can affect the reliability and performance of the RBF approximation. Therefore, considering and potentially avoiding whole-numbered shape parameters might be advisable in RBF approximation tasks to ensure more stable and consistent results.



Publication [89]:

CERVENKA, M.; SKALA, V. Behavioral Study of Various Radial Basis Functions for Approximation and Interpolation Purposes. *IEEE 18th World Symposium on Applied Machine Intelligence and Informatics, SAMI 2020*. 2020, pp. 135–140. ISBN 978-1-7281-3149-8. Available from DOI: <https://doi.org/10.1109/SAMI48414.2020.9108712>. UT WoS: 000589772600026, EID: 2-s2.0-85087093548, OBD: 43929006

Behavioral Study of Various Radial Basis Functions for Approximation and Interpolation Purposes

Martin Cervenka
University of West Bohemia
Pilsen, Czech Republic
cervemar@kiv.zcu.cz¹

Vaclav Skala
University of West Bohemia
Pilsen, Czech Republic
http://www.VaclavSkala.eu

Abstract— Both approximation and interpolation are techniques commonly used in many scientific areas. Many approaches are depending on input data type, result purpose etc. Input data can be formed in a mesh or not (meshless/meshfree data).

This contribution is oriented on meshless data approximation and interpolation using Radial Basis Functions (RBFs). Different RBFs behaves differently, but many of them have a shape parameter. This paper compares various RBFs concerning its shape parameters and provides some experimental results for each of the selected RBF.

Index Terms—RBF, radial basis function, interpolation, approximation, shape parameter, axis scaling

I. INTRODUCTION

There are many areas for approximation and interpolation in using RBF methods, despite the fact its higher computational cost. Biancolini [1], Menandro [2] and Fasshauer [3] used RBF methods in engineering practise. The RBF technique can be also used for image reconstruction [4], GIS systems [5], meteorology [6], partial differential equations [7], [8] etc.

There are two main groups of data representation i.e. mesh-based and meshfree/meshless. In the case of mesh-based data, a structure of the data is well-known apriori, in opposite to meshfree data, which are scattered in space. Meshfree data lack of connectivity information, so it is typically harder to approximate/interpolate.

Tessellation can be made to transform scattered data (mesh-free) to structured data (mesh). A common tessellation technique is Delaunay triangulation, however, its computational complexity is $O(n^{\lfloor d/2 \rfloor + 1})$ in d -dimensional space, i.e. for $d = 2$ is $O(n^2)$ and for $d = 3$ is $O(n^3)$ (more in Smolik [9]).

Dimension of the data is important, too. The higher the dimension is, the more complex and time-consuming algorithm is used. This is not completely true in the case of RBF approximation, which is nearly independent of problem dimensionality. Another advantage, which RBF technique brings, is that RBF approximation and interpolation is invariant to all rigid Euclidean transformations. It means that it is indifferent whether RBF is used and then transformation is made or the other way around.

The research was supported by the Czech Science Foundation (GACR) project No.GA 17-05534S and partially by SGS 2019-016 project.

¹ Corresponding author

The RBF method is relatively old (Hardy [10], 1971), but there are still many issues to deal with. Some of them are described in the following.

This contribution is focused to determine the behaviour of RBF approximation under certain conditions:

- when RBF shape parameter varies (see "Shape parameter selection"),
- how is approximation or interpolation affected by scaling the X (domain) axis of the original approximated function.

II. RADIAL BASIS FUNCTION

A radial basis function is a function, which value depends only on the distance from one single point, called centre. It means that distance is the only independent variable of the RBF, following notation will be used from now on:

$$\varphi(\|\mathbf{x} - \xi\|) \quad (1)$$

Where ξ is RBF centre coordinate point and \mathbf{u} is arbitrary independent point in the space. $\|\mathbf{x}\|$ denotes euclidean norm of vector \mathbf{x} . It means that the RBF is a single variable function returning a single variable as well.

There are two groups of RBF in general:

- *global* radial basis function is not limited and affects whole space
- *local* radial basis function, which has zero value from some radius and further

A. Global RBF

A typical global RBF function is Gaussian RBF, which is defined as:

$$\varphi(r) = e^{-\alpha r} \quad (2)$$

Plot of this function is a Gaussian bell curve, which has non-zero values on \mathbb{R} . It means that this function influences the whole space (will be explained later on).

Another well-known global RBF is thin-plate spline (TPS), which is defined in (3).

$$\varphi(r) = r^2 \log r = \frac{1}{2} r^2 \log r^2 \quad (3)$$

There are many various global RBF. In this paper, proposed RBF global function with variable exponent is discussed, too:

$$\varphi(r) = r^2 (r^\alpha - 1) \quad (4)$$

B. Local RBF

Local RBFs are then defined in (5). This kind of functions has been introduced by Wendland [11], more of them can be found in paper from Skala [12]. It is nothing more than multiplication of polynomial $P(r)$ with some power of cutted polynomial $1-r$ (+ sign denotes that all negative values are changed to zero), so its value will not be negative.

$$\varphi(r) = (1-r)_+^q P(r)$$

$$\varphi(r) = \begin{cases} (1-r)^q P(r) & 0 \leq r \leq 1 \\ 0 & r > 1 \end{cases} \quad (5)$$

From the group of local radial basis function, it is worth mentioning the simplest local RBF:

$$\varphi(r) = (1-r)_+ \quad (6)$$

If higher power and a reasonable polynomial is used, the resulting RBF can be defined as:

$$\varphi(r) = (1-r)_+^7 (16r^2 + 7r + 1) \quad (7)$$

In this paper, four RBFs defined above are used and plotted together in Fig.1.

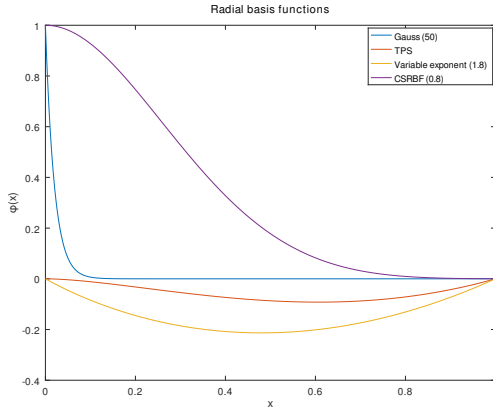


Fig. 1: Plot of the selected RBFs.

In the case of TPS and RBF with variable exponent, both functions "start" in 0 ($f(0) = 0$) and go directly through zero once more ($f(1) = 0$), then they rise to infinity ($\lim_{x \rightarrow +\infty} f(x) = +\infty$). The other two functions (Gaussian and CSRBF) "start" in 1 ($f(0) = 1$) and fall to zero $\lim_{x \rightarrow +\infty} f(x) = 0$, CSRBF with given shape parameter λ (0.8) even satisfies $f(x) = 0, \forall x \geq 1.25$.

Using a weighted sum of infinitely many these elementary functions, it is possible to describe any function. In reality, however, there is a limit on how many RBF is used. In this case, the input function is not described precisely but is only approximated.

III. RBF APPROXIMATION

Function approximation using RBF is done using formula (8). In order to be able to solve this equation, all N RBF center points ξ_j has to be known apriori as well as eventual shape parameter(s) α_i of each RBF φ . Weights λ_j are unknowns and will be computed.

$$h(\mathbf{x}) = \sum_{j=1}^M \lambda_j \varphi(\|\mathbf{x} - \xi_j\|) = \sum_{j=1}^M \lambda_j \varphi(r_j) \quad (8)$$

Where N is number of points and M is number of reference points (centres). The introduced equation can be expressed in matrix form, which leads to system of linear equations:

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \quad A_{ij} = \varphi_{ij}, \mathbf{b} = h_i, \mathbf{x} = \lambda_j, N > M, i = 1, \dots, N \quad (9)$$

Matrix \mathbf{A} is the rectangular matrix in general and the overdetermined system is obtained. There are several methods to solve the overdetermined system of equations. To minimize mean square error, the Ordinary Least Squares method (OLS) is used. Weights λ_j can be computed using OLS method by pseudoinverse as:

$$\lambda = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{h} \quad (10)$$

The solution of this equation leads to good approximation. However, Skala [13] shows that there may be some instability problems. Moreover, Majdisova [14] proved that in case of solving this equation via OLS additional polynomial conditions cannot be included.

IV. RBF INTERPOLATION

The RBF interpolation differs mathematically from approximation. In this case only distances between center points are considered. Equation for the RBF interpolation shown below:

$$h(\mathbf{x}_i) = \sum_{j=1}^N \lambda_j \varphi(\|\mathbf{x}_i - \mathbf{x}_j\|) = \sum_{j=1}^N \lambda_j \varphi(r_{ij}) \quad (11)$$

It is possible to rewrite this equation to matrix form the same way as in approximation (equation (9)). In approximation, resulting matrix is rectangular in general, whereas in this case the result \mathbf{A} is a square matrix of the linear equation system.

In opposite of approximation, the matrix \mathbf{A} can be further extended with polynomial conditions, now. The extended system is shown in formula (12).

$$\begin{bmatrix} \mathbf{A} & \mathbf{P} \\ \mathbf{P}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \lambda \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} \mathbf{h} \\ \mathbf{0} \end{bmatrix} \quad (12)$$

The matrix \mathbf{P} represents polynomial additional conditions, λ is a vector of RBF weights, vector \mathbf{a} contains resulting polynomial coefficients, N is number of points and \mathbf{h} are given values at the given points, see Skala [15].

According to Jäger [16] and Skala [17] [18], in some cases it can be counterproductive to introduce polynomial conditions, especially for large scope of domain data.

V. RBF CENTRES PLACEMENT

The placement of RBF functions (setting their centre points) is another task to solve. A naïve method is to uniformly sample input function, but it does not take function properties into account. Despite this fact sometimes this approach is used, e.g. Singh [19]. Orr [20] in his paper proposes a regularization method, Wright [21] brings an improvement of this method near function boundaries. Majdisova et. al. [22] compare different techniques of RBF placement. In this paper, the geometrical properties of input signals are considered. RBFs are placed where input signals reach minima, maxima (1st order derivative is zero), inflexion points (2nd order derivative is zero) and to locations where 3rd order derivative is zero as well. This approach is inspired by [15], whereas in this paper there is a 1½ dimensional case (single parameter function) instead of 2½D (double parameter function). An example is shown in Fig.2, red crosses denote these points.

Function boundaries are often problematic, because there may not be any geometrically important points (minima, maxima, etc.). To solve this issue, artificial centres are added to the exact border. Even with this modification, the largest errors are still located near boundaries. This is solved by adding more points near boundaries. In 1½D case, four points should solve this issue (placed to 0%, 5%, 95% and 100% ratio of the domain space).

At this point, geometrically important points are covered, but it may happen that there will be a large "gap" between two consecutive centre points. One of the solution is to force minimal constant frequency, so large gaps will be filled with one or more another centre points.

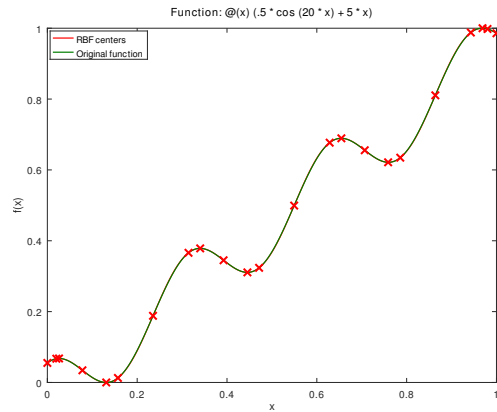


Fig. 2: Geometrically important points. The extension of this approach to the 2½D case was already explored by Vasta [23].

VI. SHAPE PARAMETER SELECTION

The selection of the shape parameter is a crucial part to do accurate interpolation or approximation. Shape parameter

can be selected for each RBF independently, which has an advantage of more precise results, on the other hand, all shape parameter values have to be stored then. A task of independent selection is an open question. Experiments from Skala, Karim and Zabran [24] showed, that there is probably no optimal global shape parameter. Suitable parameter selection is proposed by Wang and Liu [25], Afiatodust and Esmaeilbeigi [26] (using genetic algorithm) or Sarra and Sturgill [27] (random non-deterministic approach), however, none of them can find optimal shape parameter for each RBF. Optimal selection is due to this fact still an open question.

To simplify the problem, one single global shape parameter can be selected for all of RBFs, which results in less accurate approximation in general, but there is no need to store all values of shape parameters. In this case for each RBF one global shape parameter has been empirically selected in experiments described below.

VII. EXPERIMENTAL RESULTS

For testing purposes, we used 18 different functions (see Tab. I) taken from [15]. These functions, when approximated, discover various behaviour of chosen RBF approximation approach.

1	$\sin(15x^2) + 5x$
2	$0.5 \cos(20x) + 5x$
3	$50(0.4 \sin(15x^2) + 5x)$
4	$\sin(8\pi x)$
5	$\sin(6\pi x^2)$
6	$\sin(25x + 0.1) / (25x + 0.1)$
7	$2 \sin(2\pi x) + \sin(4\pi x)$
8	$2 \sin(2\pi x) + \sin(4\pi x) + \sin(8\pi x)$
9	$-2 \sin(2\pi x) + \cos(6\pi x)$
10	$2 \sin(2\pi x) + \cos(6\pi x)$
11	$-2 \sin(2\pi x) + \cos(6\pi x) - x$
12	$-2 \cos(2\pi x) - \cos(4\pi x)$
13	$\text{atan}((10x - 5)^3) + 0.5 \text{atan}((10x - 8)^3)$
14	$(4.48x - 1.88) \sin((4.88x - 1.88)^2) + 1$
15	$e^{10x-6} \sin((5x - 2)^2) + (3x - 1)^3$
16	$(1/9) \tanh(9x + 0.5)$
17	$\frac{6}{(1+16(x+0.5)^2) + \log(0.01*(x-.25)^2+10^{-6})+4}$
18	$\frac{6}{(1+16(x+0.5)^2) + \log(0.01*(x-.25)^2+10^{-10})+4}$

TABLE I: Tested artificial signals (taken from [15], extended).

For simplification of comparison, all 18 input signals have been tested for $x \in (0, 1)$ and for simple comparison of errors all values in given domain have been scaled to $f(x) \in (0, 1)$ as well. Mean square error and condition number of matrix **A** (from equation (9)) are quantitative criteria for following tests.

A. Detecting geometrically significant points

Geometrically significant points are points with special properties. Special properties can be anything, in this case

special points are points which n -th derivative is zero, meaning:

$$f^{(n)}(x) = 0 \tag{13}$$

There are two groups of methods of how to detect significant points. One of them works in discrete space, the other in continuous space. To get these points in continuous space, either analytical solution of equation (13) is needed or numerical solution is required (gradient descent etc.).

In this paper, discrete space detection is selected. In this case, the function is sampled to N points (specifically 1000 points in this paper), which are denoted a_n . To get local extrema, differentiation is done, as follows:

$$a'_n = a_{n-1} - a_n \tag{14}$$

When consecutive elements a'_n and a'_{n+1} differs in sign, then local extrema is detected (difference go opposite way in two consecutive elements). If the differentiation is done twice, inflexion points are detected. It is possible then to continue with higher-order differentiation. In this paper it is done at most three times.

Short explanation: three-element signal [3, 5, 4], which has obvious local extrema (maximum) of 5. After differentiation, [-2, 1] vector is obtained. An only consecutive pair of elements differs in sign, meaning there were a local extrema.

This simplistic approach is sufficient for smooth functions, on noisy signal it would detect much more points than it should (smoothing is required).

B. Influence of the shape parameter

To test how much dependent the resulting approximation to shape parameter selection is, the range of the parameters is used. For every shape parameter, mean square error and condition number (of the matrix **A** in equation (9) are computed and showed in the following figures.

The first RBF tested is global Gaussian RBF. There are plotted results of mean square error depending on the selected shape parameter in Fig.3. It seems that the lower shape parameter α is selected, the lower average error is reached, but some functions do not respect this trend. It is risky to select lower shape parameter (e.g. $\alpha < 20$), but for some functions it gives the best result.

Condition number values of the matrix **A** are presented in Fig.4. The higher value of shape parameter α is chosen, the better conditionality the problem has.

On the other hand, RBF $r^2 (r^\alpha - 1)$ behaves the other way around. The higher shape parameter is selected, the lower error is possible to get, but it has worse conditionality at this point. Results are plotted in Fig.5 (mean square error) and Fig.6 (conditionality). There are some peaks (singularities) on this plot, which are caused by selecting the whole number as the shape parameter α .

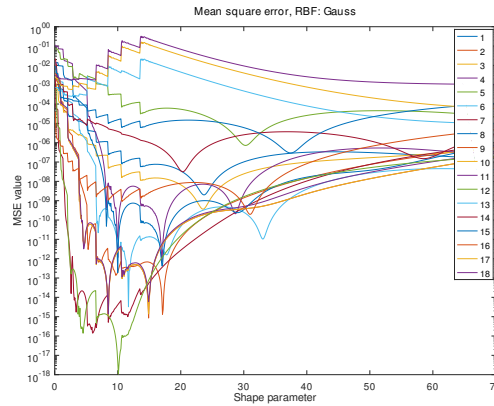


Fig. 3: Mean square error for Gaussian RBF. Note logarithmic scale on Y axis.

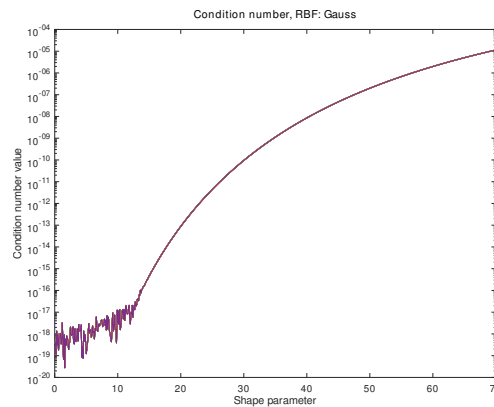


Fig. 4: Condition number values for Gaussian RBF. Note logarithmic scale on Y axis.

C. Influence of X-axis scaling

Idea is that scaling of the X-axis should influence resulting approximation error. This test stretches or squishes X-axis. Scaling X-axis is a similar operation to changing shape parameter, but these two operations are not linear in general. It should be noted that shape parameters are fixed during this test.

VIII. CONCLUSION

Different RBF functions have different properties and different behaviour in respect to its parameter(s), even using different signal which has to be approximated and does not have a significant impact. It seems there is a pattern between various

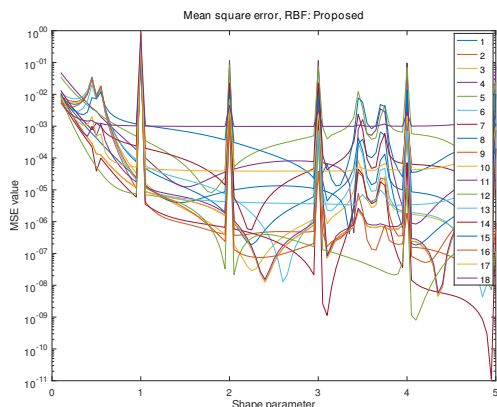


Fig. 5: Mean square error for proposed RBF. Note logarithmic scale on Y axis.

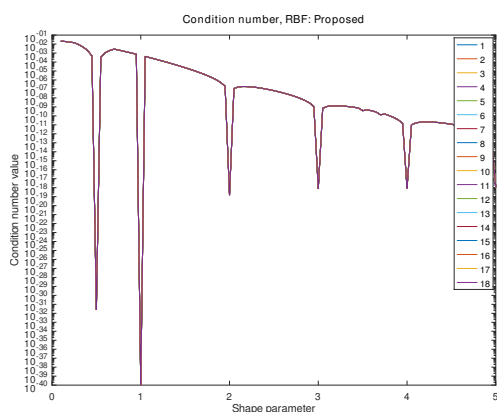


Fig. 6: Condition number values for proposed RBF. Note logarithmic scale on the Y-axis.

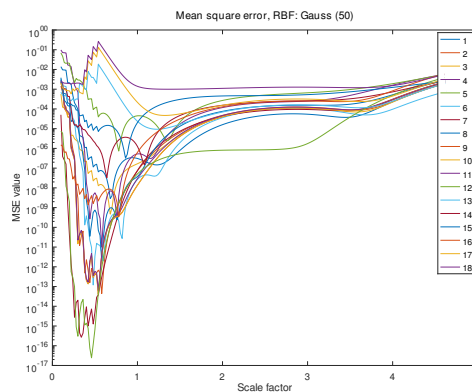


Fig. 7: Mean square error for Gaussian RBF. Note logarithmic scale on Y axis.

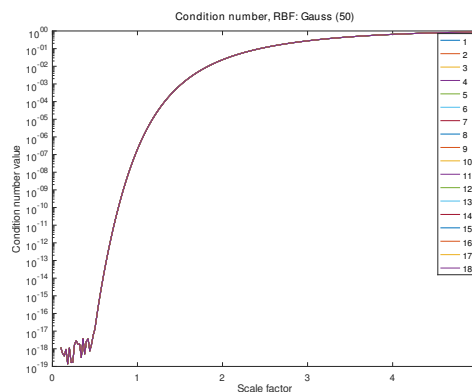


Fig. 8: Condition number values for Gaussian RBF. Note logarithmic scale on Y-axis.

approximated signals as far as RBFs with same parameters are used and it is currently under investigation.

The tests with X-axis scaling show that the Gaussian RBF tends to have worse conditionality while scaling down, but it may reach lower approximation error. Second tested RBF seems to be independent to scaling (except one singularity) if approximation error is considered, conditionality is good without scaling, it is getting worse when scaling is applied.

Presented results show that there is a tradeoff between precision and conditionality in general. Selected shape parameter is due to this fact dependent on the predefined goal, which should be achieved. If the low error is requested, scaling down the X-axis may help. If the conditionality has to be high, scale-up may help in that case. This study may help to

select appropriate scaling factor as well as shape parameter, according to the approximation goal.

ACKNOWLEDGMENT

The authors would like to thank their colleagues and students at the University of West Bohemia for their suggestions and discussion. Authors would like to thank also to anonymous reviewers for their valuable comments, provided hints, and suggestions.

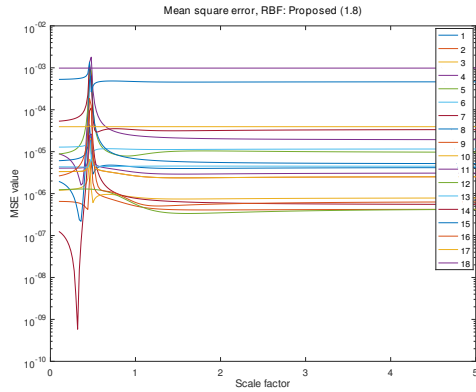


Fig. 9: Mean square error for proposed RBF. Note logarithmic scale on Y-axis.

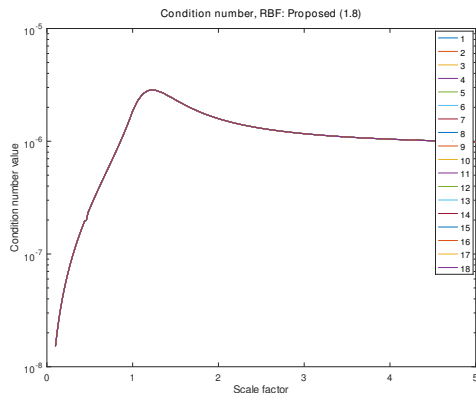


Fig. 10: Condition number values for proposed RBF. Note logarithmic scale on Y-axis.

REFERENCES

[1] M. E. Biancolini, *Fast Radial Basis Functions for Engineering Applications*, 1st ed. Springer International Publishing, 2017.
 [2] F. Menandro, "Two new classes of compactly supported radial basis functions for approximation of discrete and continuous data," *Engineering Reports*, vol. 2019:1:e12028, pp. 1–30, 2019.
 [3] G. Fasshauer, *Meshfree Approximation Methods with Matlab*, 1st ed. World Scientific, 2007.
 [4] K. Uhlir and V. Skala, "Reconstruction of damaged images using radial basis functions," *13th European Signal Processing Conference, EUSIPCO 2005*, 01 2005.
 [5] R. Pan and V. Skala, "Surface reconstruction with higher-order smoothness," *The Visual Computer*, vol. 28, no. 2, pp. 155–162, Feb 2012. [Online]. Available: <https://doi.org/10.1007/s00371-011-0604-9>
 [6] D. Pepper, C. Rasmussen, and D. Fyda, "A meshless method using global radial basis functions for creating 3-d wind fields from sparse meteorological data," *Computer Assisted Methods in Engineering and Science*, vol. 21, no. 3/4, pp. 233–243, 2017. [Online]. Available: <https://ames.ippt.pan.pl/index.php/ames/article/view/42>

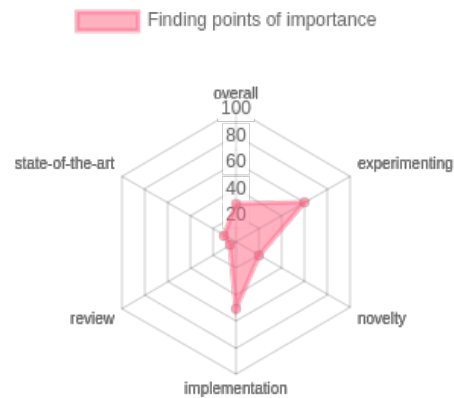
[7] E. Larsson and B. Fornberg, "A numerical study of some radial basis function based solution methods for elliptic PDEs," *Computers & Mathematics with Applications*, vol. 46, no. 5, pp. 891 – 902, 2003. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0898122103901519>
 [8] Y.-C. Hon, B. Šarler, and D. fang Yun, "Local radial basis function collocation method for solving thermo-driven fluid-flow problems with free surface," *Engineering Analysis with Boundary Elements*, vol. 57, pp. 2 – 8, 2015, rBF Collocation Methods. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0955799714002793>
 [9] M. Smolik and V. Skala, "Fast parallel triangulation algorithm of large data sets in e2 and e3 for in core and out core memory processing," vol. 8580, 07 2014.
 [10] R. L. Hardy, "Multiquadric equations of topography and other irregular surfaces," *Journal of Geophysical Research (1896-1977)*, vol. 76, no. 8, pp. 1905–1915, 1971. [Online]. Available: <https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/JB076i008p01905>
 [11] H. Wendland, "Computational aspects of radial basis function approximation," in *Topics in Multivariate Approximation and Interpolation*, ser. Studies in Computational Mathematics, K. Jetter, M. D. Buhmann, W. Haussmann, R. Schaback, and J. Stöckler, Eds. Elsevier, 2006, vol. 12, pp. 231 – 256. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1570579X06800108>
 [12] V. Skala, "RBF Interpolation with CSRBF of Large Data Sets," *Procedia Computer Science*, vol. 108, pp. 2433 – 2437, 2017, International Conference on Computational Science, ICCS 2017, 12-14 June 2017, Zurich, Switzerland. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S187705091730621X>
 [13] —, "Least square method robustness of computations: What is not usually considered and taught," in *2017 Federated Conference on Computer Science and Information Systems*, 09 2017, pp. 537–541.
 [14] Z. Majdisova and V. Skala, "A new radial basis function approximation with reproduction," *CGVCVIP 2016*, 04 2018.
 [15] V. Skala and M. Cervenka, "Novel RBF Approximation Method Based on Geometrical Properties for Signal Processing with a New RBF Function: Experimental Comparison," *Informatics 2019*, 2019.
 [16] J. Jäger, "Advances in radial and spherical basis function interpolation," Ph.D. dissertation, Justus-Liebig-Universität, Otto-Behaghel-Str. 8, 35394 Gießen, 2018.
 [17] V. Skala, "RBF interpolation and approximation of large span data sets," in *MCSI 2017 – Corfu*. IEEE, 2018, pp. 212–218.
 [18] —, "RBF interpolation with CSRBF of large data sets," in *ICCS 2017, Procedia Computer Science*, vol. 108. Elsevier, 2017, pp. 2433–2437.
 [19] B. Singh and D. Toshniwal, "MOWM: Multiple Overlapping Window Method for RBF based missing value prediction on big data," *Expert Systems with Applications*, vol. 122, pp. 303 – 318, 2019. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0957417418308340>
 [20] M. J. L. Orr, "Regularization in the selection of radial basis function centers," *Neural Computation*, vol. 7, no. 3, pp. 606–623, 1995. [Online]. Available: <https://doi.org/10.1162/neco.1995.7.3.606>
 [21] G. B. Wright, "Radial basis function interpolation: Numerical and analytical developments," Ph.D. dissertation, University of Colorado at Boulder, Boulder, CO, USA, 2003, aAI3087597.
 [22] Z. Majdisova and V. Skala, "Radial basis function approximations: comparison and applications," *Applied Mathematical Modelling*, vol. 51, pp. 728 – 743, 2017. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0307904X17304717>
 [23] J. Vasta, V. Skala, M. Smolik, and M. Cervenka, "Modified Radial Basis Functions Approximation Respecting Data Local Features," *Informatics 2019 (IEEE proceedings, in print)*, 2019.
 [24] V. Skala, S. Karim, and M. Zabran, "Radial Basis Function Approximation Optimal Shape Parameters Estimation: Preliminary Experimental Results," in *(submitted for publication)*. AIP Press, 2019.
 [25] J. Wang and G. Liu, "On the optimal shape parameters of radial basis functions used for 2-d meshless methods," *Computer methods in applied mechanics and engineering*, vol. 191, pp. 2611–2630, 2002.
 [26] F. Afiatdoust and M. Esmailbeigi, "Optimal variable shape parameters using genetic algorithm for radial basis function approximation," *Ain Shams Engineering Journal*, vol. 6, pp. 639–647, 2015.
 [27] S. A. Sarra and D. Sturgill, "A random variable shape parameter strategy for radial basis function approximation methods," *Engineering Analysis with Boundary Elements*, vol. 33, pp. 1239–1245, 2009.

9.6 Finding Points of Importance for Radial Basis Function Approximation of Large Scattered Data

This article [88] focuses mainly on describing the experimental results demonstrating the previously proposed approaches' effectiveness. The results showed high approximation precision in tests involving various functions, using only a fraction of the available data points. For instance, the proposed approximation method exhibits a compression ratio between 5% and 10%, still maintaining good precision and a high compression ratio, showing that there can be data reduction and simplification in the case of muscle modelling.

The study's conclusion highlights the simplicity and efficiency of the RBF-based approximation method, which achieves relatively low error rates and high data compression. However, there are still many other function points of importance, which may even enhance the approximation's precision by including them. However, evaluating neighbouring points is crucial for scattered data to identify the points of importance accurately. The study underscores the potential for future research, especially in analyzing approximation behaviour at interval borders in 3D scenarios, which is identified as a critical aspect for further development. Studying behaviour on borders in 3D scenarios is beneficial for muscle modelling.

The article also contributes to data approximation, particularly in handling large and complex data sets. The proposed method stands out for its ability to simplify the approximation process while maintaining high accuracy and efficiency. It mainly benefits engineering and scientific computations involving large scattered data sets.






Publication [88]:

SKALA, V.; KARIM, S.; CERVENKA, M. Finding Points of Importance for Radial Basis Function Approximation of Large Scattered Data. *Computational Science - ICCS 2020, Part VI, LNCS 12142*. 2020, pp. 239–250. Available from DOI: https://doi.org/10.1007/978-3-030-50433-5_19. OBD: 43932925, UT WoS: 000841676000019, EID: 2-s2.0-85087274721



Finding Points of Importance for Radial Basis Function Approximation of Large Scattered Data

Vaclav Skala¹ , Samsul Ariffin Abdul Karim^{1,2} ,
and Martin Cervenka¹ 

¹ Department of Computer Science and Engineering,
Faculty of Applied Sciences, University of West Bohemia, Univerzitni 8,
301 00 Plzen, Czech Republic

skala@kiv.zcu.cz, samsul_ariffin@utp.edu.my

² Fundamental and Applied Sciences Department and Centre for Smart Grid
Energy Research (CSMER), Institute of Autonomous System,
Universiti Teknologi PETRONAS, Bandar Seri Iskandar, 32610 Seri Iskandar,
Perak DR, Malaysia

Abstract. Interpolation and approximation methods are used in many fields such as in engineering as well as other disciplines for various scientific discoveries. If the data domain is formed by scattered data, approximation methods may become very complicated as well as time-consuming. Usually, the given data is tessellated by some method, not necessarily the Delaunay triangulation, to produce triangular or tetrahedral meshes. After that approximation methods can be used to produce the surface. However, it is difficult to ensure the continuity and smoothness of the final interpolant along with all adjacent triangles. In this contribution, a meshless approach is proposed by using radial basis functions (RBFs). It is applicable to explicit functions of two variables and it is suitable for all types of scattered data in general. The key point for the RBF approximation is finding the important points that give a good approximation with high precision to the scattered data. Since the compactly supported RBFs (CSRBF) has limited influence in numerical computation, large data sets can be processed efficiently as well as very fast via some efficient algorithm. The main advantage of the RBF is, that it leads to a solution of a system of linear equations (SLE) $Ax = b$. Thus any efficient method solves the systems of linear equations that can be used. In this study is we propose a new method of determining the importance points on the scattered data that produces a very good reconstructed surface with higher accuracy while maintaining the smoothness of the surface.

Keywords: Meshless methods · Radial Basis Functions · Approximation

1 Introduction

Interpolation and approximation techniques are used in the solution of many engineering problems. However, the interpolation of unorganized scattered data is still a severe problem. In the one dimensional case, i.e., curves represented as $y = f(x)$, it is possible to order points according to the x -coordinate. However, in a higher dimensionality this is not possible. Therefore, the standard approaches are based on the tessellation of the domain in x, y or x, y, z spaces using, e.g. Delaunay triangulation [7], etc. This approach is applicable for static data and t -varying data, if data in the time domain are “framed”, i.e. given for specific time samples. It also leads to an increase of dimensionality, i.e. from triangulation in E^2 to triangulation in E^3 or from triangulation in E^3 to triangulation in E^4 , etc. It results in significant increase of the triangulation complexity and complexity of a triangulation algorithm implementation. This is a significant factor influencing computation in the case of large data sets and large range data sets, i.e. when x, y, z values are spanned over several magnitudes.

On the contrary, meshless interpolations based on Radial Basis Functions (RBF) offer several significant advantages, namely:

- RBF interpolation is applicable generally to d -dimensional problems and does not require tessellation of the definition domain
- RBF interpolation and approximation is especially convenient for scattered data interpolation, including interpolation of scattered data in time as well
- RBF interpolation is smooth by a definition
- RBF interpolation can be applied for interpolation of scalar fields and vector fields as well, which can be used for scalar and vector fields visualization
- If the Compactly Supported RBFs (CSRBF) are used, sparse matrix data structures can be used which decreases memory requirements significantly.

However, there are some weak points of RBF application in real problems solution:

- there is a real problem for large data sets with robustness and reliability of the RBF application due to high conditionality of the matrix A of the system of linear equations, which is to be solved
- numerical stability and representation is to be applied over a large span of x, y, z values, i.e. if values are spanned over several magnitudes
- problems with memory management as the memory requirements are of $O(N^2)$ complexity, where N is a number of points in which values are given
- the computational complexity of a solution of the linear system, which is $O(N^3)$, resp. $O(kN^2)$, where k is a number of iteration if the iterative method are used, but k is relatively high, in general.
- Problems with unexpected behavior at geometrical borders

Many contributions are solving some issues of the RBF interpolation and approximation available. Numerical tests are mostly made using some standard testing functions and restricted domain span, mostly taking interval $\langle 0, 1 \rangle$ or similar. However, in many physically based applications, the span of the domain is higher, usually over

several magnitudes and large data sets need to be processed. Also large data sets are to be processed.

As the meshless techniques are easily scalable to higher dimensions and can handle spatial scattered data and spatial-temporal data as well, they can be used in many engineering and economical computations, etc. Polygonal representations (tessellated domains) are used in computer graphics and visualization as a surface representation and for surface rendering. In time-varying objects, a surface is represented as a triangular mesh with constant connectivity.

On the other hand, all polygonal based techniques, in the case of scattered data, require tessellations, e.g. Delaunay triangulation with $O(N^{\lfloor d/2+1 \rfloor})$ computational complexity for N points in d -dimensional space or another tessellation method. However, the complexity of tessellation algorithms implementation grows significantly with dimensionality and severe problems with robustness might be expected, as well.

In the case of data visualization smooth interpolation or approximation on unstructured meshes is required, e.g. on triangular or tetrahedral meshes, when physical phenomena are associated with points, in general. This is quite a difficult task especially if the smoothness of interpolation is needed. However, it is a natural requirement in physically-based problems.

2 Meshless Interpolation

Meshless (meshfree) methods are based on the idea of Radial Basis Function (RBF) interpolation [1, 2, 22, 23], which is not separable. RBF based techniques are easily scalable to d -dimensional space and do not require tessellation of the geometric domain and offer smooth interpolation naturally. In general, meshless techniques lead to a solution of a linear system equations (LS) [4, 5] with a full or sparse matrix.

Generally, meshless methods for scattered data can be split into two main groups in computer graphics and visualization:

- “implicit” – $F(\mathbf{x}) = 0$, i.e. $F(x, y, z) = 0$ used in the case of a surface representation in E^3 , e.g. surface reconstruction resulting into an implicit function representation. This problem is originated from the implicit function modeling [15] approach,
- “explicit” – $F(\mathbf{x}) = h$ used in interpolation or approximation resulting in a functional representation, e.g. a height map in E^2 , i.e. $h = F(x, y)$.

where: \mathbf{x} is a point represented generally in d -dimensional space, e.g. in the case of 2-dimensional case $\mathbf{x} = [x, y]^T$ and h is a scalar value or a vector value.

The RBF interpolation is based on computing of the distance of two points in the d -dimensional space and it is defined by a function:

$$f(\mathbf{x}) = \sum_{j=1}^M \lambda_j \varphi(\|\mathbf{x} - \mathbf{x}_j\|) = \sum_{j=1}^M \lambda_j \varphi(r_j) \quad (1)$$

where: $r_j = \|\mathbf{x} - \mathbf{x}_j\|_2 \stackrel{\text{def}}{=} \sqrt{(x - x_j)^2 + (y - y_j)^2}$ (in 2-dimensional case) and λ_j are weights to be computed. Due to some stability issues, usually a polynomial $P_k(\mathbf{x})$ of a degree k is added [6]. It means that for the given data set $\{(\mathbf{x}_i, h_i)\}_1^M$, where h_i are associated values to be interpolated and \mathbf{x}_i are domain coordinates, we obtain a linear system of equations:

$$h_i = f(\mathbf{x}_i) = \sum_{j=1}^M \lambda_j \varphi(\|\mathbf{x}_i - \mathbf{x}_j\|) + P_k(\mathbf{x}_i) \quad i = 1, \dots, M \quad \mathbf{x} = [x, y : 1]^T \quad (2)$$

For a practical use, a polynomial of the 1st degree is used, i.e. linear polynomial $P_1(\mathbf{x}) = \mathbf{a}^T \mathbf{x}$ in many applications. Therefore, the interpolation function has the form:

$$\begin{aligned} f(\mathbf{x}_i) &= \sum_{j=1}^M \lambda_j \varphi(\|\mathbf{x}_i - \mathbf{x}_j\|) + \mathbf{a}^T \mathbf{x}_i \quad h_i = f(\mathbf{x}_i) \quad i = 1, \dots, M \\ &= \sum_{j=1}^M \lambda_j \varphi_{i,j} + \mathbf{a}^T \mathbf{x}_i \end{aligned} \quad (3)$$

and additional conditions are to be applied:

$$\sum_{j=1}^M \lambda_j \mathbf{x}_i = \mathbf{0} \quad \text{i.e.} \quad \sum_{j=1}^M \lambda_j x_i = 0 \quad \sum_{j=1}^M \lambda_j y_i = 0 \quad \sum_{j=1}^M \lambda_j = 0 \quad (4)$$

It can be seen that for the d -dimensional case a system of $(M + d + 1)$ linear system has to be solved, where M is a number of points in the dataset and d is the dimensionality of data. For $d = 2$ vectors \mathbf{x}_i and \mathbf{a} are in the form $\mathbf{x}_i = [x_i, y_i, 1]^T$ and $\mathbf{a} = [a_x, a_y, a_0]^T$, we can write:

$$\begin{bmatrix} \varphi_{1,1} & \dots & \varphi_{1,M} & x_1 & y_1 & 1 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ \varphi_{M,1} & \dots & \varphi_{M,M} & x_M & y_M & 1 \\ x_1 & \dots & x_M & 0 & 0 & 0 \\ y_1 & \dots & y_M & 0 & 0 & 0 \\ 1 & \dots & 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_M \\ a_x \\ a_y \\ a_0 \end{bmatrix} = \begin{bmatrix} h_1 \\ \vdots \\ h_M \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (5)$$

This can be rewritten in the matrix form as:

$$\begin{bmatrix} \mathbf{B} & \mathbf{P} \\ \mathbf{P}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\lambda} \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \mathbf{0} \end{bmatrix} \quad \mathbf{A}\mathbf{x} = \mathbf{b} \quad \mathbf{a}^T \mathbf{x}_i = a_x x_i + a_y y_i + a_0 \quad (6)$$

For the two-dimensional case and M points given a system of $(M + 3)$ linear equations has to be solved. If “global” functions, e.g. $\varphi(r) = r^2 \lg r$, are used, then the matrix \mathbf{B} is “full”, if “local” functions CSRBFs are used, the matrix \mathbf{B} can be sparse.

The RBF interpolation was originally introduced by Hardy as the multiquadric method in 1971 [5], which was called Radial Basis Function (RBF) method. Since then many different RFB interpolation schemes have been developed with some specific properties, e.g. 4 uses $\varphi(r) = r^2 \lg r$, which is called Thin-Plate Spline (TPS), a function $\varphi(r) = e^{-(\epsilon r)^2}$ was proposed in [23]. However, the shape parameter ϵ might leads to an ill-conditioned system of linear equations [26].

The CSRBFs were introduced as:

$$\varphi(r) = \begin{cases} (1 - r)^q P(r), & 0 \leq r \leq 1 \\ 0, & r > 1 \end{cases} \quad (7)$$

where: $P(r)$ is a polynomial function and q is a parameter. Theoretical problems with numerical stability were solved in [4]. In the case of global functions, the linear system of equations is becoming ill conditioned and problems with convergence can be expected. On the other hand, if the CSRBFs are taken, the matrix \mathbf{A} is becoming relatively sparse, i.e. computation of the linear system will be faster, but we need to carefully select the scaling factor α (which can be “tricky”) and the final function might tend to be “blobby” shaped, see Table 1 and Fig. 1.

Table 1. Typical examples of “local” functions – CSRBF (“+” means – value zero out of $(0, 1)$)

ID	Function	ID	Function
1	$(1 - r)_+$	6	$(1 - r)_+^6 (35r^2 + 18r + 3)$
2	$(1 - r)_+^3 (3r + 1)$	7	$(1 - r)_+^8 (32r^3 + 25r^2 + 8r + 3)$
3	$(1 - r)_+^5 (8r^2 + 5r + 1)$	8	$(1 - r)_+^3$
4	$(1 - r)_+^2$	9	$(1 - r)_+^3 (5r + 1)$
5	$(1 - r)_+^4 (4r + 1)$	10	$(1 - r)_+^7 (16r^2 + 7r + 1)$

The compactly supported RBFs are defined for the “normalized” interval $r \in 0, 1$, but for the practical use a scaling is used, i.e. the value r is multiplied by shape parameter α , where $\alpha > 0$.

Meshless techniques are primarily based on the approaches mentioned above. They are used in engineering problem solutions, nowadays, e.g. partial differential equations, surface modeling, surface reconstruction of scanned objects [13, 14], reconstruction of corrupted images [21], etc. More generally, meshless object representation is based on specific interpolation or approximation techniques [1, 6, 23].

The resulting matrix A tends to be large and ill-conditioned. Therefore, some specific numerical methods have to be taken to increase the robustness of a solution, like preconditioning methods or parallel computing on GPU [9, 10], etc. In addition, subdivision or hierarchical methods are used to decrease the sizes of computations and increase robustness [15, 16, 27].

It should be noted, that the *computational complexity* of meshless methods actually covers the complexity of tessellation itself and interpolation and approximation methods. This results in problems with large data set processing, i.e. numerical stability and memory requirements, etc.

If global RBF functions are considered, the RBF matrix is full and in the case of 10^6 of points, the RBF matrix is of the size approx. $10^6 \times 10^6$! On the other hand, if CSRBF used, the relevant matrix is sparse and computational and memory requirements are decreased significantly using special data structures [8, 10, 20, 27].

In the case of physical phenomena visualization, data received by simulation, computation or obtained by experiments usually are oversampled in some areas and also numerically more or less precise. It seems possible to apply approximation methods to decrease computational complexity significantly by adding virtual points in the place of interest and use analogy of the least square method modified for the RBF case [3, 12, 17, 25].

Due to the CSRBF representation the space of data can be subdivided, interpolation, resp. the approximation can be split to independent parts and computed more or less independently [20]. This process can be also parallelized and if appropriate computational architecture is used, e.g. GPU, etc. it will lead to faster computation as well. The approach was experimentally verified for scalar and vector data used in the visualization of physical phenomena.

3 Points of Importance

Algorithms developed recently were based on different specific properties of “global” RBFs or “local” compactly supported RBFs (CS-RBFs) and application areas expected, e.g. for interpolation, approximation, solution of partial differential equations, etc., expecting “reasonable” density of points. However, there are still some important problems to be analyzed and hopefully solved, especially:

- What is an acceptable compromise between the precision of approximation and compression ratio, i.e. reduction of points, if applicable?
- What is the optimal constant shape parameter, if does exist and how to estimate it efficiently [26]?

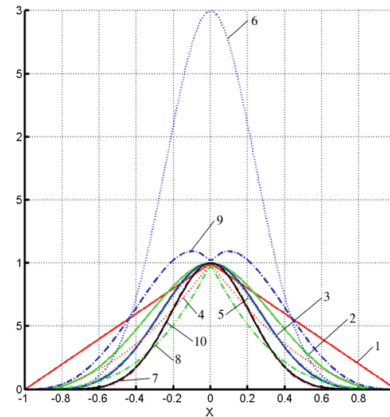


Fig. 1. Properties of CSRBFs

- What are optimal shape parameters α for every single $\varphi(r, \alpha)$ [24, 26]?
- What is the robustness and stability of the RBF for large data and large range span of data with regard to shape parameters [16, 17]?

In this contribution, we will analyze a specific problem related to the first question.

Let us consider given points of a curve (samples of a signal), described by explicit function $y = f(x)$. According to the Nyquist-Shannon theorem, the sampling frequency should be at least double the frequency of the highest frequency of the original signal. The idea is, how “points of importance”, i.e. points of inflection and extrema can be

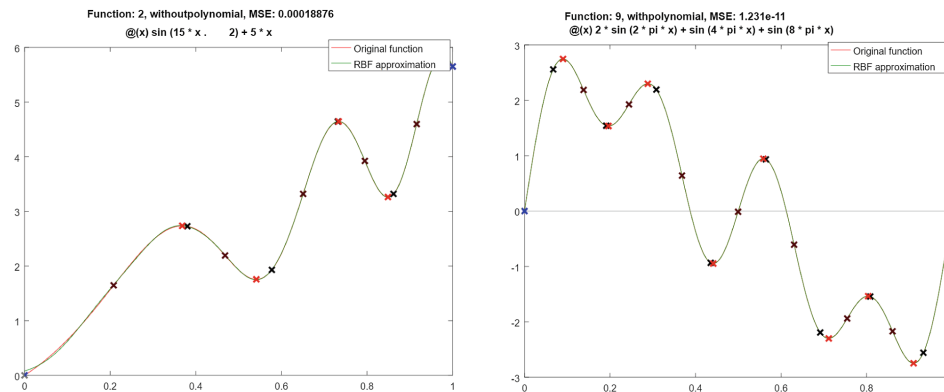


Fig. 2. Testing functions and resulting approximation based on the points of importance (red points are extrema, black points are additional points of importance) (Color figure online)

used for smooth precise curve approximation.

Let us consider sampled curves in Fig. 2, i.e. a signal without noise (the blue points are values at the borders, red are maxima, the black are inflection and added points. It can be seen that the reconstruction based on radial basis functions (RBF) has to pass:

- points at the interval borders
- points at extremes, maxima and minima
- some other important points, like points of inflection etc., and perhaps some additional points of the given data to improve signal reconstruction.

However, there several factors to be considered as well, namely:

- extensibility from 2 D to 3 D for explicit functions of two variables, i.e. $z = f(x, y)$ and hopefully to higher dimension robustness of computation as given discrete data are given.

For extrema finding, the first derivative $f'(x)$ is to be replaced by a standard discrete scheme. At the left, resp. right margin, forward, resp. the backward difference is to be used. Inside of the interval, the central difference scheme is recommended, as it also “filters” high frequencies. The simple scheme for the second derivative estimation is shown, too. It can be seen, that this is easily extensible for the 3 D case as well.

$$\begin{aligned} f'(x) &\approx \frac{f(x_{i+1})-f(x_i)}{x_{i+1}-x_i} & f'(x) &\approx \frac{f(x_i)-f(x_{i-1}))}{x_i-x_{i-1}} \\ f''(x) &\approx \frac{(x_{i+1})-f(x_{i-1}))}{2(x_{i+1}-x_{i-1})} & f''(x) &\approx \frac{f(x_{i+1})-2f(x_i)+f(x_{i-1}))}{(x_{i+1}-x_i)(x_i-x_{i-1})} \end{aligned} \quad (8)$$

So far, a finding of extrema is a simple task, now. However, due to the discrete data, the extrema is detected by

$$\text{sign}(f(x_{i+1}) - f(x_i)) \neq \text{sign}(f(x_i) - f(x_{i-1})) \quad (9)$$

as we need to detect the change of the sign, only. This increases the robustness of computation as well. The points of inflections rely on a second derivative, i.e. $f''(x) = 0$; a similar condition can be derived from (8).

Now, all the important points, i.e. points at the interval borders, maxima, minima and points of inflection, are detected and found. However, it is necessary to include some more points at the interval borders (at least one on each side) to respect the local behavior of the curve and increase the precision of approximation. It is recommended to include at least one or two points which are closest to the borders to respect a curve behavior at the beginning and end of the interval. Also, if additional points are inserted ideally between extreme and inflection points, the approximation precision increases. Now, the standard RBF interpolation scheme can be applied.

$$\begin{bmatrix} \mathbf{B} & \mathbf{P} \\ \mathbf{P}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\lambda} \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \mathbf{0} \end{bmatrix} \quad \mathbf{A}\mathbf{x} = \mathbf{b} \quad \mathbf{a}^T \mathbf{x}_i = a_x x_i + a_0 \quad (10)$$

where: \mathbf{B} represents the RBF submatrix, $\boldsymbol{\lambda}$ the weights of RBFs, \mathbf{P} represents points for the polynomial \mathbf{a} represents coefficients of the polynomial, \mathbf{f} given function values.

It should be noted, that in the case of scattered data, neighbors for each point are to be found, before the estimation of the derivative is made. In the 2 D case, ordering is possible, in the 3 D case computation is to be made on neighbors found. If the regular sampling in each dimension (along the axis) is given, computation simplifies significantly.

It is necessary to note that the curve reconstruction is at the Nyquist-Shannon theorem boundary and probably limits of the compression were obtained with very low relative error, which is less than 0.1%. However, we have many more points available and if a higher precision is needed, the approximation based on Least Square Error (LSE) computational scheme with Lagrange multipliers might be used [11]. The RBF methods usually lead to an ill-conditioned system of linear equations [26]. In the case of approximation, it can be partially improved by geometry algebra in projective space [18, 19] approach.

4 Experimental Results

The presented approach was tested on several testing functions used for evaluation of errors, stability, robustness of computation, see Table 2:

Table 2. Examples of testing functions

ID	Function	ID	Function
1	$y = \sin(15x^2 + 5x)$	2	$y = \cos(20x)/2 + 5x$
3	$y = 50(0.4 \sin(15x^2) + 5x)$	4	$y = \sin(8\pi x)$
5	$y = \sin(6\pi x^2)$	6	$y = \sin(25x + 0.1)/(25x + 0.1)$
7	$y = 2 \sin(2\pi x) + \sin(4\pi x)$	8	$y = 2 \sin(2\pi x) + \sin(4\pi x) + \sin(8\pi x)$
9	$y = 2 \sin(\pi(2x - 1)) + \sin(3\pi(2x - 1/2))$	10	$y = 2 \sin(\pi(1 - 2x)) + \sin(3\pi(2x - 1/2))$
11	$y = 2 \sin(\pi(2x - 1)) + \sin(3\pi(2x - 1/2)) - x$	12	$y = 2 \sin(2\pi x - \frac{\pi}{2}) + \sin(3\pi(2x - 1/2))$
13	$y = \operatorname{atan}(10x - 5)^3 + \operatorname{atan}(10x - 8)^3/2$	14	$y = (4.88x - 1.88) * \sin(4.88x - 1.88)^2 + 1$
15	$y = \exp(10x - 6) * \sin(5x - 2)^3 + (3x - 1)^3$	16	$y = \tanh(9x + 1/2)/9$

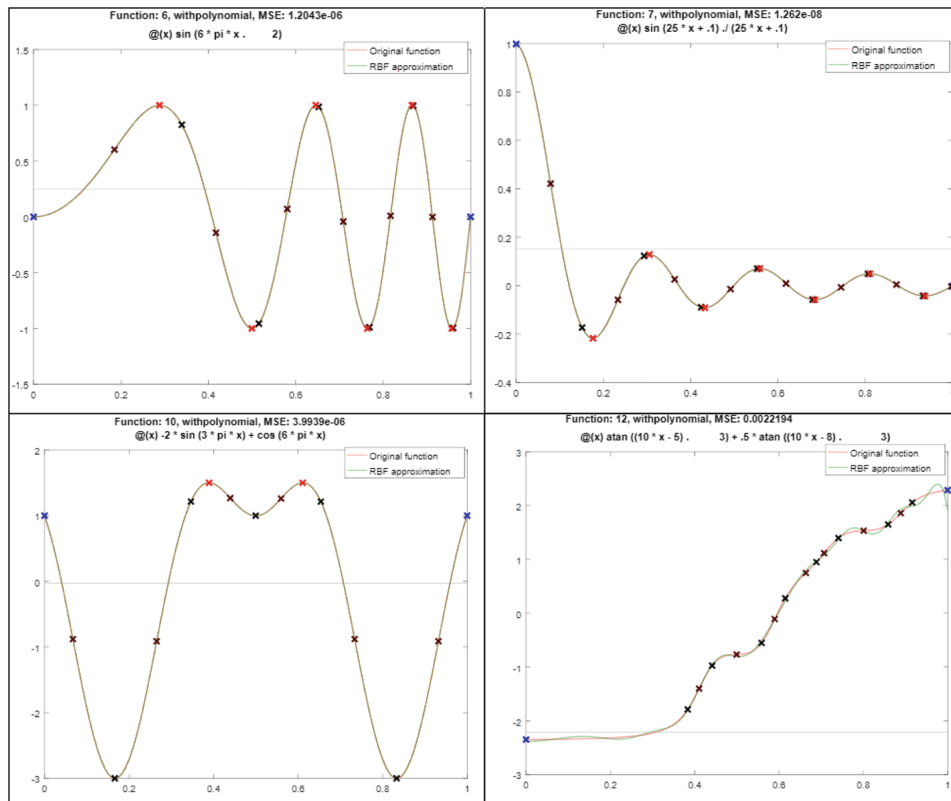


Fig. 3. Examples of approximation for selected functions.

The experiments have also proven, that for large data and data with a large span of data a polynomial $P_k(x)$ should be $P_k(x) = a_0$, i.e. $k = 0$, see [16, 17].

Selected results of the approximation of some functions are presented at Fig. 3. It can be seen, that the proposed approximation based actually on RBF interpolation scheme using points of importance offers good precision of approximation with good compression ratio. The functions were sampled in 200 points approx. and 10–20 points are actually used for the proposed approximation method.

5 Conclusion

This contribution briefly describes a method for efficient RBF approximation of large scattered data based on finding points of importance. This leads to a simple RBF based approximation of data with relatively low error with high compression. The precision of approximation can be increased significantly by covering some additional points. The approach is easily extensible to the 3D case, especially if data are ordered. However, if data are scattered, the neighbor points must be evaluated to find points of importance.

Experiments proved relatively high precision of approximation based on RBF interpolation using found points of importance leading to high data compression as well.

In future, deep analysis of an approximation behavior at the interval borders is expected as it is a critical issue for the 3D case, i.e. $z = f(x, y)$, as the first already made experiments shown. Also, the discrete points of curves of inflection are to be taken into account, i.e. discrete points of implicit curves $F(x, y) = 0$.

Acknowledgments. The authors would like to thank their colleagues and students at the University of West Bohemia and Universiti Teknologi PETRONAS for their discussions and suggestions; especially to Michal Smolik, Zuzana Majdisova and Jakub Vasta from the University of West Bohemia. Thanks belong also to anonymous reviewers for their valuable comments and hints provided.

This research was supported by the Czech Science Foundation (GACR) project GA 17-05534S and partially by SGS 2019-016.

References

1. Biancolini, M.E.: Fast Radial Basis Functions for Engineering Applications. Springer, Cham (2017). <https://doi.org/10.1007/978-3-319-75011-8>
2. Buhmann, M.D.: Radial Basis Functions: Theory and Implementations. Cambridge University Press, Cambridge (2008)
3. Cervenka, M., Smolik, M., Skala, V.: A new strategy for scattered data approximation using radial basis functions respecting points of inflection. In: Misra, S., et al. (eds.) ICCSA 2019. LNCS, vol. 11619, pp. 322–336. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-24289-3_24

4. Duchon, J.: Splines minimizing rotation-invariant semi-norms in Sobolev space. In: Schempp, W., Zeller, K. (eds.) *Constructive Theory of Functions of Several Variables*. LNCS, vol. 571. Springer, Heidelberg (1997). <https://doi.org/10.1007/BFb0086566>
5. Hardy, L.R.: Multiquadric equation of topography and other irregular surfaces. *J. Geophys. Res.* **76**(8), 1905–1915 (1971)
6. Fasshauer, G.E.: *Meshfree Approximation Methods with MATLAB*. World Scientific Publishing, Singapore (2007)
7. Karim, S.A.A., Saaban, A., Skala, V.: Range-restricted interpolation using rational bi-cubic spline functions with 12 parameters. **7**, 104992–105006 (2019). ISSN: 2169-3536. <https://doi.org/10.1109/access.2019.2931454>
8. Majdisova, Z., Skala, V.: A new radial basis function approximation with reproduction. In: *CGVCVIP 2016*, Portugal, pp. 215–222 (2016). ISBN 978-989-8533-52-4
9. Majdisova, Z., Skala, V.: Radial basis function approximations: comparison and applications. *Appl. Math. Model.* **51**, 728–743 (2017). <https://doi.org/10.1016/j.apm.2017.07.033>
10. Majdisova, Z., Skala, V.: Big geo data surface approximation using radial basis functions: a comparative study. *Comput. Geosci.* **109**, 51–58 (2017). <https://doi.org/10.1016/j.cageo.2017.08.007>
11. Majdisova, Z., Skala, V., Smolik, M.: Determination of stationary points and their bindings in dataset using RBF methods. In: Silhavy, R., Silhavy, P., Prokopova, Z. (eds.) *CoMeSySo 2018*. AISC, vol. 859, pp. 213–224. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-00211-4_20
12. Majdisova, Z., Skala, V., Smolik, M.: Determination of reference points and variable shape parameter for RBF approximation. *Integr. Comput.-Aided Eng.* **27**(1), 1–15 (2020). <https://doi.org/10.3233/ICA-190610>. ISSN 1069-2509
13. Pan, R., Skala, V.: A two level approach to implicit modeling with compactly supported radial basis functions. *Eng. Comput.* **27**(3), 299–307 (2011). <https://doi.org/10.1007/s00366-010-0199-1>. ISSN 0177-0667
14. Pan, R., Skala, V.: Surface reconstruction with higher-order smoothness. *Vis. Comput.* **28**(2), 155–162 (2012). <https://doi.org/10.1007/s00371-011-0604-9>. ISSN 0178-2789
15. Ohtake, Y., Belyaev, A., Seidel, H.-P.: A multi-scale approach to 3D scattered data interpolation with compactly supported basis functions. In: *Shape Modeling*, pp. 153–161. IEEE, Washington (2003). <https://doi.org/10.1109/smi.2003.1199611>
16. Skala, V.: RBF interpolation with CSRBF of large data sets, ICCS 2017. *Procedia Comput. Sci.* **108**, 2433–2437 (2017). <https://doi.org/10.1016/j.procs.2017.05.081>
17. Skala, V.: RBF interpolation and approximation of large span data sets. In: *MCSI 2017 – Corfu*, pp. 212–218. IEEE (2018). <https://doi.org/10.1109/mcsi.2017.44>
18. Skala, V., Karim, S.A.A., Kadir, E.A.: Scientific computing and computer graphics with GPU: application of projective geometry and principle of duality. *Int. J. Math. Comput. Sci.* **15**(3), 769–777 (2020). ISSN 1814-0432
19. Skala, V.: High dimensional and large span data least square error: numerical stability and conditionality. *Int. J. Appl. Phys. Math.* **7**(3), 148–156 (2017). <https://doi.org/10.17706/ijapm.2017.7.3.148-156>. ISSN 2010-362X
20. Smolik, M., Skala, V.: Large scattered data interpolation with radial basis functions and space subdivision. *Integr. Comput.-Aided Eng.* **25**(1), 49–62 (2018). <https://doi.org/10.3233/ica-170556>
21. Uhlir, K., Skala, V.: Reconstruction of damaged images using radial basis functions. In: *EUSIPCO 2005 Conference Proceedings*, Turkey (2005). ISBN 975-00188-0-X
22. Wenland, H.: *Scattered Data Approximation*. Cambridge University Press (2010). <http://doi.org/10.1017/CBO9780511617539>

250 V. Skala et al.

23. Wright, G.B.: Radial basis function interpolation: numerical and analytical developments. Ph.D. thesis, University of Colorado, Boulder (2003)
24. Skala, V., Karim, S.A.A., Zabran, M.: Radial basis function approximation optimal shape parameters estimation: preliminary experimental results. In: ICCS 2020 Conference (2020)
25. Vasta, J., Skala, V., Smolik, M., Cervenka, M.: Modified radial basis functions approximation respecting data local features. In: Informatics 2019, IEEE Proceedings, Poprad, Slovakia, pp. 445–449 (2019). ISBN 978-1-7281-3178-8
26. Cervenka, M., Skala, V.: Conditionality analysis of the radial basis function matrix. In: International Conference on Computational Science and Applications ICCSA (2020)
27. Smolik, M., Skala, V.: Efficient speed-up of radial basis functions approximation and interpolation formula evaluation. In: International Conference on Computational Science and Applications ICCSA (2020)

9.7 Conditionality Analysis of the Radial Basis Function Matrix

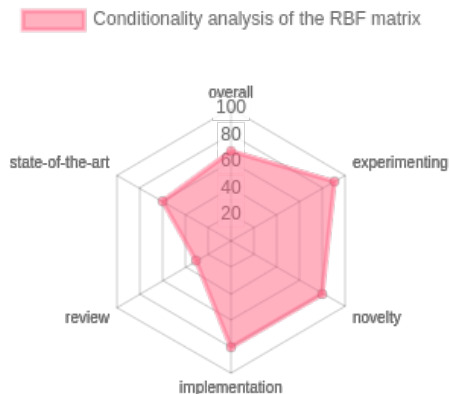
The conditionality analysis [91] paper is a more in-depth study of a Gaussian RBF. The goal was to take the knowledge from the previous paper and take the approach to the higher (2D) dimensions to ensure there are no problems with the approach. Also, the uniform centre point distribution was tested thoroughly.

The research aimed to figure out the most suitable shape parameter, and the testing scenario was simplified for that purpose. The testing scenario involves a $\langle 0, 1 \rangle \times \langle 0, 1 \rangle$ domain, where variable number of RBF centres N was put uniformly with a variable (global) shape parameter β .

During the research, two significant outcomes emerged. The first outcome is that if the uniform distribution is used, some shape parameters lead to an ill-conditional linear equation system. We were also able to find those experimentally and analytically. The second one is that the uniform distribution is unsuitable for the RBF approximation because the resulting linear equation system is ill-conditioned, in contrast to using some pseudorandom, e.g., Halton distribution.

Also, previously less discussed TPS RBF was rigorously tested. In the case of this RBF, a shape parameter also exists, which leads to ill-conditionality; however, there is only one for each number of RBFs tested.

This research proved the facts from the previous ones, that the pseudorandom distribution on points (where no other viable placement option exists) is better than the uniform one, considering the conditionality of the RBF equation system to solve. Due to that fact, further research did not consider the uniform distribution for placement at all.



Publication [91]:

CERVENKA, M.; SKALA, V. Conditionality Analysis of the Radial Basis Function Matrix. *ICCSA 2020 proceedings, part II, LNCS*. 2020, pp. 30–43. Available from DOI: https://doi.org/10.1007/978-3-030-58802-1_3. UT WoS: 000719685200003, EID: 2-s2.0-85093112881, OBD: 43932697



Conditionality Analysis of the Radial Basis Function Matrix

Martin Červenka^() and Václav Skala^()

Faculty of Applied Sciences, University of West Bohemia, Pilsen, Czechia
 {cervemar, skala}@kiv.zcu.cz

Abstract. The global Radial Basis Functions (RBFs) may lead to ill-conditioned system of linear equations. This contribution analyzes conditionality of the Gauss and the Thin Plate Spline (TPS) functions. Experiments made proved dependency between the shape parameter and number of RBF center points where the matrix is ill-conditioned. The dependency can be further described as an analytical function.

Keywords: Radial basis function · System of linear equations · Condition number · Matrix conditionality

1 Introduction

Interpolation and approximation of scattered data is a common problem in many engineering and research areas, e.g. Oliver et al. [1] use interpolation (kriging) method on geographical data, Kaymaz [2] finds usage of this technique in structural reliability problem. Sakata et al. [3] model wing structure with an approximation method, Joseph et al. [4] even create metamodels. The RBF methods are also used in the solution of partial differential equations (PDE) especially in connection with engineering problems.

To solve interpolation and approximation problems, we use two main approaches:

- Tessellated approach – it requires tessellation of the data domain (e.g. Delaunay triangulation) to generate associations between pairs of points in the tessellated cloud of points. Some algorithms were developed (Lee et al. [5] show two of them, Smolik et al. [6] show a fast parallel algorithm for triangulation of large datasets, Zeu et al. [7] recently use tessellation for seismic data etc.) for triangulation and tessellation. Even though it seems simple, tessellation is a slow process in general¹.

¹ The Delaunay triangulation has time complexity of $O(n^{\lceil d/2 \rceil + 1})$, where d is number of tessellated dimensions.

The research was supported by projects Czech Science Foundation (GACR) No. 17-05534S and partially by SGS 2019-016.

© Springer Nature Switzerland AG 2020
 O. Gervasi et al. (Eds.): ICCSA 2020, LNCS 12250, pp. 30–43, 2020.
https://doi.org/10.1007/978-3-030-58802-1_3

- Meshless approach – a method based on RBFs can be used, which does not require any form of tessellation. Hardy [8] shown that the complexity of this approach is nearly independent to the problem dimensionality, therefore it is a better alternative to tessellation in higher dimensions. On the other hand, RBF methods require solving a system of linear equations which leads to some problems as well.

There are several meshless approaches e.g. Fasshauer [9] implements some of the meshless algorithms in MATLAB, Franke [10] compares some interpolation methods of the scattered data.

Conditionality of the matrix of a linear system of equation is a key element to determine whether the system is well solvable or not.

RBF research was recently targeted:

- to find out RBF applicability for large geosciences data, see Majdisova [11],
- to interpolate and approximate vector data, see Smolik [12],
- to study robustness of the RBF data for large datasets, see Skala [13,14].
- to find out optimal variable shape parameters, see Skala [15].

This research is aimed to find optimal (or at least suboptimal) shape parameters of the RBF interpolation. This contribution describes briefly analysis of some of the most commonly used RBFs and determines its problematic shape parameters, causing ill-conditionality of the equation system matrix.

2 RBF Approximation and Interpolation

The basic idea behind the RBF approach is the partial unity approach, i.e. summing multiple weighted radial basis functions together to obtain complex interpolating function. The Fig. 1 presents two RBFs (marked by red color) forming an interpolating final function (blue one).

The RBF approach was introduced by Hardy [8] and modified in [16]. Since then, this method has been further developed and modified. Majdisova et al. [17] and Cervenka et al. [18] proposed multiple placement methods. There are also some behavioural studies of the shape parameters, e.g. searching the optimal ones from Wang et al. [19], Afatdoust et al. [20] or using different local shape parameters from Cohen et al. [21], Sarra et al. [22], Skala et al. [15].

This contribution analyzes the worst cases of the RBF matrix conditionality in order to avoid bad shape parameters, therefore the bad shape parameters can be avoided.

2.1 RBF Method Principle

The RBF interpolation is defined by Eq. 1,

$$h(\mathbf{x}_i) = \sum_{j=1}^N \lambda_j \varphi(\|\mathbf{x}_i - \mathbf{x}_j\|) = \sum_{j=1}^N \lambda_j \varphi(r_{ij}) \quad (1)$$

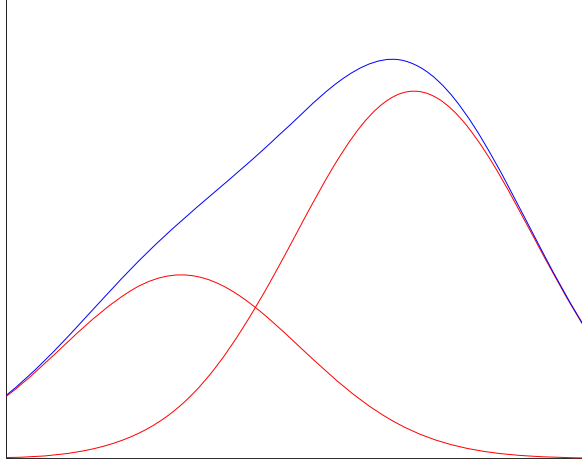


Fig. 1. Two RBFs (in red) and result of the addition (in blue). (Color figure online)

where $h(\mathbf{x}_i)$ is the resulting interpolant, N is the number of RBFs, λ_i is a weight of the i -th RBF, φ is the selected RBF and r_{ij} is a distance between points \mathbf{x}_i and \mathbf{x}_j . The points \mathbf{x}_j are all the points on the sampled original function, where the function value is known.

The RBF approximation is slightly different, see Eq. 2. The notation is the same as above, however, \mathbf{x}_j are replaced by reference points $\xi_j, j = 1, \dots, M$. Some arbitrary (sufficiently small $M \ll N$) number of points from the data domain are taken instead. More details can be found in Skala [23].

$$h(\mathbf{x}_i) = \sum_{j=1}^M \lambda_j \varphi(\|\mathbf{x}_i - \xi_j\|), \quad i = 1, \dots, N \tag{2}$$

In both cases, i.e. approximation and interpolation, the equations can be expressed in a matrix form as:

$$\mathbf{A}\boldsymbol{\lambda} = \mathbf{b}, \quad \mathbf{b} = h(\mathbf{x}), \mathbf{A}_{ij} = \varphi_{ij} \tag{3}$$

In the interpolation case, the matrix \mathbf{A} is a square matrix, while in the approximation case, the matrix \mathbf{A} is rectangular and the result is an overdetermined system of linear equations. In this case, we do not obtain exact values for the already calculated reference points ξ_j .

2.2 RBF Classification

There are many RBFs and still new ones are being proposed e.g. Menandro [24]. In general, we can divide the RBFs into two main groups, “global” and “local” ones, see Fig. 3 and Fig. 2.

- **Global** RBFs influence the interpolated values globally. The matrix \mathbf{A} will be dense and rather ill-conditioned. Typical examples of the global RBF are the Gaussian, the TPS or the inverse multiquadric RBFs.
- **Local** RBFs have limited influence to a limited space near its centre point (hypersphere, in general). The advantage of the local RBFs is that they lead to a sparse matrix \mathbf{A} . RBFs belonging to this group are called “Compactly Supported” RBFs (CS-RBFs, in short).

Global RBFs are functions, which influence is not limited and its value may be nonzero for each value in its domain. The well-known ones are the Gaussian or the TPS functions. However, there are other functions, see e.g. Table 1 or Lin et al. [25]. Mentioned functions are illustrated in Fig. 2.

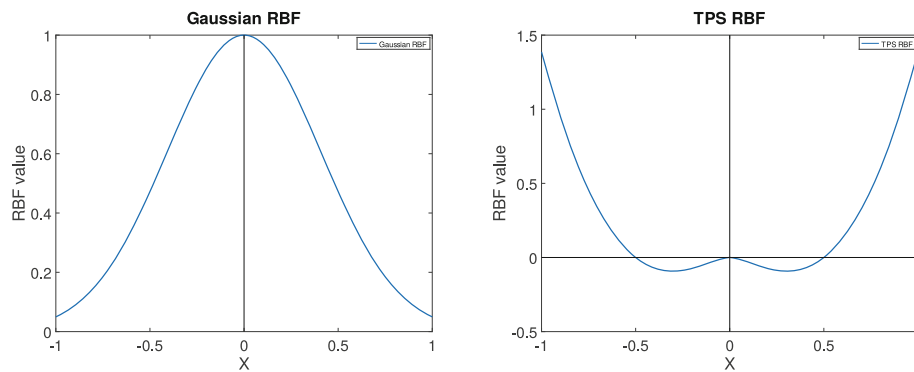


Fig. 2. Some of the global RBF functions.

Table 1. Various global RBF functions.

Name	Expression
Gaussian RBF	$e^{-\alpha r^2}$
TPS RBF	$\frac{1}{2} r^2 \log(\beta r^2)$
Multiquadric RBF	$\frac{1}{1+(\epsilon r)^2}$
Inverse Multiquadric RBF	$\frac{1}{\sqrt{1+(\epsilon r)^2}}$

The CS-RBF or compactly supported radial basis function is a function limited to a given interval. Some of CS-RBFs are presented on Fig. 3. Generally, these functions are limited to an interval (usually $r \in (0, 1)$) otherwise the value equals zero. These functions are defined by Eq. 4, where $P(r)$ is a polynomial function, r is the distance of two points and q is a parameter.

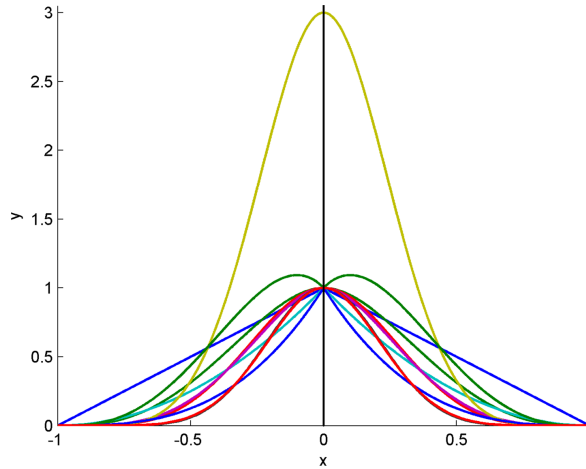


Fig. 3. Some of the CS-RBF functions. [26] (edited)

$$\varphi(r) = \begin{cases} (1-r)^q P(r) & 0 \leq r < 1, \\ 0 & r \geq 1 \end{cases} \quad (4)$$

It should be noted that some new CS-RBFs have been recently defined by Menandro [24].

3 Matrix Conditionality

Assuming a linear system of equations $\mathbf{Ax} = \mathbf{b}$, the condition number of the matrix \mathbf{A} describes how the result (vector \mathbf{b}) can change when the input vector \mathbf{x} is slightly modified. This number describes sensitivity to changes in the input vector. We aim for the lowest possible sensitivity, in order to get reasonable results. In terms of linear algebra, we can define conditionality of a normal matrix \mathbf{A} using eigenvalues $\lambda_i \in \mathbb{C}^1$ as:

$$\kappa(\mathbf{A}) = \frac{|\lambda_{max}(\mathbf{A})|}{|\lambda_{min}(\mathbf{A})|} \quad (5)$$

where $\kappa(\mathbf{A})$ is the condition number of the normal matrix \mathbf{A} , $|\lambda_{max}(\mathbf{A})|$ is the highest absolute eigenvalue of the matrix \mathbf{A} and $|\lambda_{min}(\mathbf{A})|$ is the lowest absolute eigenvalue of the matrix.

The higher the value $\kappa(\mathbf{A})$ is, the more sensitive the matrix \mathbf{A} is, meaning that $\kappa(\mathbf{A}) = 1$ is the best option, forcing all eigenvalues λ to have the same value.

It is worth noting that the conditionality is closely related to the matrix determinant. In the case when the determinant is zero, we have at least one eigenvalue equaling zero, so the conditionality will be infinite, see Eq. 6.

$$\det(\mathbf{A}) = 0 \rightarrow |\lambda_{min}(\mathbf{A})| = 0 \rightarrow \kappa(\mathbf{A}) = +\infty \Leftrightarrow |\lambda_{max}(\mathbf{A})| \neq 0 \quad (6)$$

This is only a brief introduction to the matrix conditionality. Details can be found in e.g. Ikramov [27] or Skala [14], some experimental results can be found in Skala [28].

4 Experimental Results of RBF Approximation

In the RBF approximation problem, we normally have two main issues to deal with – selecting number of RBFs and its global shape parameter. To obtain a robust solution, the matrix \mathbf{A} of the linear system of equations should not be ill-conditioned. We did some experiments to show how the condition number of the matrix \mathbf{A} depends on the number of RBFs (N) used and a shape parameter (α or β , see below). To make things easier, all RBFs have been distributed uniformly on $x \in \langle 0, 1 \rangle$ interval and have the same constant shape parameter.

4.1 Gaussian RBF

The Gaussian RBF is defined by Eq. 7. It is the unnormalized probability density function of a Gaussian distribution centred at zero and with a variance of $\frac{1}{2\alpha}$. Variable r denotes the distance from its centre points and α is the shape parameter.

$$\varphi(r, \alpha) = e^{-\alpha r^2} \quad (7)$$

Figure 4 presents dependence of matrix conditionality on Gaussian RBF shape parameter α and number of uniformly distributed RBF reference points.

A hyperbolic function (Eq. 8) was used to fit extremal points of each curve (Table 2).

Table 2. Analytical form of first 9 hyperboles.

Hyperbole	a	b	c	Hyperbole	a	b	c
1	7.64	38.36	-3.58	6	8.47	1387.35	-30.84
2	13.49	1.93	-7.98	7	17.98	1218.46	-49.14
3	9.17	277.29	-11.95	8	49.16	278.29	-78.53
4	9.44	509.55	-18.37	9	93.81	63.73	16.11
5	12.02	545.66	-31.8				

$$\beta = a + \frac{b}{N + c} \quad (8)$$

The plot at Fig. 5 describes the situation. These curves describe number of RBFs N and shape parameter α when the matrix is ill-conditioned.

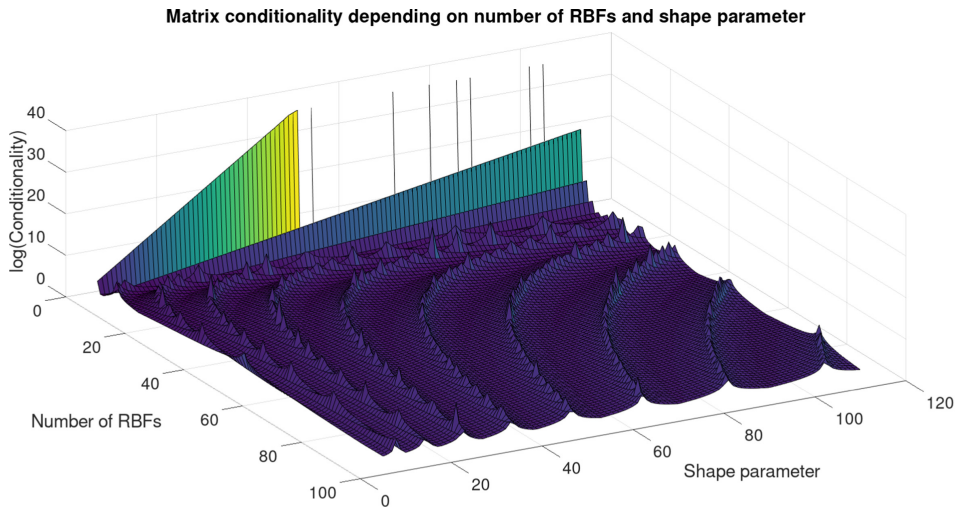


Fig. 4. Matrix conditionality values for Gaussian RBF.

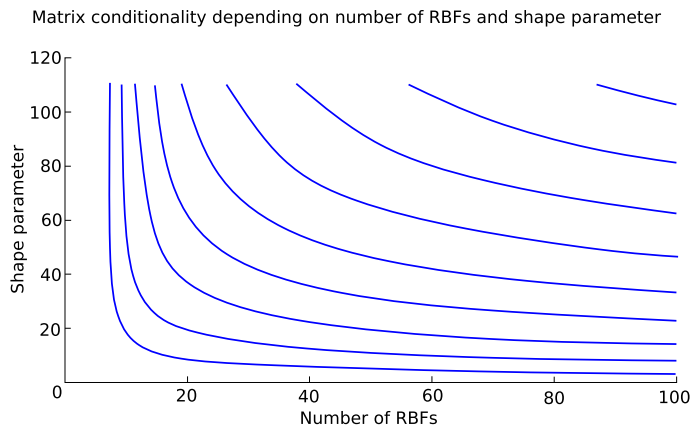


Fig. 5. Worst conditionality shape parameters α for Gauss RBF.

4.2 Thin Plate Spline RBF

The Thin Plate Spline (TPS) radial basis function is defined by the Eq. 9. The TPS was introduced by Duchon [29] and used for RBF approximation afterwards. Variable r is the same as in the Gaussian RBF – the distance from its centre point and parameter β is the shape parameter.

$$\varphi(r, \beta) = \frac{1}{2} r^2 \log(\beta r^2) \tag{9}$$

The Fig. 6 presents a result for a simulated experiment to the recent Gaussian RBF case using the TPS function instead. There is only one curve which has a hyperbolic shape similar to the Gaussian RBF case.

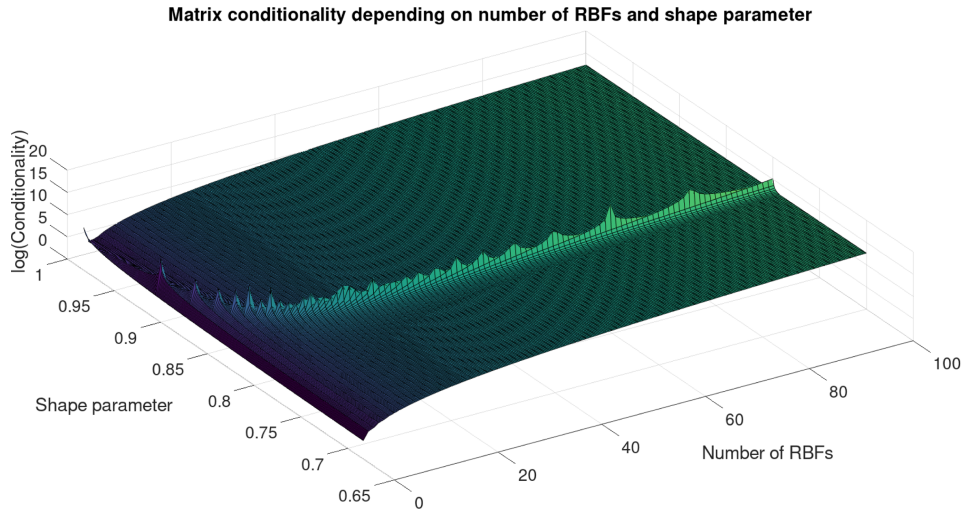


Fig. 6. Matrix conditionality values for TPS RBFs.

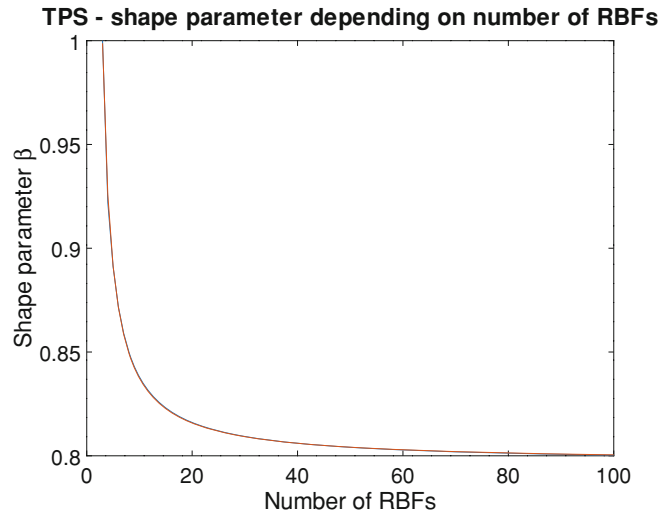


Fig. 7. Worst conditionality shape parameters β for the TPS RBF.

The Fig. 7 also represents the curve, when the matrix \mathbf{A} is close to singular. The Table 3 presents dependency of the β_{exp} shape parameter for different N as a function when the matrix \mathbf{A} is significantly ill-conditioned.

We obtained a hyperbolic function from the graph on Fig. 7 (coefficients are rounded to 2 decimal places).

$$\beta = 0.79 + \frac{0.36}{N - 1.24} \quad (10)$$

The Table 3 presents the shape parameters β_{calc} evaluated for small numbers of RBF functions according to Eq. 10.

The experimental results presented above led to a question, how the results are related from the analytical side. This led to the validation of experiments with two analytical results described in this section.

5 Theoretical Analysis

Let us calculate values of the TPS shape parameter β for $N = 3$ and $N = 4$ in a way that the matrix \mathbf{A} will be ill-conditioned ($\kappa(\mathbf{A}) = +\infty$).

It should be noted that the multiplicative constant $\frac{1}{2}$ is omitted in the Eq. 11 as it has no influence to the conditionality evaluation. In the first case, i.e. $N = 3$, the RBF matrix \mathbf{A} has the form (using equidistant distribution of RBF center points):

$$\mathbf{A}_3 = \begin{bmatrix} 0 & r^2 \log(\beta r^2) & (2r)^2 \log(\beta 4r^2) \\ r^2 \log(\beta r^2) & 0 & r^2 \log(\beta r^2) \\ (2r)^2 \log(\beta 4r^2) & r^2 \log(\beta r^2) & 0 \end{bmatrix} \quad (11)$$

Let us explore singularity of the matrix \mathbf{A}_3 , when $\det(\mathbf{A}_3) = 0$, the determinant will have the form:

$$r^6 \begin{vmatrix} 0 & \log(\beta r^2) & 4 \log(\beta 4r^2) \\ \log(\beta r^2) & 0 & \log(\beta r^2) \\ 4 \log(\beta 4r^2) & \log(\beta r^2) & 0 \end{vmatrix} = 0 \quad (12)$$

As $r \neq 0$ for all pairs of different points, $\lim_{r \rightarrow 0} r^2 \log(r^2) = 0$ and equidistant point distribution.

For the sake of simplicity, we substitute $q = \log(\beta r^2)$, $a = \log 4$ and use formula $\log(ab) = \log a + \log b$ so we get:

$$\begin{aligned} & \begin{vmatrix} 0 & q & 4(q+a) \\ q & 0 & q \\ 4(q+a) & q & 0 \end{vmatrix} = 0 \\ & 8(q+a)q^2 = 0 \rightarrow q = 0 \vee q = -a \\ & \log(\beta r^2) = -\log 4 = \log \frac{1}{4} \\ & \beta r^2 = \frac{1}{4} \\ & \beta = \frac{1}{4r^2} \end{aligned} \quad (13)$$

In the experiments, we used interval $x \in \langle 0, 1 \rangle$ and with three points $(0, 0.5, 1)$. The distance between two consecutive points r is 0.5, which led to $\beta = 1$. This exact value we obtained from experiments as well (see Table 3).

Table 3. β_{exp} -values for TPS RBF for some small N (number of RBFs) obtained by experiment as well as β_{calc} values calculated by Eq. 10

N	β_{exp}	β_{calc}	N	β_{exp}	β_{calc}	N	β_{exp}	β_{calc}
3	1.00000	0.99874	23	0.81338	0.81319	43	0.80535	0.80536
4	0.92206	0.92564	24	0.81264	0.81247	44	0.80515	0.80516
5	0.89118	0.89141	25	0.81197	0.81182	45	0.80496	0.80497
6	0.87182	0.87155	26	0.81135	0.81121	46	0.80477	0.80479
7	0.85909	0.85858	27	0.81078	0.81065	47	0.80459	0.80462
8	0.85002	0.84945	28	0.81025	0.81014	48	0.80442	0.80445
9	0.84324	0.84268	29	0.80976	0.80966	49	0.80426	0.80429
10	0.83799	0.83744	30	0.80930	0.80921	50	0.80410	0.80414
11	0.83379	0.83329	31	0.80888	0.80880	51	0.80395	0.80399
12	0.83037	0.82990	32	0.80848	0.80841	52	0.80380	0.80385
13	0.82753	0.82709	33	0.80811	0.80804	53	0.80366	0.80372
14	0.82512	0.82472	34	0.80776	0.80770	54	0.80353	0.80359
15	0.82306	0.82269	35	0.80743	0.80738	55	0.80340	0.80346
16	0.82128	0.82094	36	0.80711	0.80708	56	0.80328	0.80334
17	0.81973	0.81941	37	0.80682	0.80679	57	0.80316	0.80322
18	0.81835	0.81807	38	0.80654	0.80652	58	0.80304	0.80311
19	0.81713	0.81687	39	0.80628	0.80626	59	0.80293	0.80300
20	0.81605	0.81581	40	0.80603	0.80602	60	0.80282	0.80290
21	0.81507	0.81485	41	0.80579	0.80579	61	0.80272	0.80280
22	0.81418	0.81398	42	0.80557	0.80557	62	0.80262	0.80270

In the second case, i.e. $N = 4$, a similar approach has been taken. In this case the matrix \mathbf{A}_4 is defined as:

$$\mathbf{A}_4 = \begin{bmatrix} 0 & r^2 \log(\beta r^2) & (2r)^2 \log(\beta 4r^2) & (3r)^2 \log(\beta 9r^2) \\ r^2 \log(\beta r^2) & 0 & r^2 \log(\beta r^2) & (2r)^2 \log(\beta 4r^2) \\ (2r)^2 \log(\beta 4r^2) & r^2 \log(\beta r^2) & 0 & r^2 \log(\beta r^2) \\ (3r)^2 \log(\beta 9r^2) & (2r)^2 \log(\beta 4r^2) & r^2 \log(\beta r^2) & 0 \end{bmatrix} \quad (14)$$

Similarly as in the case for $N = 3$, we can write the $\det(\mathbf{A}_4)$ and declare the matrix singular if:

$$r^8 \begin{vmatrix} 0 & \log(\beta r^2) & 4 \log(\beta 4r^2) & 9 \log(\beta 9r^2) \\ \log(\beta r^2) & 0 & \log(\beta r^2) & 4 \log(\beta 4r^2) \\ 4 \log(\beta 4r^2) & \log(\beta r^2) & 0 & \log(\beta r^2) \\ 9 \log(\beta 9r^2) & 4 \log(\beta 4r^2) & \log(\beta r^2) & 0 \end{vmatrix} = 0 \quad (15)$$

40 M. Červenka and V. Skala

Using the substitutions $q = \log(\beta r^2)$, $a = \log 4$ and $b = \log 9$, we obtain:

$$\begin{vmatrix} 0 & q & 4(q+a) & 9(q+b) \\ q & 0 & q & 4(q+a) \\ 4(q+a) & q & 0 & q \\ 9(q+b) & 4(q+a) & q & 0 \end{vmatrix} \quad (16)$$

This can be further expressed as:

$$\begin{aligned} & (4(q+a))^4 + q^4 + q^2(9(q+b))^2 \\ & = -2q^3(9(q+b)) - 2q(4(q+a))^2(9(q+b)) - 2q^2(4(q+a))^2 \\ & = 256(q+a)^4 + q^4 + 81q^2(q+b)^2 - 18q^3(q+b) \\ & \quad - 288q(q+a)(q+b)^2 - 32q^2(q+a)^2 \end{aligned} \quad (17)$$

This leads to the cubic equation:

$$\begin{aligned} & (383a - 144b)q^3 + (1216a^2 + 81b^2 - 576ab)q^2 \\ & \quad + (1024a^3 - 288a^2b)q + 256a^4 = 0 \end{aligned} \quad (18)$$

Solving this cubic equation (Eq. 18), one real and two complex (complex conjugate) roots are obtained:

$$\begin{aligned} q_1 & \approx -2.2784 \\ q_2 & \approx -1.1149 + 0.8239i \\ q_3 & \approx -1.1149 - 0.8239i \end{aligned} \quad (19)$$

As we have four points distributed uniformly on the interval $x \in (0, 1)$, the distance between two adjacent nodes is $r = \frac{1}{3}$. Now, using the real root of the Eq. 19, i.e. $q = -2.2784$, we can estimate the shape parameter β as follows:

$$\begin{aligned} q & = \log(\beta r^2) \approx -2.2784 \\ \beta r^2 & \approx e^{-2.2784} \approx 0.10245 \\ \beta & \approx \frac{e^{-2.2784}}{r^2} \\ \beta & \approx \frac{e^{-2.2784}}{\left(\frac{1}{3}\right)^2} = 9e^{-2.2784} \approx 0.92206 \end{aligned} \quad (20)$$

From the experiments, we obtained value $\hat{\beta} = 0.92206$ which is consistent with this theoretical estimation. Both these analytical examples support the argument that the experiments made are correct.

It should be noted, that if irregular point distribution is used, i.e. using Halton points distributions, the ill-conditionality get slightly worse.

6 Conclusion

In this paper, we discussed some properties of the two well-known RBFs. We find out that there are some regularities in the shape parameters, where the RBF matrix is ill-conditioned. Our experiments proved that there are no global optimal shape parameters from the RBF matrix conditionality point of view.

In the future, the RBF conditionality problem is to be explored for higher dimension, especially for $d = 2$, $d = 3$ and in the context of partial differential equations.

Acknowledgement. The authors would like to thank their colleagues and students at the University of West Bohemia for their discussions and suggestions, and especially to Michal Smolik for valuable discussion and notes he provided. The research was supported by projects Czech Science Foundation (GACR) No. 17-05534S and partially by SGS 2019-016.

References

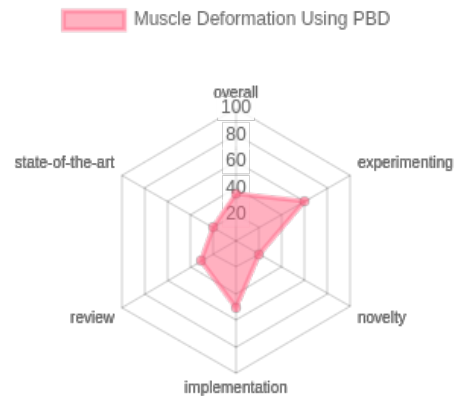
1. Oliver, M.A., Webster, R.: Kriging: a method of interpolation for geographical information systems. *Int. J. Geograph. Inf. Syst.* **4**(3), 313–332 (1990). <https://doi.org/10.1080/02693799008941549>
2. Kaymaz, I.: Application of kriging method to structural reliability problems. *Struct. Saf.* **27**(2), 133–151 (2005). <https://doi.org/10.1016/j.strusafe.2004.09.001>
3. Sakata, S., Ashida, F., Zako, M.: An efficient algorithm for kriging approximation and optimization with large-scale sampling data. *Comput. Meth. Appl. Mech. Eng.* **193**(3–5), 385–404 (2004). <https://doi.org/10.1016/j.cma.2003.10.006>
4. Joseph, V.R., Hung, Y., Sudjianto, A.: Blind kriging: a new method for developing metamodells. *J. Mech. Des.* **130**(3), 031102 (2008). <https://doi.org/10.1115/1.2829873>
5. Lee, D.T., Schachter, B.J.: Two algorithms for constructing a Delaunay triangulation. *Int. J. Comput. Inf. Sci.* **9**(3), 219–242 (1980). <https://doi.org/10.1007/BF00977785>
6. Smolik, M., Skala, V.: Fast parallel triangulation algorithm of large data sets in E^2 and E^3 for in-core and out-core memory processing. In: Murgante, B., et al. (eds.) ICCSA 2014. LNCS, vol. 8580, pp. 301–314. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-09129-7_23
7. Zeu, Y., Youngseok, S., Joongmoo, B., Soon-Jee, S., Ki-Young, K.: Regularisation of multidimensional sparse seismic data using Delaunay tessellation. *J. Appl. Geophys.* (2019). <https://doi.org/10.1016/j.jappgeo.2019.103877>
8. Hardy, R.L.: Multiquadric equations of topography and other irregular surfaces. *J. Geophys. Res.* **76**, 1905–1915 (1971). <https://doi.org/10.1029/JB076i008p01905>
9. Fasshauer, G.E.: Meshfree Approximation Methods with MATLAB, vol 6. World Scientific (2007). <https://doi.org/10.1142/6437>
10. Franke, R.: A critical comparison of some methods for interpolation of scattered data. Technical report, Naval Postgraduate School Monterey CA (1979)
11. Majdisova, Z., Skala, V.: Big geo data surface approximation using radial basis functions: a comparative study. *Comput. Geosci.* **109**, 51–58 (2017). <https://doi.org/j.cageo.2017.08.007>

12. Smolik, M., Skala, V., Majdisova, Z.: Vector field radial basis function approximation. *Adv. Eng. Softw.* **123**, 117–129 (2018). <https://doi.org/10.1016/j.advengsoft.2018.06.013>
13. Skala, V.: RBF interpolation with CSRBF of large data sets, ICCS. *Procedia Comput. Sci.* **108**, 2433–2437 (2017). <https://doi.org/10.1016/j.procs.2017.05.081>
14. Skala, V.: Conditionality of linear systems of equations and matrices using projective geometric algebra. In: Murgante, B., et al. (eds.) *ICCSA 2020, LNCS*, vol. 12250, pp. 3–17. Springer, Heidelberg (2020)
15. Skala, V., Karim, S.A.A., Zabran, M.: Radial basis function approximation optimal shape parameters estimation. In: Krzhizhanovskaya, V.V., et al. (eds.) *ICCS 2020. LNCS*, vol. 12142, pp. 309–317. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-50433-5_24
16. Hardy, R.L.: Theory and applications of the multiquadric-biharmonic method 20 years of discovery 1968–1988. *Comput. Math. Appl.* **19**(8–9), 163–208 (1990). [https://doi.org/10.1016/0898-1221\(90\)90272-L](https://doi.org/10.1016/0898-1221(90)90272-L)
17. Majdisova, Z., Skala, V.: Radial basis function approximations: comparison and applications. *Appl. Math. Model.* **51**, 728–743 (2017). <https://doi.org/10.1016/j.apm.2017.07.033>
18. Červenka, M., Smolik, M., Skala, V.: A new strategy for scattered data approximation using radial basis functions respecting points of inflection. In: Misra, S., et al. (eds.) *ICCSA 2019. LNCS*, vol. 11619, pp. 322–336. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-24289-3_24
19. Liu, G., Wang, J.: On the optimal shape parameters of radial basis functions used for 2-d meshless methods. *Comput. Meth. Appl. Mech. Eng.* **191**(23–24), 2611–2630 (2002). [https://doi.org/10.1016/S0045-7825\(01\)00419-4](https://doi.org/10.1016/S0045-7825(01)00419-4)
20. Afiatdoust, F., Esmailbeigi, M.: Optimal variable shape parameters using genetic algorithm for radial basis function approximation. *Shams Eng. J.* **6**(2), 639–647 (2015). <https://doi.org/10.1016/j.asej.2014.10.019>
21. Cohen-Steiner, D., Alliez, P., Desbrun, M.: Variational shape approximation. *ACM Trans. Graph. (ToG)* **23**(3), 905–914 (2004). <https://doi.org/10.1145/1015706.1015817>
22. Sarra, S.A., Sturgill, D.: A random variable shape parameter strategy for radial basis function approximation methods. *Eng. Anal. Boundary Elem.* **33**(11), 1239–1245 (2009). <https://doi.org/10.1016/j.enganabound.2009.07.003>
23. Skala, V.: Fast interpolation and approximation of scattered multidimensional and dynamic data using radial basis functions. *WSEAS Trans. Math.* **12**(5), 501–511 (2013). E-ISSN 2224–2880
24. Menandro, F.C.M.: Two new classes of compactly supported radial basis functions for approximation of discrete and continuous data. *Eng. Rep.* (2019). <https://doi.org/10.1002/eng2.12028>
25. Lin, J., Chen, W., Sze, K.Y.: A new radial basis function for Helmholtz problems. *Eng. Anal. Bound. Elem.* **36**, 1923–1930 (2012). <https://doi.org/10.1016/j.enganabound.2012.07.010>
26. Smolik, M., Skala, V.: Large scattered data interpolation with radial basis functions and space subdivision. *Integr. Comput. Aided Eng.* **25**(1), 49–62 (2018). <https://doi.org/10.3233/ICA-170556>
27. Ikramov, K.D.: Conditionality of the intermediate matrices of the Gauss, Jordan and optimal elimination methods. *USSR Comput. Math. Math. Phys.* **18**, 1–16 (1978). [https://doi.org/10.1016/0041-5553\(78\)90159-3](https://doi.org/10.1016/0041-5553(78)90159-3)

28. Skala, V.: High dimensional and large span data least square error: numerical stability and conditionality. *Int. J. Appl. Phys. Math.* **7**(3), 148–156 (2017). <https://doi.org/10.17706/ijapm.2017.7.3.148-156>
29. Duchon, J.: Splines minimizing rotation-invariant semi-norms in Sobolev spaces. In: Schempp, W., Zeller, K. (eds.) *Constructive Theory of Functions of Several Variables*, vol. 571, pp. 85–100. Springer, Heidelberg (1977). <https://doi.org/10.1007/BFb0086566>

9.8 Muscle Deformation Using Position Based Dynamics

In this article [71], we followed our recent research [65] on muscle deformation using a triangular mesh model of a musculoskeletal system, already described in Section 9.4. The approach emphasises the volume preservation constraint, evident in the results where the volume is well preserved with less than 1% error in all cases. This emphasis on volume preservation is a key improvement, ensuring more accurate and realistic muscle simulations. It also presents a detailed analysis of the average displacement of points in the more complex adductor brevis muscle during hip flexion with different numbers of PBD solver iterations. This analysis contributes to the muscle deformation dynamics, testing the previous method [65] more rigorously. Also, multiple issues were described:



1. The approach relies on detecting muscle points that should move with bones based on the information about attachment areas of the muscle. However, muscle attachment sites cannot be automatically extracted from medical images, and their manual specification by an expert is time-consuming.
2. The collision handling method is inaccurate, leading to sharp spikes on the surface of the muscle, especially when using a coarse voxel representation of bones. As the memory grows cubically, using a refined voxel representation becomes impractical.
3. The simulation results are susceptible to the parameter settings. Although the simulation runs in real-time, even with an unoptimized sequential algorithm, careful adjustment and calibration of parameters are required to achieve accurate results.

These disadvantages highlight the challenges in accurately simulating muscle deformation, particularly in efficiently handling collision and the complexity of setting up accurate muscle attachment points. The first issue (attachment area specification) was further discussed in our research in Section 9.11. The issues described in the second point were improved significantly (see Section 9.10). The third issue is at the time of writing a work in progress.

Publication [71]:

KOHOUT, J.; CERVENKA, M. Muscle Deformation Using Position Based Dynamics. *Ye X. et al. (eds) Biomedical Engineering Systems and Technologies. BIOSTEC 2020. Communications in Computer and Information Science.* 2021, vol. 1400. Available from DOI: https://doi.org/10.1007/978-3-030-72379-8_24. EID: 2-s2.0-85107281398, OBD: 43932927



Muscle Deformation Using Position Based Dynamics

Josef Kohout¹ and Martin Červenka²

¹ NTIS - New Technologies for the Information Society, Faculty of Applied Sciences, University of West Bohemia, Univerzitní 8, Plzeň, Czech Republic

`besoft@ntis.zcu.cz`

² Department of Computer Science and Engineering, Faculty of Applied Sciences, University of West Bohemia, Univerzitní 8, Plzeň, Czech Republic

`cervemar@kiv.zcu.cz`

Abstract. This paper describes an approach to personalized musculoskeletal modelling, in which the muscle represented by its triangular mesh is subject to deformation, based on a modified position-based dynamic (PBD) method, followed by decomposition of its volume into a set of muscle fibres. The PBD was enhanced by respecting some muscle-specific features, mainly its anisotropy. The proposed method builds no internal structures and works only with the muscle surface model. It runs in real-time on commodity hardware while maintaining visual plausibility of the resulting deformation. For decomposition, the state-of-the-art Kukačka method is used. Experiments with the gluteus maximus, gluteus medius, iliacus and adductor brevis deforming during the simulation of the hip flexion and decomposed into 100 fibres of 15 line segments show that the approach is capable of achieving promising results comparable with those in the literature, at least in the term of muscle fibre lengths.

Keywords: Position based dynamics · Musculoskeletal system · Muscle deformation · Muscle fibres · Personalised model

1 Introduction

For decades, musculoskeletal modelling has been an important topic of research interest because of its ability to estimate internal loading on the human skeleton, which cannot be measured *in-vivo*. These estimations are useful, e.g., for preoperative surgical planning and postoperative assessment in orthopaedic surgery, rehabilitation procedures, prosthesis design, or prevention of injuries in professional sport.

Musculoskeletal models used in common practice (see, e.g., [1, 2, 6, 8, 11]) represent a muscle (or even a group of muscles) as one or more Hill-type one-dimensional structures, commonly referred as lines of action or fibres, connecting

This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic, project SGS-2019-016 and project PUNTIS (LO1506).

© Springer Nature Switzerland AG 2021

X. Ye et al. (Eds.): BIOSTEC 2020, CCIS 1400, pp. 486–509, 2021.

https://doi.org/10.1007/978-3-030-72379-8_24

the origin and insertion points of the muscle, i.e., the sites at which the muscle is attached to the bone by a tendon, and passing through a couple of predefined via points, fixed to the underlying bone, or wrapping around predefined parametric objects (e.g. spheres, cylinders, or ellipsoids). Due to apparent difficulties with the specification of the locations of insertion, origin, and via points, it is common that there are no more than three fibres per muscle and they penetrate the bones in some poses. Figure 1 shows an example of models of this kind. An advantage of this approach, which makes it so popular, is its simplicity and rapid processing speed.

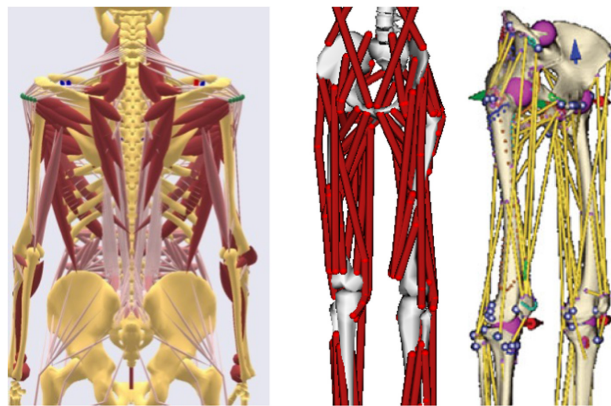


Fig. 1. Musculoskeletal models used in common practice: left – Anybody (<http://www.anybodytech.com/>) default model, middle – OpenSim (<https://simtk.org/home/opensim>) gait2392 model, right – LHDL model [22].

As acquiring complete patient-specific or subject-specific data is nearly impossible due to technological limitations of scanning devices, these musculoskeletal models have anatomical parameters derived from cadaver experiments. However, to answer specific subject-related questions, it is generally believed that a patient-specific or subject-specific model is needed. The current practice is, therefore, to take some of these generic models and adapt it to get a personalized model, which most typically consists of a non-uniform scaling (see, e.g., [38]) and a change of optimum fibre length.

Bolsterlee et al. [3] pointed out that many parameters in a model are inter-related. Adapting the model to the subject by scaling improves the anatomical resemblance between the model and the subject but may not improve force prediction. Unfortunately, it is not known how to adapt the other parameters. Several studies, e.g., [10,12,31], warn that attachment sites of muscles show high inter-subject variability, which may considerably affect muscle moment-arms because it has been shown that small differences in location of muscle attachment points often affect muscle force predictions to a great extent (see e.g., [4]).

Valente et al. [34] showed that representing a muscle, especially, a complex one, e.g., the gluteus medius, by a single line segment can produce errors up to 75% suggesting thus that the number of fibres in musculoskeletal models being in used might not be enough. Recently, Weinhandl & Bennett [37] confirmed that high number of fibres are required for the muscle surrounding the hip joint to provide an accurate estimation of joint contact forces. Modenese et al. [26] found out that representing the muscles surrounding the hip joint by fibres with none or a few via points only may limit the accuracy of hip contact force predictions.

To reduce the human effort associated with the construction of subject-specific musculoskeletal models, some researchers proposed algorithms to generate the fibres automatically providing that the surface model of a muscle is available [18,20,30]. The problem is how to update the shape of these fibres in reaction to the movement of bones. One approach to this problem is to express their vertices to be relative to the vertices of the surface mesh of the muscle, first, and then use some of the existing algorithms for surface mesh deformation proposed in the context of musculoskeletal modelling, e.g., [9,16,17,32].

In our conference paper [9], we proposed a new algorithm for muscle mesh deformation, based on position-based dynamics [28], and demonstrated its features on three hip muscles deforming during flexion of the right leg. In this paper, which is an extended version of that paper, we newly include:

- a description of the implementation details of our algorithm such as its initialization for muscle deformation, constraints calculations,
- a proposal of alternative algorithms for detecting the muscle points that should move with the bones,
- new experiments demonstrating the sensitivity of the results on its various parameters (e.g., anisotropy, number of iterations, resolution of the mesh),
- new experiments showing the lengths of fibres generated in the volume of hip muscles, and comparing them with those obtained by other approaches.

2 Position-Based Dynamics

Position-based dynamics (PBD), which is the core part of our approach, was firstly introduced in [28] as a fast, stable, and controllable alternative to mass-spring systems used in computer graphics algorithm. Since then, it has been further developed (e.g., [25] proposed recently some speed and accuracy improvements) and has found many (close to) real-time applications, not only in computer graphics, e.g., for simulations of cloth or fluids [33], but even in other domains. For example, Kotsalos et al. use PBD to model blood cells [24].

PBD represents a dynamic object, e.g., a muscle, by a set of N points, having associated mass and velocity, and a set of M constraints restricting the freedom of the movement of these points during the simulation. In their paper [28], Müller et al. presented the restraints to maintain distances among the points, the shape of the object and its volume, and to avoid collisions with other objects, however, one can use any constraint that is meaningful in their application context. Mathematically speaking, assuming that every point has the same mass,

the PBD method solves Eq. 1 that describes a movement of a single point \mathbf{p}_i restricted by a constraint function C with cardinality n , where $\Delta\mathbf{p}_i$ denotes the difference in position of i th point and $\nabla_{\mathbf{p}_i} C$ is the gradient of the function C with respect to point \mathbf{p}_i .

$$\Delta\mathbf{p}_i = -\frac{\nabla_{\mathbf{p}_i} C(\mathbf{p}_1, \dots, \mathbf{p}_n)}{\sum_{j=1}^n |\nabla_{\mathbf{p}_j} C(\mathbf{p}_1, \dots, \mathbf{p}_n)|^2} \cdot C(\mathbf{p}_1, \dots, \mathbf{p}_n) \quad (1)$$

2.1 Distance Constraint

Distance constraint is restricting each model point to change the distance from the others in its neighbourhood. It is described by Eq. 2, where d is the original distance between points \mathbf{p}_1 and \mathbf{p}_2 .

$$C(\mathbf{p}_1, \mathbf{p}_2) = |\mathbf{p}_1 - \mathbf{p}_2| - d \quad (2)$$

At this point, the gradient of this function has to be determined. Calculation procedure of determining the gradient of the vector norm is shown in (3).

$$\begin{aligned} \nabla_{\mathbf{p}_1} C(\mathbf{p}_1, \mathbf{p}_2) &= \nabla_{\mathbf{p}_1} (|\mathbf{p}_1 - \mathbf{p}_2| - d) \\ &= \frac{\left[\frac{\partial(p_{1x}-p_{2x})^2}{\partial p_{1x}} \quad \frac{\partial(p_{1y}-p_{2y})^2}{\partial p_{1y}} \quad \frac{\partial(p_{1z}-p_{2z})^2}{\partial p_{1z}} \right]}{2|\mathbf{p}_1 - \mathbf{p}_2|} \\ &= \frac{\mathbf{p}_1 - \mathbf{p}_2}{|\mathbf{p}_1 - \mathbf{p}_2|} = u \\ \nabla_{\mathbf{p}_2} C(\mathbf{p}_1, \mathbf{p}_2) &= \frac{\mathbf{p}_2 - \mathbf{p}_1}{|\mathbf{p}_1 - \mathbf{p}_2|} = -u \end{aligned} \quad (3)$$

Coincidentally, the result is the unit directional vector u of given edge.

2.2 Volume Constraint

Volume constraint restricts the object to change its volume during the simulation process. Assuming that this object is a triangular mesh model, the constraint function is:

$$C(\mathbf{p}_1, \dots, \mathbf{p}_n) = \sum_{i=1}^m \left(\mathbf{p}_{t_1}^i \cdot (\mathbf{p}_{t_2}^i \times \mathbf{p}_{t_3}^i) \right) - V_0 \quad (4)$$

where m is the number of triangles forming the mesh, V_0 is its original volume, and $p_{t_j}^i$ is j th vertex of triangle i .

To obtain complete gradient of volume constraint function, all triangles are treated independently and their results are just summed together:

$$\nabla_{\mathbf{p}_i} C(\mathbf{p}_1, \dots, \mathbf{p}_n) = \sum_{h=1}^t \mathbf{p}_j \times \mathbf{p}_k; \quad i \neq j \neq k \quad (5)$$

2.3 Local Shape Constraint

Above described constraints are not enough to prevent the triangular mesh model from becoming noisy, full of unrealistic spikes. One possible solution to this problem is to use the distance constraint not only to keep the distances between adjacent points but also between the pairs of points lying on the opposite sides of the model. This would, however, need to create a 3D mesh model first, which would be quite complex to do. Another option is to ensure that the local shape is maintained. To achieve this, the dihedral angles between neighbouring triangles should stay the same during deformation.

Equation 6 presents the local shape constraint function of triangles $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3$ and triangle $\mathbf{p}_2, \mathbf{p}_1, \mathbf{p}_4$ sharing points \mathbf{p}_1 and \mathbf{p}_2 . In this equation, \mathbf{n}_1 and \mathbf{n}_2 are normal vectors of these triangles and φ_0 is the original dihedral angle between them. Gradients are defined in (7).

$$\begin{aligned} C(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4) &= \arccos(\mathbf{n}_1 \cdot \mathbf{n}_2) - \varphi_0 \\ &= \arccos\left(\frac{(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_1)}{|\mathbf{p}_2 - \mathbf{p}_1| \times |\mathbf{p}_3 - \mathbf{p}_1|} \cdot \frac{(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_4 - \mathbf{p}_1)}{|\mathbf{p}_2 - \mathbf{p}_1| \times |\mathbf{p}_4 - \mathbf{p}_1|}\right) - \varphi_0 \end{aligned} \quad (6)$$

$$\begin{aligned} d &= \mathbf{n}_1 \cdot \mathbf{n}_2 \\ \nabla_{\mathbf{p}'_4} C &= -\frac{1}{\sqrt{1-d^2}} (\nabla_{\mathbf{p}'_4}(\mathbf{n}_2) \cdot \mathbf{n}_1) \\ \nabla_{\mathbf{p}'_3} C &= -\frac{1}{\sqrt{1-d^2}} (\nabla_{\mathbf{p}'_3}(\mathbf{n}_1) \cdot \mathbf{n}_2) \\ \nabla_{\mathbf{p}'_2} C &= -\frac{1}{\sqrt{1-d^2}} (\nabla_{\mathbf{p}'_2}(\mathbf{n}_1) \cdot \mathbf{n}_2 + \nabla_{\mathbf{p}'_2}(\mathbf{n}_2) \cdot \mathbf{n}_1) \\ \nabla_{\mathbf{p}'_1} C &= -\sum_{i=2}^4 \nabla_{\mathbf{p}'_i} C \end{aligned} \quad (7)$$

3 Our Approach

The requested inputs of our approach are 1) a set of bones, each of which is represented by a triangular mesh and has an associated time-dependent transformation describing its movement, and 2) a muscle, also represented by a triangular mesh. We note that the first input is standard when creating any subject-specific musculoskeletal model. A muscle model is obtainable with a little effort from the medical images by segmentation (similarly as models of bones). Optionally, the user may specify a set of muscle fibres, represented by polylines, obtained, e.g., by Kohout & Kukačka [19], Kohout & Cholt [21], or Otake et al. [30] method. Furthermore, the user may also specify a set of attachment areas that describes the sites where the muscle attaches to the bones. As the muscle attachment sites

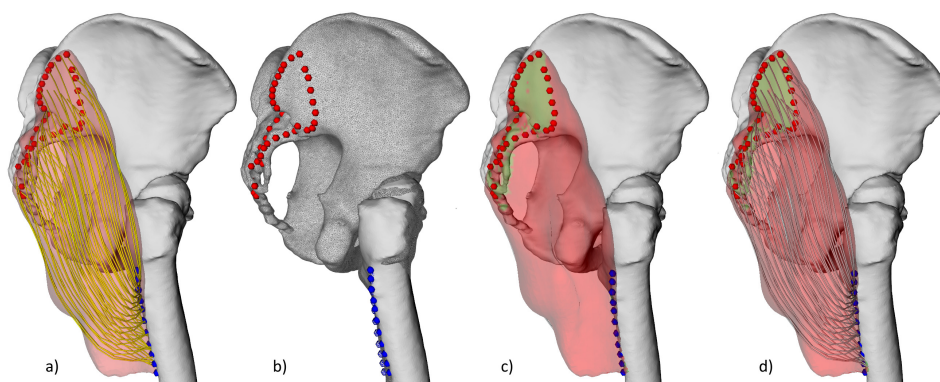


Fig. 2. Gluteus Maximus deformed by our approach: a) the input (origin and insertion attachment sites denoted by red and blue spheres), b) bones move from their initial rest-pose to the current pose (wireframe), c) the muscle surface adapted to the change of bones by PBD, d) the output. (color figure online)

are not apparent from the medical images, these are traditionally identified manually by an expert, typically as a set of landmarks fixed to the bones. Figure 2a shows an example of a typical input.

At each vertex of the muscle mesh, we create one PBD point with the mass being 1.0 and the initial velocity 0. For each pair of PBD points corresponding to the vertices connected in the muscle mesh by an edge, we establish the distance constraint (see Sect. 2.1) modified to support the anisotropic feature of muscles – see Sect. 3.1. Similarly, we create the local shape constraint (see Sect. 2.3) and the volume constraint (see Sect. 2.2). We note that we do not create any distance constraint between points of opposite sides of the muscle to avoid an unnatural change of the muscle shape during the simulation but rely solely on the latter two restraints in that.

We distinguish between two classes of PBD points: fixed and moveable, automatically detected as described in Sect. 3.3. A fixed point is bound to a single bone and moves with it at the beginning of the PBD simulation. The movement of the fixed points violates the equilibrium of the entire system, as described by the constraints, and the PBD attempts to restore it by updating, iteratively, the position of the moveable points while avoiding the penetration with the moved bones using the mechanism for collision detection and response described in Sect. 3.2 – see Fig. 2b,c.

Providing that the muscle fibres are specified, we compute the mean value coordinates of every vertex of polylines representing the fibres in the domain described by the triangular mesh of the muscle using the algorithm by Ju et al. [15]. Mathematically speaking, this operation maps the position of a muscle fibre vertex from E^3 to E^n , where n is the number of vertices of the muscle mesh. When the muscle surface deforms, the inverse mapping provides new positions of fibre vertices within the deformed domain (see Fig. 2d):

$$\mathbf{v}'_i = \sum_{j=1}^n w_j \cdot \mathbf{p}'_j \quad (8)$$

In the equation above, \mathbf{v}'_i denotes the i -th fibre vertex, w_j are its mean value coordinates and \mathbf{p}'_j is the position of the deformed muscle vertex \mathbf{p}_j .

The entire algorithm written in pseudocode is in Algorithm 1 and 2.

3.1 Anisotropy

The PBD algorithm has been originally proposed in the computer graphics field to model isotropic materials (e.g., cloths). However, muscles are anisotropic (may behave differently in two distinct directions), so it is appropriate to take anisotropy into account. The main idea is that muscle surface is stiffer in the direction perpendicular to the muscle fibres and more flexible in the direction parallel to these fibres. Mathematically speaking, we multiply the distance constraint (see Eq. 2) with the result of the following equation:

Algorithm 1. Pre-processing stage of our algorithm.

```

1: procedure INIT( $M, S_B, S_F, S_A$ )      ▷  $M$  is a muscle triangular mesh,  $S_F$  is a
                                         set of muscle fibres,  $S_B$  is a set of bones,
                                         and  $S_A$  is a set of attachment areas
2:   for all vertices  $\mathbf{v}_i \in S_F$  do
3:      $\mathbf{w}_i = \text{computeMVC}(\mathbf{v}_i, M)$       ▷ compute the mean value coordinates
4:   end for
5:   for all bones  $B \in S_B$  do
6:      $\text{generateCollisionDataStructure}(B)$     ▷ see Section 3.2
7:   end for
8:   for all vertices  $\mathbf{p}_i \in M$  do
9:      $\mathbf{x}_i = \mathbf{p}_i, \mathbf{v}_i = 0, m_i = 1$       ▷ initialize a PBD point
10:  end for
11:  detect fixed points                    ▷ see Section 3.3
12:  for all edges  $e_i \in M$  do
13:     $\text{generateDistanceConstraint}(e_i)$       ▷ compute the original distance  $d$ 
14:    if  $S_F = \emptyset$  then
15:       $k_i = 1$                                ▷ no anisotropy used
16:    else
17:       $k_i = \text{computeAnisotropyStiffness}(e_i)$     ▷ see Section 3.1
18:    end if
19:     $\text{generateLocalShapeConstraint}(e_i)$     ▷ compute the dihedral angle  $\varphi_0$ 
20:  end for
21:   $\text{generateVolumeConstraint}(M)$           ▷ compute the original volume  $V_0$ 
22: end procedure

```

Algorithm 2. Runtime stage of our algorithm.

```

1: procedure EXECUTE(simFrame) ▷ simFrame is the index of simulation frame
2:   for all bones  $B \in S_B$  do                                     ▷ see also Algorithm 1
3:      $T = \text{getTransform}(B, \text{simFrame})$                        ▷ get the transformation matrix
4:      $\text{transformMesh}(B, T)$ 
5:   end for

6:   for all PBD points  $i$  do
7:     if isFixed( $i$ ) then
8:        $B = \text{getAttachmentBoneForPoint}(i)$ 
9:        $\mathbf{p}_i = \text{transformPoint}(\mathbf{x}_i, \text{getTransform}(B, \text{simFrame}))$ 
10:    else
11:       $\mathbf{v}_i = \mathbf{v}_i + \Delta t \cdot \mathbf{f}_{ext}(\mathbf{x}_i) / m_i$            ▷ update velocities by external forces
12:       $\mathbf{v}_i = \mathbf{v}_i \cdot c_{damp}$                                    ▷ apply some damping
13:       $\mathbf{p}_i = \mathbf{x}_i + \Delta t \cdot \mathbf{v}_i$ 
14:    end if
15:  end for

16:  loop solverIterations times
17:    for all edges  $e_i \in M$  do
18:       $\text{projectDistanceConstraintWithAnisotropy}(e_i, k_i)$        ▷ updates  $\mathbf{p}_i$ 
19:    end for
20:     $\text{projectVolumeConstraint}()$ 
21:    for all edges  $e_i \in M$  do
22:       $\text{projectLocalShapeConstraint}(e_i)$ 
23:    end for
24:    for all vertices  $i$  do
25:      for all bones  $B \in S_B$  do
26:         $T = \text{getTransform}(B, \text{simFrame})$ 
27:         $\text{generateCollisionConstraints}(B, T^{-1}, \mathbf{x}_i, \mathbf{p}_i)$ 
28:      end for
29:       $\text{projectCollisionConstraints}()$ 
30:    end for
31:  end loop

32:  for all vertices  $i$  do
33:    if NotIsFixed( $i$ ) then
34:       $\mathbf{v}_i = \frac{\mathbf{p}_i - \mathbf{x}_i}{\Delta t}$                                ▷ compute the velocity
35:       $\mathbf{x}_i = \mathbf{p}_i$                                        ▷ update the position
36:    end if
37:  end for

38:  for all vertices  $\mathbf{p}_i \in M$  do
39:     $\mathbf{p}_i = \mathbf{x}_i$                                            ▷ update the muscle mesh
40:  end for
41:  for all vertices  $\mathbf{v}_i \in S_F$  do
42:     $\mathbf{v}_i = \text{reconstructPositionFromMVC}(\mathbf{w}_i, M)$ 
43:  end for
44: end procedure

```

$$k_i = 1 - \mathbf{u}_i \cdot \mathbf{v}_i \quad (9)$$

The direction of i th edge is described by normalized vector \mathbf{u}_i , \mathbf{v}_i denotes tangential direction normal vector of nearest fibre on the surface. If both vectors are collinear, the result k_i will be zero, meaning no distance is preserved. If these two vectors are perpendicular, then k_1 is equal to one and edge length will be preserved.

3.2 Collision Handling

The moving muscle and bones should not intersect each other. From several approaches to this issue we considered (see our conference paper [9]), we have opted for voxelization because of its simplicity and processing speed. In this approach, the bounding box of a bone is divided into a uniform grid of $n_x \times n_y \times n_z$ equally sized cells. For each triangle in the bone mesh, we detect the cells intersected by it and mark them as the boundary. Assuming that the mesh is closed, we mark the cells that are inside the bone using the flood-fill algorithm with 8-directions. All other cells are outside. Figure 3 shows the visualization of boundary cells when the constants n_x , n_y , and n_z are equal and when they are automatically determined from the sizes of the bounding box so that the overall number of cells is roughly equal to some given constant n_{max} .

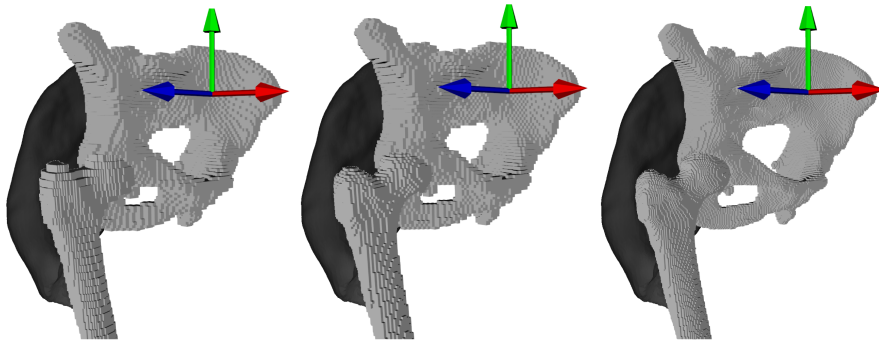


Fig. 3. Voxel representation of pelvis and femur. From left to right: $n_x = n_y = n_z = 64$ (262 144 cells, 256 KB min), $n_{max} = 262\ 144$ – pelvis = $47 \times 64 \times 85$ (255 680 cells, 250 KB min) femur = $42 \times 176 \times 34$ (251 328 cells, 245 KB min), $n_{max} = 8 \cdot 262\ 144 = 2\ 097\ 152$ – pelvis = $95 \times 128 \times 171$ (2 079 360 cells, ≈ 2 MB min) femur = $85 \times 352 \times 69$ (2 064 480 cells, ≈ 2 MB min).

During the simulation, the algorithm identifies the cell in which a PBD point \mathbf{p}_i lies. If this cell is outside the bone, the point does not collide with the bone. Otherwise, its position needs to be updated. Two scenarios have to be distinguished. In the first one, the muscle moves (e.g. because of surrounding forces) and hits a bone. As it is, the previous position of this point (\mathbf{x}_i) is outside the bone. The algorithm, therefore, traverses the collision data structure along the

line segment from \mathbf{p}_i to \mathbf{x}_i until it does not find an outside cell. If this cell is the cell of \mathbf{x}_i , \mathbf{p}_i moves back to \mathbf{x}_i ; otherwise, it moves to the point on the line segment where the traversal stopped.

In the second scenario, a bone moves into the muscle. Therefore, even the previous position of the point (\mathbf{x}_i) no longer lies outside the bone. We propose a solution where \mathbf{p}_i moves to \mathbf{x}_i transformed by the same transformation that caused the collision.

3.3 Detecting the Fixed Points

Assuming that the muscle is, in fact, a musculotendon unit, i.e., its surface touches the bones at the attachment sites, there are three approaches to detecting the muscle points that should be fixed to some bone and move with it, each of which has its pros and cons. In our previous work [9], we used the constructed data structure for collision detection also for the identification of the fixed points. However, recent analysis shows that this algorithm may inappropriately fix also the points that are close to some bone but should slide along it – see Fig. 4. That is the real reason for the unacceptable behaviour of the iliacus muscle during the flexion of the right leg reported in the original paper.

We, therefore, have experimented with another approach. We fix all points lying in the proximity of some bone, i.e., having their distances to the surface of a bone smaller or equal to a predefined threshold. An obvious choice is to compute the average length of edges in the muscle mesh and use it as this threshold. Figure 5, however, shows that the results are not very different from the results obtained by the original algorithm. Specification of the threshold value by the user may help. Nevertheless, this is very sensitive. For example,

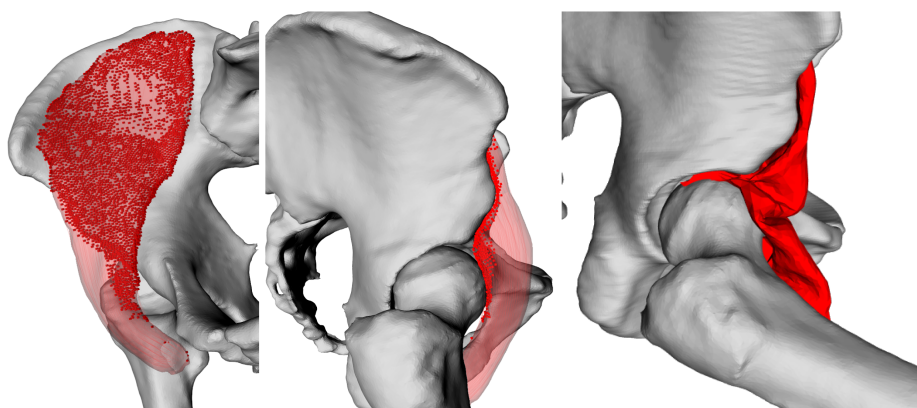


Fig. 4. Muscle vertices (red cubes) of Iliacus fixed to some bone, i.e., preserving their relative position to the bone during the simulation, as identified by the original CD-based algorithm ($n_x = n_y = n_z = 64$) causing an unsatisfactory result of the deformation (right). (Color figure online)

while 1 mm threshold seems perfect for gluteus maximus, for iliacus, a value less than 0.5 mm is needed to get something at least reasonable.

The third approach exploits the idea that muscle attachment areas are typically required for the construction of muscle fibres and, the user, therefore, have them readily available also for detection of the fixed points. We assume that an attachment area is defined by a set of landmarks that are fixed to a bone and, furthermore, they are specified in an order such that interconnecting every pair of adjacent landmarks by a line segment would produce a closed non-intersecting poly-line corresponding to the boundary of the attachment area. Following the idea described by Kohout & Kukačka in [19], we detect the patch on the muscle having the boundary corresponding to the boundary of the attachment area projected onto the muscle surface and fix all the points of this patch. As Fig. 5 demonstrates, this approach provides us with the best results.

4 Experimental Results

In this paper, a subset of a comprehensive female cadaver anatomical dataset (81 y/o, 167 cm, 63kg) is used. Specifically, pelvic and femur bones together with several muscles from the pelvic region have been selected.

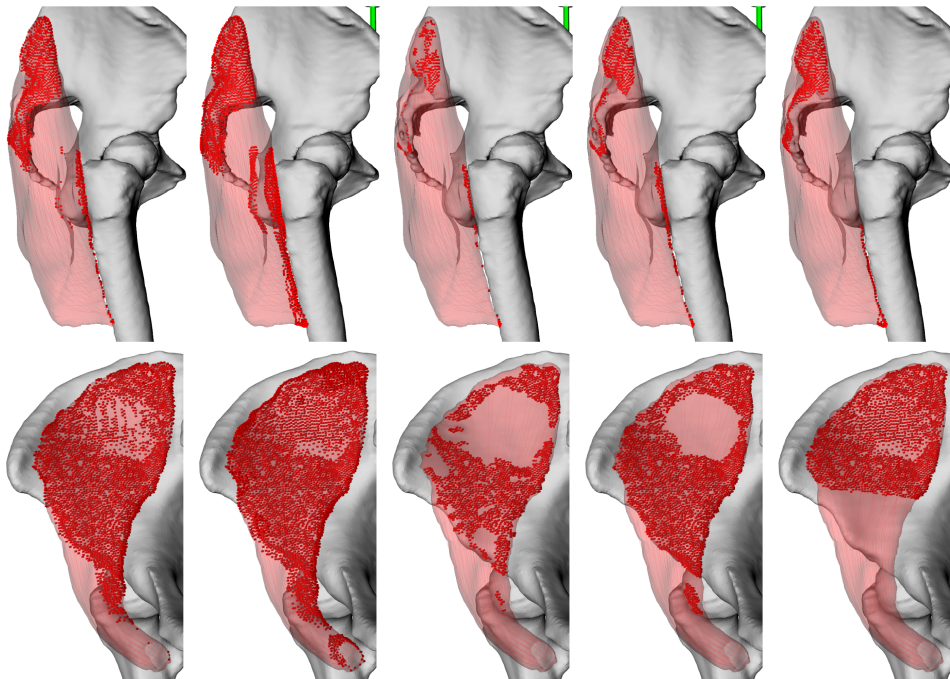


Fig. 5. Muscle vertices (red cubes) of gluteus maximus (top) and iliacus (bottom) fixed to some bone, i.e., preserving their relative position to the bone during the simulation, as identified by the original CD-based algorithm ($n_x = n_y = n_z = 64$), the muscle-bone proximity algorithm with the thresholds: average edge length, 0.5 mm, and 1 mm, and by the algorithm using muscle attachment areas input data. (Color figure online)

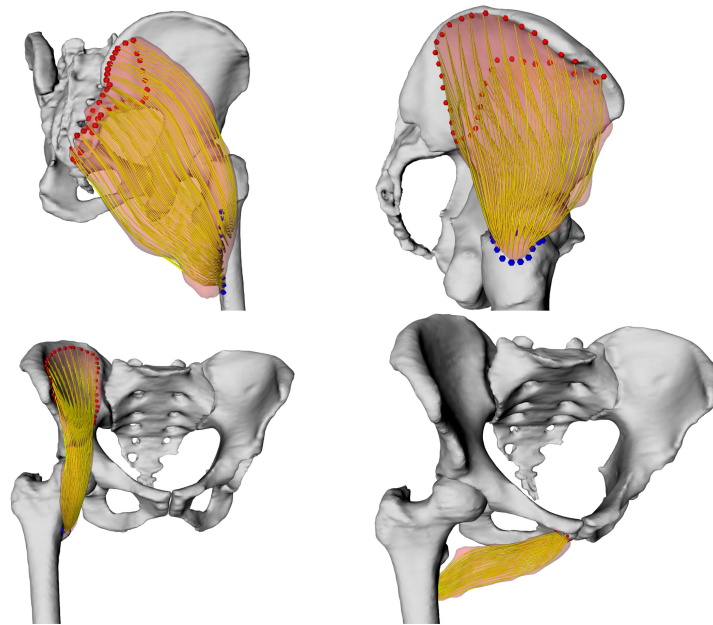


Fig. 6. Gluteus maximus (19 752 triangles (Δ)), gluteus medius (10 622 Δ), iliacus (13 858 Δ), and adductor brevis (17 124 Δ) decomposed into a set of 100 fibres composed of 15 line segments.

The complete data are publicly available in LHDLD dataset [36] and has been selected because it includes high-quality surface meshes of bones and muscles. Furthermore, the dataset was improved by removing non-manifold edges, duplicated vertices and degenerate triangles followed by surface smoothing in both muscle and bone models using MeshLab [5]. The dataset also contains muscle attachment areas and geometrical paths of superficial fibres obtained from dissection [35].

To decompose the muscles into fibres, we use the Kohout & Kukačka method [19] with a slight modification: establishing the inter-contour correspondence is done by minimizing the sum of square distances between the corresponding points. This modification increases the robustness of the method even in cases when the user-specified number of straight-line segments per fibre is low.

We decomposed the surface meshes of gluteus maximus, gluteus medius, iliacus and adductor brevis into models of 100 fibres using a template with parallel fibres composed of 15 line segments – see Fig. 6. The decomposition took less than 50 ms in all cases on HP EliteDesk 800 G3 TWR (Intel Core i7-7700K @ 4.2 GHz, 64 GB RAM, Windows 10 64-bit).

Simulations of hip flexion (0° to 90°) were performed in steps of 2° via inverse kinematics. Inverse kinematics means that the location and movement of all bones are known, and muscle actual shape has to be determined according to these situations. We note this is exactly the opposite to what can be seen in real situation, where muscles control bone movement.

The default parameters in all our experiments were: $n_{max} = 8 \cdot 64 \cdot 64 \cdot 64$, the damping constant $c_{damp} = 0.99$, anisotropy on, local shape constraint stiffness = 0.9 (i.e., the solver was allowed to violate this constraint to preserve the volume and avoid the penetration between the muscle and bones).

4.1 Number of Solver Iterations

In the first experiment, we investigated the influence of the number of iterations of constraint projections (see the loop on line 16 in Algorithm 2) on the quality of the results and overall time required for the simulation. From Fig. 7, it is apparent that the average displacement of points between individual iterations quickly decreases. In a few iterations, it drops below 0.1 mm; with just 10 iterations it is below 0.01 mm.

Average time required by one simulation step (Algorithm 2) on HP EliteDesk 800 G3 TWR (Intel Core i7-7700K @ 4.2 GHz, 64 GB RAM, Windows 10 64-bit) using our, mostly unoptimized, C++ implementation is in Table 1.

Table 1. Times needed for one simulation step on average for adductor brevis using various number of PBD solver iterations (NoIters).

NoIters	1	3	5	10	25	50	100
Time [ms]	53.04	62.55	74.66	104.96	193.66	441.72	658.71

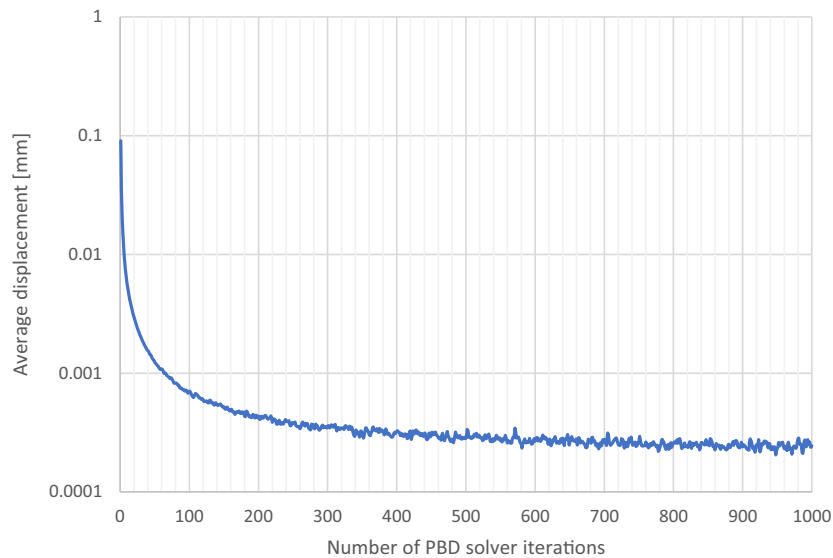


Fig. 7. The average displacement of points of adductor brevis between individual iterations of the PBD solver during the hip flexion. Note the logarithmic scale on the y-axis.

Figure 8 brings a visual comparison of the results obtained using a various number of iterations. An unrealistic bending of the muscle is apparent, especially when using a few iterations only. This behaviour has three reasons. First, the speed of the femur bone is quite high; it is 2° per simulation step, which represents the movement of the fixed points of 3.78–6.75 mm. Next, at the beginning of the simulation, the moving femur hits an unfixed part of the muscle giving it a large velocity pulling it in the direction opposite to the natural movement. Finally, the muscle mesh contains 17126 triangles, i.e., it is pretty accurate, and, therefore, a lot of iterations are required to propagate the movement from the points on the femur to those on the pelvis.

Hence, we reduced the number of triangles using the quadric edge collapse decimation implemented in MeshLab software down to 3000 (L1) and 1000 (L2). Not only visual realism improves, as Fig. 9 illustrates, but also the overall required time decreases since there are less PBD points and consequently also fewer constraints to satisfy. For L2 mesh, 1000 iterations need 349.80 ms per simulation step on average, which is even faster than 100 iterations for the original, high-resolution mesh. Naturally, this higher number of iterations improves visual appearance considerably. We note, however, that increasing the number of iterations further, e.g., to 10000, does not bring any substantial change.

4.2 Fixed Points

Figure 8 also demonstrates the effect of the algorithm used to detect the points to fix on the results of the deformation. Due to inaccuracies during the extraction of the musculotendon unit, only a very small part of the adductor brevis muscle is touching the femur. When using the original algorithm, which exploits the collision detection mechanism, a significant area on the muscle is, therefore, not fixed. As a result, the deformation algorithm produces the mesh with an unrealistic sharp spike. There is no such issue with the detection algorithm exploiting the knowledge of muscle attachment areas.

A different case happens with the iliacus muscle – see Fig. 10. Despite the relatively high resolution of the voxel data structure, many muscle points in proximity of the femur ball are fixed incorrectly to the femur. As a result, this part of the muscle moves unrealistically into the narrow space between the femur and pelvis. Using the attachment areas improves the situation but only slightly because the points in the proximity of the femur ball typically collide with the coarse voxel representation of the femur and they are, therefore, transformed using the same transformation. After turning this collision handling mechanism off, the algorithm provides us with acceptable results with a small muscle-bone penetration.

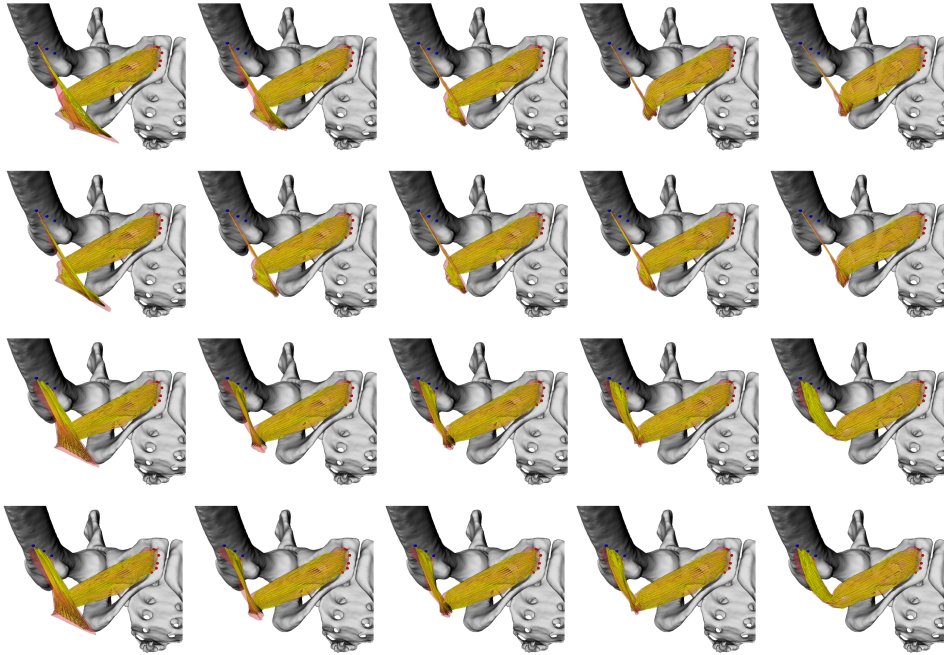


Fig. 8. Adductor brevis at flexion of 40° using 1, 3, 5, 10 or 100 PBD solver iterations (from left to right) with anisotropy off (odd rows) and on (even rows) when fixing the points by the original CD-based algorithm with $n_{max} = 8 \cdot 262144$ (the first two rows) and by the algorithm using muscle attachment areas (the last two rows).

4.3 Anisotropy, Volume and Other Constraints

The impact of the anisotropy on the results is apparent in Fig. 8. Surprisingly, it is barely observable. Most probably, this is because the other constraints (especially the volume constraint) play a dominant role. Volume preservation constraint was tested by determining ratio between both original and actual volumes. Figure 11 show us the volume preservation results. As we can see, the volume is well preserved (the error is less than 1% in all cases). Other quantitative tests, e.g., preservation of the dihedral angles between two adjacent triangles and average edge extension, are presented in our original conference paper [9].

4.4 Muscle Fibre Lengths

Last but not least, we analyzed the lengths of fibres reconstructed at the end of the deformation step. To remove any noise that might be present in the data, we performed a smoothing process, repeated five times, that updates the length l_i according to the equation: $l'_i = (l_{i-1} + 4 \cdot l_i + l_{i+1})/6$. The results are present in Figs. 12, 13, and 14. Both gluteus maximus and gluteus medius behave during the hip flexion as expected. The lengths of all the gluteus maximus fibres increase. In the case of the gluteus medius, only the surface fibres extend, while the deep

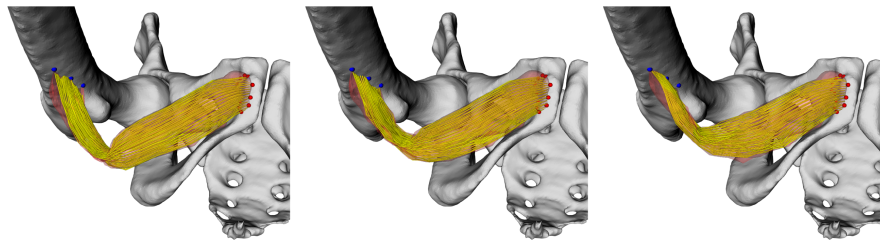


Fig. 9. Adductor brevis at flexion of 40° using 100 PBD solver iterations, anisotropy on, fixing the points at muscle attachment areas when a surface mesh with 17 124, 3 000, and 1 000 triangles is used.

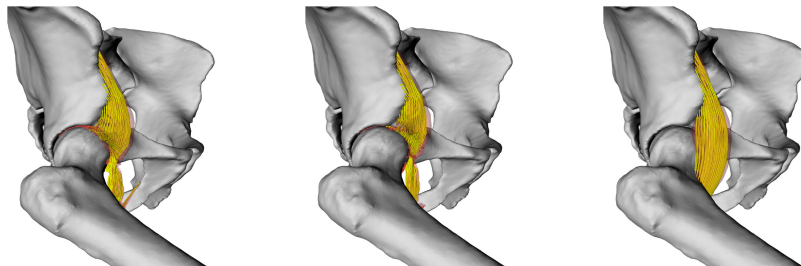


Fig. 10. Iliacus at flexion of 40° using 5 PBD solver iterations, anisotropy on, fixing the points by the original CD-based algorithm with $n_{max} = 8.262144$ (left) and by the algorithm using muscle attachment areas with (middle) and without (right) collision handling when a bone hits the muscle.

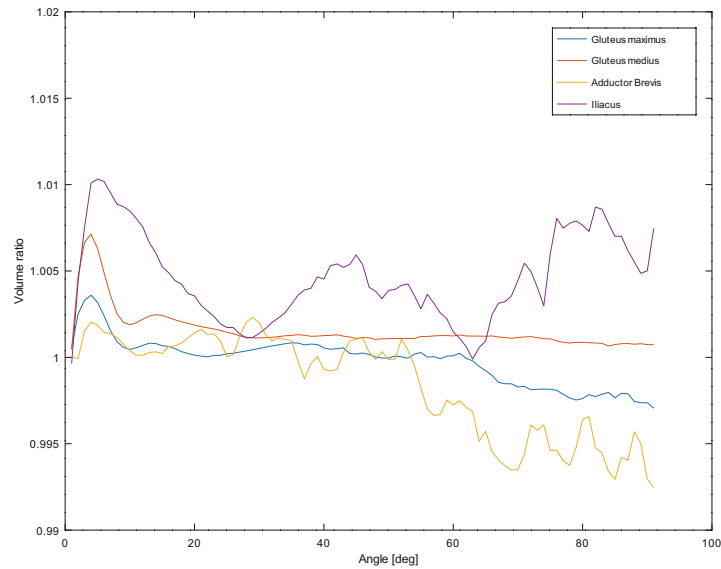


Fig. 11. Volume preservation during hip flexion using 3 PBD solver iterations.

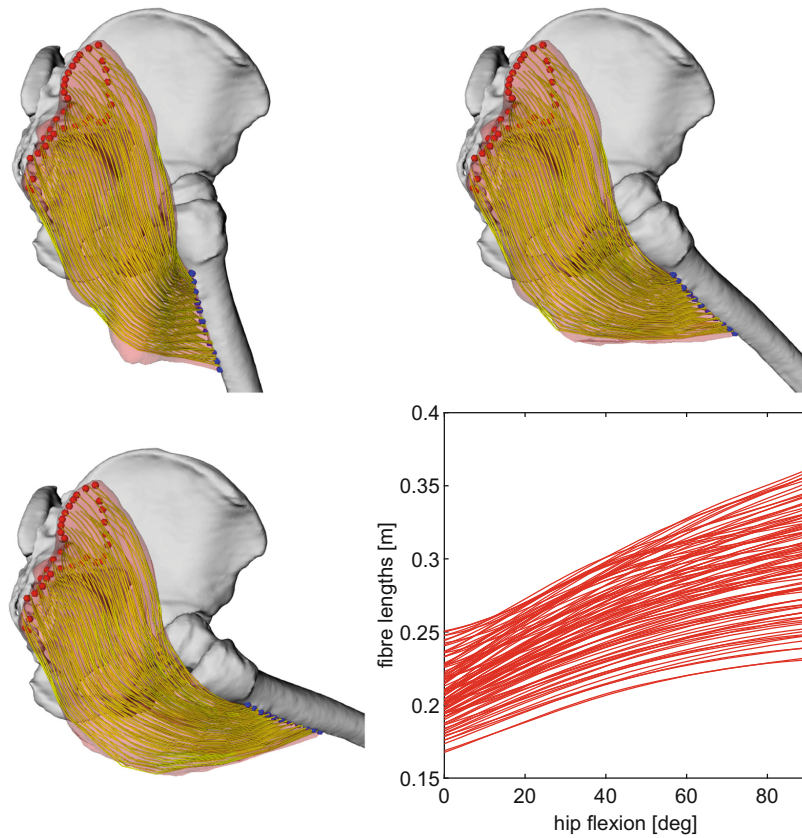


Fig. 12. Total length of each individual fibre during simulation in the gluteus maximus muscle. The visual results at 20°, 50°, and 70° are shown for illustration.

fibres contract. For the iliacus muscle, we can observe an unrealistic increase in the lengths when the flexion is greater than 70°, which is caused by the above-described issue of pushing a part of the muscle into the joint space.

4.5 Deformation Speed

The proposed method was designed to be not only precise, but mainly, fast. It was implemented in C++ using VTK toolkit. Its current version is publicly available at <https://github.com/cervenkam/muscle-deformation-PBD>.

Using the collision detection structure with $n_x = n_y = n_z = 64$ and three PBD solver iterations, we measured the processing speed of our implementation. All tests were performed on Intel[®] Core[™] i7-4930K 3.40 GHz CPU, Radeon HD 8740 GPU and WDC WD40EURX-64WRWY0 4TB HDD. The results, given in FPS (Frames Per Second), i.e., the number of simulation steps per second, are listed in Table 2. As it can be seen, the FPS strictly depends on number of triangles (Spearman's $\rho = -1$). The more triangles is used, the slower the method

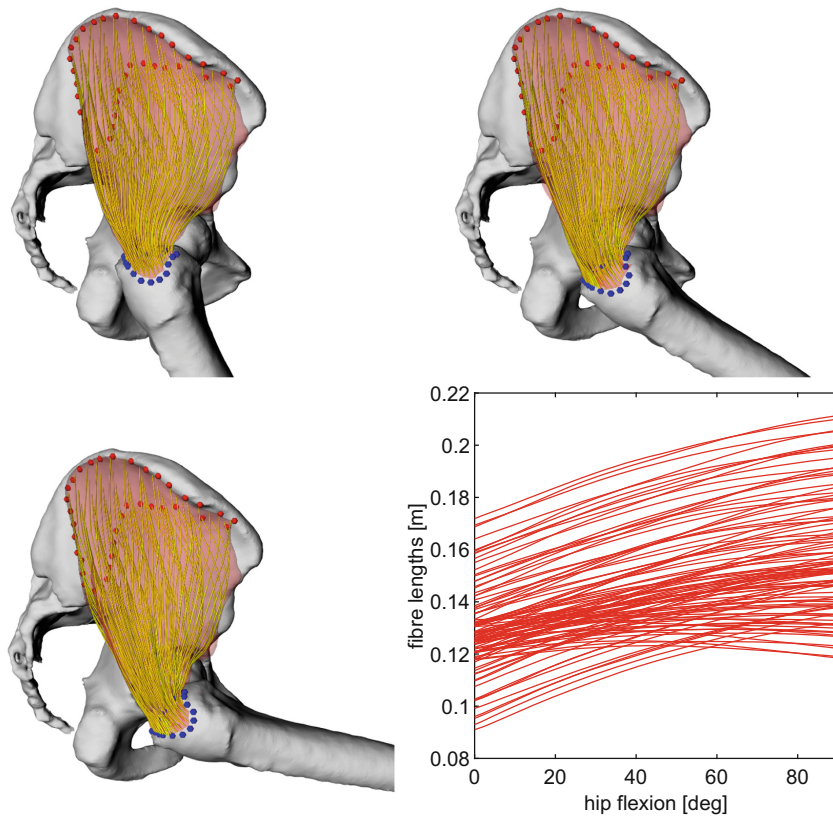


Fig. 13. Total length of each individual fibre during simulation in the gluteus medius muscle. The visual results at 20°, 50°, and 70° are shown for illustration.

Table 2. FPS of each simulation.

Deforming object	Triangle count	FPS
Gluteus maximus	19752	33.85
Abductor brevis	17124	35.89
Iliacus	13858	47.21
Gluteus medius	10622	57.12

is. Even though the program is mostly unoptimized and runs sequentially at the moment, the FPS is sufficient for considered purposes in general.

5 Discussion

In the past, several algorithms for the deformation of the surface mesh of a muscle were proposed. Most of these algorithms, however, have unreal requirements on the input, e.g., [16, 17] rely on existence of a muscle skeleton (centroid)

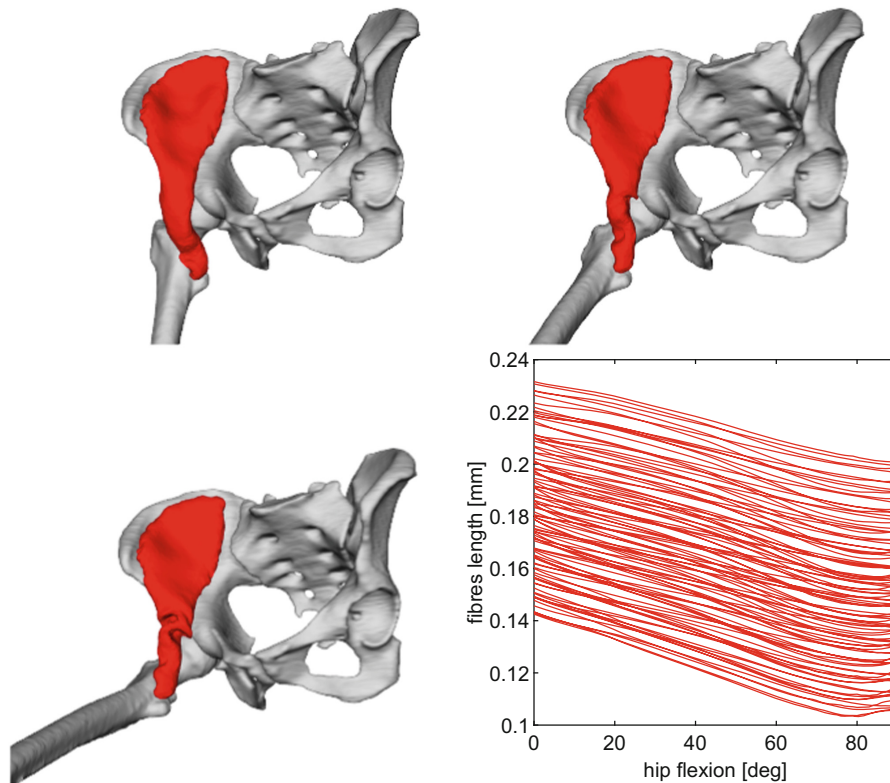


Fig. 14. Total length of each individual fibre during simulation in the iliacus muscle. The visual result at 20° , 50° , and 70° are shown for illustration. For clarity, we do not show the fibres. Readers are referred to Fig. 10 to see the produced fibres of the iliacus muscle.

having known a physiologically correct deformation, or they ignore important physiological properties such as impenetrability with bones and other muscles (e.g., [17,32]), muscle volume preservation, and anisotropy of muscles during their contraction.

Romeo et al. [32] independently to our work developed an approach similar to ours. The main differences are as follows. First, the authors build a complex internal muscle structure to better preserve the shape and volume of the muscle, while we work with the surface geometry only. Next, they do not include any mechanism to prevent penetration of muscles and bones, relying thus on manually defined various mesh-to-mesh constraints, which not only complicates the setup but also does not guarantee impenetrability. We implemented a simple and fast collision handling that avoids muscle-bone penetration in most cases. Finally, their aim is to have a visually plausible skin deformation but what is going on inside the body is not of their interest. We, on the other hand, focus on the representation of muscles for mechanical assessments.

Janak et al. [14] proposed a technique based on the mass-spring system to deform the fibres while preventing their penetration with bones and fibres of other muscles. To get reasonable results, a lot of particles are required, which causes high time and memory complexity. More importantly, the muscle volume is not preserved. This could be probably solved using the approach described in [13], however, it would increase computational time dramatically. Finally, our experiments show that although this method retains the smooth shape of iliacus muscle during flexion, it twists the part of the muscle close to the insertion. This is because, unlike our approach, the particles are in the entire volume of the muscle, which results in a model that is much more rigid, and as anisotropy is not exploited, rigid in all directions. Our method supports anisotropy, preserves the volume and runs in a fraction of time while requiring no extra parameter or input in comparison with this method.

The most complex way to solve muscle dynamics is by using the finite element method (FEM). This approach is physically the most accurate one if the muscle is well discretized (see e.g., [7]). However, computational complexity is high, meaning the FEM-based methods are unsatisfactorily slow. Therefore, it is quite impractical for real-time application or even clinical assessments. Next issue is a difficult set up of FEM methods, making them unsuitable for personalised musculoskeletal method deformation. Despite these facts, these methods can be seen in the movie industry, see e.g. Ziva VFX¹ plugin for Maya, and in muscle physiology research, see e.g. [29] or [23]. In comparison with these methods, our method is quite simple to set up and runs fast providing the promising results in most cases.

Recently, Modenese & Kohout [27] presented a simple method that calculates the kinematic position of a vertex of the fibre as a linear combination of the transformations of its rest-pose position with respect to the bones with the attachment sites of the muscle this fibre belongs to, whereas the blending weight is chosen as a function of the relative distance of this vertex from the origin point of the fibre with one user-specific parameter to minimize the penetration of the fibre with bones. Using the approach described in [18] to highly discretize the muscles of pelvic region (up to 100 fibres of 15 line segments), the fibres' moment arms of hip flexion, adduction, and internal rotation were validated against measurements and models of the same muscles from the literature with promising outcomes. Nevertheless, extending the method for muscles wrapping around multiple bones, such as *rectus femoris*, is not straightforward. Furthermore, a muscle-bone penetration cannot be avoided and in the case of the iliacus muscle, the fibres are also unrealistically pushed into the hip joint. Similarly to [14], the volume of a muscle cannot be preserved.

We compared the length of the fibres produced by Modenese & Kohout [27] with those produced by our approach using the same data. Figure 15 shows a good match between the results for the gluteus medius and the iliacus. A significant difference is apparent for the gluteus maximus. The range of lengths of our fibres is much bigger than theirs, whereas our fibres tend to be longer. One of

¹ <https://zivadynamics.com/>.

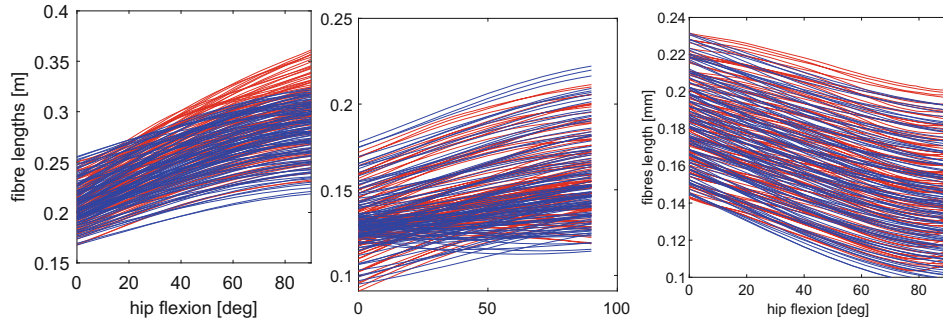


Fig. 15. Comparison of the lengths of the fibres of the gluteus maximus (left), gluteus medius (middle), and iliacus (right) muscle produced by our approach (red) and by the approach described in [27] (blue). (Color figure online)

the reasons for this difference is that our approach guarantees impenetrability between muscle and bones. As a result, all the fibres have to wrap around the joint and, naturally, they must be longer than the fibres produced by the other approach, where some fibres penetrate the femur in extreme positions. The volume preservation constraint prevents the flattening of the muscle at the greater trochanter of the femur, which means that the surface fibres are more distant from the bone than in the other approach. Consequently, they are longer.

There are some limitations of the proposed approach. First of all, the experiments have shown that detecting the muscle points that should move with bones exploiting the information about attachment areas of the muscle is superior in most cases when compared with proximity or collision-based detection. The muscle attachment sites, however, cannot be extracted automatically from the medical images and their manual specification, by an expert in anatomy, is time-consuming. Nevertheless, Fukuda et al. [10] proposed an approach to the automatic estimation of the muscle attachments that is based on applying a non-rigid transformation of the surface model of a normalized (average) bone with a normalized attachment site specified onto the surface model of the subject-specific bone. When the normalized attachment site is obtained from a probabilistic atlas built as suggested by the authors, the estimations are quite accurate, with dice coefficients reaching up to 70%.

Next, the proposed collision handling is inaccurate, which leads to an appearance of sharp spikes on the surface of the muscle, especially, when using a coarse voxel representation of bones. Naturally, as the memory complexity of this representation grows cubically, it is obvious that using a refined voxel representation is impractical. In the scenario when a bone moves into a muscle, setting the velocities of the colliding points to zero instead of using the formula on line 34 (in Algorithm 2) could help.

Finally, the results are very sensitive to the settings of the parameters. Fortunately, as the simulation runs in real-time, even using an unoptimized sequential

implementation, the user may tune the values of these parameters until they are satisfied with the visual output of our approach.

6 Conclusion and Future Work

The presented approach is capable of performing a visually plausible and physically correct real-time deformation of muscles represented by triangular meshes in most cases we tested. The main issue is with the iliacus muscle, which (when deformed) looks unrealistic. Nevertheless, the qualitative and quantitative results (e.g., the length of the fibres produced in the volume of the deformed muscle) are comparable with the other state-of-the-art methods. In the future, the iliacus muscle deformation will be further analyzed and the issue with muscle tissue entering the joint is to be solved.

The implementation is written in C++ and partially included in OpenSim (a state-of-the-art simulation software) as a plugin. Its source code is available at <https://github.com/cervenkam/muscle-deformation-PBD>.

Acknowledgment. Authors would like to thank their colleagues and students for valuable discussion, worthwhile suggestions and constructive comments. Authors would like to thank also anonymous reviewers for their hints and notes provided.

References

1. Arnold, E.M., Ward, S.R., Lieber, R.L., Delp, S.L.: A model of the lower limb for analysis of human movement. *Ann. Biomed. Eng.* **38**(2), 269–279 (2009). <https://doi.org/10.1007/s10439-009-9852-5>. <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2903973/>
2. Audenaert, A., Audenaert, E.: Global optimization method for combined spherical-cylindrical wrapping in musculoskeletal upper limb modelling. *Comput. Methods Programs Biomed.* **92**(1), 8–19 (2008). <https://doi.org/10.1016/j.cmpb.2008.05.005>. <http://www.ncbi.nlm.nih.gov/pubmed/18606476>
3. Bolsterlee, B., Veeger, D.H.E.J., Chadwick, E.K.: Clinical applications of musculoskeletal modelling for the shoulder and upper limb. *Med. Biol. Eng. Comput.* **51**(9), 953–963 (2013). <https://doi.org/10.1007/s11517-013-1099-5>
4. Carbone, V., van der Krogt, M., Koopman, H., Verdonschot, N.: Sensitivity of subject-specific models to errors in musculo-skeletal geometry. *J. Biomech.* **45**(14), 2476–2480 (2012). <https://doi.org/10.1016/j.jbiomech.2012.06.026>
5. Cignoni, P., Callieri, M., Corsini, M., Dellepiane, M., Ganovelli, F., Ranzuglia, G.: MeshLab: an open-source mesh processing tool. *Computing* **1**, 129–136 (2008). <https://doi.org/10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136>
6. Delp, S.L., Loan, J.P., Hoy, M.G., Zajac, F.E., Topp, E.L., Rosen, J.M.: An interactive graphics-based model of the lower extremity to study orthopaedic surgical procedures. *IEEE Trans. Biomed. Eng.* **37**(8), 757–767 (1990). <https://doi.org/10.1109/10.102791>
7. Delp, S.: Three-dimensional representation of complex muscle architectures and geometries 1. *Ann. Biomed. Eng.* **33**, 1134 (2005). <https://doi.org/10.1007/s10439-005-1433-7>

8. Delp, S.L., Loan, J.P.: A computational framework for simulating and analyzing human and animal movement. *Comput. Sci. Eng.* **2**(5), 46–55 (2000)
9. Červenka, M., Kohout, J.: Fast and realistic approach to virtual muscle deformation. In: *Proceedings of the 13th International Joint Conference on Biomedical Engineering Systems and Technologies*. SCITEPRESS - Science and Technology Publications (2020). <https://doi.org/10.5220/0009129302170227>
10. Fukuda, N., et al.: Estimation of attachment regions of hip muscles in CT image using muscle attachment probabilistic atlas constructed from measurements in eight cadavers. *Int. J. Comput. Assist. Radiol. Surg.* **12**(5), 733–742 (2017). <https://doi.org/10.1007/s11548-016-1519-8>
11. Garner, B., Pandey, M.: The obstacle-set method for representing muscle paths in musculoskeletal models. *Comput. Methods Biomech. Biomed. Eng.* **3**(1), 1–30 (2000)
12. Herteleer, M., et al.: Variation of the clavicle's muscle insertion footprints - a cadaveric study. *Sci. Rep.* **9**(1), 1–8 (2019). <https://doi.org/10.1038/s41598-019-52845-8>
13. Hong, M., Jung, S., Choi, M.H., Welch, S.: Fast volume preservation for a mass-spring system. *IEEE Comput. Graph. Appl.* **26**, 83–91 (2006). <https://doi.org/10.1109/MCG.2006.104>
14. Janák, T., Kohout, J.: Deformable muscle models for motion simulation. In: *Proceedings of the 9th International Conference on Computer Graphics Theory and Applications*. pp. 301–311. SCITEPRESS - Science and Technology Publications (2014). <https://doi.org/10.5220/0004678903010311>
15. Ju, T., Schaefer, S., Warren, J.: Mean value coordinates for closed triangular meshes. *ACM Trans. Graph.* **24**(3), 561–566 (2005). <http://portal.acm.org/citation.cfm?doid=1073204.1073229>
16. Kellnhofer, P., Kohout, J.: Time-convenient deformation of musculoskeletal system. In: *ALGORITMY 2012, 19th Conference on Scientific Computing*, Vysoké Tatry, Slovakia, 09–14 Sep 2012, pp. 239–249. Slovak Univ Technology, Bratislava (2012)
17. Kohout, J., et al.: Patient-specific fibre-based models of muscle wrapping. *Interface Focus* **3**(2), 20120062 (2013). <https://doi.org/10.1098/rsfs.2012.0062>
18. Kohout, J., Kukačka, M.: Real-time modelling of fibrous muscle. *Comput. Graph. Forum* **33**(8), 1–15 (2014). <https://doi.org/10.1111/cgf.12354>
19. Kohout, J., Kukačka, M.: Real-time modelling of fibrous muscle. In: *Computer Graphics Forum* [18], pp. 1–15. <https://doi.org/10.1111/cgf.12354>
20. Kohout, J., Cholt, D.: Automatic reconstruction of the muscle architecture from the superficial layer fibres data. *Comput. Methods Programs Biomed.* **150**, 85–95 (2017). <https://doi.org/10.1016/j.cmpb.2017.08.002>
21. Kohout, J., Cholt, D.: Automatic reconstruction of the muscle architecture from the superficial layer fibres data. In: *Computer Methods and Programs in Biomedicine* [20], pp. 85–95. <https://doi.org/10.1016/j.cmpb.2017.08.002>
22. Kohout, J., Clapworthy, G.J., Martelli, S., Viceconti, M.: Fast realistic modelling of muscle fibres. In: Csurka, G., Kraus, M., Laramee, R.S., Richard, P., Braz, J. (eds.) *VISIGRAPP 2012*. CCIS, vol. 359, pp. 33–47. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38241-3_3
23. Kojic, M., Mijailovic, S., Zdravkovic, N.: Modelling of muscle behaviour by the finite element method using Hill's three-element model. *Int. J. Numer. Meth. Eng.* **43**(5), 941–953 (1998). [https://doi.org/10.1002/\(SICI\)1097-0207\(19981115\)43:5<941::AID-NME435>3.0.CO;2-3](https://doi.org/10.1002/(SICI)1097-0207(19981115)43:5<941::AID-NME435>3.0.CO;2-3)

24. Kotsalos, C., Latt, J., Chopard, B.: Bridging the computational gap between mesoscopic and continuum modeling of red blood cells for fully resolved blood flow. *J. Comput. Phys.* **398**, 108905 (2019). <https://doi.org/10.1016/j.jcp.2019.108905>. cited By 0
25. Macklin, M., et al.: Small steps in physics simulation. In: Proceedings of the 18th Annual ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA 2019, pp. 2:1–2:7. ACM, New York (2019). <https://doi.org/10.1145/3309486.3340247>
26. Modenese, L., Gopalakrishnan, A., Phillips, A.: Application of a falsification strategy to a musculoskeletal model of the lower limb and accuracy of the predicted hip contact force vector. *J. Biomech.* **46**(6), 1193–1200 (2013). <https://doi.org/10.1016/j.jbiomech.2012.11.045>
27. Modenese, L., Kohout, J.: Automated generation of three-dimensional complex muscle geometries for use in personalised musculoskeletal models. *Ann. Biomed. Eng.* **48**, 1793–1804 (2020). <https://doi.org/10.1007/s10439-020-02490-4>
28. Müller, M., Heidelberger, B., Hennix, M., Ratcliff, J.: Position based dynamics. *J. Vis. Commun. Image Represent.* **18**, 109–118 (2007). <https://doi.org/10.1016/j.jvcir.2007.01.005>
29. Oberhofer, K., Mithraratne, K., Stott, N.S., Anderson, I.A.: Anatomically-based musculoskeletal modeling: prediction and validation of muscle deformation during walking. *Vis. Comput.* **25**(9), 843–851 (2009). <https://doi.org/10.1007/s00371-009-0314-8>
30. Otake, Y., et al.: Patient-specific skeletal muscle fiber modeling from structure tensor field of clinical CT images. In: Descoteaux, M., Maier-Hein, L., Franz, A., Jannin, P., Collins, D.L., Duchesne, S. (eds.) MICCAI 2017. LNCS, vol. 10433, pp. 656–663. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66182-7_75
31. Pellikaan, P., et al.: Evaluation of a morphing based method to estimate muscle attachment sites of the lower extremity. *J. Biomech.* **47**(5), 1144–1150 (2014). <https://doi.org/10.1016/j.jbiomech.2013.12.010>
32. Romeo, M., Monteagudo, C., Sánchez-Quirós, D.: Muscle and fascia simulation with extended position based dynamics. *Comput. Graph. Forum* **39**(1), 134–146 (2019). <https://doi.org/10.1111/cgf.13734>
33. Shao, X., Liao, E., Zhang, F.: Improving SPH fluid simulation using position based dynamics. *IEEE Access* **5**, 13901–13908 (2017). <https://doi.org/10.1109/ACCESS.2017.2729601>
34. Valente, G., Martelli, S., Taddei, F., Farinella, G., Viceconti, M.: Muscle discretization affects the loading transferred to bones in lower-limb musculoskeletal models. *Proc. Inst. Mech. Eng. Part H J. Eng. Med.* **226**(2), 161–169 (2012)
35. Van Sint Jan, S.: Introducing anatomical and physiological accuracy in computerized anthropometry for increasing the clinical usefulness of modeling systems. *Crit. Rev. Phys. Rehabil. Med.* **17**, 149–174 (2005). <https://doi.org/10.1615/CritRevPhysRehabilMed.v17.i4.10>
36. Viceconti, M., Clapworthy, G., Van Sint Jan, S.: The virtual physiological human - a European initiative for in silico human modelling. *J. Physiol. Sci.: JPS* **58**, 441–446 (2008). <https://doi.org/10.2170/physiolsci.RP009908>
37. Weinhandl, J.T., Bennett, H.J.: Musculoskeletal model choice influences hip joint load estimations during gait. *J. Biomech.* **91**, 124–132 (2019). <https://doi.org/10.1016/j.jbiomech.2019.05.015>
38. Zhao, Y., et al.: Laplacian musculoskeletal deformation for patient-specific simulation and visualisation. In: 2013 17th International Conference on Information Visualisation. IEEE (2013). <https://doi.org/10.1109/iv.2013.67>

9.9 Geometry Algebra and Gauss Elimination method for solving a linear system of equations without division

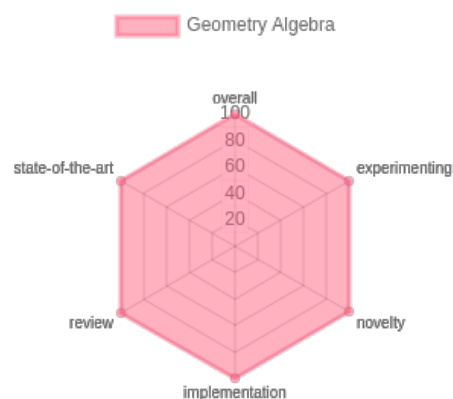
The results of this paper [97] can be applied to the RBF equation system solving if Gaussian elimination (GE) is used. The research introduces an innovative approach to GE that eliminates the need for division operations. This method is significant in computational contexts where division is expensive, not optimized, or inconvenient.

The GE process includes division operations, generating computational expense and numerical instabilities. The division avoidance approach is produced, reducing computational cost and maintaining numerical stability.

The proposed method involves additional multiplication and addition steps to substitute for division operations. By avoiding division, the process reduces computational expense, with only a slight increase in execution time compared to the standard approach on modern computers. Experiments were conducted using the Hilbert matrix, which is known for its numerical instability during inversion. The paper compares the proposed method with the traditional GE and another approach for reducing division operations. The results demonstrate that the new method maintains numerical stability.

The proposed method maintains accuracy and stability while being slightly slower than the GE method on a traditional PC because the division on this hardware is already optimised. The performance and correctness of the approach were evaluated using the normalized Frobenius norm and conditionality of the inverses.

This paper contributes to computational mathematics by offering an alternative approach to a fundamental mathematical process, potentially impacting areas where division operations are not feasible.



Publication [97]:

CERVENKA, M.: Geometry Algebra and Gauss Elimination method for solving a linear system of equations without division. *Informatics 2022, IEEE proceedings*. 2022, pp. 55–59. Available from DOI: <https://doi.org/10.1109/Informatics57926.2022.10083445>. OBD: 43937872, EID: 2-s2.0-851533-55665

Geometry Algebra and Gauss Elimination method for solving a linear system of equations without division

Martin Cervenka
Faculty of Applied Sciences
University of West Bohemia
 Pilsen, Czech Republic
 cervemar@kiv.zcu.cz
 0000-0001-9625-1872

Abstract—This paper aims to calculate the Gaussian elimination method without division operation, which is useful for cases where the division operation is considerably expensive, not optimised or inconvenient. To substitute the division, more multiplication steps are performed. The division is completely avoided, reaching only 7% longer execution time on a modern computer. Memory savings and also less multiplication has been reached in comparison to the state-of-the-art approach.

Index Terms—Gaussian Elimination Division-free Linear equation system Geometry algebra

I. INTRODUCTION

The Linear system of equations $\mathbf{Ax} = \mathbf{b}$ is defined by matrix \mathbf{A} of size $N \times N$, where A_{ij} is a coefficient of i^{th} equation and j^{th} independent variable. The b_i is the i^{th} equation constant coefficient (also called right-hand side of the equation) and x_j are the dependent variable to solve. If the matrix is rectangular, the system is over-determined/under-determined. The over-determined system has more rows/equations than columns/independent variables, and the under-determined is the opposite. For the sake of simplicity, only square and regular (with matrix \mathbf{A} of size $N \times N$ and order N) equation systems will be considered.

II. GAUSS ELIMINATION METHOD

Although the Gauss elimination method (GEM) with the complexity of $O(N^3)$ [1] is not optimal in theory [2], it is commonly used to solve reasonably small matrices, where its higher asymptotic complexity is suppressed. The basic idea behind the Gaussian elimination method is the matrix conversion from its initial form to the identity matrix form, using only the following operations:

- Multiplying a matrix row with a nonzero scalar value
- Adding a row to another row (or its arbitrary nonzero multiple)

To solve for vector \mathbf{x} , the \mathbf{b} will be transformed in the same fashion as \mathbf{A} (often, \mathbf{b} is written inside \mathbf{A} , separated by a vertical line). Finding the inversion of the \mathbf{A} matrix is the same

This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic, project SGS-2022-015

problem as the linear equation solution using orthogonal unit vectors \mathbf{b} . All of them will form the identity matrix. Another way to describe the problem is to apply the same operations on the identity matrix, as there were applied on the original matrix \mathbf{A} .

The procedure is such that a combination of allowed operations zeroes the first column below the diagonal, then the second column, the third and so on. In the next step (backward cycle), the part above the diagonal in the last column is zeroed, then the second to last till the second column. The final step is multiplying all rows to obtain the ones on the diagonal.

A "shortcut" also solves the equation system but does not produce an inverse matrix. The shortcut lies in the backward cycle that can be solved directly from the last variable to the first. Further in the text, this shortcut will be called the "half-way" Gaussian elimination method, which will be stated explicitly. Otherwise, the complete Gaussian elimination method is meant.

In the case of the rectangular matrix, the approach would be the same as described here for the square matrix. However, the same restrictions apply to this approach as to the original GEM method (mainly the rank of the matrix should equal the number of variables in obtaining a single solution).

A. Partial pivoting

Partial pivoting is the computational step performed in the forward cycle to improve computational accuracy. When a new column is zeroed below the diagonal, the remaining rows (with higher indices than the column index) are swapped so that the pivot will have interesting properties. It is appropriate to select the pivot with the highest absolute value so that the roundoff errors will be less significant [3].

B. Algorithm

The Gaussian elimination algorithm is described in Listing 1. The \mathbf{a} denotes the one-based indexed matrix \mathbf{A} , with the right-hand side column vector appended to the matrix (so its final dimension is $N \times (N + 1)$).

```

for k := 1 to n-1 do
{
  # Finds the maximum pivot, #
  # partial pivoting #
  ii := max_arg(abs(a[i, k]), i=k..n);
  if abs(a[ii, k]) <= eps
    ERROR ("Matrix_is_singular!");
    exit;
  swap_rows (k, ii); # swaps k, ii #
  # for all rows below pivot #
  for i := k + 1 to n do
  { # for all remaining elements #
    # in the current row #
    for j := k + 1 to n+1 do
      a[i, j] := a[i, j] - a[k, j]
        * (a[i, k] / a[k, k]);
    # fill lower triangular matrix #
    # with zeros if needed #
    a[i, k] := 0
  }
}
for i := n downto 1 do # backward loop #
{
  s:=0;
  for j := i+1 to k
    s := s + a[i, j] * x[j];
  x[i] := (a[i, n+1] - s) / a[i, i];
}

```

Listing 1. Gaussian elimination algorithm [4] (modified). The backward cycle does not edit the matrix but effectively calculates the result ("half-way" Gaussian elimination method).

III. PROPOSED APPROACH

The main problem in the Gaussian elimination method is that the "Gaussian elimination to solve a system of n equations for n unknowns requires $\frac{n(n+1)}{2}$ divisions, $\frac{2n^3+3n^2-5n}{6}$ multiplications, and $\frac{2n^3+3n^2-5n}{6}$ subtractions" [5]. Be aware that the mentioned approach is just a "half-way" Gaussian elimination because it will create just the upper triangular matrix and solve it. In the case of the complete Gaussian elimination (and able to obtain matrix inverse), it will be even more costly than that. Luckily, the division count can be reduced or, even better, unnecessary.

Skala's article [6] proposed a projective extension of the Euclidean space to reduce the number of division operations; however, the right-hand side vector can be used instead of the homogenous coordinate for this purpose, discarding the homogenous coordinate multiplication step and saving some

memory (not significant in asymptotic case, though). The only advantage is that there are two variables instead of a single one, allowing the possibility for better numerical stability.

The idea is that in each step, every other row then a selected one b can be multiplied by the pivot value from the row p , using the factor a_{pp} . Then, the row multiple can be added more simply, because the factor will be a whole number instead of a real one. The idea will work for rational numbers and irrational ones, as the irrational number can be used as a factor for other rows.

Operation	Complexity
Gauss. elim.	
/ (Division)	$\frac{1}{2}(N^3 - N)$
* (Multiplication)	$\frac{1}{2}(N^3 - N)$
- (Subtraction)	$\frac{1}{2}(N^3 - N)$
∧ (Bitwise AND)	0
∨ (Bitwise OR)	0
⊕ (Bitwise XOR)	0
Memory	N^2
Skala's approach	
/ (Division)	0
* (Multiplication)	$N^3 + 2N^2 - 3N$
- (Subtraction)	$\frac{3}{2}(N^3 + 2N^2 + 3N)$
∧ (Bitwise AND)	$\frac{1}{2}(N^3 + 2N^2 - 3N)$
∨ (Bitwise OR)	$\frac{1}{2}(N^3 + 2N^2 - 3N)$
⊕ (Bitwise XOR)	$\frac{1}{2}(N^3 + 2N^2 - 3N)$
Memory	$N^2 + N$
Proposed approach	
/ (Division)	0
* (Multiplication)	$N^3 + N^2 - 2N$
- (Subtraction)	$\frac{3}{2}(N^3 + 2N^2 + 3N)$
∧ (Bitwise AND)	$\frac{1}{2}(N^3 + 2N^2 - 3N)$
∨ (Bitwise OR)	$\frac{1}{2}(N^3 + 2N^2 - 3N)$
⊕ (Bitwise XOR)	$\frac{1}{2}(N^3 + 2N^2 - 3N)$
Memory	N^2

TABLE I
COMPARISON OF OPERATION COUNT FOR ALL OF THE COMPLETE GAUSSIAN ELIMINATION METHODS. THE PROPOSED APPROACH IMPROVES THE MEMORY SIZE AND NUMBER OF MULTIPLICATION OPERATIONS OVER SKALA'S APPROACH [6].

A. Example of direct GEM

Let us assume an example linear equation system:

$$\begin{aligned} 2x_1 - x_2 &= 5 \\ 3x_1 - 4x_2 &= 6 \end{aligned} \quad (1)$$

The classical Gauss Elimination method would look like the following:

$$\begin{aligned} \left[\begin{array}{cc|c} 2 & 1 & 5 \\ 3 & 4 & 6 \end{array} \right] -\frac{3}{2}\mathbf{I} &\sim \left[\begin{array}{cc|c} 2 & 1 & 5 \\ 0 & \frac{5}{2} & -\frac{3}{2} \end{array} \right] -\frac{2}{5}\mathbf{II} \\ \sim \left[\begin{array}{cc|c} 2 & 0 & \frac{28}{5} \\ 0 & \frac{5}{2} & -\frac{3}{2} \end{array} \right] \times \frac{1}{2} &\sim \left[\begin{array}{cc|c} 1 & 0 & 2.8 \\ 0 & 1 & -0.6 \end{array} \right] \end{aligned} \quad (2)$$

In the first step, the $\frac{3}{2}$ of the first row is subtracted from the second row to eliminate A_{21} . In the second step, $\frac{2}{5}$ of the second row is subtracted from the first row to eliminate A_{12} . Finally, both rows are multiplied to get the identity matrix on the left. The solution to the problem can be found on the right.

B. Example of our approach

Let us assume the same example as it was in the III-A section with linear equation system $2x_1 - x_2 = 5$ and $3x_1 - 4x_2 = 6$. The issue is the division has to be done in every computation step. To avoid that problem, a different method is proposed, where each row $i, i \neq p$ is multiplied by the pivot (excluding the row containing the pivot) from a chosen row p (element a_{pp}) as follows:

$$\begin{aligned} \left[\begin{array}{cc|c} 2 & 1 & 5 \\ 3 & 4 & 6 \end{array} \right] \times 2 &\sim \left[\begin{array}{cc|c} 2 & 1 & 5 \\ 6 & 8 & 12 \end{array} \right] -3\mathbf{I} \\ \sim \left[\begin{array}{cc|c} 2 & 1 & 5 \\ 0 & 5 & -3 \end{array} \right] \times 5 &\sim \left[\begin{array}{cc|c} 10 & 5 & 25 \\ 0 & 5 & -3 \end{array} \right] -\mathbf{II} \\ \sim \left[\begin{array}{cc|c} 10 & 0 & 28 \\ 0 & 5 & -3 \end{array} \right] \times \frac{1}{10} &\sim \left[\begin{array}{cc|c} 1 & 0 & 2.8 \\ 0 & 1 & -0.6 \end{array} \right] \end{aligned} \quad (3)$$

This method multiplies each row beforehand, so there is no need for division. The approach required only N divisions at the end to determine the result vector $(2.8, -0.6)$. The main issue here is that the matrix elements may grow/decay exponentially, making the approach useless for practical use on the computer. This phenomenon can be seen in Equ. (3), where diagonal elements grow, for bigger matrix will grow even more.

IV. IMPLEMENTATION

The proposal has been tested using the C++ programming language. Skala's row normalization [6] step has been adopted, which can also be done without division, mainly using bitwise operations.

```
#define FLT_MASK 0x800FFFFFFFFFFFFFFFL
#define EXP_MIDDLE 0x3FF0000000000000L
// everything else than the exponent part
// of the "b" vector
unsigned right_data = a[i][N] & FLT_MASK;
//exponent part of the IEEE-754
// double precision variable
unsigned right_exp = (right_data ^
    a[i][N]) - EXP_MIDDLE;
for(int j=k; j<N; j++){
    //exponent part of each element
    // in the matrix row
    unsigned data = a[i][j] & FLT_MASK;
    //shift the exponent to the opposite
    //direction of the b vector exponent
    a[i][j] = data |
        ((data^a[i][j])-right_exp);
}
// make exponent of the
// "b" vector element "zero"
a[i][N] = right_data | EXP_MIDDLE;
```

Listing 2. Exponent normalization step, bitwise implementation, for every operation (OR, XOR, AND and subtraction), there is one of them performed inside the loop and one outside.

The normalize function deals with growing/decaying pivots in such a way that no division is needed. It is done by the exponent modification, using the same modification for each row element. As seen on Listing 2, the modification can be easily done on IEEE 754 floating point values, using just bitwise operations and a subtraction.

The a variable represents the matrix \mathbf{A} , last column of the a contains also right-hand side of the equation (column vector \mathbf{b}). FLT_MASK will mask bits belonging to the exponent part of the IEEE 754 floating point value, EXP_MIDDLE is a value with zero exponent (exponent in IEEE 754 is shifted by 127).

V. EXPERIMENTAL RESULTS

Experiments were performed on the Hilbert matrix, where each element of the matrix is given by:

$$H_{ij} = \frac{1}{i+j-1} \quad (4)$$

This particular matrix is well-known for its numerical instability during its inversion. The condition numbers of the matrix increase significantly with increasing N (see Tab. II). For the conditionality, 2-norm condition number of a matrix with respect to inversion was used ($\|\mathbf{A}\|_2 \cdot \|\mathbf{A}^{-1}\|_2$).

Luckily, the Hilbert inversion matrix is well-known analytically [7] and can be evaluated by the Equ. (5), so the p -norm can be calculated more precisely using this inversion.

9. Author's contribution

M. Cervenka · Geometry Algebra and Gauss Elimination method for solving a linear system of equations without divis...

N	Conditionality	N	Conditionality
1	1.0000e+00	8	1.5257e+10
2	1.9281e+01	9	4.9315e+11
3	5.2406e+02	10	1.6024e+13
4	1.5513e+04	11	5.2227e+14
5	4.7660e+05	12	1.7515e+16
6	1.4951e+07	13	3.3441e+18
7	4.7536e+08	14	6.2008e+17

TABLE II
CONDITION NUMBER OF THE HILBERT MATRIX FOR GIVEN DIMENSION N .

The normalization step has been tested. It has been shown that the normalization step does not need to be performed in each computational step, but only if necessary (if the exponent is big/small enough to make it worthwhile). This approach is on average about 7% slower than the original, keeping all of the advantages (no division, no additive memory), see Fig. 1, Fig. 2 and Fig. 3.

$$H_{ij}^{-1} = (-1)^{i+j} (i+j-1) \binom{n+i-1}{n-j} \binom{n+j-1}{n-i} \binom{i+j-2}{i-1}^2 \quad (5)$$

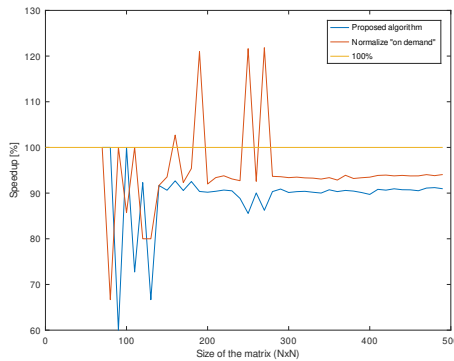


Fig. 1. Running speed of the proposed algorithm in proportion to original Gaussian elimination method. The Hilbert matrix inverse has been performed of the given size. The algorithm is slower than the original one, about 10%. It should be noted that the results may be inaccurate for higher N . The peaks are caused by the low computation time of all methods.

In Fig. 1 it can be seen that the proposed algorithm is about 10% slower than the original algorithm. The correctness of the execution can be seen in Fig. 2, where all of the algorithms provide results with (nearly) the same conditionality. Moreover, these results are the same as in Tab. II, caused by

the fact that the conditionality of the matrix is equal to the conditionality of its inverse due to the commutativity property.

Forward cycle
<p>Gaussian elimination</p> <pre>for(int j=k+1; j<=n+1; j++) a[i][j] = a[i][j] - a[i][k] * a[k][j] / a[k][k]</pre>
<p>Skala's algorithm [6]</p> <pre>for(int j=k+1; j<=n+1; j++) a[i][j] = a[i][j] * a[k][k] - a[i][k] * a[k][j]; a[i][HOMOG] = a[i][HOMOG] * a[k][k]; normalize(a, k+1, n)</pre>
<p>Proposed algorithm</p> <pre>for(int j=k+1; j<=n+1; j++) a[i][j] := a[i][j] * a[k][k] - a[i][k] * a[k][j]; normalize(a, k+1, n)</pre>
Backward cycle
<p>Gaussian elimination</p> <pre>for(int j=k+1; j<=n+1; j++) a[i][j] = a[i][j] - a[i][k] * a[k][j] / a[k][k]</pre>
<p>Skala's algorithm [6]</p> <pre>for(int j=k+1; j<=n+1; j++) a[i][j] = a[i][j] * a[k][k] - a[i][k] * a[k][j]; a[i][i] := a[i][i] * a[k][k] - a[i][k] * a[k][i]; a[i][HOMOG] = a[i][HOMOG] * a[k][k]; normalize(a, k+1, n)</pre>
<p>Proposed algorithm</p> <pre>for(int j=k+1; j<=n+1; j++) a[i][j] := a[i][j] * a[k][k] - a[i][k] * a[k][j]; a[i][i] := a[i][i] * a[k][k] - a[i][k] * a[k][i]; normalize(a, k+1, n)</pre>

TABLE III
COMPARISON OF THE GAUSSIAN ELIMINATION ALGORITHM AND ITS MODIFICATIONS. IN THE BACKWARD CYCLE, THE DIVISION-FREE APPROACHES HAVE TO MODIFY THE DIAGONAL AS WELL.

The output error has been measured in Fig. 3. The normalized Frobenius norm of the difference of the matrix pair has been used:

$$L_2(\mathbf{X}) = \frac{\sum_{i=1}^N \sum_{j=1}^N (H_{ij}^{-1} - X_{ij})^2}{N^2} \quad (6)$$

The \mathbf{H}^{-1} has been computed analytically; see Equ. (5).

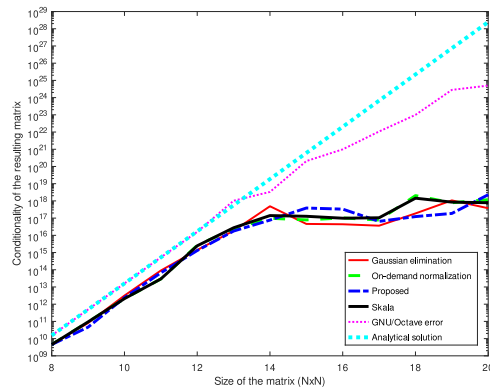


Fig. 2. Conditionality of the resulting inverse matrix. The results show that the conditionality is nearly the same for all algorithms. The differences between algorithms are pointless for $N > 12$ due to the high conditionality. The conditionality should be even higher, as the analytical solution shows. GNU/Octave inverse is approaching due to LU decomposition for matrix inversion.

The difference between the original Gaussian elimination method, Skala's modification [6], and the proposed one is shown on Tab. III.

VI. CONCLUSION & FUTURE WORK

The Gaussian elimination method can be done without division and additive memory requirements, which is particularly useful in scenarios where the division operation is expensive. There will be no division if it is sufficient to obtain the result in projective space or N division in the Euclidean space. The improvement from Skala's publication [6] is that there is no need for storage for homogenous coordinates, saving memory for N variables and saving $N^2 - N$ multiplication operations. Despite these facts, the desired running time improvement has not been reached. Considering execution time, the division operation probably causes this to be no longer crucial.

Andrilli et al. state, "The partial pivoting technique is used to avoid roundoff errors that could be caused when dividing every entry of a row by a pivot value that is relatively small in comparison to its remaining row entries." [3]. Because division is unnecessary for this paper, the future task is to explore possibilities when partial pivoting will be avoided. It is also true that multiplication is still present. The true challenge will be to prevent the "nearly" zero factor, which may cause numerical instabilities.

This approach is particularly useful when the inversion is made not over a field of real numbers (matrix containing real numbers), but over a ring, where not all elements have multiplicative inverses, so dividing, in general, is impossible. However, the application of this approach is also future work to be done.

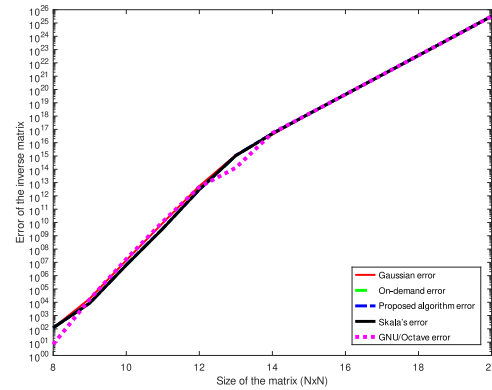


Fig. 3. The result difference between analytical inversion and a computed one. The differences between the algorithms are negligible.

ACKNOWLEDGMENTS

The author thanks students and colleagues at the University of West Bohemia, Plzen. Special thanks belong to Vaclav Skala for his critical comments and discussion. This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic, project SGS-2022-015.

REFERENCES

- [1] X. G. Fang and G. Havas, "On the worst-case complexity of integer gaussian elimination," in *Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation*, ser. ISSAC '97. New York, NY, USA: Association for Computing Machinery, 1997, p. 28–31. [Online]. Available: <https://doi.org/10.1145/258726.258740>
- [2] V. Strassen, "Gaussian Elimination is Not Optimal (1969)," in *Ideas That Created the Future: Classic Papers of Computer Science*. The MIT Press, 02 2021. [Online]. Available: <https://doi.org/10.7551/mitpress/12274.003.0032>
- [3] S. Andrilli and D. Hecker, "Chapter 9 - numerical techniques," in *Elementary Linear Algebra (Fifth Edition)*, 5th ed., S. Andrilli and D. Hecker, Eds. Boston: Academic Press, 2016, pp. 607–666. [Online]. Available: www.sciencedirect.com/science/article/pii/B9780128008539000098
- [4] R. Bronson and G. B. Costa, "Chapter 3 - the inverse," in *Matrix Methods (Fourth Edition)*, 4th ed., R. Bronson and G. B. Costa, Eds. Academic Press, 2021, pp. 93–129. [Online]. Available: www.sciencedirect.com/science/article/pii/B9780128184196000034
- [5] N. S. Rani, "Nondeterministic procedure of solving simultaneous equations," pp. 49–54, 02 2013. [Online]. Available: www.researchinventy.com/papers/v2i3/G023049054.pdf
- [6] V. Skala, "Modified gaussian elimination without division operations," *AIP Conference Proceedings*, vol. 1558, no. 1, pp. 1936–1939, 2013. [Online]. Available: aip.scitation.org/doi/abs/10.1063/1.4825912
- [7] K. Neshat, "Proof that the hilbert matrix is invertible with integer entries," 05 2020. [Online]. Available: www.researchgate.net/publication/341215305_Proof_that_the_Hilbert_Matrix_is_Invertible_with_Integer_Entries

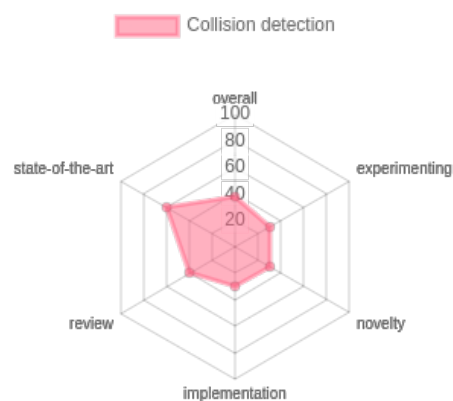
9.10 Collision detection and response approaches for computer muscle modelling

In our research, addressing collision detection and response presented significant challenges (refer to the second point in Section 9.8 and also see Section 9.4), primarily due to the lack of an effective solution initially. The study highlighted in this article [72] sought to improve collision handling by substituting the original voxelization method with a scalar distance field (SDF) approach.

Initially, we considered two alternatives: the scalar distance field (SDF) and the flexible collision library (FCL). Our findings indicated that SDF was the superior option, offering enhanced computational speed and reduced memory usage without compromising quality.

The implementation of SDF in this study markedly enhanced our collision handling technique. Its success is primarily attributed to its lower discretization resolution than voxelization, with trilinear interpolation in each voxel providing sufficient accuracy.

Furthermore, we successfully resolved a persistent issue where muscles would get stuck in joints due to the coarse surface of the voxel grid, hindering smooth movement out of the joint. The new method also significantly reduces the likelihood of muscles entering narrow spaces between bones.



Publication [72]:

HAVLICEK, O.; CERVENKA, M.; KOHOUT, J. Collision detection and response approaches for computer muscle modelling. *Informatics 2022, IEEE proceedings*. 2022, pp. 120–125. Available from DOI: <https://doi.org/10.1109/Informatics57926.2022.10083500>. EID: 2-s2.0-85153333554, OBD: 43937869

Collision detection and response approaches for computer muscle modelling

Ondrej Havlicek

Department of Computer Science
and EngineeringFaculty of Applied Sciences,
University of West BohemiaPilsen, Czech Republic
0000-0002-6944-7084

Martin Cervenka

Department of Computer Science
and EngineeringFaculty of Applied Sciences,
University of West BohemiaPilsen, Czech Republic
0000-0001-9625-1872

Josef Kohout

NTIS - New Technologies
for the Information SocietyFaculty of Applied Sciences,
University of West BohemiaPilsen, Czech Republic
0000-0002-3231-2573

Abstract—Computer muscle modelling is used for many purposes, from injury recovery and treatment of chronic diseases to disease prediction. These predictions often involve computing the muscle's internal forces to determine further how fast something may happen (e.g. how quickly the muscle joint wears out). During the simulation of such a model, collisions of soft and rigid bodies inevitably occur. This paper tests various state-of-the-art collision handling methods: voxelisation, one using Signed Distance Fields and one based on Bounding Volume Hierarchies. These methods are tested in the context of muscle modelling with the previously proposed position-based dynamics approach. Compared to the other options, using the Discregrid library for Signed Distance Field generation shows the best results, mainly due to its accuracy to the speed of execution ratio. In contrast to the current system, visually pleasant improvements are significant.

Index Terms—Collision detection, Discregrid, Signed Distance Fields, Fast Collision Library, Voxelization, Muscle modelling, Position-based dynamics

I. INTRODUCTION

With diseases such as osteoporosis (prevalence up to 34.3% for females 50 years old or more in the USA, and about 10% among the average population [1]), where bone density decreases or osteoarthritis disease, where there may even emerge the need for joint replacement due to bone structure degeneration, the desire for a realistic musculoskeletal model arises. Such a model could be used to estimate various forces acting around the muscles and bones, which then may be used for prediction and prevention of the named diseases and many more.

Some state-of-the-art models (e.g. [2], [3]) use a generic model (from cadaveric studies or measured on completely different patients and edited with lengthy and exhaustive manual labour). Realistically, the human body varies greatly; with this diversity, the need for patient-specific models becomes increasingly apparent. This presumption leads to a new method, a statistical model. An example would be a statistical model built from 26 patients using Principal Component Analysis created

The authors would like to thank their colleagues and students at the University of West Bohemia for their discussions and suggestions. This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic, project SGS-2022-015 New Methods for Medical, Spatial and Communication Data.

by [4]. Another personalized model based on Position-Based Dynamics (PBD) was presented by [5]. This particular model exhibits promising results in terms of model simplicity, speed and accuracy for biomechanical studies, in comparison with musculoskeletal models used commonly in practice. However, in some cases, e.g. extreme flexion around the hip joint, the muscles in this model behave unrealistically. The authors suggest it might be due to the used collision handling system.

The aim of our research, therefore, was to propose a new collision detection (CD) and response (CR) mechanism that would behave adequately even in various extreme scenarios.

In this paper, we present the results of our analysis of the current CD & CR mechanism used in the PBD muscle modelling by [5], and propose two principally different approaches for CD and a couple of minor improvements for CR. The proposed method surpasses the former approach (based on voxelisation) in accuracy, mainly around the problematic hip joint area where muscles no longer get unrealistically stuck.

II. POSITION BASED DYNAMICS APPROACH

In the PBD framework presented by [5], a scene exists consisting of a set of bones, each of which is represented by a triangular mesh surface and has an associated time-dependent transformation describing its movement, and a muscle, also represented by a triangular mesh surface. Each vertex of the muscle is interpreted as a PBD primitive node, having associated mass and velocity, and a set of constraints restricting the freedom of the movement of these points during the simulation. The constraints represent external forces acting on the muscle, including gravitation and fixation to a bone attachment area, as well as internal forces, including local shape and volume preservation.

Each of these constraints can be interpreted as a cost function, resulting in a nonlinear system of differential equations, for which the PBD tries to find their global optimum in the sense of gradient descent using an iterative Gauss-Seidel solver [6]. In each iteration, every PBD node (i.e., a vertex of the triangular muscle surface) is moved from its original position \mathbf{x} to a position \mathbf{p} , satisfying the constraints. This new position may end up inside a bone, which means a collision has occur-

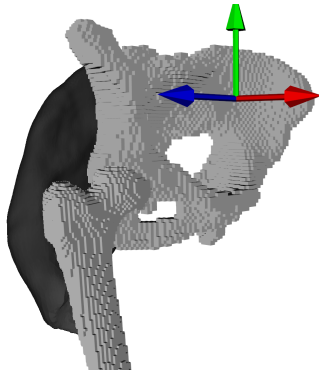


Fig. 1. An example of the voxelisation structure in the pelvic area [5].

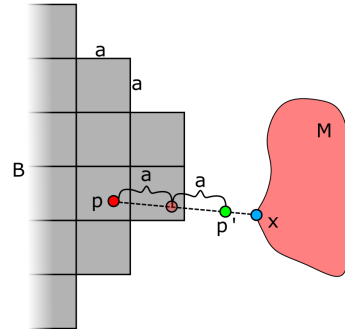


Fig. 2. Collision response mechanism of the voxelisation approach: the point is moving from the desired position p towards its initial position x using the constant step of the voxel size a until it reaches free space (position p').

red. These collisions need to be detected and resolved after each solver iteration, i.e. the new position p of each node gets corrected to a noncolliding position p' .

A. Collision handling

The collision handling in the work [5] was done by creating a voxelised model (a uniform grid) of the bones, as illustrated in Fig. 1. According to the authors, the reasons for choosing this approach were its simplicity and feasible time requirements.

The CD had been designed to further satisfy the run time and memory needs via leaving the bone collision models in their initial rest-pose, while the nodes to test for collisions after each solver iteration would get inversely transformed to the rest-pose coordinate system, then checked whether or not they are contained in any of the bone voxels. If so, then a collision would get detected and the CR process would begin. The result of the CR would then get transformed back to the visual model coordinates. Otherwise, no collision would occur.

The CR is a process where given voxel size a , the position p of the colliding point is incrementally set by the distance of a towards the original position x , until it is no longer contained in any of the bone voxels and therefore not colliding. The final position is the denoted position p' . The process of CD & CR can be seen in Fig. 2.

III. PROBLEMS IDENTIFIED

A. Tunnelling effect

The main issue to consider is that the discrete bone movement may be too fast to simulate the behaviour properly. Consider a femur bone performing the flexion. When it rotates about just 2° (a typical step in simulations), the displacement of the distal part of this bone is nearly 3 cm (considering the average length of the bone being 41.61 cm [7]). Such a displacement might result in muscle penetrating the bone to the other side, as illustrated in Fig. 3.

B. Unavoidable collisions

Due to modelling inaccuracies, the geometrical models of bones and muscles typically slightly penetrate even at the beginning of the simulation, most often at the places where the muscle is connected to the bone. In the original approach, therefore, when the point collides even in its initial position x , its "noncolliding" position p' is obtained by transforming x using the same transformation that was applied to the bone with which the point collides.

The problem is that the voxelisation approach falsely identifies the muscle points closer to the bone surface than the voxel size a as colliding. As a result, these points are bound to the bone and move with it even though they should move freely. For example, the central part of the iliacus muscle is close to the femur head and, therefore, is bound to this bone. During hip flexion, the consequence is dire: a part of the muscle is wound into the free space in the hip joint.

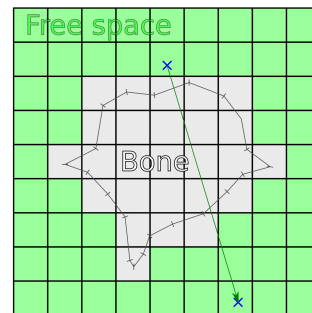


Fig. 3. The muscle vertex (in blue) may penetrate the bone all the way through, because of the big displacement due to the angular motion of the bone.

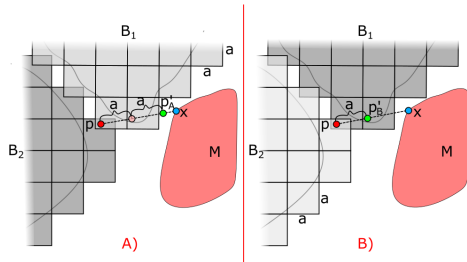


Fig. 4. Collision of a muscle with two different bones. The problem is the collision response as far as both bones force the vertex to move to a different position.

C. Voxelisation problems

Another unsolved problem is a collision with multiple bones simultaneously (see Fig. 4). One of the bones forces the muscle vertex to be in p'_A on left and the second one forces the vertex to the other position p'_B . Applying these CRs consecutively would then lead to only the latter one of them being resolved, which would sometimes leave the point in a collision.

The last problem with voxelisation is that the muscle vertex may get stuck in a gap in between the voxels. As shown in Fig. 5, the blue vertex may move only upwards, but the side-to-side motion is suppressed unless a much bigger force is applied, forcing the blue vertex out. The consequence of the issue is an unrealistic laggy movement and possibly the addition to the problem of muscle wounding in the joint.

IV. PROPOSED METHODS

Considering the simplicity of the voxelisation method, the urge for a more sophisticated collision handling system manifests itself through some of the described problems.

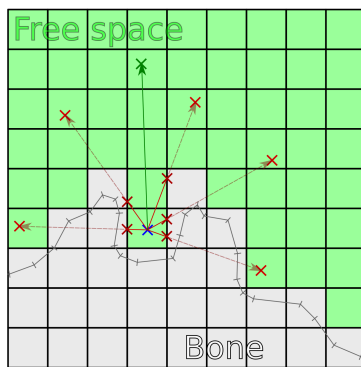


Fig. 5. The blue muscle vertex is stuck in between the voxels occupied by a bone. To get the vertex out sideways, a large force has to be applied.

The general problem seems to be that the voxelisation CR receives on the input only the information about the original and the colliding position of a given node and a rough approximation of the bone surface. Therefore, we propose two different CD & CR approaches, which both provide broader information about the detected collision state, allowing the CR to be more reflective of the underlying physical reality and represent the bone more accurately.

A. Discregrid

Discregrid library is a C++ library based on a Signed Distance Field generated for a bounded finite subspace, allowing to tell for any point in space (x, y, z) the shortest distance and direction to the given triangular surface residing in that subspace. Moreover, the sign of the result adds additional information if the point is inside or outside the surface.

For the method to work, the bounded finite subspace is firstly discretized [8] into a user-defined resolution cuboid grid where each cuboid is a Serendipity type with 32 nodes [9]. At each node, the shortest path to a given surface, as well as the sign, are computed as described by [10], who addresses the problem of discontinuity of a mesh at the edges and the vertices by defining an adequate pseudo-normal for them based on the surrounding angles.

Once the distance, direction and sign are known at every node of the grid, this set of values can be quickly estimated at any point inside any of the cuboids using interpolation by cubic Lagrange polynomials [9].

A collision is detected if the interpolation of a vertex position has a negative sign. This collision can then easily be resolved by pushing the vertex along the shortest distance direction to the surface by this distance instead of the direction it came from.

B. Flexible Collision Library

Flexible Collision Library (FCL) is a CD library written in C++ programming language, providing multiple CD approaches, such as convex polytope-based CD, bounding volume hierarchy (BVH) CD, continuous CD, broad-phase CD, point cloud CD and parallel CD with proximity computation. For speedup, the Sweep and Prune approach over BVH has been used. The library can also handle eight basic shapes: general triangle meshes, convex triangle meshes, spheres, AABB cuboids, cones, cylinders, ellipsoids and capsules.

The library provides so-called managers, which are objects taking care of updating the built structures and detecting collisions among them. The collision information also contains the directional vector between the centroids of the colliding basic shapes, which can be further used for CR.

The dynamic AABB tree collision manager has been used for testing purposes. The bounding box parallel to the euclidean space axes encapsulates each primitive of the object (e.g. the triangle in triangular mesh), which can be checked much faster if it collides with another primitive. The hierarchical structure of these AABB boxes forms a tree, which allows for even faster CD.

C. Collision with multiple bones

Assuming the improved accuracy of the collision models with the proposed methods, collisions with multiple bones should be extremely rare as the bone collision models should not overlap in any way.

Still, in the case of Discregrid, it is possible to virtually add margin to the collision models in hopes to prevent even edge collisions, which could eventually result in this case. For that case, a naive approach of adding the given gradients together was implemented.

V. EXPERIMENTAL RESULTS

The CD approaches have been tested using an existing PBD library originating in [5] on the LHD dataset [11]. This dataset was chosen because it contains the most refined surface triangle meshes of bones and muscles. The advantage is also its public availability. All nonmanifold edges, degenerate triangles and duplicated vertices and the smoothing of all muscle and bone models have also been done by [11] using MeshLab [12]. Dissection data from [13] are also included, containing muscle attachment areas and geometrical paths of superficial fibres.

For the sake of testing, four muscle models have been used: *adductor brevis*, *gluteus maximus*, *gluteus medius* and *iliacus*. Their vertices count are 502, 9878, 5313 and 6931, respectively.

The experiments were conducted on CPU Core-i5-7200U 2.5GHz, GPU NVIDIA GeForce 930MX, Windows 10, and were compiled in C++ release mode.

A. Visual comparison

The visual comparison between all three CD algorithms shows a clear difference. The Discregrid and FCL are superior, resolving the problem of incorrect muscle shape near joints, which is problematic for the voxelisation approach (see Sections III-B and III-C). To observe the difference, see Fig. 6. We note that the FCL provides (not shown in the figure) an even smoother surface than Discregrid, which is not surprising considering that it works directly with the triangular mesh of the bone.

B. Run time

The execution time shows that Discregrid and FCL are slower than the voxelisation approach since these methods are more complicated. As shown in Fig. 7, the FCL is performing poorly, requiring about one second to detect the collisions. However, we note that the FCL approach has been tested using a single thread only, and multi-threading could improve the performance.

The voxelisation has been originally proposed to ensure real-time simulation (at least 30 FPS on a common PC). The slowdown of the voxelisation means that in the same setup, it would read 6 FPS, which is not precisely real-time but can either be rendered directly with the visible delays or precalculated quickly and rendered as a video for other purposes. FCL is slower (approximately 1 frame per 7 seconds), but still applicable.

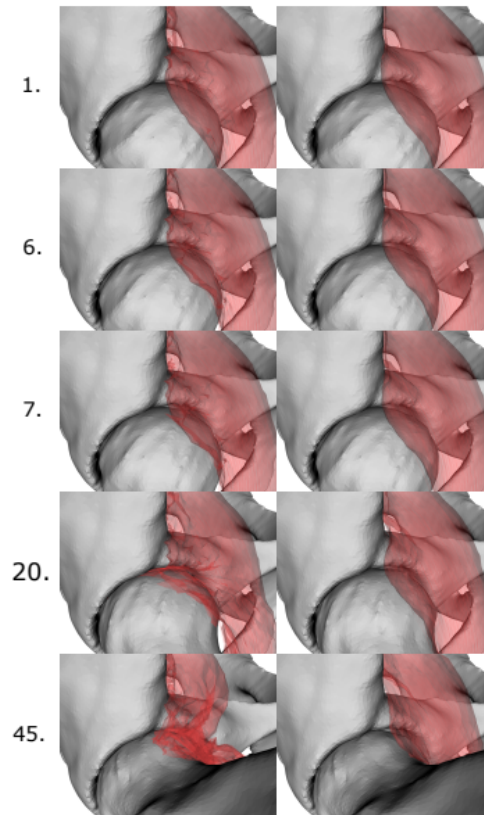


Fig. 6. Iliac muscle deformation in detail. The voxelisation approach (on the left) shreds the surrounding muscle tissue into the hip joint. Discregrid (on the right) has only a slightly rough surface near the joint. The number on the left denotes the number of the time frame in the simulation of flexion.

One of the main differences between the methods is that in the voxelisation and Discregrid approaches, no update of the collision model has to be made. The collision models are also built only for the rigid bones which can then be queried for arbitrary points whether it collides or not and even the collision information at a constant time.

With the FCL library, BVH collision models have to be built for every object in the scene (bones and muscles). In contrast with the prior methods, these collision models need to be updated each iteration - both for the bones in case of rigid translation and for the muscles due to likely geometry deformation. Furthermore, no arbitrary point CD seems to be supported by the management system. As a result, the collision information has to be post-processed to become usable for CR, which hinders this method's time effectiveness.

9.10. Collision detection and response approaches for computer muscle modelling

O. Havlicek et al. · Collision detection and response approaches for computer muscle modelling

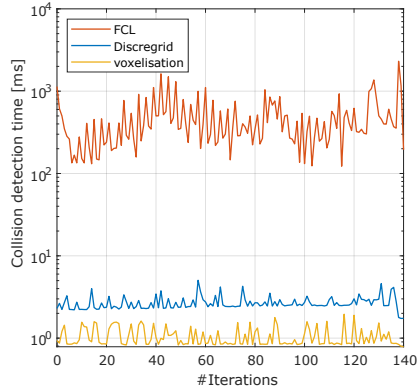


Fig. 7. Run time of collision detection algorithms on the iliacus muscle. The voxelisation is the fastest to execute in general. The Discregrid is approximately 3x slower and the FCL is more than 200x slower on average.

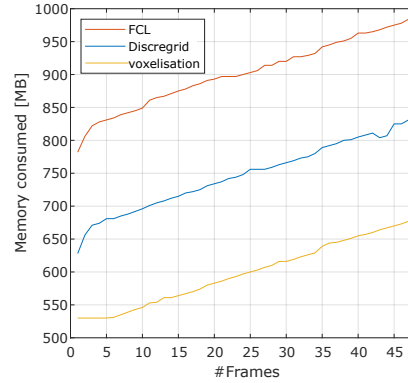


Fig. 8. Memory consumption of all of the algorithms when *iliacus* muscle is used. The voxelisation required the least amount of data, but the Discregrid and FCL seem to be in the same memory complexity level (shifted up only by a constant).

C. Memory consumption

The memory consumption while using the iliacus muscle is shown in Fig. 8. The smallest amount of memory has been used by the original voxelisation algorithm, followed by the Discregrid and FCL algorithms.

Table I shows the results for each tested muscle. All of the consumed memory has been measured using the Visual Studio 2019 16.11.7 Performance Profiler before the first PBD iteration (after the CD structure allocation) and the last PBD iteration (after the whole simulation).

As can be seen in Fig. 8, the different approaches mainly differ in the amount of memory allocated for the collision structures in frame 1. The rise of the memory in the graph reflects the memory consumed by the deformation process while the different CD & CR algorithms used do not seem to have a great impact on the memory used during the course

TABLE I
THE MEMORY USED FOR DIFFERENT CD & CR ALGORITHMS WHILE THE MUSCLE IS DEFORMING. ALL MEASUREMENTS ARE IN MB.

Muscle	First iteration	Last iteration
Voxelization		
Adductor brevis	389	534
Gluteus maximus	526	735
Gluteus medius	458	660
Iliacus	537	688
Discregrid		
Adductor brevis	540	740
Gluteus maximus	677	888
Gluteus medius	609	810
Iliacus	632	839
FCL		
Adductor brevis	685	885
Gluteus maximus	835	1048
Gluteus medius	761	962
Iliacus	786	991

of the simulation. The small deviations in fact represent the number of collisions detected and resolved.

VI. DISCUSSION

The mentioned CD and CR approaches vary significantly in the underlining data structure. The grid approaches (Discregrid and voxelisation) can be simply set to the desired accuracy (changing the grid resolution) to increase the computational speed. This can be done similarly in FCL by allowing various approximations or modifying the narrow-phase solvers, which is also a place for run time improvement of the FCL method in the future since, in this paper, only the default FCL setup has been used.

One big advantage of the FCL is that its hierarchies are designed to be updated during runtime, allowing for CD and CR even among muscles, which could be deformed at any time during the simulation. This is generally hard to achieve using the grid approaches with respect to run time requirements, making them feasible for handling collisions against rigid bones only. This is where the FCL approach could be taken advantage of in the future, e.g. only handling the collisions among muscles. This would, however, result in a heterogeneous CD and CR, increasing the intricacy of the solution, which may be harder to maintain or extend.

On the other hand, Discregrid seems a good successor to voxelisation since it provides feasible accuracy and more information about collision state can be used for CR. CR in the direction of the bone surface results in a more realistic behaviour, resembling the effect of the muscle sliding against the bone surface. This does not change the fact that bone is discretized, which is mitigated to some degree by the Signed Distance Field interpolation, providing better accuracy than voxelisation while running relatively fast compared to the single-threaded FCL implementation.

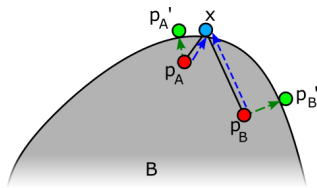


Fig. 9. Simplified illustration of the Discregrid approach respecting the PBD constraints more than the voxelisation approach.

In comparison to the voxelisation approach, where CR tries to revert the colliding point back to the presumably noncolliding original position, the Discregrid approach could be argued to respect the PBD constraints more as illustrated in Fig. 9, where from position x , PBD may move the point either to position p_A or p_B (in red). Voxelisation CR (in blue) tries to return the point toward the original position, reducing the two different PBD constraint results into one point. Discregrid CR (in green) adjust the PBD positions more respectfully (points p'_A and p'_B), merely projecting them onto the bone surface.

Additionally, as previously pointed out, FCL can run multi-threaded, which is currently not viable in our PBD implementation. This is also a possibility for future research.

A. Tunnelling effect

From the Fig. 9, it is easy to imagine that the CR could cause the point to travel to the other side of the bone since the shortest path to the bone surface would reside in that direction. This would cause the same situation as illustrated in Fig. 3. This problem is not directly addressed in this paper, but one possible solution could be to use Discregrid once again and get the gradients of both the former and final positions x and p' , respectively. Then it could be stated that if the angle between them is bigger than some set angle e.g. 135° , the point p' ended up on the other side of the bone and thus the tunnelling occurs. If that happens, the final position could be set to an intersection of the bone surface with the path defined by the points x and p .

B. Unavoidable collisions

To overcome the roughness of voxelisation discretisation, FCL can be utilised since it does not omit any information and does CD at the level of the actual bone triangles and not their grid approximation. Nevertheless, due to the impractical running times of the currently implemented FCL solution, it cannot be used. Instead, the Discregrid results display enough accuracy in the iliacus muscle case and therefore seem a reasonable compromise overall.

Because of the better CD accuracy and the fact, that the method provides a gradient to the bone surface, there is no longer a need for the colliding points to follow the bone transform in case the bone moves into the muscle, as this transform is already encompassed by the gradient.

C. Voxelisation problems

Furthermore, the problem of the muscle being stuck exposed in Fig. 5 has not been fully solved in this paper but is to an acceptable extent mitigated by the greater accuracy of the proposed methods.

VII. CONCLUSION

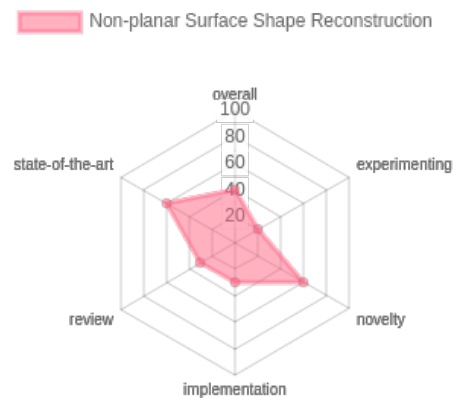
This paper presented the analysis of CD & CR system used by [5] and proposed one new CD approach based on Signed Distance Fields and another one based on Bounding Volume Hierarchies to overcome the shortcomings of a former method in the context of a state-of-the-art PBD muscle modelling approach. Moreover, slight modifications to CR were made to introduce more realistic muscle sliding behaviour. The Discregrid approach, based on the Signed Distance Fields, proved to be a suitable trade-off between improved accuracy and time requirements.

REFERENCES

- [1] S. W. Wade, C. Strader, L. A. Fitzpatrick, M. S. Anthony, and C. D. O'Malley, "Estimating prevalence of osteoporosis: examples from industrialized countries," *Archives of Osteoporosis*, vol. 9, no. 1, p. 182, May 2014. [Online]. Available: <https://doi.org/10.1007/s11657-014-0182-3>
- [2] S. L. Delp, J. P. Loan, M. G. Hoy, F. E. Zajac, E. L. Topp, and J. M. Rosen, "An interactive graphics-based model of the lower extremity to study orthopaedic surgical procedures," *IEEE Transactions on Biomedical Engineering*, vol. 37, no. 8, pp. 757-767, Aug 1990.
- [3] E. Arnold, S. Ward, R. Lieber, and S. Delp, "A model of the lower limb for analysis of human movement," *Annals of biomedical engineering*, vol. 38, pp. 269-79, 12 2009.
- [4] J. Zhang, J. Fernandez, J. Hislop-Jambrich, and T. F. Besier, "Lower limb estimation from sparse landmarks using an articulated shape model," *Journal of Biomechanics*, vol. 49, no. 16, pp. 3875 - 3881, 2016. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0021929016311186>
- [5] J. Kohout and M. Červenka, "Muscle deformation using position based dynamics," in *Biomedical Engineering Systems and Technologies*, X. Ye, F. Soares, E. De Maria, P. Gómez Vilda, F. Cabitza, A. Fred, and H. Gamboa, Eds. Cham: Springer International Publishing, 2021, pp. 486-509.
- [6] J. Bender, M. Müller, and M. Macklin, "A Survey on Position Based Dynamics," in *EG 2017 - Tutorials*, A. Bousseau and D. Gutierrez, Eds. The Eurographics Association, 2017.
- [7] A. Vasilopoulos, G. Tsoucalas, E. Panagouli, G. Trypsianis, V. Thomaidis, and A. Fiska, "Odontoid process and femur: A novel bond in anatomy," *Cureus*, vol. 12, 03 2020.
- [8] B. Angles, D. Rebain, M. Macklin, B. Wyvill, L. Barthe, J. Lewis, J. V. D. Pahlen, S. Izadi, J. Valentin, S. Bouaziz, and A. Tagliasacchi, "VIPER," *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, vol. 2, no. 2, pp. 1-26, jul 2019.
- [9] D. Koschier and J. Bender, "Density maps for improved sph boundary handling," in *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, ser. SCA '17. New York, NY, USA: Association for Computing Machinery, 2017. [Online]. Available: <https://doi.org/10.1145/3099564.3099565>
- [10] A. Bærentzen and H. Aanaes, "Generating signed distance fields from triangle meshes," *Technical University of Denmark*, 01 2002.
- [11] M. Viceconti, G. Clapworthy, and S. V. S. Jan, "The virtual physiological human - a european initiative for in silico human modelling," *The Journal of Physiological Sciences*, vol. 58, no. 7, pp. 441-446, 2008.
- [12] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia, "Meshlab: an open-source mesh processing tool," *Computing*, vol. 1, pp. 129-136, 01 2008.
- [13] S. Van Sint Jan, "Introducing anatomical and physiological accuracy in computerized anthropometry for increasing the clinical usefulness of modeling systems," *Critical Reviews in Physical and Rehabilitation Medicine*, vol. 17, pp. 149-174, 01 2005.

9.11 Non-planar Surface Shape Reconstruction from a Point Cloud in the Context of Muscles Attachments Estimation

The attachment estimation described in the following paper [13] plays a crucial role in muscle modelling. The article mainly tackles the issue described in Section 9.8, already described in that paper. The option of automatically searching for the attachment area according to the information about the closeness of the muscle to the bone proved insufficient because it often happens that the muscle is adjacent to some bone but not attached to it. Hence, additional data is required. Those data already exist as vertices measured at the border of the attachment area as a part of the TLEM 2.0 dataset [18]. The main issue is to find all points inside the attachment area, which would be subsequently fixed to the adjacent bone.



The article tested 15 different curve reconstruction algorithms to reconstruct the whole attachment boundary on the bone, and the subsequent surface bounded by it can be restored by the radial basis function (RBF) approach, which has already been researched before. In this research, we also tested the RBF approach for curve reconstruction, which worked to some extent but was unstable in terms of the parameters selected, which must be chosen carefully with the prior and deep knowledge of the RBF approximation technique.

The major outcome of this research is that the RBF technique is useful and works very well for muscle modelling and muscle attachment area approximation; however, the parameter selection is a crucial part of the success.

Publication [13]:

KOHOUT, J.; CERVENKA, M. Nonplanar Surface Shape Reconstruction from a Point Cloud in the Context of Muscles Attachments Estimation. *Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*. 2022, pp. 236–243. ISBN 978-989-758-555-5. Available from DOI: <https://doi.org/10.5220/0010869600003124>. UT WoS: 000774795400024, OBD: 43936004

Non-planar Surface Shape Reconstruction from a Point Cloud in the Context of Muscles Attachments Estimation*

Josef Kohout^a and Martin Cervenka^b

Faculty of Applied Sciences, University of West Bohemia, Technická 8, Plzeň, Czech Republic

Keywords: Shape Reconstruction, Point Cloud, Multidimensional Scaling, Muscle Attachments Estimation, Fast Marching, Scalar Distance Field.

Abstract: Knowledge of muscle attachments on bones is essential for musculoskeletal modelling. A muscle attachment is often represented by points (in 3D) obtained by a manual digitisation system during dissection. Although this representation suffices for many purposes, sophisticated musculoskeletal models commonly require representing a muscle attachment by a surface patch or at least by a closed boundary curve. In this paper, therefore, we propose an approach to automatic shape reconstruction from such point sets. It is based on iso-contour extraction from a scalar field of distances to geodetics connecting the pairs of points (from the input set) as identified by a state-of-the-art algorithm for 2D curve reconstruction running on the input points transformed to 2D. We investigated the performance of 15 existing state-of-the-art algorithms with public implementations on the TLEM 2.0 data set of muscle attachments. The best results were obtained for the lenz algorithm with just one unacceptable reconstruction when standard projection onto a best-fit plane was used to transform the input 3D points to 2D. The second algorithm was α -shape with three unacceptable reconstructions, whereas in this case, the multidimensional scaling technique was exploited to transform the points.

1 INTRODUCTION

Shape reconstruction from a point cloud is an important computational geometry problem with various applications in computer graphics, computer vision, medical image analysis, pattern recognition, computer-aided design, cultural heritage, and others. During the past decades of research, many algorithms for shape reconstruction have been proposed. Some work with points sampled on the boundary of an object whose shape is to reconstruct, while others work with the points sampled in its interior. Some algorithms can deal (to some extent) with non-uniform or sparse sampling, noise or outliers, while others assume a dense uniform sampling. Some focus on specific kinds of objects, e.g., CAD objects with sharp edges or terrain data. Some require additional information, such as normals in points. However, most importantly, some work in 2D, processing 2D point

clouds to reconstruct the outlining contour of the object, while others work in 3D, processing 3D point clouds to reconstruct the outlining surface of the object. A good survey of algorithms of the former category can be found in (Ohrhallinger et al., 2021). For a survey of the algorithms of the latter category, we refer the reader to (Berger et al., 2016).

In this paper, we propose a novel algorithm for reconstructing a space curve from a set of 3D points. It employs the multidimensional scaling technique (Cox and Cox, 2008) to transform the points from 3D to 2D and then uses a suitable algorithm for 2D curve reconstruction to get the connectivity of the input points.

Motivation for our work lies in muscle attachment estimation. Knowledge of muscle attachments on bones is essential for musculoskeletal modelling. A muscle attachment is often represented by points obtained by a manual digitisation system during dissection. Due to the apparent effort associated with this process, no wonder that the sampling is sparse. Commonly, the points are unordered and exhibit a non-uniform distribution because it is natural to sample the upper side of the attachment area from left to right, then cut off the muscle-tendon unit and sample the lower side from left to right. The points are sub-

^a <https://orcid.org/0000-0002-3231-2573>

^b <https://orcid.org/0000-0001-9625-1872>

*This Work Was Supported by the Meys of the Czech Republic, Project SGS-2019-016.

†Corresponding author

ject to various errors, introduced during the dissection (e.g., movements of the limbs, the ambiguity in defining the attachment boundary) or during the registration process. Sometimes a few points are even sampled in the interior of the attachment area. Figure 1 shows an example of such data.

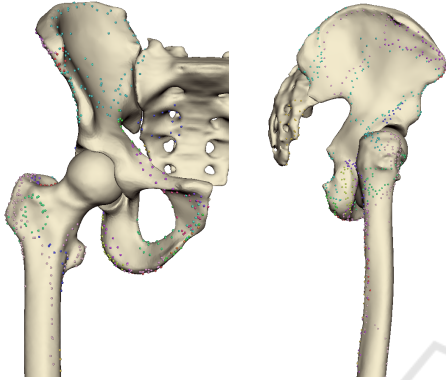


Figure 1: TLEM 2.0 (Carbone et al., 2015) data set containing point clouds defining various muscle attachments on lower limbs.

It can be seen that the attachment areas are only slightly curved. Therefore, a naive approach would be to project the points onto the best-fitted plane (obtained, e.g., by using the least-squares method) and then proceed with some of the existing state-of-the-art algorithms for reconstructing 2D curves. This paper investigates if exploiting the multidimensional scaling (MDS) technique would not improve the obtained results when comparing them with the ground truth. We experimented with 15 different algorithms.

Our other contribution is as follows. Since the 3D models of bones are available, we propose an approach to convert a non-manifold curve, which may result from the process, into a closed manifold one. It is based on iso-contour extraction from the scalar field on the surface of the bones that encodes distances to geodetics connecting the adjacent points of curves.

2 RELATED WORK

The problem of muscle attachments estimation was addressed in (Fukuda et al., 2017). The authors dissected individual muscles in the hip region of eight cadaver specimens while tracing the boundary of the muscle attachments using an optical tracker. The recorded points were manually refined to remove outlier measurements due to tracking noise. In this paper, we investigate if we could get the boundary of an at-

tachment automatically without the necessity of such manual refinement.

In their work, (Kohout and Kukačka, 2014) described a fully automatic algorithm for extraction of a closed region from a triangular model of a muscle, where region boundary is specified by a set of points lying on the muscle surface or in its vicinity. The points had to be specified in an order such that interconnecting every pair of adjacent points by a line segment would produce a closed non-intersecting polyline corresponding to the boundary of the region to extract. However, typical data sets of attachment areas do not comply with this requirement, as shown in Figure 1. In this paper, we investigate how to filter out the input points and order them to satisfy the requirement of this algorithm.

Approximating or interpolating the input points by an analytical function may be considered relevant to this problem. Most suitable seems to be radial basis function (RBF) approximation since the points are scattered and unordered. RBF was used for surface reconstruction of watertight 3D objects by (Carr et al., 2001). It is also commonly used for scattered data approximation in general (Cervenka et al., 2019). In this paper, we address the idea of transforming the input 3D points to 2D, finding the curve there (by exploiting RBF approximation) and returning to 3D space. One option for the points dimension reduction is the multidimensional scaling (MDS) technique (Cox and Cox, 2008), which is widely used in many different scientific fields.

Recently, (Ohrhallinger et al., 2021) designed a benchmark for a comprehensive quantitative evaluation of algorithms for 2D curves reconstruction. It consists of 14 curve reconstruction algorithms, including the recent ones, implemented in C++. Most of these algorithms construct a graph from the points and then filter the outline by some criteria. Most of them are parameterless, but only some are robust to noise and outliers.

3 OUR APPROACH

Given a set $S = (P_i)$ of n points in 3D, sampled on a smooth curve on a non-planar smooth surface, including potentially noise or outliers, our task is to find an ordered set $S' \subseteq S$ such that S' represents a closed non-intersecting space curve. The other points $(S \setminus S')$ which do not lie on the curve are considered as outliers. Figure 2 shows an example of the input data.

Our basic idea is to exploit the multidimensional scaling (MDS) technique (Cox and Cox, 2008) to construct n points Q_i in 2D such that:

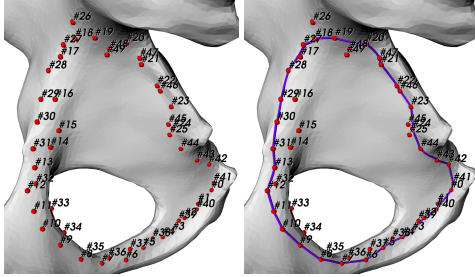


Figure 2: Obturator internus origin data set. Left - the input point set together with the order in which the points were sampled, right - the ground truth closed curve we specified manually according to anatomical atlases.

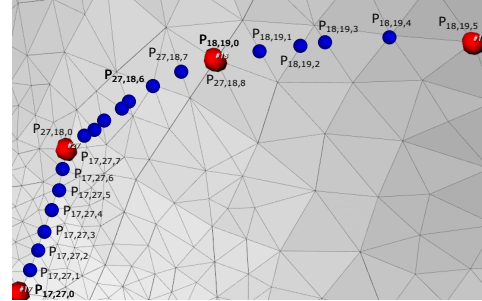


Figure 3: Refined edges $P_{17}, P_{27}, P_{27}, P_{18}$, and P_{18}, P_{19} of the obturator internus origin data set (see Figure 2) on the surface of the pelvis bone.

- every point Q_i is uniquely associated with just one point P_i so we are able to return back to 3D, and
- the distance between a pair of points Q_i, Q_j are as close to the distance between a pair of associated points P_i, P_j as possible.

Executing a 2D curve reconstruction algorithm on the set of points Q_i produces the sought connectivity between the points P_i . If the output connectivity forms a single closed curve, we are ready. Otherwise, the output must be filtered: some edges might need to be removed, some edges to close the curve might need to be inserted. We assume that the surface from which the data were sampled is available and is represented by a triangular model, or can be reconstructed from the input points, e.g., by using the RBF approach (see Section 3.1). It allows us to solve the filtering step in a rather unorthodox but straightforward approach.

Suppose the points P_i and P_j should be connected. At first, we subdivide the triangles of the surface mesh that contain these points in their vicinity, introducing thus these points as new vertices of the mesh. Then, we trace the shortest path connecting newly introduced vertices to a set of surface points $P_{i,j,k}$, as illustrated in Figure 3. The Dijkstra algorithm can be used for it, providing that the triangular model is fine enough. Some of the fast marching methods, see, e.g., (Peyré, 2009), is an alternative suitable in all cases.

We construct a scalar field $SDF(V)$ on the surface of the mesh such that it returns the geodesic distance between the given surface point V and the nearest $P_{i,j,k}$ point – see Figure 4. This field can be constructed easily using a bread-first search algorithm starting at $P_{i,j,k}$ points. We adopt a fast marching method described in (Peyré, 2009) for this purpose.

An algorithm for iso-contours extraction is executed with the iso-value about the average length of edges $P_{i,j,k}, P_{i,j,k+1}$. In our experiments, we specify this value to $0.5 \cdot (\min \|P_{i,j,k}, P_{i,j,k+1}\| + \max \|P_{i,j,k}, P_{i,j,k+1}\|)$. Multiple contours are usually

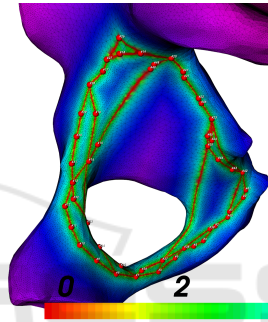


Figure 4: Scalar field constructed on the surface of the pelvis bone for the refined connectivity (obtained by (Lenz, 2006) algorithm) of the obturator internus origin data set.

extracted (e.g., one from the exterior of a closed curve, the other from the interior). The one with the largest perimeter is selected as a result – see Figure 5 for an example. We note that the final contour does not go through the input points but providing that the refinement of the primary edges is sufficient, this does not stand for a problem in many applications (including the muscle attachments estimation).

3.1 Radial Basis Functions (RBF)

Radial basis function interpolation and approximation is defined as follows:

$$h_i(x) = \sum_{j=1}^N \lambda_j \varphi(\|x_i - x_j\|), \quad (1)$$

$$\text{or also: } h = A\lambda, \quad A_{i,j} = \varphi(\|x_i - x_j\|)$$

The λ_i variable is the weight of a single RBF, φ denotes radial basis function, x_i and x_j are the positions of the input vertices (maybe attachment area vertices in our case), and h_i are values in the vertices.

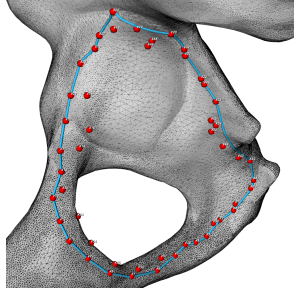


Figure 5: Obturator internus origin closed curve extracted from the scalar field in Figure 4. Compare it with the ground truth curve in Figure 2 right.

We experimented with a novel RBF described in (Skala and Cervenka, 2020) defined as:

$$\varphi(r) = r^2(r^a - 1) \quad (2)$$

Variable a is a shape parameter that has to be identified accurately to get good results. We also used $a = 1.8$ as proposed by the original authors. Figure 6 shows the surface reconstructed by this approach from the input points. This surface can be used as an alternative to the triangulated model of a bone.

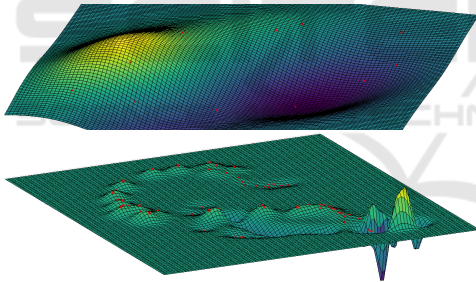


Figure 6: The surface of the femur bone reconstructed by the RBF method from the input points of biceps femoris origin (top) and vastus intermedius origin (bottom).

4 EXPERIMENTS AND RESULTS

The approach described above was implemented in C++ 14 using:

- the Visualization Toolkit (VTK)¹ for loading the data, curve reconstruction by the α -shape algorithm (see (Edelsbrunner et al., 1983)), iso-lines extraction, and visualization of results,
- the benchmarking by (Ohrhallinger et al., 2021) for curve reconstruction by 14 different algo-

rithms – connect2d, hnn crust, fitconnect, stretchdenoise, discour, vicur, crawl, peel, crust, nncrust, ccrust, gathan1, gathang, and lenz,

- the code by Yuki Koyama² for the multidimensional scaling,
- the geodesic computation on surfaces by (Krishnan, 2013), based on the fast marching method,
- and the Muscle Decomposition by (Kohout and Kukačka, 2014) for extracting the muscle attachment area bounded by the reconstructed curve from the surface mesh.

We note that the disk radius parameter of the α -shape algorithm was set to 0.5625 times the maximal shortest distance between pairs of points, i.e., just enough to guarantee that the output will have one component only. The implementations of the other reconstruction algorithms were used with their default parameters.

We experimented with the point sets representing muscle attachments of a comprehensive TLEM 2.0 data set of lower limbs (Carbone et al., 2015). After performing initial analyses, we selected, more or less randomly, a couple of representative point sets for further experiments – see Figure 7. For each of the 27 point sets we ended with, we specified the ground-truth connection of the points according to depictions in anatomical atlases. We note that in some cases, the task of finding a proper connection has proven to be complicated even for a human expert.

We inserted the points into the surface mesh of the appropriate bone and used the geodesic computation to obtain the final closed ground-truth curve. An example of such a curve is in Figure 2, right.

Using the code for the muscle decomposition by (Kohout and Kukačka, 2014), we extracted the part of the mesh belonging to the attachment area bounded by the ground-truth closed curve. In three cases (adductor longus insertion, obturator internus origin and gluteus maximus inferior origin), the implementation failed to provide us with an acceptable result. This was caused by the fact that the input data violated the assumptions of the original method. Figure 8 shows examples of extracted ground-truth attachments.

We then ran our implementation. It provided us with 15 contours for each dataset, one for every curve reconstruction algorithm. For each output contour, the surface patch was extracted from the bone model in the same way as described above. Dice similarity coefficient (DSC) was computed to measure the dissimilarity between the outcome and the ground truth. $DSC = 1$ means a perfect match, while $DSC = 0$ means that the patches do not intersect. Naturally, the value of DSC depends on the sampling frequency.

¹<https://vtk.org/>

²<https://github.com/yuki-koyama/multidimensional-scaling/blob/master/mds.h>

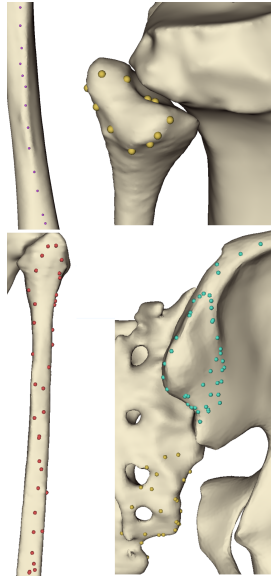


Figure 7: Representative examples of TLEM 2.0 point sets of muscle attachments. From top to bottom, left to right: biceps femoris origin, biceps femoris insertion, soleus lateralis origin, and gluteus maximus superior + inferior origin.

Our samples included all vertices of the bone mesh triangles plus some points sampled randomly in every triangle with an area larger than ϵ . The number of samples in such a triangle was determined as its area divided by ϵ . In the experiments, we used $\epsilon = \pi \cdot 0.1 \cdot 0.1$, which means our sampling frequency was about 0.2 mm.

Figure 9 shows the results we obtained. For closed curves with almost uniform sampling without apparent outliers and noise, represented, e.g., for biceps femoris insertion, the differences between 2D curve reconstruction algorithms are negligible. It is also apparent that only the α -shape algorithm was robust enough to process every data set. Connect2d, fitconnect, discor, and vicur algorithms could process only about 70.8% of data sets, stretchdenoise only about 58.3%. The rest failed to process the gluteus medius posterior insertion, which is not surprising considering that this data set contains many outliers (see Figure 10). The poor performance, generally, showed discor and vicur algorithms.

Further inspection reveals that none of the algorithms could provide acceptable results for the gluteus maximus inferior insertion and vastus intermedius origin data sets. In the former case, the reason is simply that the data contains three outliers outside the attachment region (probably introduced during an er-

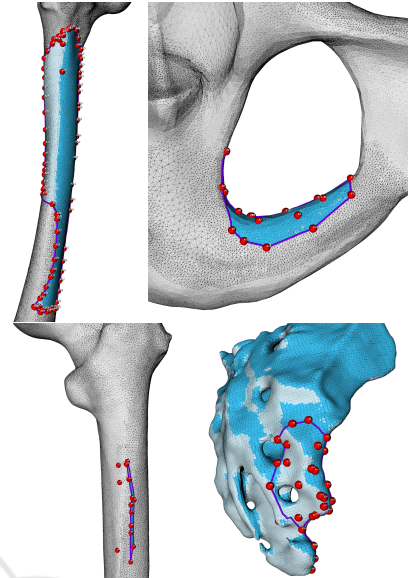


Figure 8: Ground-truth closed curves (purple) for vastus intermedius origin, obturator externus lateral origin, gluteus maximus inferior origin, and gluteus maximus inferior insertion. Surface patches representing the attachment areas, extracted using the implementation by (Kohout and Kukačka, 2014) are shaded in blue.

roneous registration process) – see Figure 10. In the latter case, the explanation is more complicated. Although the data of vastus intermedius origin seems quite OK for a human observer (see Figure 8), the algorithms yield multiple connections between the points on the left side with those on the right one. It might be because the sampling frequency on the boundary is insufficient, and thus the influence of the single apparent outlier is not negligible. As the femur bone resembles a cylinder, the geodetics computed for these incorrect edges are not on the same side, but some lie in the front, others in the back. As a result, the bone is effectively cut into several parts, as shown in Figure 10 and, therefore, only a part of the attachment area is extracted.

On average, the best performance reached α -shape (72.00%), followed by connect2d (69.35%), and lenz (67.04%). However, it is needed to point out that the dice similarity coefficient is not a reliable indicator for very narrow attachments represented by slightly curved lines, for which values as low as 30% are often visually acceptable. This is the case of adductor longus insertion, adductor magnus mid insertion, adductor magnus proximal insertion, biceps femoris CB origin, iliopsoas superior in-

9.11. *Non-planar Surf. Shape Rec. from a Point Cloud in the Context of Muscles Attach. Est.*

Non-planar Surface Shape Reconstruction from a Point Cloud in the Context of Muscles Attachments Estimation

Muscle attachment	alpha shape	connect2d	hncrust	fileconnect	stretchdenoise	discur	vicur	crawl	peel	crust	ncrust	ccrust	gathan1	gathang	lens
adductor magnus distal insertion	94.64%	87.58%	28.79%	92.49%	92.49%	0.48%	14.31%	34.14%	28.37%	24.80%	93.70%	14.57%	26.01%	26.01%	93.38%
adductor magnus mid insertion	78.88%	51.98%	28.83%	28.83%	28.83%	28.83%	28.83%	28.83%	25.86%	28.83%	28.83%	28.83%	52.22%	28.83%	72.11%
adductor magnus proximal insertion	85.06%	81.97%	24.83%	24.83%	24.83%	24.83%	24.83%	22.88%	24.72%	21.36%	21.36%	21.36%	81.83%	21.36%	86.38%
biceps femoris origin	84.72%	54.63%	19.09%	19.09%	19.09%	19.09%	20.48%	19.58%	20.48%	20.48%	20.48%	20.48%	54.84%	20.48%	26.02%
biceps femoris insertion	98.44%	97.23%	97.23%	97.23%	97.23%	97.23%	97.23%	97.23%	97.23%	97.23%	97.23%	97.23%	97.23%	97.23%	97.95%
gluteus maximus inferior insertion	16.46%		31.96%					28.91%	28.68%	30.09%	27.92%	27.92%	31.64%	35.42%	17.31%
gluteus maximus superior insertion	89.67%		21.92%					33.59%	25.10%	29.46%	91.06%	14.44%	15.27%	18.40%	83.66%
gluteus maximus superior origin	79.92%	87.56%	5.78%	76.52%	76.52%	3.88%	3.55%	84.14%	8.10%	6.89%	81.34%	3.19%	4.26%	7.18%	77.98%
gluteus medius anterior insertion	91.48%	83.15%	7.05%	89.35%	89.35%	16.23%	11.39%	20.65%	12.97%	10.01%	65.61%	14.41%	8.62%	15.88%	84.84%
gluteus medius anterior origin	96.84%		7.22%					13.09%	10.72%	5.59%	93.02%	2.91%	4.24%	5.21%	97.47%
gluteus medius posterior insertion	79.40%														
gluteus medius posterior origin	97.31%	98.11%	98.11%	97.98%	97.98%	3.73%	98.08%	98.11%	98.11%	98.11%	98.11%	4.00%	98.11%	98.11%	98.36%
iliopsoas inferior insertion	96.17%	77.17%	8.82%	54.97%	54.97%	0.91%	16.68%	25.79%	17.46%	18.71%	47.84%	11.75%	9.54%	17.81%	95.59%
iliopsoas superior insertion	71.98%	65.20%	49.47%	49.47%	49.47%	49.47%	49.47%	32.39%	32.39%	49.47%	49.47%	49.47%	49.47%	49.47%	66.23%
obturator externus lateral origin	89.09%	62.48%	9.62%	11.78%	11.78%	5.23%	12.28%	17.84%	10.30%	17.15%	65.52%	7.38%	45.24%	6.08%	78.82%
obturator externus medial origin	97.89%	78.36%	6.40%	80.84%	80.84%	6.48%	4.82%	10.11%	6.14%	6.62%	89.87%	5.68%	1.50%	7.51%	96.77%
sartorius insertion	48.95%		20.88%					61.61%	34.42%	35.86%	74.81%	40.69%	35.02%	48.11%	59.02%
semimembranosus insertion	78.31%	72.93%	34.34%	34.34%	34.34%	34.34%	34.34%	24.14%	34.85%	24.14%	24.14%	24.14%	72.80%	24.14%	78.88%
semimembranosus origin	70.16%	87.90%	12.20%	8.95%	8.95%	1.89%	9.28%	19.69%	12.62%	23.40%	56.29%	8.91%	13.53%	79.03%	73.13%
soleus lateralis origin	71.05%	45.73%	15.31%	15.52%	15.52%	3.64%	7.07%	15.55%	14.89%	11.36%	15.56%	10.60%	51.43%	11.66%	16.24%
soleus medialis origin	15.38%	12.28%	48.25%	46.16%	46.16%	15.54%	47.63%	48.43%	44.07%	45.44%	19.33%	51.41%	12.27%	12.27%	14.24%
vastus intermedius origin	0.06%		6.73%					11.19%	6.42%	10.56%	36.09%	4.49%	22.82%	4.00%	39.64%
vastus medialis inferior origin	36.76%		31.47%					41.63%	42.55%	40.60%	41.45%	41.45%	38.42%	41.45%	33.87%
vastus medialis superior origin	58.85%	34.74%	14.56%	25.18%	25.18%	20.83%	8.65%	23.99%	13.34%	13.32%	44.78%	12.14%	13.00%	15.50%	53.97%

Figure 9: Dice similarity coefficients of various muscle attachments, obtained for the MDS with different curve reconstruction algorithms. Missing values mean that the reconstruction algorithm failed. The best performances are marked in bold.

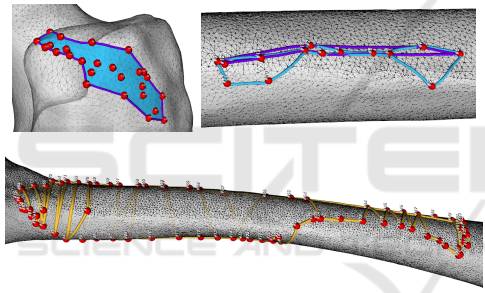


Figure 10: The data sets causing troubles during the processing (see the text): gluteus medius posterior insertion with a lot of internal points (top left), gluteus maximus inferior insertion with three outer points (top right), and vastus intermedius origin with an insufficient sampling frequency (bottom). The ground-truth curve is purple, the final contour obtained from α -shape curve is light blue, and geodesics computed from the connectivity found by the lenz algorithm are yellow.

sertion, semimembranosus insertion, soleus medialis origin, and vastus medialis inferior origin. If we exclude the results of these data sets, we get four algorithms whose performance exceeds 70% on average: α -shape (78.74%), lenz (76.45%), connect2d (76.36%), and nncrust (70.05%). Considering that connect2d failed repeatedly, we can recommend only α -shape or lenz algorithms for the curve reconstruction in our context.

For the data sets mentioned above, a better indicator might be the difference in the perimeter of the reconstructed and ground-truth curves. Table 1 shows that according to this indicator, the best performing

algorithm is connect2d, with the error of 1.25% on average but one failure, followed by gathan1 (2.60% on average), α -shape (3.03%), and lenz (3.43%). All other algorithms exhibited a worse performance with multiple failures or average errors exceeding 4% (in absolute values). We note that median errors were below 4% in all cases. The order of the five best-performing algorithms remains the same even when median errors are considered.

Therefore, it can be concluded that α -shape or lenz algorithms are universal algorithms suitable for all cases. This result is also confirmed by a subjective test, in which a human volunteer assessed all the reconstructed contours visually, classifying them into three categories:

- A = no issue or a minor one only without any considerable impact on musculoskeletal modelling,
- B = acceptable but with some issues that might have some undesirable impacts on musculoskeletal modelling, and
- C = unacceptable.

The α -shape and lenz algorithms have almost half of their contours (48.1% precisely in both cases) in category A. While hncrust, discur, vicur, peel, crust, ccrust, gathan1, and gathang algorithms have more than half of the contours they produced in category C, α -shape and lenz have there only 4 and 5 contours, which corresponds to 11.1% and 18.5%, respectively. Two of these contours belong to gluteus maximus inferior insertion, and vastus intermedius origin, already discussed above (see also Figure 10). α -shape further failed to provide acceptable results for vastus medialis superior origin, lenz for obturator externus

Table 1: Differences between the perimeters of the reconstructed and ground-truth contours for selected 2D curve reconstruction algorithms. The best results are bold.

Name	α -shape	connect2d	fitconnect	nncrust	gathan1	lenz
adductor longus insertion	7.73%	5.82%	7.85%	7.84%	5.82%	8.26%
adductor magnus mid insertion	1.54%	0.00%	1.07%	1.07%	0.00%	0.92%
adductor magnus proximal insertion	2.31%	-0.20%	3.79%	3.71%	-0.20%	3.11%
biceps femoris origin	1.25%	-0.28%	1.37%	0.73%	-0.28%	1.41%
iliopsoas superior insertion	6.69%	0.70%	8.35%	8.35%	8.35%	7.05%
semimembranosus insertion	1.85%	-0.04%	2.48%	2.68%	-0.04%	1.97%
soleus medialis origin	-1.31%	-2.98%	-5.75%	-0.22%	-2.99%	-0.84%
vastus medialis inferior origo	1.59%			7.63%	3.09%	3.86%
Avg(abs(error))	3.03%	1.25%	3.83%	4.03%	2.60%	3.43%

lateral origin, soleus lateralis origin, and soleus medialis origin. In all four cases, the attachments are much more stretched in one dimension than in the other. Due to sparse sampling, the MDS increased further this ratio, producing points visually lying almost on a one-dimensional object (see Figure 11). All algorithms then struggled with such data.



Figure 11: Soleus lateralis origin after being transformed into 2D using projection onto the best fit plane (left) and using the multidimensional scaling technique (right).

We, therefore, compared the results with those obtained when the input points were projected onto the plane fitted to the data by the least-squares method instead of using the multidimensional scaling technique. Table 2 show that although some algorithms benefit from the MDS technique (e.g., fitconnect, stretchdenoise, or peel), others, without any doubt, perform better without it (e.g., lenz or nncrust). As for the subjective tests, lenz algorithm came in first with 15 (i.e., 55.6%) muscle attachments in category A, 11 (i.e., 40.7%) in category B and only vastus intermedius origin in category C. The second place was taken by α -shape with 10 (i.e., 37%) muscle attachments in category A, 13 (i.e., 48.1%) in category B, and 4 (i.e., 14.8%) in category C. Clearly, while α -shape achieved more acceptable results with the MDS, lenz demonstrated different behaviour.

However, we must point out that a projection of points onto a common plane is not suitable when the curve to be reconstructed bends several times, e.g., like in the case of a narrow saddle. Although such cases are pretty rare in the context of muscle attachments, they seem to be frequent in the aneurysm neck identification problem.

Table 2: Difference of the average performance when using the MDS and when using the projection onto a common plane. Positive values mean that the MDS outperforms the projection. Dice similarity coefficients (DSC) are used as a performance indicator for the data for which DSC is a reliable indicator (see the text for explanation). For the rest, errors in the muscle attachments perimeters (PER) are used.

Algorithm	DSC	PER
α -shape	-0.37%	-0.31%
connect2d	-8.80%	0.23%
hnnncrust	8.83%	0.00%
fitconnect	11.54%	0.25%
stretchdenoise	17.10%	-0.26%
discur	-8.46%	6.38%
vicur	-5.70%	-0.01%
crawl	-6.69%	-0.03%
peel	9.61%	0.05%
crust	4.31%	-0.08%
nncrust	-6.63%	-0.57%
ccrust	-0.17%	-6.31%
gathan1	3.87%	-2.11%
gathang	-12.59%	-11.09%
lenz	-12.66%	-0.72%

We also did some preliminary testing of the RBF approach for ordering the vertices and smoothing the curve. The primary purpose of this test is to check whether the RBF approach will be capable of creating a closed and non-self-intersecting curve in 2D.

If the outliers or apparent nonuniformity are present, the resulting curve is far from expectations (e.g. vastus intermedius origin on the left of Figure 12). Luckily, this approach gives better results for many other attachment areas (e.g. gluteus medius posterior on the right of the Figure 12). The polar coordinate system for the dimension reduction causes problems in some cases, mainly if there is a wide angle without any vertex (left image of the Figure 12, bottom part), causing a single or even multiple self-intersectional loops. Approximating the curve instead of interpolating may solve these issues.

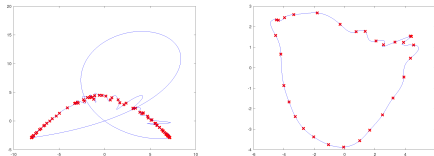


Figure 12: Two radial basis function approximation results. Vastus intermedius origin is on the left. The connection between both ends of the attachment area does not turned out well due to looping the curve. Gluteus medius posterior origin is on the right. The shape is approximated by expectations.

5 CONCLUSION AND FUTURE WORK

This paper investigated the options of reconstructing a closed space curve from the points sampled on that curve, supporting sparse sampling and noisy data with multiple outliers. Our extensive experiments, performed on the TLEM 2.0 data sets (Carbone et al., 2015) in the context of muscle attachments estimation, lead us to the following recommendations. If the curve to be reconstructed is not expected to have a shape of a narrow saddle or be otherwise strangely bent, the points should be projected onto the plane that best fit the input data. The lenz algorithm (Lenz, 2006) should be used on the projected points to find the primary connectivity between the input points. Suppose this algorithm is unavailable or the expectations on the curve shape do not hold. In that case, the input data should be transformed onto the plane using the multidimensional scaling (MDS) technique (Cox and Cox, 2008). The α -shape algorithm (Edelsbrunner et al., 1983) should be then used on the transformed points (instead of the lenz algorithm), with the disc radius being slightly above half of the maximal shortest distance between pairs of transformed points. If neither algorithm is available, connect2d or nncrust (see (Ohrhallinger et al., 2021)) are a decent choice.

Providing that the surface on which the space curve lies is available, the reconstructed curve can be refined by tracing the shortest paths between each pair of points connected by an edge. A non-manifold curve, i.e., a curve containing vertices of valence larger than 2, can be converted into a manifold one using the algorithm proposed in the paper, based on iso-contour extraction from a scalar field describing for each point on the surface its distance to the curve. If the object bounded by the curve covers only a tiny portion of the surface in any direction or the surface is open, this conversion is reliable.

REFERENCES

- Berger, M., Tagliasacchi, A., Seversky, L. M., Alliez, P., Guennebaud, G., Levine, J. A., Sharf, A., and Silva, C. T. (2016). A survey of surface reconstruction from point clouds. *Computer Graphics Forum*, 36(1):301–329.
- Carbone, V., Fluit, R., Pellikaan, P., van der Krogt, M., Janssen, D., Damsgaard, M., Vigneron, L., Feilkas, T., Koopman, H., and Verdonshot, N. (2015). TLEM 2.0 – a comprehensive musculoskeletal geometry dataset for subject-specific modeling of lower extremity. *Journal of Biomechanics*, 48(5):734–741.
- Carr, J. C., Beatson, R. K., Cherrie, J. B., Mitchell, T. J., Fright, W. R., McCallum, B. C., and Evans, T. R. (2001). Reconstruction and representation of 3d objects with radial basis functions. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. ACM Press.
- Cervenka, M., Smolik, M., and Skala, V. (2019). A new strategy for scattered data approximation using radial basis functions respecting points of inflection. *Computational Science and Its Applications*.
- Cox, M. A. A. and Cox, T. F. (2008). Multidimensional scaling. In *Handbook of Data Visualization*, pages 315–347. Springer Berlin Heidelberg.
- Edelsbrunner, H., Kirkpatrick, D., and Seidel, R. (1983). On the shape of a set of points in the plane. *Information Theory, IEEE Transactions on*, 29:551 – 559.
- Fukuda, N., Otake, Y., Takao, M., Yokota, F., Ogawa, T., Uemura, K., Nakaya, R., Tamura, K., Grupp, R. B., Farvardin, A., Armand, M., Sugano, N., and Sato, Y. (2017). Estimation of attachment regions of hip muscles in CT image using muscle attachment probabilistic atlas constructed from measurements in eight cadavers. *Int J Comput Assist Radiol Surg.*, 12(5):733–742.
- Kohout, J. and Kukačka, M. (2014). Real-time modelling of fibrous muscle. *Computer Graphics Forum*, 33(8):1–15.
- Krishnan, K. (2013). Geodesic computations on surfaces. *The VTK Journal*.
- Lenz, T. (2006). How to sample and reconstruct curves with unusual features. In *Proceedings of the 22nd European Workshop on Computational Geometry (EWCG)*, pages 29–32.
- Ohrhallinger, S., Peethambaran, J., Parakkat, A. D., Dey, T. K., and Muthuganapathy, R. (2021). 2d points curve reconstruction survey and benchmark. *Computer Graphics Forum*, 40(2):611–632.
- Peyré, G. (2009). Geodesic methods in computer vision and graphics. *Foundations and Trends® in Computer Graphics and Vision*, 5(3-4):197–397.
- Skala, V. and Cervenka, M. (2020). Novel rbf approximation method based on geometrical properties for signal processing with a new rbf function: Experimental comparison. *2019 IEEE 15th International Scientific Conference on Informatics*.

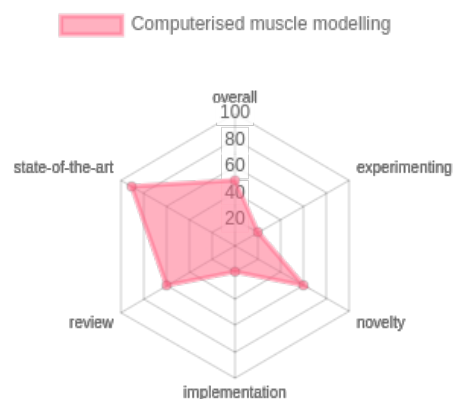
9.12 Computerised muscle modelling and simulation for interactive applications

In our subsequent study [98], we conducted comprehensive evaluations of our previously developed muscle modelling technique, addressing a future challenge and introducing a method enabling muscles to "slide" over bones using a position-based dynamics (PBD) virtual edges.

A problem identified during our tests occurs when two bones move close to each other with a muscle sandwiched in between, akin to a muscle caught in shears. This seemingly rare scenario occurs on a microscale in joint areas. Another problem arises when bone movement is so rapid between iterations that the muscle might penetrate the bone, complicating shape restoration. This issue, while appearing unlikely, is common in long bones like the femur, where a small angular change can result in substantial displacement at the bone's opposite end. Finally, some further directions are outlined:

1. Expanding on the PBD concept to include advancements like XPBD, an enhanced version of PBD. This area is currently under exploration, though it has become a smaller focus of my research.
2. Integrating another algorithm with PBD to refine outcomes. The ARAP algorithm tested in this paper is a candidate for this integration. However, we decided to diverge from this research path due to unresolved issues.
3. Adopting a different geometric model than triangular meshes could mitigate certain problems, particularly surface roughness. Each alternative geometry, however, presents its own set of challenges.

This final examination of the triangular mesh model prompted me to shift my focus to a different geometric approach (according to the third point), employing radial basis functions rather than the triangular mesh.



Publication [98]:

CERVENKA, M.; HAVLICEK, O.; KOHOUT, J.; VASA, L. Computer muscle modelling. *Computerised muscle modelling and simulation for interactive applications, Proceedings of the 18th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, VISIGRAPP 2023, Volume 1: GRAPP*. 2023, pp. 214–221. ISBN 978-989-758-634-7. Available from DOI: <https://doi.org/10.5220/0011688000003417>. UT WoS: 001066254400019, OBD: 43940148

Computerised muscle modelling and simulation for interactive applications^a

Martin Cervenka¹^b, Ondrej Havlicek³^c, Josef Kohout³^d, Libor Váša⁴^e,

^{1,2,4}The University of West Bohemia, Faculty of Applied Sciences, Department of Computer Science and Engineering

³The University of West Bohemia, Faculty of Applied Sciences, NTIS - New Technologies for the Information Society

¹cervemar@kiv.zcu.cz, ²onhavlic@students.zcu.cz, ³besoft@ntis.zcu.cz, ⁴lvasa@kiv.zcu.cz

Keywords: Muscle modelling, Collision detection, Collision response, Position Based Dynamics, Discregrid, Scalar Distance Field, As-Rigid-As-Possible, Radial basis functions

Abstract: The main challenges of collision detection and handling in muscle modelling are demonstrated. Then, a collision handling technique is tested, exploiting the issue of muscle penetrating the bone in some circumstances, mainly when the movement is too rapid or the displacement of the bone is too high. Our approach also detects the problem, using Discregrid to see the immediate direction change towards the penetrated bone. Some alternatives to the described PBD (Position-Based dynamics) technique are presented: PBD with As-Rigid-As-Possible modification and radial basis function approach.

1 INTRODUCTION

Osteoporosis [Wade et al., 2014], osteoarthritis [Oatis, 2017], patellar dislocation [Barzan et al., 2017], or hemiplegic diseases [Zhang et al., 2021] are leading researchers to develop a satisfactory model of the musculoskeletal system. Creating such a model is a complex procedure with many issues. The problem of collision detection (CD) between muscle and bone models and its response (CR) is critical.

In this paper, we follow our previous work and newly present the parameter upper bound, where the simulation still works as expected. Another contribution of this paper is exploring novel modelling methods to overcome the current limitations. This paper also briefly describes some techniques of CD and CR.

The paper is structured as follows. The next section gives an idea of the whole muscle modelling overview and the steps to obtain a usable computerised muscle model. The state-of-the-art approaches to modelling and simulation of the muscles, based on

the position-based dynamics, are described in Section 3. The description of existing CD and CR techniques employed during these simulations follows. Sections 5 and 6 present the current approach's limitations and propose several improvements to overcome them. Discussion of future work and concluding remarks follow.


2 MUSCLE MODELLING PIPELINE


The muscle modelling procedure involves many steps, including acquiring relevant raw data and its subsequent transformation into a useful form. The last step is formulating the mathematical model, where the main concern is (among others) the definition of the muscle-bone interaction.


An example of a complex pipeline (consisting of data acquisition, model building and inverse kinematics) includes the following steps:


1. obtaining raw data of the patient at rest (such as medical images) representing anatomical objects (bones, muscles, muscle attachment areas, etc.), and movement data,
2. extraction and transformation of the raw data into a useful form, using:
 - (a) segmentation – separation of different types of

^aThis work was supported by the Ministry of Education, Youth and Sports of the Czech Republic, project SGS-2022-015.

^b <https://orcid.org/0000-0001-9625-1872>

^c <https://orcid.org/0000-0002-6944-7084>

^d <https://orcid.org/0000-0002-3231-2573>

^e <https://orcid.org/0000-0002-0213-3769>

¹ Corresponding author

- tissues present in medical images (if they are distinguishable). Segmentation can be manual, semi-automatic or automatic (depending on the complexity of the segmentation),
- (b) extraction – the conversion of segmented data into geometrical models,
 - (c) registration – mapping of data and models from different measurements and modalities into the common reference space
 - (d) approximation and interpolation – reconstruction of missing parts or partially corrupted data.
3. acquiring some general apriori knowledge, determined by human anatomies, such as
 - (a) how the attachment areas will be determined (whether based on apriori knowledge only or the measured muscle attachment areas),
 - (b) defining how a bone is connected by a joint to another bone or how a muscle is connected to a set of bones (attachment areas), etc.,
 - (c) defining physiological parameters of studied muscles, such as internal muscle architecture (e.g. figure arrangement: parallel, pennate, etc.), optimal (resting) length and others.
 4. creating a mathematical model that requires:
 - (a) defining the space (discretised or continuous),
 - (b) defining the shape of the data (triangular surface mesh, tetrahedral volumetric mesh, scattered data... / surface defined by Fourier series, implicit RBF,...)
 - (c) defining the interaction between muscle and bone models and thus determining whether or not a transformation of the measured data is necessary.
 5. transformation of the simulation output from its model representation into the final form, e.g., from a triangular surface mesh into a set of internal fibres.

We recommend the following papers for a more detailed view: the foremost step is well described by [Fukuda et al., 2017] for the attachment area acquisition strategy, [Lee et al., 2014] to determine the pennate angle from the source data. There are also data from invasive measurements, e.g. Visible Human Project (the National Library of Medicine) or The Chinese Visible Human [Zhang et al., 2004]. The second step (registration) is also well described in [Zhao et al., 2013]. There is also an approach from [Li et al., 2008], with promising results. The other steps are highly dependent on the considered application, whether the purpose is a plausible visualisation of muscles in movement (see. e.g. [Romeo et al., 2018]) or to calculate some physical phenomena of the muscle (see. e.g. [Modenese and Kohout, 2020]), and whether the user should be able to change

the modelling or simulation parameters interactively or not.

We focus on interactive applications which do not require specialised hardware. Consequently, any model developed in the fourth step needs to trade off some accuracy for the speed of simulation. In this paper, we consider position-based dynamics (PBD) suitable for generating models of such a kind.

3 POSITION BASED DYNAMICS

Position-based dynamics (PBD) [Müller et al., 2007] is a fast approach used mainly in the animation industry to model elastic object (and cloth) deformations. Nowadays, the PBD is making its way into physical simulations as well. The original algorithm does not consider the possibility of object anisotropy as far as the algorithm has been developed for general objects. The method accepts a manifold surface mesh and produces its deformed variant as the output.

The PBD also exists in the xPBD (eXtended version of PBD, which respects the concept of elastic potential energy) form. The xPBD incorporates elastic potential energy and eliminates the necessity to know the time step and iteration count [Macklin et al., 2016].

Romeo et al. [Romeo et al., 2018] are the first who proposed using the xPBD algorithm for muscle modelling problems. Their fundamental idea is to build an internal structure above the surface mesh to respect the anisotropy of the muscle (the internal structure respects the general direction of the muscle fibres). They can form a volumetric model better suitable for the PBD algorithm with an intelligent edge-creation process. However, the paper needs to describe their collision detection and handle thoroughly. According to their video of the technique outcome, many collisions occur, suggesting their approach did not address the apparent requirement of avoiding muscle-bone penetrations.

Angles et al. [Angles et al., 2019] developed a PBD-based approach for muscle modelling in 2019. Their approach virtually decomposes the muscle into "rods" (which may approximate the muscle fibres). These rods can adjust their diameter wherever they want to preserve their volume. Their main contribution is the ability to provide real-time simulation, which Romeo's approach cannot because "its ≈ 40 s/frame of processing time causes it unfitting to interactive applications" [Angles et al., 2019]. They adopt "Particle simulation using CUDA" from [Green, 2010] for collision detection between rods and response. Again, the problem of muscle-bone

penetration is not addressed, though in this case, the extension is relatively straightforward.

The position-based dynamics for muscle modelling is also described in the paper "Fast and Realistic Approach to Virtual Muscle Deformation." [Cervenka. and Kohout., 2020] where the PBD approach has been proposed (finished the same year as Romeo's article [Romeo et al., 2018], working concurrently on the same). The paper "Muscle Deformation using Position Based Dynamics" [Kohout and Červenka, 2021] follows, which tests the approach and compares the results to an existing FEM technique. The primary benefit of our suggested system is that no interior is needed. The anisotropy is computed on the surface of the mesh only, utilising muscle fibres on the mesh surface, representing the fibre direction. The voxelisation technique has been used for collision detection and response purposes.

4 COLLISION DETECTION AND RESPONSE

Various approaches to detect and respond to an occurring collision have been proposed. The most common algorithms exploit the D&C (divide & conquer) paradigm. The bounding volume hierarchy is one of them [Teschner et al., 2005], using a primitive (often an axis-aligned bounding box AABB or a sphere) hierarchy to enclose the model and its parts. The spatial hashing [Turk, 1990] is its generalisation over the whole model space.

If there is a necessity to know not only if the collision occurs but how far from the collision the model is, the (signed) distance field approach is an excellent way to go. Numerous techniques can be used to construct such a field. The vast majority use voxelisation to obtain a cell array and then use an interpolation method to determine the value between the cells. Some of these techniques are well described in an older work by [Bærentzen and Aanæs, 2002].

In our research of the PBD approach, the first decision was to use a simple voxelisation method to simplify the collision detection problem. This simple idea, however, leads to some things that could be improved. Luckily, some ideas have emerged to entirely improve or even fix some problems, using more complex collision detection algorithms. Havlicek [Havlicek et al., 2022] changed the collision detection to Discregrid (using a signed distance field) and FLC (using a binary search tree), beating the voxelisation approach in terms of accuracy. However, there is still some work because even those methods only work correctly in extreme conditions, mainly if the

movement is rapid.

The collision response is a complicated task as well. Assume that two bones move towards each other and narrowly miss each other (like shear blades). If a muscle is attached to both of the bones and appears to be in between the bones, there is no such room for the bones to move into. This problem often happens on a smaller scale, near joints, especially where two bones move close. Our former solution [Cervenka and Skala, 2020] was to assume, in this particular case, only one of the bones and move a muscle in the direction opposite of it, but it proved insufficient. Havlicek [Havlicek et al., 2022] targets this problem primarily, and he proposed a better approach of considering all adjacent bones and moving opposite to the sum of all collision vectors. Even this approach, however, does not always guarantee collision resolution.

Our current contribution, proposed in this paper, follows our recent articles, mainly [Cervenka. and Kohout., 2020, Kohout and Červenka, 2021] and also [Havlicek et al., 2022]. In the first article, we developed a PBD-based approach for muscle modelling. The issue with a muscle stuck inside a joint was shown in that article. We believed that "better collision detection can fix the issue" [Cervenka. and Kohout., 2020]. In the second article, the voxelisation collision detection approach was proven inaccurate in some cases, mainly in the case of more "complicated" (e.g. concave) bone surfaces, which are located near the joint areas more frequently. The last article explores two existing collision detection algorithms for the PBD approach: Discregrid and Flexible Collision Library. The Discregrid was shown to be more suitable for the problem.

4.1 Discregrid

Discregrid library can be considered a Signed Distance Field generator written in C++. The algorithm computes for each point in 3D space the shortest distance and direction towards a given nearby surface represented by a triangular mesh. Also, assuming that the input mesh is at least watertight, the method can make the inside/outside decision because the algorithm provides the sign. A finite bounded subspace is required for the approach to work.

The bounded surface is firstly divided (like in the voxelisation method) into a rectangular grid with a user-defined resolution, where each voxel is a 32-node Serendipity type [Koschier and Bender, 2017]. For each node, the distance and the direction towards the nearest bounded surface are computed, see [Bærentzen and Aanæs, 2002]. The article describes

the problem of discontinuity of the mesh (where the normal vectors have to be estimated differently).

The field creation process is time-consuming (about half a minute for bone meshes consisting of up to 45 000 vertices with the grid resolution of $64 \times 64 \times 64$ on standard hardware) and unfeasible for deformable objects like muscles, where recalculation is often needed. There is no such problem with the bone models because their movement is only rigid (allowing for Discregrid results to be transformed using a global transformation).

In the case of muscle modelling, a situation may happen when the muscle collides with a bone or another muscle, even at the start of the simulation. This situation is caused by the different modalities used for the data measurements (see section 2). To fix the issue, the colliding surface vertices are pushed according to the Discregrid result directional vectors so no collision would occur at the start of the simulation.

5 MUSCLE SIMULATION

In this paper, we experimented with more types of motion of the hip joint, not just flexion but also rotation and adduction. We also tested a hip extension scenario in the described PBD approach to find the maximum amount of problematic cases possible. The tests will be done on muscles and bones depicted in Fig. 1.

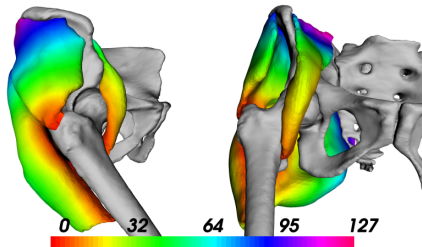


Figure 1: One view from the side and one from the front of a muscle group surrounding the hip joint in rest pose. The vertices of the muscles are coloured by their distance to the femur bone (pointing down) in millimetres given by Discregrid.

5.1 Test of motion types

For the test of different motion types, we tested the original hip flexion (from 0° to $+90^\circ$ with the step of 2°), hip rotation (from 0° to $+45^\circ$ and also to -45° with the approximate degree of 1°) and hip adduction (from 0° to $+60^\circ$ with the approximate step of 2°).

The results of hip flexion are shown in Fig. 3. Near the joint area, the muscle nearly touches the femur bone; however, no collision occurs. Fig. 4 shows the results for the rotational motion. As before, the muscle nearly touches the femur's upper extremity without collisions. The adduction is demonstrated in Fig. 5. In this case, the muscle is further from the hip joint, lowering the possibility of collisions. The collision detection and response approach solved all the apriori collisions, so no collisions happened.

5.2 Test of motion speeds

To test for bone movement speed impact, we chose the extension movement. The initial step of 2° was increased to 4° , 5° and finally 10° . We also increased the target angle to $+80^\circ$ for rapid movement to have time and space to show up fully.

In this case, when the angle finally reaches 72° with the angle step of 4° , first bone penetration occurs (see Fig. 6). The muscle is not as fast as needed to keep up with the bone movement, causing the distal part of the muscle to enter the bone volume and go through the whole cross-section. The muscle is also being unnaturally pushed into itself by the bone.

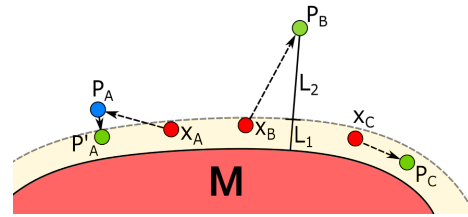


Figure 2: Simplified 2D illustration of the sliding mechanism. Points X_A , X_B and X_C are being evaluated for the distance from the other muscle M . Consider the same threshold distance for each of them (yellow margin). Point X_A is pushed to the position P_A by previous PBD constraints but returned towards the other muscle to the position P'_A . Point X_B is pushed away by the PBD so much it makes sense to leave it to go ($L_2 > L_1$). Point X_C is free to roam inside the strip.

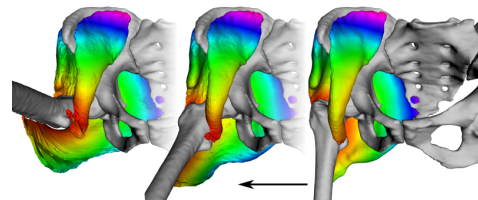


Figure 3: The result of the hip flexion progressing from right to left.

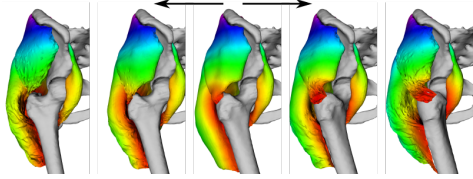


Figure 4: The result of the hip rotation progressing from the centre to the sides.

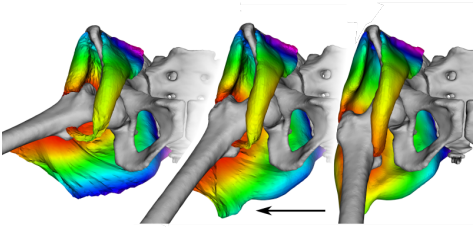


Figure 5: The result of the hip adduction progressing right to the left during the simulation.

5.3 Tunnelling detection

The tunnelling problem (muscle “jumps” in a single iteration through the whole bone to the opposite side) appears at places where the displacement of the bone between two consecutive simulation frames exceeds half the size of a muscle with which the bone collides. According to [Havlicek et al., 2022], “when [the femur] rotates about just 2° (a typical step in simulations), the displacement of the distal part of this bone is nearly 3 cm”. In our experiments with the muscles of the hip, i.e., no part of the muscle is near the distal portion of the bone, this problem arises when the angle step is higher than 4° .

This problem is relatively standard and does not arise only with this particular approach. For example, [Janák, 2012] notes that “if the movement of the object is too fast in relation to the discrete time step, the collision may not be detected”.

The scenario of the muscle movement is so rapid that the whole muscle volume could go through the entire bone model, which is possible (due to the already described displacement issue concerning the change of the angle of the bone). Our solution would be moving a muscle to the bone (rotate around the exact centre of rotation and about the same angle), so the muscle is closer to where it should lie. However, before applying this correction, such an event must be detected.

Generally, a continuous collision detection method could be employed instead of the currently used discrete one. However, such methods are expensive. We, therefore, propose a simple (and fast) test based on a comparison of the directional vectors

to the nearest bone surface, provided automatically by the Discregrid.

If the direction of one muscle vertex suddenly changes “too much”, we may expect that the penetration through the whole bone has happened. The “too much” is defined as when the angle between the directional vector from the previous step and the new one is greater than 135° , as described in Equ. 1, where d_i is the direction to the bone in this computational step and d'_i is the direction to the bone in the previous computational step, $\|\mathbf{a}\|_2$ is the euclidean norm of the vector \mathbf{a} . A tunnelling case can be seen in Fig. 6.

$$\begin{aligned} \arccos \left| \mathbf{d}_i \cdot \mathbf{d}'_i \right|_2 &> 135^\circ \\ \left| \mathbf{d}_i \cdot \mathbf{d}'_i \right|_2 &> -0.75 \end{aligned} \quad (1)$$

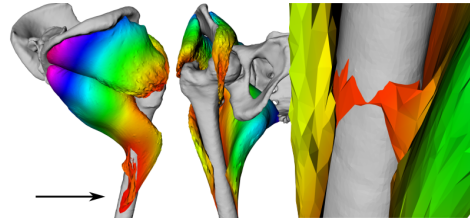


Figure 6: Due to the lack of contraction modelling, the muscles are being dragged during the hip extension (left and centre images). Finally, under the degree of 72° , first tunnelling occurs (right) and is detected successfully by the proposed algorithm.

5.4 Problem of multiple muscles

When multiple muscles are simulated in the scene, they may also intersect each other. As noted in section 4.1, building the Discregrid structure for the muscle meshes is unusable for the interactive application due to time requirements. An appropriate collision handling system could be, e.g. a BVH structure (see section 4), which would have to be updated each time any muscle moves. As the nearby muscles often touch each other - see Figure 1, this solution would probably be inefficient. Therefore, we propose the “sliding” technique using the so-called “virtual edges” to keep the muscles at a certain distance from each other to prevent collisions and unrealistic detachments.

5.4.1 Sliding over surfaces

The main idea is to allow the muscles to slide over each other using the virtual edges between the muscles. We may keep chosen vertices up to a certain distance away from the other muscles, i.e. a threshold. The vertices to keep close to the other muscles

can be selected by their initial distance and kept in a list. This list could also be updated if the PBD displacement of a particular vertex away from the other muscle would be greater than its initial distance, letting the point go. Similarly, collecting more vertices into this list could be achieved via checking, for example, the neighbourhood vertices, which would be the likely candidates.

In case the PBD forces are less in magnitude than this *attraction force* but still point away from the muscle, the vertex would be pushed in the closest direction to the other muscle surface, effectively making the point slide along the other muscle surface and hopefully preventing it from colliding or unrealistically spanning out, as illustrated in Fig. 2.

This functionality is implemented preliminarily, using the Discregrid library to preserve the distances between a bone and a muscle. The resulting virtual edges can be seen in Figure 7. The reason for this measure is that the original idea implementation would be beyond the scope of this paper.

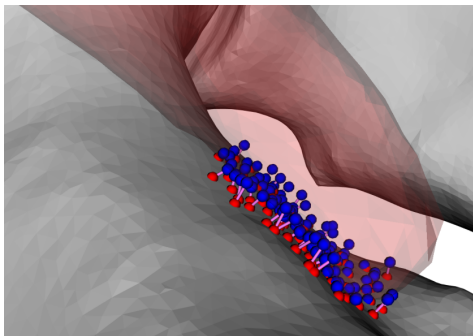


Figure 7: An example of virtual edges between a muscle and a bone with the participating vertices in blue and red, respectively. The points in blue are muscle vertices, which are close enough to the bone at the start of the simulation. For each of them, the closest bone vertex (red) is found. The pairs of vertices are connected with a straight line segment, symbolizing a virtual edge.

6 FUTURE WORK

The PBD approach on its own brings some problems to muscle modelling. The shape is not well preserved (see Fig. 5, on the middle image, the bottom central part of the gluteus maximus muscle is unrealistically deformed), and bone penetration happens. Any problems stem from low solver iteration count, essential for real-time model interaction. The options to solve these are to:

- increases the number of PBD iterations, effectively slowing down the simulation, which would become no longer interactive [Kohout and Červenka, 2021];
- uses the eXtended PBD, which converges more consistently (the iteration count is not as significant) but does not solve the penetration issues;
- uses a different muscle model.

6.1 RBF representation

The radial basis function model opens a new possibility to develop a new approach to the deformation of this model, which would allow smooth and rapid muscle simulation. Collision detection and response, volume preservation, and muscle anisotropy are the challenges for future work.

The critical decision for the suitable model is to select the suitable radial basis function and shape parameters (if any). A comprehensive study of some well-known RBFs has already been made (see, e.g. [Majdisova and Skala, 2017]), and the shape parameters were explored (e.g. in [Skala and Červenka, 2019] or [Afiatdoust and Esmaeilbeigi, 2015]).

6.2 ARAP & PBD

Because of the deformed and unrealistic shape of the model during the simulation, the As-Rigid-As-Possible (ARAP) approach from computer graphics is proposed for merge with PBD to deform an object respecting its original shape [Sorkine and Alexa, 2007] to obtain "the best of both worlds".

As the preliminary experiment, we tried to use a single PBD iteration to preserve the muscle's original volume, followed by a single iteration from ARAP, which should restore the initial shape of the muscle. The problem is that these two restrictions force most vertices to go in the opposite direction, resulting in a rough surface. (see results in Fig. 8). The volume preservation constraint is not solvable by introducing a new condition into the system since the interleaving approach does not work.

Dvorak et al. [Dvořák et al., 2022] show how to apply the ARAP approach to volume preservation. However, their approach is not used directly for muscle modelling problems. Our goal is to avoid the introduction of an internal muscle structure to reach lower computational complexity, meaning that their approach would have to be altered drastically.

We see two options for fixing the mentioned issues. The first is to start with PBD and replace the

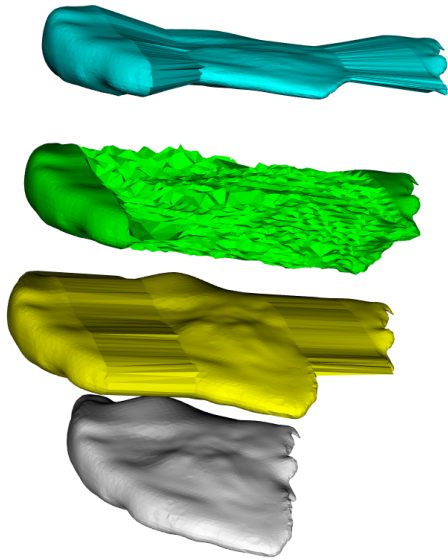


Figure 8: ARAP approach with a simple volume preservation constraint. The initial model is in white, the user deformation is in yellow. The volume preserving model with user defined constraint is in lime blue, the green model is the simple interweaving approach of the ARAP and PBD.

shape preservation constraint with the shape preservation constraint from the ARAP approach. A mathematical reformulation of the shape constraint and finding a gradient expression would be required for the ARAP shape preservation constraint. The other option is to start with ARAP and replace the interleaving approach with gradient descent from PBD. Then, a volume constraint can be added. Either way, both methods should end up with the same result.

7 CONCLUSION

All of the described techniques for muscle modelling provide good outcomes; however, each has some drawbacks. Some are inaccurate (Hill-type model [Hill, 1938], incorrect according to [Burzyński et al., 2021], Via-points too approximate according to [Modenese and Kohout, 2020]), some are accurate but difficult to set up or simply too slow to be useful (Finite element methods used by, e.g. [Delp and Blemker, 2005], proved difficult to set up by [Romeo et al., 2018]). Other methods are "compromise solutions" in terms of accuracy and computational complexity (Mass-Spring system, [Georgii and Westermann, 2005, Aubel and Thalmann, 2001,

Janák, 2012], PBD, As-Rigid-As-Possible [Sorkine and Alexa, 2007, Fasser et al., 2021, Wang et al., 2021]). As the reader can probably imagine, many open problems still exist.

ACKNOWLEDGEMENTS

The authors thank their colleagues and students at the University of West Bohemia for their discussions and suggestions. This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic, project SGS-2022-015 New Methods for Medical, Spatial, and Communication Data.

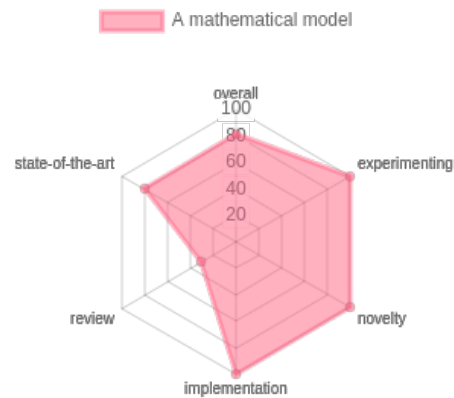
REFERENCES

- Afiatdoust, F. and Esmailbeigi, M. (2015). Optimal variable shape parameters using genetic algorithm for radial basis function approximation. *Ain Shams Engineering Journal*, 6(2):639–647.
- Angles, B., Rebain, D., Macklin, M., Wyvill, B., Barthe, L., Lewis, J., Von Der Pahlen, J., Izadi, S., Valentin, J., Bouaziz, S., and Tagliasacchi, A. (2019). Viper: Volume invariant position-based elastic rods. *Proc. ACM Comput. Graph. Interact. Tech.*, 2(2).
- Aubel, A. and Thalmann, D. (2001). Interactive modeling of the human musculature. In *Proceedings Computer Animation 2001. Fourteenth Conference on Computer Animation (Cat. No.01TH8596)*, pages 167 – 255.
- Barzan, M., Carty, C., Maine, S., Brito da Luz, S., Lloyd, D., and Modenese, L. (2017). Subject-specific knee kinematics during walking in children and adolescents with recurrent patellar dislocation. In *23rd Australian & New Zealand Orthopaedic Research Society*.
- Burzyński, S., Sabik, A., Witkowski, W., and Łuczkiwicz, P. (2021). Influence of the femoral offset on the muscles passive resistance in total hip arthroplasty. *PLOS ONE*, 16(5):1–12.
- Bærentzen, A. and Aanæs, H. (2002). *Generating Signed Distance Fields From Triangle Meshes*. Informatics and Mathematical Modelling.
- Cervenka, M. and Kohout, J. (2020). Fast and realistic approach to virtual muscle deformation. In *Proceedings of the 13th International Joint Conference on Biomedical Engineering Systems and Technologies - HEALTHINF*, pages 217–227. INSTICC, SciTePress.
- Cervenka, M. and Skala, V. (2020). Behavioral study of various radial basis functions for approximation and interpolation purposes. In *2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*, pages 135–140.
- Delp, S. and Blemker, S. (2005). Three-dimensional representation of complex muscle architectures and geometries. *Annals of biomedical engineering*, 33:661–73.
- Dvořák, J., Káčereková, Z., Vaněček, P., Hruša, L., and

- Váša, L. (2022). As-rigid-as-possible volume tracking for time-varying surfaces. *Computers & Graphics*, 102:329–338.
- Fasser, M., Jokeit, M., Kalthoff, M., Gomez Romero, D. A., Trache, T., Snedeker, J. G., Farshad, M., and Widmer, J. (2021). Subject-specific alignment and mass distribution in musculoskeletal models of the lumbar spine. *Frontiers in Bioengineering and Biotechnology*, 9. Cited By :2.
- Fukuda, N., Otake, Y., Takao, M., Yokota, F., Ogawa, T., Uemura, K., Nakaya, R., Tamura, K., Grupp, R., Farvardin, A., Sugano, N., and Sato, Y. (2017). Estimation of attachment regions of hip muscles in ct image using muscle attachment probabilistic atlas constructed from measurements in eight cadavers. *International Journal of Computer Assisted Radiology and Surgery*, 12.
- Georgii, J. and Westermann, R. (2005). Mass-spring systems on the gpu. *Simulation Modelling Practice and Theory*, 13:693–702.
- Green, S. (2010). Particle simulation using cuda. In *Particle Simulation using CUDA*.
- Havlicek, O., Cervenka, M., and Kohout, J. (2022). Collision detection and response approaches for computer muscle modelling. accepted for the IEEE 16th International Scientific Conference on Informatics.
- Hill, A. (1938). The heat of shortening and the dynamic constants of muscle. *Proc. R. Soc. Lond. B*, 126:612–745.
- Janák, T. (2012). Fast soft-body models for musculoskeletal modelling. Technical report, University of West Bohemia, Faculty of Applied Sciences.
- Kohout, J. and Červenka, M. (2021). Muscle deformation using position based dynamics. In Ye, X., Soares, F., De Maria, E., Gómez Vilda, P., Cabitza, F., Fred, A., and Gamboa, H., editors, *Biomedical Engineering Systems and Technologies*, pages 486–509, Cham. Springer International Publishing.
- Koschier, D. and Bender, J. (2017). Density maps for improved sph boundary handling. In *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation, SCA '17*, New York, NY, USA. Association for Computing Machinery.
- Lee, D., Li, Z., Sohail, Q. Z., Jackson, K., Fiume, E., and Agur, A. (2014). A three-dimensional approach to pennation angle estimation for human skeletal muscle. *Computer methods in biomechanics and biomedical engineering*, 18:1–11.
- Li, H., Sumner, R., and Pauly, M. (2008). Global correspondence optimization for non-rigid registration of depth scans. *Computer Graphics Forum*, 27.
- Macklin, M., Müller, M., and Chentanez, N. (2016). Xpbd: Position-based simulation of compliant constrained dynamics. In *Proceedings of the 9th International Conference on Motion in Games, MIG '16*, page 49–54, New York, NY, USA. Association for Computing Machinery.
- Majdisova, Z. and Skala, V. (2017). Radial basis function approximations: comparison and applications. *Applied Mathematical Modelling*, 51:728–743.
- Modenese, L. and Kohout, J. (2020). Automated generation of three-dimensional complex muscle geometries for use in personalised musculoskeletal models. *Annals of Biomedical Engineering*, 48.
- Müller, M., Heidelberger, B., Hennix, M., and Ratcliff, J. (2007). Position based dynamics. *Journal of Visual Communication and Image Representation*, 18(2):109–118.
- Oatis, C. A. (2017). *Biomechanics of skeletal muscle*. Lippincott Williams & Wilkins.
- Romeo, M., Monteagudo, C., and Sánchez-Quirós, D. (2018). Muscle Simulation with Extended Position Based Dynamics. In García-Fernández, I. and Ureña, C., editors, *Spanish Computer Graphics Conference (CEIG)*. The Eurographics Association.
- Skala, V. and Cervenka, M. (2019). Novel rbf approximation method based on geometrical properties for signal processing with a new rbf function: Experimental comparison. In *2019 IEEE 15th International Scientific Conference on Informatics*.
- Sorkine, O. and Alexa, M. (2007). As-Rigid-As-Possible Surface Modeling. In Belyaev, A. and Garland, M., editors, *Geometry Processing*. The Eurographics Association.
- Teschner, M., Kimmerle, S., Heidelberger, B., Zachmann, G., Raghupathi, L., Fuhrman, A., Cani, M.-P., Faure, F., Magnenat-Thalmann, N., Strasser, W., and Volino, P. (2005). Collision Detection for Deformable Objects. *Computer Graphics Forum*.
- Turk, G. (1990). Interactive collision detection for molecular graphics. Technical report, University of North Carolina at Chapel Hill, USA.
- Wade, S., Strader, C., Fitzpatrick, L., Anthony, M., and O'Malley, C. (2014). Estimating prevalence of osteoporosis: Examples from industrialized countries. *Archives of osteoporosis*, 9:182.
- Wang, B., Matcuk, G., and Barbič, J. (2021). Modeling of personalized anatomy using plastic strains. *ACM Trans. Graph.*, 40(2).
- Zhang, G., Wang, C., Liu, Q., Wei, J., Luo, C., Duan, L., Long, J., Zhang, X., and Wang, G. (2021). Development of skeletal muscle model for bridge-style movement rehabilitation. *Journal of Physics: Conference Series*, 2026:012061.
- Zhang, S.-X., Heng, P.-A., Liu, Z.-J., Tan, L.-W., Qiu, M.-G., Li, Q.-Y., Liao, R.-X., Li, K., Cui, G.-Y., Guo, Y.-L., Yang, X.-P., Liu, G.-J., Shan, J.-L., Liu, J.-J., Zhang, W.-G., Chen, X.-H., Chen, J.-H., Wang, J., Chen, W., Lu, M., You, J., Pang, X.-L., Xiao, H., Xie, Y.-M., and Cheng, J. C.-Y. (2004). The chinese visible human (cvh) datasets incorporate technical and imaging advances on earlier digital humans. *Journal of Anatomy*, 204(3):165–173.
- Zhao, Y., Clapworthy, G., Kohout, J., Dong, F., Tao, Y., Wei, S., and Mcfarlane, N. (2013). Laplacian musculoskeletal deformation for patient-specific simulation and visualisation. In *2013 17th International Conference on Information Visualisation*, pages 505–510.

9.13 A mathematical model for smooth Radial Basis Function implicit surface model for muscle modelling

The research paper titled "A mathematical model for smooth Radial Basis Function implicit surface model for the purpose of muscle modelling" [95] presents an innovative method for modelling muscle geometry. This approach is heavily theoretical and builds upon insights gleaned from our prior research. This theoretical model (but slightly improved) already had its dedicated Chapter 7.4, so I direct the dear reader to that chapter for more information.



The journal paper further develops knowledge acquired from earlier studies. Here's a summary of each publication and its key contribution to this paper:

- "A New Strategy for Scattered Data Approximation Using Radial Basis Functions Respecting Points of Inflection" [3]: Highlights the utilization of Halton point distribution and the incorporation of central points at domain boundaries.
- "Novel RBF Approximation Method Based on Geometrical Properties for Signal Processing with a New RBF Function: Experimental Comparison" [93]: Demonstrates limitations of local RBFs in certain approximations, suggesting three global RBF alternatives: Gaussian, TPS, and a newly developed one.
- "Modified Radial Basis Functions Approximation Respecting Data Local Features" [94]: Reveals how edge detection and curvature can enhance RBF approximation, with the concept of curvature preservation being adopted from this study.
- "Fast and Realistic Approach to Virtual Muscle Deformation" [65]: This foundational paper on muscle modelling principles greatly influenced the journal article by outlining the requirements of the muscle model.
- "Behavioural Study of Various Radial Basis Functions for Approximation and Interpolation Purposes" [89]: Concludes that while a novel RBF might be preferable for approximation, caution is needed in shape parameter selection due to potential instabilities in RBF matrix conditionality. The paper suggests Gaussian RBF is a potentially better choice.

- "Finding Points of Importance for Radial Basis Function Approximation of Large Scattered Data" [88]: Summarizes and tests previous research [3, 93, 94], contributing the insight that high compression ratios can maintain accuracy.
- "Conditionality Analysis of the Radial Basis Function Matrix" [91]: Provides valuable insights into shape parameter selection for Gaussian RBF through analytical study.
- "Muscle Deformation Using Position Based Dynamics" [71]: Examines PBD methods on a triangular mesh muscle model, highlighting the need to address muscle tissue entering the joint.
- "Geometry Algebra and Gauss Elimination method for solving a linear system of equations without division" [97]: Delves into solving the RBF equation system, focusing on the methodological intricacies.
- "Collision detection and response approaches for computer muscle modelling" [72]: Enhances understanding of collision handling approaches, relevant for future work with RBF geometric models.
- "Non-planar Surface Shape Reconstruction from a Point Cloud in the Context of Muscles Attachments Estimation" [13]: Discusses applying RBF to real musculoskeletal data and its unique challenges.
- "Computerised muscle modelling and simulation for interactive applications" [98]: Summarizes, tests, and expands the existing approach, mainly focusing on future research opportunities, which the journal paper explores. The paper is at the time of writing in the stage of submission.

Publication [95]:

CERVENKA, Martin; KOHOUT, Josef; LIPUS, Bogdan. A mathematical model for smooth Radial Basis Function implicit surface model for the purpose of muscle modelling. *INFORMATICA*. 2024, submitted

A novel radial basis function description of a smooth implicit surface for musculoskeletal modelling

Martin CERVENKA^{1,*}, Josef KOHOUT¹, Bogdan LIPUS²

¹ *Department of Computer Science and Engineering, Faculty of Applied Sciences, Technicka 8, 301 00 Pilsen, University of West Bohemia, Czech Republic*

² *Faculty of Electrical Engineering and Computer Science, Koroska Cesta 46, SI-2000 Maribor, University of Maribor, Slovenia*

e-mail: cervemar@kiv.zcu.cz

Abstract. As musculoskeletal illnesses continue to increase, practical computerised muscle modelling is crucial. This paper addresses this concern by proposing a mathematical model for a dynamic 3D geometrical surface representation of muscles using a Radial Basis Function (RBF) approximation technique. The objective is to obtain a smoother surface while minimising data use, contrasting it from classical polygonal (e.g., triangular) surface mesh models or volumetric (e.g., tetrahedral) mesh models. The paper uses RBF implicit surface description to describe static surface generation and dynamic surface deformations based on its spatial curvature preservation during the deformation. The novel method is tested on multiple data sets, and the experiments show promising results according to the introduced metrics.

Key words: radial basis function, muscle model, gradient descent, curvature, mean curvature, Gaussian RBF.

1. Introduction

Computerised muscle modelling garners increasing attention with the rising prevalence of musculoskeletal illnesses (Cieza *et al.* (2020)). As of the latest Scopus index, 31 papers on "musculoskeletal illnesses" out of 102 were published after 2020. On the term "musculoskeletal modelling", there are over one-third of papers published after 2020 compared to the total. This fast development in the field signifies that musculoskeletal modelling approaches play a significant role in musculoskeletal illness treatment. This paper aims to contribute significantly to the field by proposing a new mathematical muscle description. To provide context, we begin with a brief overview of the evolution of critical contributions.

Muscle modelling has a rich history, with Hill (1938) presenting the first formal mathematical representation, a three-element model for muscle fibre. As technology advanced,

*Corresponding author.

the need for precision led to the development of more intricate models. Initially, fibres were individually represented, as seen, e.g. in the alpine skiing context (Heinrich *et al.* (2023)). Subsequently, a "Via-points approach" emerged, linking multiple fibres in series, e.g., in shoulder muscle modelling (Abderrazak and Benabid (2023)).

While these methods often operate in one dimension, moving to higher dimensions, especially in 2D space, evolved crucial because the 1D model cannot accurately define the original shape, producing errors up to 75% (Valente *et al.* (2012)). Employing 2D space allows the description of muscles using surfaces, a more intuitive technique for reflecting external structures. This paper focuses mainly on 2D models, as detailed in Section 2.

The open, ultimate problem is developing an accurate, simple, fast, and smooth musculoskeletal model using the least possible parameters. All of the state-of-the-art methods lack one or another or give some compromises.

The paper primarily emphasises a detailed description of two-dimensional techniques, allowing faster calculations while keeping the ability to infer internal muscle composition. Two main objectives include achieving more immediate model deformation with fewer updated parameters and data reduction. The goal is achieved using a mathematically described RBF implicit surface geometric model, providing infinitely differentiable smooth surfaces and using fewer parameters than the "traditional" triangular mesh approaches.

The contribution of this paper lies not just in the overall RBF model but also in a novel metric to determine the approximation accuracy presented. This metric can also be used in different research fields where two surfaces must be compared. The RBF solver is another contribution, and its usage can also be extended to applications where some smooth objects (without sharp edges) need to be approximated, such as fluid dynamics, aerodynamics, computer graphics and more.

Section 2 describes various models already used for muscle modelling, followed by Section 3, including the description of RBF in general and muscle modelling. Section 4 then describes the paper's contribution to static and dynamic models. Finally, in Section 5, experiments prove the model works as expected in theory.

2. Related methods

Having established the theoretical background, we now advance to a detailed investigation of some techniques relative to muscle modelling.

The origin of a musculoskeletal model is deeply rooted in empirical data. The primary raw materials for these models are medical scans, such as Magnetic Resonance Imaging and Computed Tomography screenings, originating from living subjects and cadavers. Recently, the processing has evolved, transitioning from semi-automatic to fully automatic techniques. This progression involves the segmentation of images, a critical step where the relevant structures, such as bones, joints and muscles, are isolated and extracted from the rest of the image.

2.1. *Bones and joints*

Following segmentation, surface models of bones are created. This process is not just about assembling a skeletal structure; it involves the intricate specification of joints, setting constraints on the degrees of freedom within these joints. Here, e.g., the STAPLE algorithm by Modenese and Renault (2021) can significantly automate and streamline the process.

Triangular meshes are the most common way to approximate bone surfaces by connecting triangles to define the object's overall shape. The methodology for obtaining a triangular muscle model from a person entails a comprehensive process that begins with acquiring high-resolution, segmented medical images, commonly from MRI or CT scans. These images provide a detailed view of the bones (and surrounding tissues). The next step involves the extraction of a triangular mesh from these segmented images, a task typically accomplished using algorithms such as Marching Cubes (Lorenson and Cline (1987)) or Marching Triangles (Hilton *et al.* (1996)). These algorithms traverse the voxel grid of the images, forming vertices, edges, and faces that approximate the bone surface, effectively converting the 2D segmented images into a 3D surface representation.

Following the initial mesh extraction, the mesh undergoes a series of refinement procedures to enhance its quality and anatomical accuracy. These procedures include the removal of non-manifold vertices and edges to eliminate anomalies that do not adhere to the criteria of a well-defined surface. Concurrently, any gaps or holes in the mesh are meticulously filled to ensure continuity of the bone surface, a critical step for maintaining anatomical fidelity. The mesh is further refined by optimising the shapes of the triangles to more accurately align with the bone's contours and by reducing the mesh to lower its complexity while keeping the model's overall shape and intricate details.

Additionally, Laplacian smoothing is applied to the mesh. This process adjusts the position of each vertex based on the average of its neighbouring vertices, effectively smoothing out irregularities and noise, resulting in a uniform and more realistic representation of the bone. The refined mesh is then rigorously reviewed and compared against the original segmented images by medical professionals to validate its accuracy, ensuring the model precisely mirrors the anatomical structure.

While the forces used on bones during the movement can induce deformations, these deformations are generally so minute that, for practical purposes, they can be disregarded. This assumption yields the bones as rigid bodies in motion. This assumption is in contrast to the muscle-tendon units.

2.2. *Muscles and tendons*

Joint data about the muscles and tendons (together form muscle-tendon units - MTU) is acquired because it is difficult to distinguish the muscle and the tendon apart from the imaging; otherwise, the data can be obtained in the same fashion as in the case of bones. However, the problem is that these tissues are less visible in the imaging. Fortunately, the segmentation can also be automated, but more complex strategies are required, with

machine learning methods emerging as promising approaches (Goyanes *et al.* (2024)). Despite their potential, these techniques have yet to become a staple in routine practice, and semi-automatic approaches are used instead.

If the deformation is discussed, muscles behave differently than bone structures; they exhibit elastic deformations, presenting a significant challenge in modelling their behaviour accurately to the known movement of bones. Various models attempt to address this, often employing many oversimplifications.

Also, in the real-world scenario, muscle deformation results from the contraction or relaxation of muscle fibres, driven by the sliding of actin and myosin filaments within muscle fibres. This contraction leads to changes in muscle shape, causing deformation. In inverse kinematics, muscle deformation is pretended to be caused by adjacent bone movement, posing the challenge of figuring out the shape of unknown parts of the muscle model. If we do not consider measurements of the physiological signals (e.g. electromyography), it is required to perform the inverse kinematics because the bone displacements are known; however, the shape of the muscles during the movement is not (due to the problematic data acquisition).

The first models created were one-dimensional and were formed by a straight line, polyline, or curve and their multiples.

2.2.1. 1D models

The simplest models might represent a muscle as a straight line connecting two points on different bones, blatantly ignoring any potential intersection with the bone itself. This approach relies heavily on the assumption that the attachment points are accurately chosen (Kohout. and Cervenka. (2022)). A slight improvement to this model is to replace the straight line with a polyline or a curve that either passes through predetermined points relative to a bone or traces the surface of a parametric body, aiming to minimise the curve length. However, fine-tuning these models to mirror reality is arduous (Hájková and Kohout (2014)).

It's also imperative to represent muscles not just as mere lines or curves but as substantial higher-dimensional models (Kedadria *et al.* (2023)) to enhance the realism of musculoskeletal models, i.

2.2.2. 3D models

Geometrical 3D models have been explored, utilising the fact that a large set of muscle fibres creates the muscle. Modelling approaches using mass-spring systems (Janák. and Kohout. (2014)) and the finite element (FE) method are currently gaining prominence (Delp and Blemker (2005)). The FE method, utilising a template of internal structure projected onto the surface shape, proves superior accuracy. The main issue of those methods is their large number of parameters, which need to be figured out, and their computational complexity. Therefore, this paper concentrates more on the two-dimensional (surface) models instead, which is a reasonable compromise between a simple but inaccurate 1D modelling and 3D accurate but too complex 3D modelling Macklin *et al.* (2016).

2.2.3. Discrete 2D models

Despite muscles lying in the 3D space, modelling them in lower dimensions is viable. Straight lines or polylines can be employed in one dimension but inaccurately. In two dimensions, already obtained triangular surfaces are limited in the original object description but can be accurate enough. Each "well-behaved" 3D surface model constructed by a single component manifold can be parameterised in two dimensions or described utilising a 2D representation.

In the case of model dynamics, numerous approaches have been developed, primarily working directly on the triangular meshes that have already been obtained. Here, we provide a brief overview of the most significant ones.

Position-based dynamics. Position-based dynamics, introduced by Müller *et al.* (2007), is a rapid technique widely utilised in the animation industry for simulating deformations of elastic objects. Initially created for generic shapes, the method inputs a smooth manifold surface mesh and generates its deformed counterpart.

An extended version, XPBD (Macklin *et al.* (2016)), introduces the concept of elastic potential energy, removing the need to determine time steps and iteration counts. XPBD has been applied to muscle modelling challenges by Romeo *et al.* (2018), who addressed limitations regarding convergence speed, setup simplicity, intuitive controls, and artistic control. They constructed an internal structure inside the surface mesh, accounting for muscle anisotropy. However, the approach lacked attention to collision detection and resolution, later addressed by Cervenka. *et al.* (2023).

In 2019, Angles *et al.* (2019) introduced an XPBD-based method for muscle modelling, virtually decomposing the muscle into flexible "rods" approximating muscle fibres. These rods adjust their diameter to maintain volume, enabling real-time simulation. The main issue is the same: the problem of muscle-bone penetration is not adequately addressed or apparent in the paper or its supplementary material.

As-rigid-as-possible surface modelling. The as-rigid-as-possible approach (ARAP), developed by Sorkine and Alexa (2007), is a technique for determining minimal non-rigid transformations in a surface mesh. Unlike the method by Kellnhofer and Kohout (2012), which involves volume constraints and an internal skeleton, "it distinguishes itself by not requiring an internal structure." Fasser *et al.* (2021) applied ARAP in a medical context to morph the template of a pelvic bone onto subject-specific landmarks, though overlooking certain critical features like volume preservation. Wang *et al.* (2021) investigated ARAP but chose not to use it due to non-smooth shapes and spikes in results. ARAP has also found application in time-varying meshes by Dvořák *et al.* (2021).

ARAP minimises shape deformation by discouraging non-rigid transformation via a cost function. Mathematically, this is characterised as the search for a solution to a non-homogeneous system of linear equations. The matrix corresponds to the discrete Laplace operator of the mesh, while the right-hand side vector encompasses the second differences of each vertex concerning its neighbours.

The main drawback of ARAP and similar approaches lies in recalculating each mesh parameter in every iteration, proving time-consuming for fine triangular meshes. Also, the

triangular mesh has the smoothness issue mentioned before. Therefore, a continuous 2D model better approximates smoothness and is better suited for the deformation approaches.

2.2.4. Continuous 2D models

The Non-uniform Rational Basis Spline (NURBS) is an interpolation and approximation method (Nie *et al.* (2022)), describing a surface via a set of B-spline curves. NURBS is useful for interactive applications, offering higher smoothness than triangular meshes, allowing intuitive deformation of specific parts of the geometrical model (see, e.g. Clapés *et al.* (2008) or Sánchez-Reyes and Chacón (2020)). For muscle modelling, the NURBS is not required to form a structure as triangular meshes when creating the data and then approximate those data, meaning that after applying the Marching Cubes or Triangles step, the vertices will not be connected by triangles but rather by individual NURBS patches and the smoothness is accomplished automatically. Also, if a part of the segmentation is missing, the NURBS techniques can restore those parts more precisely than in the triangular mesh case.

D-NURBS (Terzopoulos and Qin (1994)) (Dynamic-NURBS) can be used for dynamics. It extends the NURBS approach by incorporating physical phenomena for more intuitive interactive shape deformation relevant to various purposes such as high-resolution digital terrain (Ye *et al.* (2020)), virtual reality (Lavoie *et al.* (2006)), or CAD applications (Zhang and Qin (2001)).

Na *et al.* (2023) used D-NURBS for muscle shape parametrisation and deformation. They achieved satisfactory results by defining muscle shape with a set of neighbouring 2D NURBS plates, reaching a mean square difference of approximately 1.5mm and a maximum volume error of 0.75%.

While these approaches offer valuable insights, a common challenge is recalculating every parameter in each iteration, which can be time-consuming. Also, the neighbourhood of vertices needs to be defined for all the techniques mentioned.

3. Radial basis functions

Even the previously described NURBS techniques require some relations between the vertices (neighbourhood), but Radial Basis Function (RBF) techniques do not require that knowledge. The task of estimating and fitting scattered data is widespread across various fields of engineering and scientific research. To list only a limited subset of the RBF usage possibilities, Zhang *et al.* (2022) shows its usability for better predicting the sub-cellular location of long non-coding RNA. Oliver and Webster (1990) demonstrates this by applying the kriging method for interpolating geographical data, while Kaymaz (2005) demonstrate its effectiveness in solving issues related to structural reliability. The process has further applications in simulation, e.g., in Sakata *et al.* (2004) study on wing structures or Joseph *et al.* (2008) development of metamodels. RBF techniques address partial differential equations, especially in engineering.

The RBF method was presented by Hardy (1971) and improved by him (Hardy (1990)) later. Majdisova and Skala (2017a) have offered different strategies for RBF position-

ing. Significant studies have been conducted in terms of 'shape parameters' in RBF approaches, including, e.g. search for optimal settings by Wang *et al.* (2021), investigations by Afiatdoust and Esmailbeigi (2015), and research into varying local shape parameters methods by Cohen *et al.*, Sarra *et al.*, and Skala *et al.* (2020).

The radial basis function approach uses "centre points" instead of surface vertices, serving as descriptors for spatial function locations. Unlike NURBS, modifying a single RBF centre location can affect the entire surface because of its "blending" behaviour with other RBFs nearby, propagating recursively throughout the volume. To address the RBF placement possibilities, Majdisova and Skala (2017b) conducted a comparative study of various RBF placement strategies, showing that uniformly sampled input functions tend to lead to ill-conditioned RBF equation systems. They seek the solution by proposing a quasi-random sampling, such as the Halton distribution. Alternative methods have also been suggested, such as regularisation by Orr (1995) and enhancements close to function boundaries by Wright (2003).

The RBF approach offers advantages over triangular mesh representations of muscle surfaces, including:

- There is no need to define the connectivity between control points.
- RBFs can generate C^n -smooth surfaces, with the user-defined required degree of smoothness denoted by n .
- RBF may use fewer parameters compared to triangular meshes in general.
- Deforming a muscle may involve changing only a portion of the parametric space while maintaining smoothness.

For each application, the correct RBF must be chosen first to reach desirable properties and satisfactory results. Two global RBFs are commonly considered: the Gaussian RBF or thin-plate spline (TPS). The Gaussian RBF is defined as $e^{-\alpha r}$, and TPS as $r^2 \log r = \frac{1}{2} r^2 \log r^2$. The decision of which one to use cannot be made by the differentiability criterion (because both are infinitely differentiable). The Gaussian can also be adjusted using the shape parameter; however, TPS (in its original form) cannot. There are also other options for global RBFs (such as in Skala and Cervenka (2019)) to consider; however, Gaussian RBF and TPS provide stability and a good understanding of their behaviour and are already the most analysed by others.

3.1. *Mathematical description*

A single RBF is a mathematical function defined solely by the distance from a designated point, denoted as the centre. Mathematically, it can be expressed as $\varphi(\|\mathbf{x}_i - \boldsymbol{\xi}_j\|)$, where $\boldsymbol{\xi}_j$ represents the center point's coordinates, and \mathbf{x}_i is an arbitrary independent point.

The expression of RBF approximation, defining the function value at any specific input points, is as follows:

$$h(\mathbf{x}_i) = \sum_{j=1}^N \lambda_j \varphi(r_{ij}) \quad (1)$$

Here, the approximation h at the location of \mathbf{x}_i is described by $r_{ij} = \|\mathbf{x}_i - \boldsymbol{\xi}_j\|$ the distance between the input point \mathbf{x}_i and the centre point $\boldsymbol{\xi}_j$; and $\boldsymbol{\lambda}_j$ is the weight of a single RBF φ , which can be e.g. already discussed Gaussian RBF or TPS. The equation (1) can be formulated in a matrix form, resulting in a square matrix \mathbf{A} within the system of linear equations. A simple example of RBF approximating a function can be seen in Zhao and San (2011).

The idea can be extended using polynomial conditions, which can improve accuracy. The extended system of equations, considering these conditions, is expressed as:

$$\begin{bmatrix} \mathbf{A} & \mathbf{P} \\ \mathbf{P}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\lambda} \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} \mathbf{h} \\ \mathbf{0} \end{bmatrix} \quad (2)$$

Here, \mathbf{P} are the polynomial conditions, $\boldsymbol{\lambda}$ represents RBF weights, Big \mathbf{A} is the RBF matrix without the polynomial constraints, small \mathbf{a} contains resulting coefficients of the polynomial, and \mathbf{h} encompasses values at the input points.

Suppose we disregard the polynomial conditions for a while. In that case, RBF approximation involves solving the equation (1). The equation can also be described as an overdetermined system of linear equations $\mathbf{A}\mathbf{x} = \mathbf{b}$, where \mathbf{A} is a rectangular matrix, \mathbf{b} and \mathbf{x} are vectors, and the number of rows N is greater than the number of columns M .

The Ordinary Least Squares (OLS) method can choose weights λ_j , minimising mean square error (MSE). This involves computing weights utilising the inverse or pseudoinverse: $\boldsymbol{\lambda} = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{h}$. Although typically satisfactory, OLS can encounter stability problems, as noted by Skala (2017), which needs to be kept in mind while solving. It is often beneficial to even choose another solver instead. Furthermore, an issue when combining values of different units in the OLS method to solve the equation system with polynomial conditions emerges, which needs to be properly addressed (Skala (2017)). The different units can be found in matrices \mathbf{A} and \mathbf{P} , see equation (2).

3.2. RBF for muscle modelling

Due to the often intricate shape of muscles/bones (e.g., with many triangles, quadrilaterals, etc.), approximating the muscle/bone rather than interpolating is advantageous for subsequent calculations. When approximating, spatial placement of individual RBF is critical. A naive approach involves uniformly sampling the input shape (scalar distance field of the volumetric data or triangular mesh in the case of muscle modelling), but this may not adequately capture the muscle/bone underlying properties.

Even though utilising the polynomial conditions improves the outcomes and enhances the matrix conditionality in some cases, the presumption is that the input data behaves like the selected polynomial. However, this is not the case in muscle modelling, where the data's overall shape resembles somewhat an ellipsoidal shape rather than a polynomial form, further diminishes the relevance of polynomial constraints, which will not be discussed further.

4. Novel RBF Mathematical model

The number of parameters of the triangular mesh is often large. To address the issue of reducing parameters during deformation, a new mathematical model is proposed, encompassing both static muscle shape and its dynamic configuration during bone movement.

4.1. Geometrical static model

Even though the number of parameters will be reduced, many will still be present. Therefore, the extensive emerging RBF linear equation system would be ill-conditioned. The static model is constructed using a greedy approach, adding one RBF after another to solve the issue. In each iteration, the algorithm identifies the position and shape of a single RBF to minimise the approximation error effectively.

The objective is to minimise the squared error of the approximation to the original triangle mesh across all vertices throughout the bounded space $\Omega \subseteq \mathbb{R}^d$:

$$\int_{\mathbf{x} \in \Omega} \sum_{j=1}^k \left\| \sum_{i=1}^N \lambda_i \phi(\|\mathbf{x} - \boldsymbol{\xi}_i\|) - \mathbf{v}_j \right\|_2^2 d\mathbf{x} \quad (3)$$

Here, N denotes the current number of employed RBFs, k is the number of vertices in the triangular meshes, d is the number of dimensions, and \mathbf{v}_j is the respective vertex. The notation $\|\mathbf{x}\|_2$ represents the L2 norm of a vector \mathbf{x} .

Creating the geometrical static model initiates the generation of a scalar distance field (SDF) on the triangular mesh. This field is later sampled using a (quasi-)random sampling method. Throughout the research, various distributions, including uniform, random uniform, Gaussian, and Halton, were tested. The findings indicated that the Halton distribution emerged as the most effective option, which is following other research (see, e.g. Majdisova and Skala (2017b), Cervenka *et al.* (2019)).

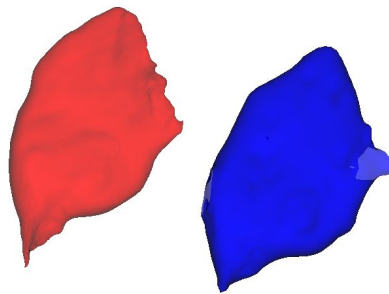


Fig. 1. The geometrical static model depicts a gluteus maximus muscle. The red represents the original triangular mesh, while the blue represents its RBF approximation.

4.1.1. RBF placement

In this study, we followed the placement of each RBF by Carr *et al.* (2001), placing centres at the three level-sets of the isosurface. However, in the case of muscle modelling, only a subset of the proposed set is required. This subset is determined through a straightforward trial-and-error method. It involves evaluating an objective function and dynamically minimising it. Therefore, this placement approach can be described as greedy. The process involves checking the objective function of each RBF for every vertex across all samples within the input space (the bounding box of a mesh), as well as for each predefined shape parameter (refer to section 4.1.2 for more details).

Greedy placement is a sub-optimal approach because two independently and subsequently placed centres independently select a shape parameter for each RBF compared to two "cooperating" centres, which are not required to be optimal at the time of their selection. However, this approach avoids the stability issue by using an extensive system of linear equations as a solution.

The optimal position is for each centre selected, and the RBF function is subtracted from the volume bounded by, e.g., the AABB box and the process is iterated until either the allotted number of RBFs is utilised or the predetermined level of accuracy is achieved. This centre point distribution approach was designed to avoid the OLS stability issue altogether.

4.1.2. Shape parameter selection

Choosing the radius of action, also called a shape parameter, is pivotal in achieving precise interpolation or approximation. While one option is to select a shape parameter for each RBF independently, this can lead to many stored parameters. On the other hand, determining an optimal global shape parameter for all RBFs leads to inaccuracies and using it as a compromise remains an open question. Different approaches have been suggested, such as those by Wang *et al.* (2021), Afiatdoust and Esmaeilbeigi (2015), and Sarra and Sturgill (2009). Still, they may not always identify the optimal shape parameter for each RBF. During testing, we discovered that the single global shape parameter is not exhaustive. Search from a limited list of possible shape parameters is suitable because the number of parameters is low. The condensed set of parameters reduces the storage capacity and computational complexity while searching for the most suitable one.

4.1.3. Objective functions

The task at hand for evaluating the placement of individual RBFs is to assess how accurately the placement represents the original shape. One possible metric for this assessment is the Jaccard index; the second is the mean square error.

Jaccard index. The Jaccard index quantifies the disparity in volume* between two objects: the original mesh and the surface formed by RBF. It is mathematically defined as follows:

*Strictly speaking, it is a disparity between the area inside and outside both objects and the areas inside one object but outside the other.

$$J_i = \frac{i}{u} = \frac{i}{i + d} \quad (4)$$

In this context, i represents the intersection volume, u symbolises the union of both meshes, and d represents the difference between both meshes. Calculating the exact volume of intersections can be complex, so a rough approximation might be a more practical choice. Rather than using volumes, the number of samples (using the already described Halton distribution) in each group can be used. A higher number of samples leads to more significant computational accuracy. It is worth noting that while the Jaccard index helps specify the difference between two shapes, it does not provide an understanding of how much is "inside" or "outside" something, which is a feature to consider.

Mean square error. The Mean Square Error (MSE) provides a more detailed insight into both muscles' interior and exterior rather than a binary assessment. Calculating MSE requires knowledge of interior information. Interestingly, no additional information is necessary for the RBF representation, as it can be evaluated at any point within the volume. For the original surface mesh, interior information can be acquired using a scalar distance field (SDF) already obtained for its creation, allowing us to determine the distance of each vertex from the surface mesh.

The computation of the MSE objective function entails integrating the squared L2 norm across the subset Ω of the original space \mathbb{R}_d and can be expressed as:

$$M_{se} = \int_{\Omega} \|s(\mathbf{x}) - h(\mathbf{x}_i)\|_2^2 d\mathbf{x} \quad (5)$$

Here, the function s represents the Scalar Distance Field (SDF) constructed over the mesh, while h represents the sum of all currently utilised RBFs. When computed, a more efficient approach is subtracting each RBF from the function s and then calculating the updated s norm.

It's important to note that these objective functions may encounter challenges when dealing with the translational motion of the shape. This paper assumes that the muscle movement occurs only in part of the muscle while the rest remains static (for instance, the muscle is attached to two bones, but only one moves). This assumption holds as long as the deformation of a muscle is done in the local scope, with one static and one moving bone (meaning both bones cannot translate together during deformation).

Delving further into the challenge of selecting the appropriate objective function, opting for the MSE tends to yield a smoother surface. At the same time, the Jaccard index aligns more closely with the original model at the expense of introducing a less smooth surface. Hence, a combination of both proves to be a favourable approach.

12

4.2. Metrics

The metric used to compare the original surface with a new one is a weighted sum between the Jaccard index and MSE. This metric is proposed because the smooth surface provided by MSE and the surface location accuracy provided by the Jaccard index are required. The weight σ is called as a *smoothness* factor:

$$C_m = (1 - \sigma) (1 - J_i) + \sigma \frac{M_{se}}{\max M_{se}} \quad (6)$$

The equation calculates the final metric C_m using the smoothness factor σ , Jaccard index J_i , and MSE value denoted as M_{se} . The Jaccard index is expressed as $1 - J_i$ to penalise the lower Jaccard index more than the higher one. The MSE variable M_{se} must also be normalised into zero to one interval, which is accomplished by dividing the MSE total value by the maximum one possible.

4.2.1. Regularisation

Regularisation optimises how the RBFs are placed, increasing the likelihood of placing RBF centres in a predefined position or formation. When forming the RBF surface, it may be advantageous to position RBFs within the muscle volume and less outside, resulting in a more accurate curvature field, including fewer local extrema near the surface. The central concept involves penalising the RBFs placed significantly outside the desired region more heavily than the others.

When utilising the Signed Distance Field (SDF), the sign indicates whether the target vertex is inside or outside. Let's denote the signed distance as d_{v_i} . To obtain relative values rather than absolute distances, we can divide d_v by the maximum possible distance d_{\max} . This yields $r_d = \frac{d_{v_i}}{d_{\max}}$, which falls within the interval $(-\infty, 1)$.

Additionally, to enable the use of wider RBFs (avoiding the potential replacement by less advantageous numerous RBFs with minimal overall influence), a similar penalisation for the RBF shape parameters can be implemented: $r_s = \frac{\alpha_i}{\alpha_{\max}}$. Since $\forall i, \alpha_i > 0$, the r_s value will fall within the interval of $(0, 1)$. The wider RBFs are then useful in forcing fewer parameters to move to deform the overall shape.

To summarise, the parameters for the following need to be determined. The regularisation factor γ represents the ratio of how much of the original cost function is employed.

$$C_r = \gamma C_m + (1 - \gamma) \frac{d_{v_i}}{d_{\max}} \frac{\alpha_i}{\alpha_{\max}} \quad (7)$$

4.3. Mathematical dynamic model

The bones undergo a rigid transformation in inverse kinematics in our studied scenario, typically following the desired real-world movement. Subsequently, in an inverse manner, the shape of all the muscles related to the bone needs to be reconstructed. The portions of

the muscle attached to the bones are mandated to move in conjunction with these bones, while the remaining parts of all muscles require reconstruction. In the following text, the muscle is meant to be the whole muscle-tendon unit to ensure that the "muscle" is correctly attached to a set of bones. Also, for the simplicity of the following text, just a singular muscle involved in deformation is considered.

The main idea of the novel dynamic model is to find a new shape according to the curvature of the original one, preserving its initial shape as much as possible. The mathematical description to maintain the initial shape as much as possible can be described in many ways. In the case of muscle modelling, the muscle's initial curvature throughout the whole volume is a suitable shape descriptor. Mathematically speaking, we define the cost function as the total difference of the original ($\kappa_{\mu f_i}$) and current curvature ($\kappa_{\mu f}$) over the bounded space $\Omega \subseteq \mathbb{R}^d$, and we are looking to obtain its gradient:

$$C_f = \int \dots \int \|\kappa_{\mu f} - \kappa_{\mu f_i}\|_2^2 dx_1 \dots dx_d = \int_{\Omega} \|\kappa_{\mu f} - \kappa_{\mu f_i}\|_2^2 d\mathbf{x} \quad (8)$$

Then we use the definition of the 3D RBF approximation (and for the sake of simplicity, use g_i as the individual weighted RBF):

$$f(\mathbf{x}) = \sum_{i=1}^N \lambda_i \phi(\|\mathbf{x} - \boldsymbol{\xi}_i\|) = \sum_{i=1}^N g_i(\mathbf{x}) \quad (9)$$

Because to calculate the curvature, the eigenvalues of the Hessian matrix of f are required to find, the calculation of second partial derivatives of f is needed. From the Hessian matrix, all of the eigenvalues can be obtained.

For this paper, only global RBFs were considered at first due to their influence over the entire space, allowing the change of the whole model by altering fewer parameters. Moreover, for that purpose, the global Gaussian RBF is chosen for this paper due to its simplicity if differentiation is considered, which is beneficial when creating this dynamical model. If the Gaussian RBF is considered, then the mean of all of the eigenvalues can be calculated as:

$$\kappa_{\mu f} = \kappa_{\mu}(\mathbf{H}(f(\mathbf{x}))) = \frac{2}{d} \sum_{i=1}^N \alpha_i g_i(\mathbf{x}) \left(2\alpha_i \|\mathbf{x} - \boldsymbol{\xi}_i\|_2^2 - d \right) \quad (10)$$

Combining (10) and the gradient of (8), the complete cost function can be expressed as:

$$\nabla C_{fkj} = \frac{8}{d} \int_{\mathbb{R}^d} (\kappa_{\mu f} - \kappa_{\mu f_i}) \alpha_k^2 g_k(\mathbf{x}) (x_j - \xi_{kj}) \left(2\alpha_k \|\mathbf{x} - \boldsymbol{\xi}_k\|_2^2 - 2 - d \right) d\mathbf{x} \quad (11)$$

The calculation, including all the computational steps, can be found in Appendix C. The result implies that the direction of the gradient depends on the following factors:

1. $(\kappa_{\mu f} - \kappa_{\mu f_i})$ – the direction of the gradient is influenced by the disparity between the original and new curvature across the entire interval. The more significant this difference, the larger the magnitude of the gradient.
2. α_i^2 – the shape parameter plays a crucial role. A broader function exerts a more significant influence on the gradient of the respective centre point ξ_{kj} .
3. $g_i(\mathbf{x})$ – naturally, the RBF itself plays a significant role.
4. $(x_j - \xi_{kj})$ – as we integrate further from the centre point, its impact diminishes.
5. $(2\alpha_i \|\mathbf{x} - \xi_i\|_2^2 - 2 - d)$ – the distance of the current RBF to the integration variable is not just significant exponentially in g_i (expressed in third point) but also linearly (subject to specific constant manipulations).

The new mathematical model for the RBF shape deformation defines where all centre points should be translated (in the direction specified by its gradient). At this point, we can, according to the broader shape parameters and larger weights, decide which RBFs are more significant and calculate the deformation more often. In contrast, the less critical parameters can be recalculated only a few times, which may reduce the computational complexity significantly.

It's crucial to re-emphasise the centrality of our mathematical model in this study. Even though this paper does not adequately address any collision detection and response (CD/R) issues, similar techniques to Cani-Gascuel and Desbrun (1997) hierarchical CD/R can be used in the future. This section has encapsulated the theoretical and practical applications of radial basis functions in muscle modelling and illuminated the robustness and versatility of our mathematical approach. Through rigid exploration of complex mathematical concepts and their biomechanical applications, this research emphasises the potential for advanced modelling techniques in future studies, paving the way for groundbreaking innovations in biomechanics and beyond.

5. Experiments

The proposed approach has been tested to ensure its usefulness for musculoskeletal modelling. Here, we detail the critical experiments conducted to test the relevance and efficacy of the radial basis function approach in muscle modelling, concentrating on processes and their implications for our theoretical description.

The mathematical model of the curvatures can be evaluated using actual data from the gluteus maximus, gluteus medius, iliacus and adductor brevis muscles. The initial experiment will be conducted without regularisation, solely relying on the greedy placement approach.

5.1. Geometrical static model

Before we dive deeper into the muscle dynamics experiments, the first experiment shows the results of the static model. These experiments prove the correct centre placements.

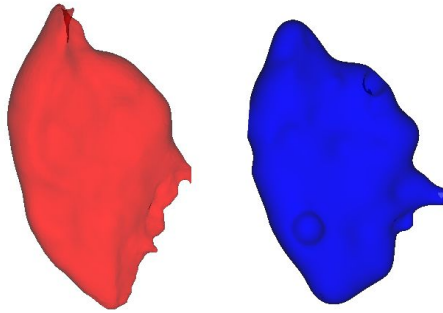


Fig. 2. The original gluteus maximus muscle in red and its RBF counterpart in blue.

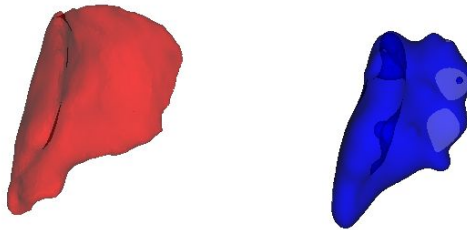


Fig. 3. The original gluteus medius muscle in red and its RBF counterpart in blue.

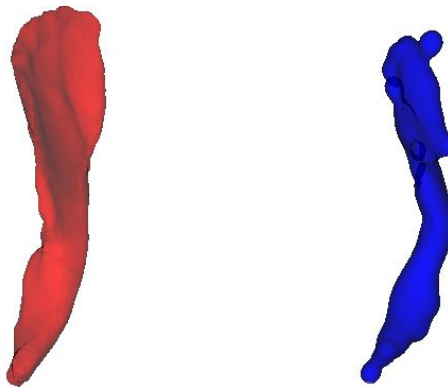


Fig. 4. The original iliacus muscle in red and its RBF counterpart in blue.

5.1.1. *Greedy centre placement*

Using a greedy approach, we employ a naive RBF placement strategy in the initial experiment. The results of such placement can be seen in Fig. 2, 3, 4 and 5. While this approach is straightforward to implement and relatively fast in execution, as depicted in Fig. 11, it exhibits issues with numerous small (but still different in size) RBFs distributed across

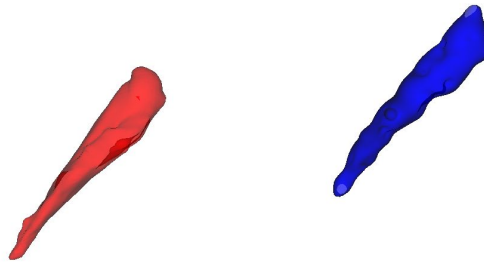


Fig. 5. The original adductor brevis muscle in red and its RBF counterpart in blue.

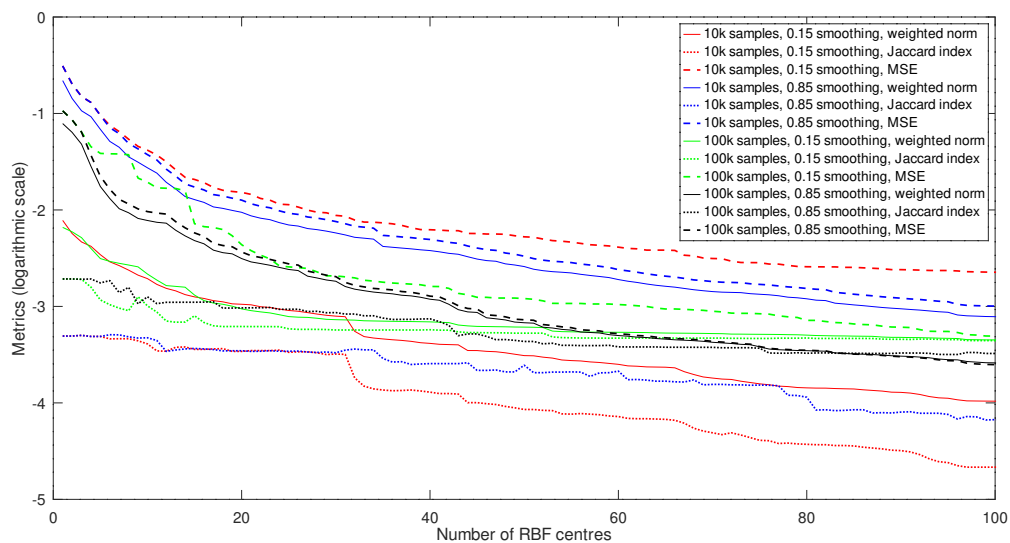


Fig. 6. The metrics evaluation on gluteus maximus muscle up to 100 RBF centres, with different sampling density and smoothness weight.

the entire space. The approach forms a complex curvature field with numerous potential local extrema. Additionally, the curvature outside the muscle boundaries (though barely visible in the image) is cluttered with unneeded curvature differences, further exacerbating the proliferation of local extrema. The problem is solved with regularisation, and the next section describes the experiment.

5.1.2. Regularisation

By including the regularisation term, higher curvatures will be concentrated at the borders of the original triangular mesh. The regularisation creates more promising conditions for the gradient descent method to achieve a superior approximation of the muscle in motion, mitigating the problem of numerous local minima in the field. Three examples are presented here: one with a regularisation factor of 0.7, another with a factor of 0.3, and the last with a regularisation factor of 0.05. The outcomes can be observed in the appendix A in Figures 12, 13, and 14, respectively.

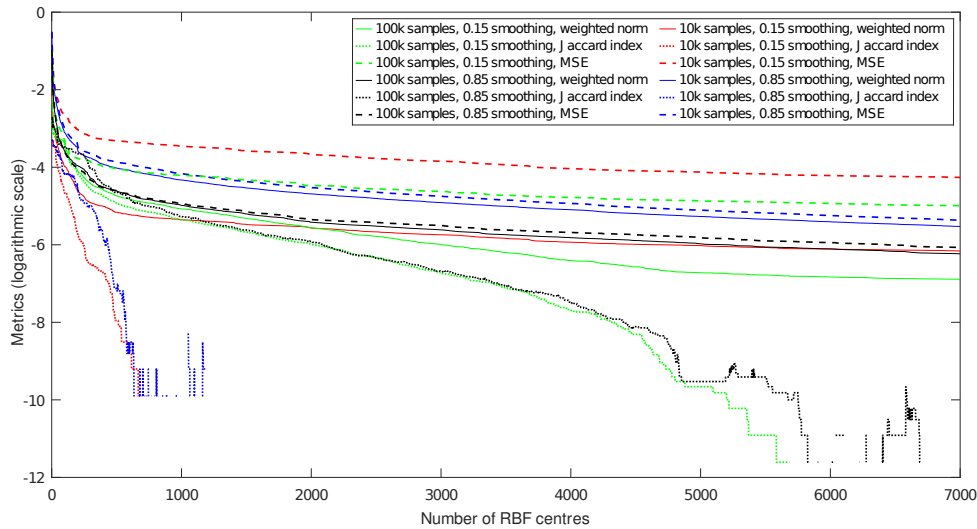


Fig. 7. The metrics evaluation on gluteus maximus muscle, with different sampling density and smoothness weight.

5.1.3. Metrics

The weighted metrics (between the MSE and JI) have been tested with two weights (0.15 and 0.85) and on two different sample resolutions (10k and 100k vertices). The evaluation for the first hundred RBF centres is shown in Fig. 6, which describes the initial part of the metrics convergence.

One may tell from the results that both metrics converge similarly, and in the end, they tend to go to a fixed value. However, if you continue further, it will be evident that this is not the case because the penalisation of the Jaccard index holds near a fixed value. Still, after a while, it goes faster towards zero (on a non-logarithmic scale). The results for more (7000) centres are visualised in Fig. 7. The other takeaway is that the finer the sampling (green and black dotted curves), the longer it takes to lower the penalisation of the Jaccard index (red and blue). Also, the lower the smoothing factor (red and green), the higher the MSE (dashed lines), confirming that both metrics describe the shape difference significantly differently.

5.1.4. Limitation

A well-known problem with the static surface RBF approximation is its ineffectiveness in accurately approximating sharp edges. This drawback is less critical in muscle modelling, as most soft tissues are relatively smooth and do not possess sharp edges, often resembling spherical shapes. However, this issue can be demonstrated using a simple 3D volumetric shape like a tetrahedron. A tetrahedron has six sharp edges (each with a dihedral angle exceeding 70 degrees) and four corners, which present significant challenges for RBF approximation.

The fundamental issue with RBFs is that each RBF inherently forms a spherical iso-surface at a given value. To create a sharp corner would theoretically require an infinite number of infinitesimal RBFs. An example of attempting to develop an RBF static surface

for a tetrahedron is shown in Fig. 8. Let's also note that in the case of the tetrahedron and similar shapes, often no data reduction is achieved, but rather the opposite.

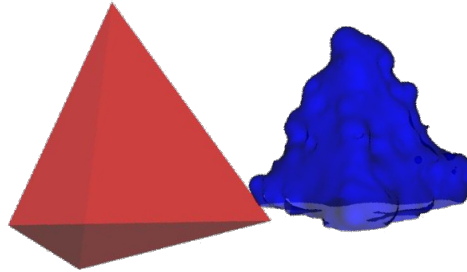


Fig. 8. An attempt to create an RBF surface for a tetrahedron demonstrates the unsuitability of RBF for approximating sharp edges. RBF is ineffective for shapes like a tetrahedron due to its inability to model the sharp edges accurately.

5.2. Deformation

Both options involving and neglecting the regularisation factor were tested. For testing, the centre points were shifted randomly, with a magnitude of 5% of the AABB box diagonal, and the expectation is that the centre points return to/near their initial positions. The successful experiment with regularisation is shown in Fig. 9 and 10 and the full results are visualised in the appendix B in the respective figures. The experiments supported the idea of using regularisation to lower the number of local extrema for the gradient descent because it converges much closer to the original shape. The first experiment without the regularisation factor was only about 0.7% faster on the standard PC than the second one, which cannot be considered a statistical difference.

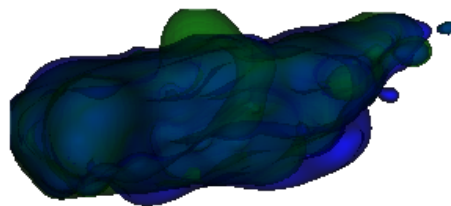


Fig. 9. The initialisation, where the randomly deformed green muscle should return to its initial shape in blue.

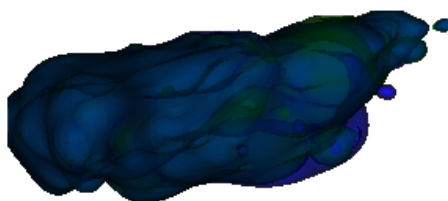


Fig. 10. The restoration copies the original blue shape with the green one, despite minor differences on the right side.

6. Conclusion & Future work

This paper shows a novel approach for modelling a surface with the radial basis function approach. The proposed method offers a way to model a static and dynamic muscle surface, altogether avoiding recalculating the geometrical model's parameters to deform it. Even though there are some examples where the approach is not applicable (e.g. the tetrahedron), the suitability for the muscle reconstruction is apparent. Regularisation helps reduce the number of local extrema. However, it also reduces the resulting precision; luckily, a compromise can be reached using a suitable regularisation term selection.

A deformation methodology respecting muscle properties (muscle volume, shape preservation, and bone avoidance) is the most obvious candidate for future work. A good starting point seems to be the work from Cani-Gascuel and Desbrun (1997) describing the geometrical model deformation in general. However, their method uses the model generated by a skeleton, which is not reasonable for modelling a muscle shape, which would need an overcomplicated skeleton to be able to work. The following work and a great review from Lee *et al.* (2012) describe the possibilities of modelling muscle deformation using skeleton-generated implicit surfaces.

It is not just the novelty but also the oversight of the approach that is apparent; the metric discussed in the article extends beyond its use in muscle modelling or general modelling applications. It is versatile enough to be applied in any situation that requires the comparison of two surfaces. Similarly, the concept of regularisation, with its mathematical underpinnings suitable for RBF surface modelling, can be adapted for broader applications. Its fundamental principle of constraining centres to remain inside applies to more general contexts, such as clustering methodologies.

Acknowledgments

The authors thank their colleagues at the University of West Bohemia and the University of Maribor. Thanks to Vaclav Skala and Ivana Kolingerova for their valuable insight and discussion.

Funding

This research was supported by the Czech Science Foundation, project number 23-04622L, by the Slovene Research Agency under Research Project J2-4458 and Research Programme P2-0041 and by the Ministry of Education, Youth and Sports of the Czech Republic, project SGS-2022-015.

References

- Abderrazak, K., Benabid, Y. (2023). Realistic Modeling of Shoulder Muscle for use in Musculoskeletal Model. In: ? (Ed.), *Ann Biomed Eng*, pp. 1079–1093. <https://doi.org/10.1007/s10439-023-03189-y>.
- Afiatdoust, F., Esmailbeigi, M. (2015). Optimal variable shape parameters using genetic algorithm for radial basis function approximation. *Ain Shams Engineering Journal*, 6(2), 639–647. <https://doi.org/10.1016/j.asej.2014.10.019>.
- Angles, B., Rebain, D., Macklin, M., Wyvill, B., Barthe, L., Lewis, J., Von Der Pahlen, J., Izadi, S., Valentin, J., Bouaziz, S., Tagliasacchi, A. (2019). VIPER: Volume Invariant Position-Based Elastic Rods. *Proc. ACM Comput. Graph. Interact. Tech.*, 2(2). <https://doi.org/10.1145/3340260>. <https://doi.org/10.1145/3340260>.
- Cani-Gascuel, M., Desbrun, M. (1997). Animation of deformable models using implicit surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 3(1), 39–50. <https://doi.org/10.1109/2945.582343>.
- Carr, J.C., Beatson, R.K., Cherrie, J.B., Mitchell, T.J., Fright, W.R., McCallum, B.C., Evans, T.R. (2001). Reconstruction and representation of 3D objects with radial basis functions. In: *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '01. Association for Computing Machinery, New York, NY, USA, pp. 67–76. 158113374X. <https://doi.org/10.1145/383259.383266>. <https://doi.org/10.1145/383259.383266>.
- Cervenka, M., Smolik, M., Skala, V. (2019). A New Strategy for Scattered Data Approximation Using Radial Basis Functions Respecting Points of Inflection. In: Misra, S., Gervasi, O., Murgante, B., Stankova, E., Korkhov, V., Torre, C., Rocha, A.M.A.C., Taniar, D., Apduhan, B.O., Tarantino, E. (Eds.), *Computational Science and Its Applications – ICCSA 2019*. Springer International Publishing, Cham, pp. 322–336. 978-3-030-24289-3.
- Cervenka, M., Havlicek, O., Kohout, J., Váša, L. (2023). Computerised Muscle Modelling and Simulation for Interactive Applications. In: ? (Ed.), *Proceedings of the 18th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2023) - GRAPP*. SciTePress, pp. 214–221. INSTICC. 978-989-758-634-7. <https://doi.org/10.5220/0011688000003417>.
- Cieza, A., Causey, K., Kamenov, K., Hanson, S., Chatterji, S., Vos, T. (2020). Global estimates of the need for rehabilitation based on the Global Burden of Disease study 2019: a systematic analysis for the Global Burden of Disease Study 2019. *The Lancet*, 396. [https://doi.org/10.1016/S0140-6736\(20\)32340-0](https://doi.org/10.1016/S0140-6736(20)32340-0).
- Clapés, M., González Hidalgo, M., Torres, A., Palmer-Rodríguez, P. (2008). Interactive Constrained Deformations of NURBS Surfaces: N-SCODEF. In: ? (Ed.), *Articulated Motion and Deformable Objects*, pp. 359–369. 978-3-540-70516-1. https://doi.org/10.1007/978-3-540-70517-8_35.
- Delp, S., Blemker, S. (2005). Three-Dimensional Representation of Complex Muscle Architectures and Geometries. *Annals of biomedical engineering*, 33, 661–73. <https://doi.org/10.1007/s10439-005-1433-7>.
- Dvořák, J., Káčereková, Z., Vanecek, P., Hruda, L., Váša, L. (2021). As-rigid-as-possible volume tracking for time-varying surfaces. *Computers & Graphics*, 102. <https://doi.org/10.1016/j.cag.2021.10.015>.
- Fasser, M.-., Jokeit, M., Kalthoff, M., Gomez Romero, D.A., Trache, T., Snedeker, J.G., Farshad, M., Widmer, J. (2021). Subject-Specific Alignment and Mass Distribution in Musculoskeletal Models of the Lumbar Spine. *Frontiers in Bioengineering and Biotechnology*, 9. Cited By :2.

- Goyanes, E., de Moura, J., Fernández-Vigo, J.I., Fernández-Vigo, J.A., Novo, J., Ortega, M. (2024). Automatic simultaneous ciliary muscle segmentation and biomarker extraction in AS-OCT images using deep learning-based approaches. *Biomedical Signal Processing and Control*, 90, 105851. <https://doi.org/10.1016/j.bspc.2023.105851>. <https://www.sciencedirect.com/science/article/pii/S1746809423012843>.
- Hardy, R.L. (1990). Theory and applications of the multiquadric-biharmonic method 20 years of discovery 1968–1988. *Computers & Mathematics with Applications*, 19(8), 163–208. [https://doi.org/10.1016/0898-1221\(90\)90272-L](https://doi.org/10.1016/0898-1221(90)90272-L).
- Hardy, R.L. (1971). Multiquadric equations of topography and other irregular surfaces. *Journal of Geophysical Research*, 76(8), 1905–1915. <https://doi.org/10.1029/JB076i008p01905>.
- Heinrich, D., Bogert, A., Mössner, M., Nachbauer, W. (2023). Model-based estimation of muscle and ACL forces during turning maneuvers in alpine skiing. *Scientific Reports*. <https://doi.org/10.1038/s41598-023-35775-4>.
- Hill, A. (1938). The heat of shortening and the dynamic constants of muscle. *Proc. R. Soc. Lond. B*, 126, 612–745.
- Hilton, A., Stoddart, A.J., Illingworth, J., Windeatt, T. (1996). Marching triangles: range image fusion for complex object modelling. In: ? (Ed.), *Proceedings of 3rd IEEE International Conference on Image Processing* (Vol. 2), pp. 381–3842. <https://doi.org/10.1109/ICIP.1996.560840>.
- Hájková, J., Kohout, J. (2014). Human Body Model Movement Support: Automatic Muscle Control Curves Computation. In: ? (Ed.), *Combinatorial Image Analysis*, pp. 196–211. 978-3-319-07147-3. https://doi.org/10.1007/978-3-319-07148-0_18.
- Janák, T., Kohout, J. (2014). Deformable Muscle Models for Motion Simulation. In: ? (Ed.), *Proceedings of the 9th International Conference on Computer Graphics Theory and Applications - GRAPP, (VISIGRAPP 2014)*. SciTePress, pp. 301–311. INSTICC. 978-989-758-002-4. <https://doi.org/10.5220/0004678903010311>.
- Joseph, V.R., Hung, Y., Sudjianto, A. (2008). Blind Kriging: A New Method for Developing Metamodels. *Journal of Mechanical Design*, 130(3), 031102. <https://doi.org/10.1115/1.2829873>.
- Kaymaz, I. (2005). Application of kriging method to structural reliability problems. *Structural Safety*, 27(2), 133–151. <https://doi.org/10.1016/j.strusafe.2004.09.001>.
- Kedadria, A., Benabid, Y., Remil, O., Benaouali, A., May, A., Ramtani, S. (2023). A Shoulder Musculoskeletal Model with Three-Dimensional Complex Muscle Geometries. *Annals of Biomedical Engineering*, 51, 1079–1093. <https://doi.org/10.1007/s10439-023-03189-y>.
- Kellnhofer, P., Kohout, J. (2012). Time-convenient Deformation of Musculoskeletal System. In: ? (Ed.), *ALGORITHMY 2012*, pp. 1–10.
- Kohout, J., Cervenka, M. (2022). Non-planar Surface Shape Reconstruction from a Point Cloud in the Context of Muscles Attachments Estimation. In: ? (Ed.), *Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2022) - GRAPP*. SciTePress, pp. 236–243. INSTICC. 978-989-758-555-5. <https://doi.org/10.5220/0010869600003124>.
- Lavoie, P., Ionescu, D., Petriu, E. (2006). Constructing 3D virtual reality objects from 2D images of real objects using NURBS. In: ? (Ed.), *2007 IEEE SYMPOSIUM ON VIRTUAL ENVIRONMENTS, HUMAN-COMPUTER INTERFACES AND MEASUREMENT SYSTEMS*. IEEE, 345 E 47TH ST, NEW YORK, NY 10017 USA, p. 117. IEEE. IEEE Symposium on Virtual Environments, Human-Computer Interfaces and Measurement Systems, Ostuni, ITALY, JUN 25-27, 2007. 978-1-4244-0819-1.
- Lee, D., Glueck, M., Khan, A., Fiume, E., Jackson, K. (2012). Modeling and Simulation of Skeletal Muscle for Computer Graphics: A Survey. *Foundations and Trends® in Computer Graphics and Vision*, 7, 229. <https://doi.org/10.1561/06000000036>.
- Lorensen, W.E., Cline, H.E. (1987). Marching cubes: A high resolution 3D surface construction algorithm. In: ? (Ed.) *NEED BOOKTITLE*. SIGGRAPH '87. Association for Computing Machinery, New York, NY, USA, pp. 163–169. 0897912276. <https://doi.org/10.1145/37401.37422>. <https://doi.org/10.1145/37401.37422>.
- Macklin, M., Müller, M., Chentanez, N. (2016). XPBD: Position-Based Simulation of Compliant Constrained Dynamics. In: ? (Ed.), *Proceedings of the 9th International Conference on Motion in Games*. MIG '16. Association for Computing Machinery, New York, NY, USA, pp. 49–54. 9781450345927. <https://doi.org/10.1145/2994258.2994272>. <https://doi.org/10.1145/2994258.2994272>.

- Majdisova, Z., Skala, V. (2017a). Radial basis function approximations: comparison and applications. *Applied Mathematical Modelling*, 51, 728–743. <https://doi.org/10.1016/j.apm.2017.07.033>.
- Majdisova, Z., Skala, V. (2017b). Radial basis function approximations: comparison and applications. *Applied Mathematical Modelling*, 51, 728–743. <https://doi.org/10.1016/j.apm.2017.07.033>.
- Modenese, L., Renault, J.-B. (2021). Automatic Generation of Personalized Skeletal Models of the Lower Limb from Three-Dimensional Bone Geometries. *Journal of Biomechanics*, 116, 110186. <https://doi.org/10.1016/j.jbiomech.2020.110186>.
- Müller, M., Heidelberger, B., Hennix, M., Ratcliff, J. (2007). Position based dynamics. *Journal of Visual Communication and Image Representation*, 18(2), 109–118. <https://doi.org/10.1016/j.jvcir.2007.01.005>.
- Na, N., Kunjin, H., Lin, W., Junfeng, J., Zhengming, C. (2023). Modeling of human muscle and its deformation. *Computer Methods in Biomechanics and Biomedical Engineering*, 0(0), 1–13. PMID: 36880856. <https://doi.org/10.1080/10255842.2023.2186160>. <https://doi.org/10.1080/10255842.2023.2186160>.
- Nie, M., Wan, Y., Zhou, A. (2022). Real-Time NURBS Interpolation under Multiple Constraints. *Computational Intelligence and Neuroscience*, 2022, 1–15. <https://doi.org/10.1155/2022/7492762>.
- Oliver, M.A., Webster, R. (1990). Kriging: a method of interpolation for geographical information systems. *International journal of geographical information systems*, 4(3), 313–332. <https://doi.org/10.1080/02693799008941549>.
- Orr, M.J.L. (1995). Regularization in the Selection of Radial Basis Function Centers. *Neural Computation*, 7(3), 606–623. <https://doi.org/10.1162/neco.1995.7.3.606>. <https://doi.org/10.1162/neco.1995.7.3.606>.
- Romeo, M., Montegudo, C., Sánchez-Quirós, D. (2018). Muscle Simulation with Extended Position Based Dynamics. In: García-Fernández, I., Ureña, C. (Eds.), *Spanish Computer Graphics Conference (CEIG)*. The Eurographics Association, pp. 134–146. 978-3-03868-067-3. <https://doi.org/10.2312/ceig.20181146>.
- Sakata, S., Ashida, F., Zako, M. (2004). An efficient algorithm for Kriging approximation and optimization with large-scale sampling data. *Computer Methods in Applied Mechanics and Engineering*, 193(3), 385–404. <https://doi.org/10.1016/j.cma.2003.10.006>.
- Sarra, S.A., Sturgill, D. (2009). A random variable shape parameter strategy for radial basis function approximation methods. *Engineering Analysis with Boundary Elements*, 33(11), 1239–1245. <https://doi.org/10.1016/j.enganabound.2009.07.003>.
- Skala, V. (2017). Least Square Method Robustness of Computations: What is not usually considered and taught. In: ? (Ed.), *2017 Federated Conference on Computer Science and Information Systems*, pp. 537–541. <https://doi.org/10.15439/2017F7>.
- Skala, V., Cervenka, M. (2019). Novel RBF Approximation Method Based on Geometrical Properties for Signal Processing with a New RBF Function: Experimental Comparison. *Informatics 2019, IEEE proceedings*, 357–362. UT WoS: 000610452900074, EID: 2-s2.0-85087090327, OBD: 43929007. 978-1-7281-3178-8. <https://doi.org/10.1109/Informatics47936.2019.9119276>.
- Skala, V., Karim, S.A.A., Zabran, M. (2020). Radial Basis Function Approximation Optimal Shape Parameters Estimation. In: ? (Ed.), *Computational Science – ICCS 2020*. Springer International Publishing, Cham, pp. 309–317. 978-3-030-50433-5.
- Sorkine, O., Alexa, M. (2007). As-Rigid-As-Possible Surface Modeling. In: Belyaev, A., Garland, M. (Eds.), *Geometry Processing*. The Eurographics Association, pp. 109–116. 978-3-905673-46-3. <https://doi.org/10.2312/SGP/SGP07/109-116>.
- Sánchez-Reyes, J., Chacón, J. (2020). How to make impossible objects possible: Anamorphic deformation of textured NURBS. *Computer Aided Geometric Design*, 78, 101826. <https://doi.org/10.1016/j.cagd.2020.101826>.
- Terzopoulos, D., Qin, H. (1994). Dynamic NURBS with Geometric Constraints for Interactive Sculpting. *ACM Trans. Graph.*, 13(2), 103–136. <https://doi.org/10.1145/176579.176580>. <https://doi.org/10.1145/176579.176580>.
- Valente, G., Martelli, S., Taddei, F., Farinella, G., Viceconti, M. (2012). Muscle discretization affects the loading transferred to bones in lower-limb musculoskeletal models. *Proc. Inst. Mech. Eng. H J. Eng. Med.*, 226(2), 161–169.
- Wang, B., Matcuk, G., Barbič, J. (2021). Modeling of Personalized Anatomy Using Plastic Strains. *ACM Trans.*

- Graph.*, 40(2). <https://doi.org/10.1145/3443703>. <https://doi.org/10.1145/3443703>.
- Wright, G.B. (2003). *Radial Basis Function Interpolation: Numerical and Analytical Developments*. PhD thesis, University of Colorado at Boulder, Boulder, CO, USA. AAI3087597.
- Ye, D., Jiang, X., Huo, G., Su, C., Lu, Z., Wang, B., Zheng, Z. (2020). A Physical Process Driven Digital Terrain Model Generating Method Based on D-NURBS. *IEEE Access*, 8, 3115–3122. Cited by: 0; All Open Access, Gold Open Access. <https://doi.org/10.1109/ACCESS.2019.2962385>.
- Zhang, M., Qin, H. (2001). Hierarchical D-NURBS surfaces and their physics-based sculpting. In: ? (Ed.), *Proceedings International Conference on Shape Modeling and Applications*, pp. 257–266. <https://doi.org/10.1109/SMA.2001.923397>.
- Zhang, Z.-Y., Sun, Z.-J., Yang, Y.-H., Lin, H. (2022). Towards a better prediction of subcellular location of long non-coding RNA. *Frontiers of Computer Science*, 16(5), 165903. <https://doi.org/10.1007/s11704-021-1015-3>. <https://doi.org/10.1007/s11704-021-1015-3>.
- Zhao, W., San, Y. (2011). RBF neural network based on q-Gaussian function in function approximation. *Frontiers of Computer Science in China*, 5(4), 381–386. <https://doi.org/10.1007/s11704-011-1041-7>. <https://doi.org/10.1007/s11704-011-1041-7>.

A. Curvature visualisations

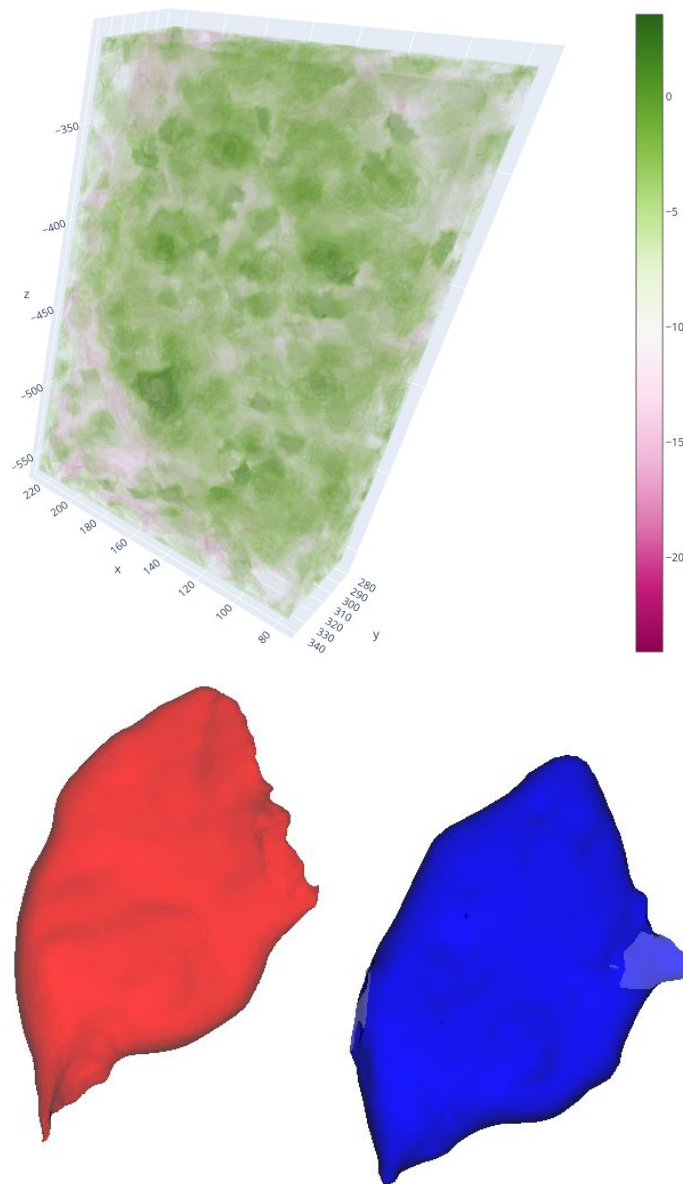


Fig. 11. The curvature of the RBF surface of the gluteus maximus muscle is depicted in the image. The prominent centres of the RBFs are represented by green spheres in the space. The curvature values are displayed in a logarithmic scale, as the differences in curvature are not linear but rather exponential. This result stems from the same experiment shown in Figure 1; it has been included here for easy comparison.

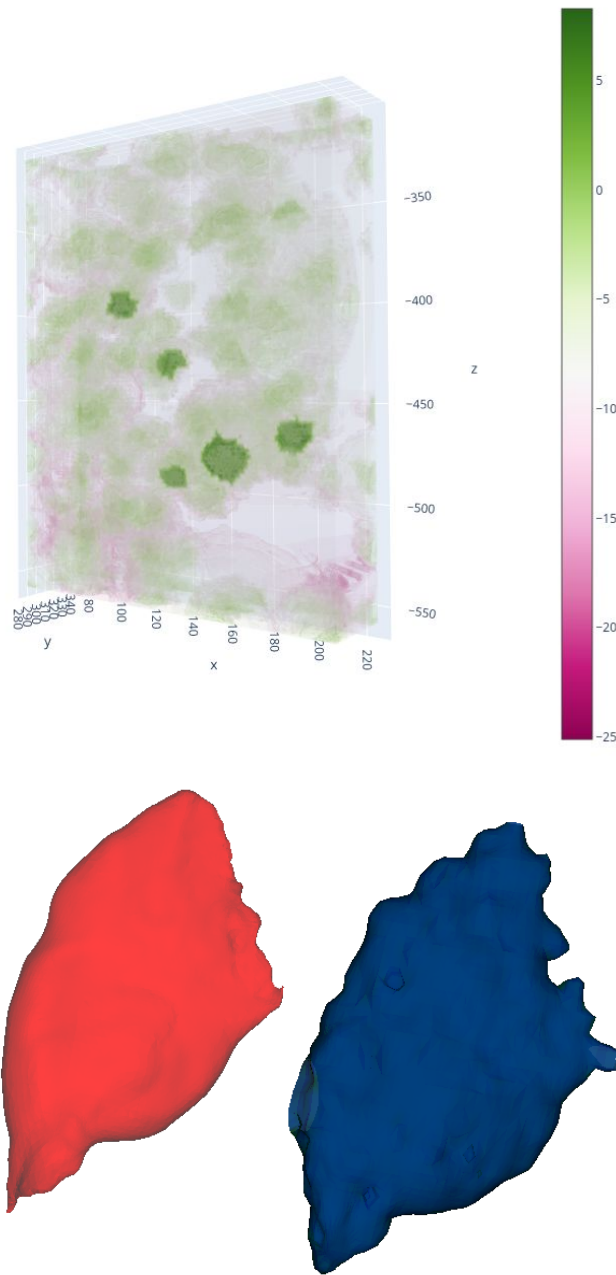


Fig. 12. On the left, we have the curvature field, while on the right, we see the approximation result with a regularisation factor of 0.7. Although there is a preference for curvature fluctuation within the muscle, there is also substantial fluctuation outside. Decreasing the local minima outside the muscle volume results in a less precise approximation of the original red muscle to the blue one.

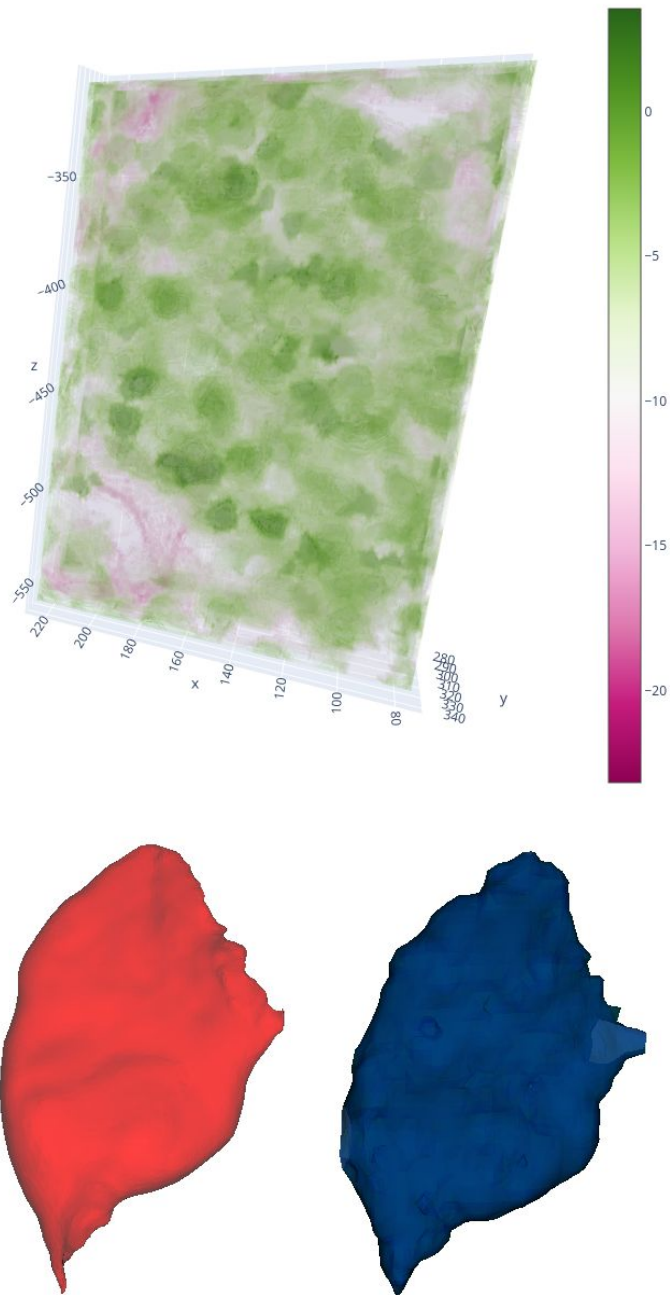


Fig. 13. On the left, we have the curvature field, while on the right, we see the approximation result with a regularisation factor of 0.3. The fluctuation in the curvature field has been further reduced, but as a consequence, the resulting geometrical surface model has become more rough in texture.

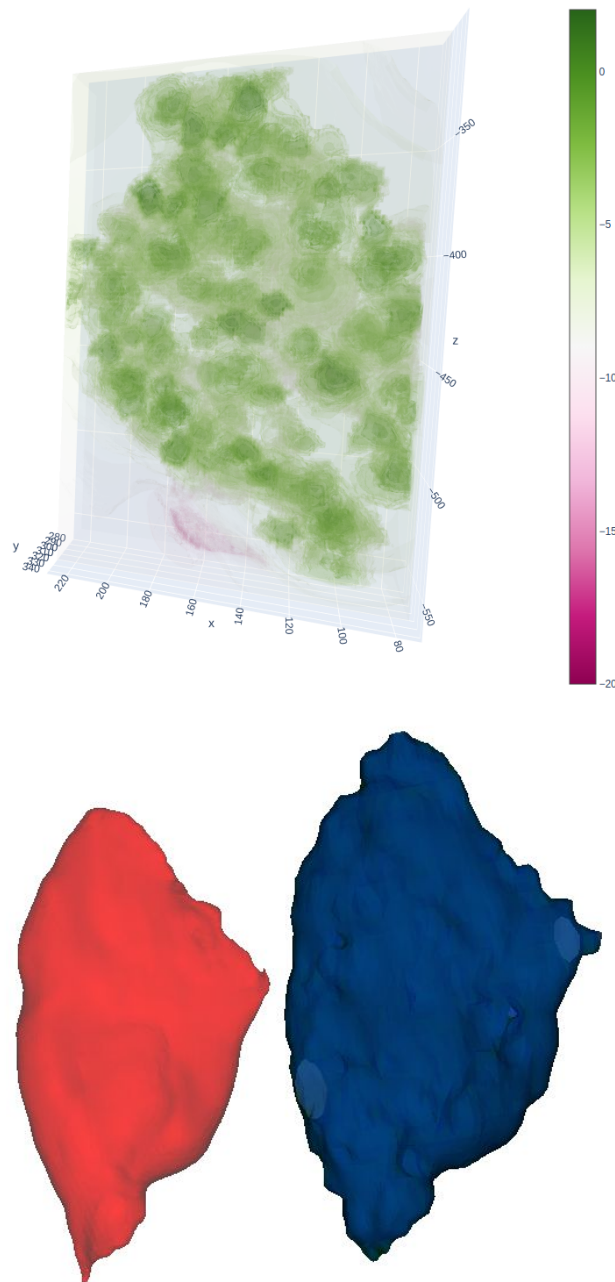


Fig. 14. The curvature field (on the left) and approximation result (on the right) with the regularisation factor of 0.05. While the fluctuation in the curvature field has been nearly eliminated, the roughness of the muscle surface has become quite pronounced.

B. Dynamic restoration visualisations

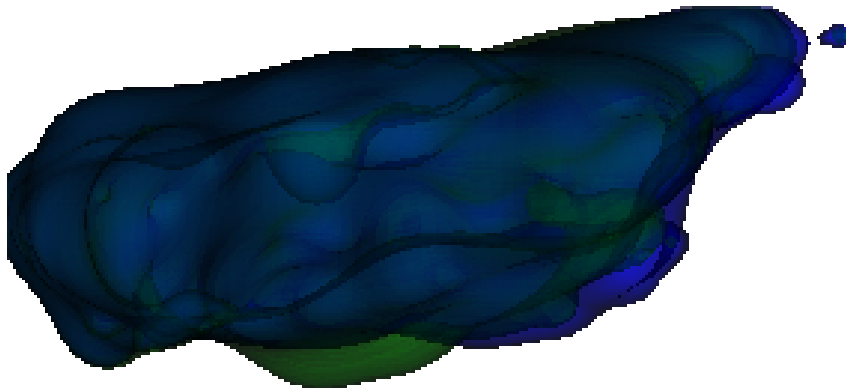


Fig. 15. The initialisation of the simulation without the regularisation factor, where the randomly deformed green muscle should return to its initial shape in blue.

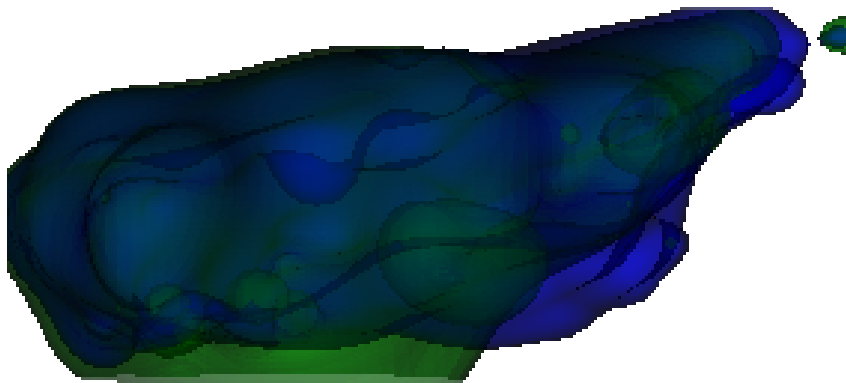


Fig. 16. The progress of the shape restoration. The green surface seems to diverge from the blue one due to omitting the regularisation factor.

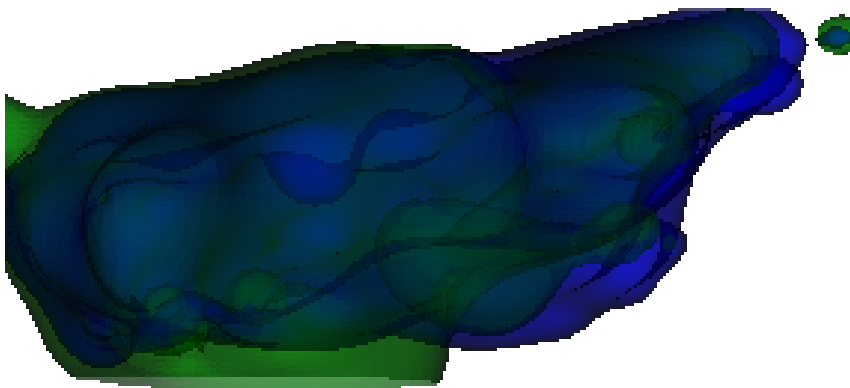


Fig. 17. The final part of the shape restoration. The green shape found other local minima than the muscle's blue (original) shape.

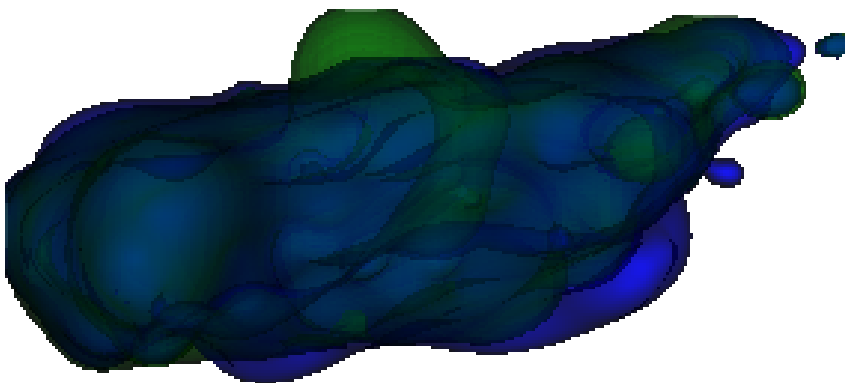


Fig. 18. The initialisation of the simulation with the regularisation factor, where the randomly deformed green muscle should return to its initial shape in blue.

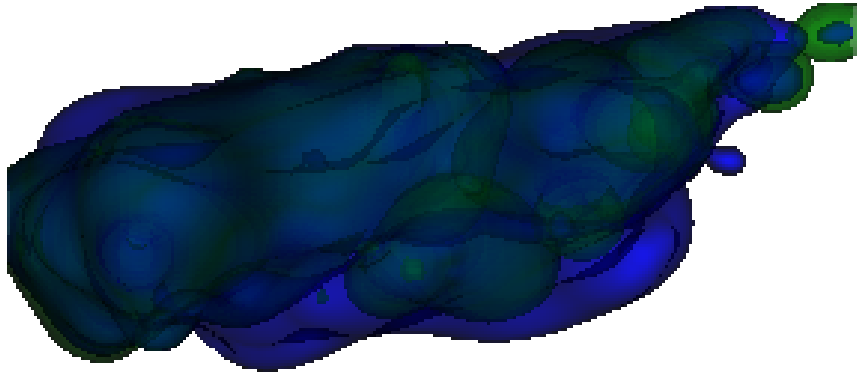


Fig. 19. The green shape seems to converge to the blue shape. The main difference is the restored top part of the green shape.

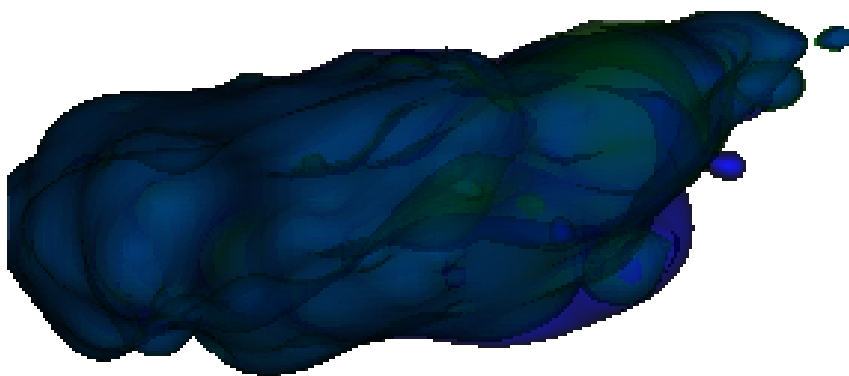


Fig. 20. The final restoration. Excluding the minor differences in the right part of the muscle, the green shape quite accurately approximates the original blue one.

C. Mathematical dynamic model

Let us assume the function f to be a 3D radial basis function approximation:

$$f(\mathbf{x}) = \sum_{i=1}^N \lambda_i \phi(\|\mathbf{x} - \boldsymbol{\xi}_i\|) \quad (12)$$

Let's assume that Gaussian Radial Basis Functions (RBFs) are utilised exclusively. Additionally, we can define $g_i(\mathbf{x})$ as a replacement for the i th RBF to streamline subsequent derivations.

$$f(\mathbf{x}) = \sum_{i=1}^N \lambda_i e^{-\alpha \|\mathbf{x} - \boldsymbol{\xi}_i\|_2} = \sum_{i=1}^N g_i(\mathbf{x}) \quad (13)$$

The shape of the geometrical model can be well described using its curvature. To compute the curvature, we need to determine the Hessian matrix, consisting of the second partial derivatives with respect to the function f . Therefore, the initial step involves finding the gradient of the function f in an n -dimensional space:

$$\nabla f(\mathbf{x}) = \left[\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \frac{\partial f}{\partial x_3}, \dots \right] \quad (14)$$

By employing the chain rule, where the expression $(x_j - \xi_{ij})$ originates from the exponent of the exponential function f , along with the shape parameter α and the constant value of -2 , the resultant gradient takes the following form:

$$\nabla f(\mathbf{x}) = -2 \sum_{i=1}^N \alpha_i g_i(\mathbf{x}) [x_1 - \xi_{i1}, x_2 - \xi_{i2}, x_3 - \xi_{i3}, \dots] \quad (15)$$

We will now construct the Hessian matrix, which will be used to calculate the curvature. As previously mentioned, its computation relies on knowing the second partial derivatives. In an n D space, it is defined as follows:

$$\mathbf{H}(f(\mathbf{x})) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \frac{\partial^2 f}{\partial x_1 \partial x_3} & \dots \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \frac{\partial^2 f}{\partial x_2 \partial x_3} & \dots \\ \frac{\partial^2 f}{\partial x_3 \partial x_1} & \frac{\partial^2 f}{\partial x_3 \partial x_2} & \frac{\partial^2 f}{\partial x_3^2} & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \quad (16)$$

To compute the second partial derivatives, we rely on the gradient of a function f . For instance, the second partial derivatives with respect to x^2 can be expressed as follows:

$$\frac{\partial^2 f}{\partial x_j^2} = \frac{\partial f}{\partial x_j} \left(-2 \sum_{i=1}^N \alpha_i g_i(\mathbf{x}) (x_j - \xi_{ij}) \right) \quad (17)$$

Applying the product and chain rules, the second partial derivative with respect to x^2 takes the form of the following expression. The derivation is performed similarly to the first derivative, which has already been carried out above in (15).

$$\frac{\partial^2 f}{\partial x_j^2} = 2 \sum_{i=1}^N \alpha_i \left[2\alpha_i (x_j - \xi_{ij})^2 - 1 \right] g_i(\mathbf{x}) \quad (18)$$

The calculation of the mixed second partial derivatives is performed similarly to before. Upon examining the outcome in 19, it becomes apparent that there is a resemblance between the mixed partial derivatives and those involving the same variables. The mixed ones are not scaled by $2\alpha_i$ and do not undergo a subtraction of 1 from them; otherwise, the remaining operations are identical for both cases.

$$\frac{\partial^2 f}{\partial x_j \partial x_k} = 2 \sum_{i=1}^N 2\alpha_i^2 (x_j - \xi_{ij}) (x_k - \xi_{ik}) g_i(\mathbf{x}) \quad (19)$$

The Hessian matrix can be populated with the second partial derivatives. Both sets of second partial derivatives involve multiplication by the distances from the centre point in their respective directions. This process can be encapsulated using an outer product. The value of -1 from the pure partial derivatives will transform into the identity matrix with dimensions $d \times d$, where d denotes the problem's dimensionality (which is 3 in our case).

$$\mathbf{H}(f(\mathbf{x})) = 2 \sum_{i=1}^N \alpha_i g_i(\mathbf{x}) \left(2\alpha_i (\mathbf{x} - \boldsymbol{\xi}_i) (\mathbf{x} - \boldsymbol{\xi}_i)^T - \mathbf{I}_d \right) \quad (20)$$

C.1. Mean curvature

For curvature calculation, one option is to employ the mean curvature, which is defined as the average of all the eigenvalues λ_i of a Hessian matrix:

$$\kappa_\mu(\mathbf{H}) = \frac{1}{d} \sum_{i=1}^d \lambda_i \quad (21)$$

There are zero eigenvalues with the multiplicity of $d - 1$ and one with a value of $\|\mathbf{x} - \boldsymbol{\xi}_i\|^2$. The result aligns with the intuition, as zero eigenvalues correspond to eigenvectors perpendicular to the hypersphere isosurface at each point. In contrast, the sole non-zero eigenvalue pertains to the eigenvector tangent to them, pointing from the RBF centre outward. When calculated over the outer product of a vector with itself, the coefficient 2α and identity matrix \mathbf{I}_d transform the eigenvalues to -1 , with a multiplicity of $d - 1$, and the final eigenvalue becomes $2\alpha\|\mathbf{x} - \boldsymbol{\xi}_i\|^2 - 1$.

$$\kappa_{\mu}(\mathbf{H}(f(\mathbf{x}))) = \kappa_{\mu f} = \frac{2}{d} \sum_{i=1}^N \alpha_i g_i(\mathbf{x}) \left(2\alpha_i \|\mathbf{x} - \boldsymbol{\xi}_i\|_2^2 - d \right) \quad (22)$$

The subsequent step involves specifying the cost function, which is essentially the squared L2 norm between the new and original curvatures across the subspace $\Omega \subseteq \mathbb{R}^d$:

$$C_f = \int \dots \int \|\kappa_{\mu f} - \kappa_{\mu f_i}\|_2^2 dx_1 \dots dx_d = \int_{\Omega} \|\kappa_{\mu f} - \kappa_{\mu f_i}\|_2^2 d\mathbf{x} \quad (23)$$

One may employ the gradient descent method to discover the optimal values of $\boldsymbol{\xi}_i$. Initially, we must compute the gradient of the curvature with respect to $\boldsymbol{\xi}_i$:

$$\nabla \kappa_{\mu f} = \left[\frac{\partial \kappa_{\mu f}}{\partial \xi_{i1}} \quad \frac{\partial \kappa_{\mu f}}{\partial \xi_{i2}} \quad \frac{\partial \kappa_{\mu f}}{\partial \xi_{i3}} \quad \dots \right] \quad (24)$$

The resulting gradient takes the following form:

$$\frac{\partial \kappa_{\mu f}}{\partial \xi_{kj}} = \frac{4}{d} \alpha_k^2 g_k(\mathbf{x}) (x_j - \xi_{kj}) \left(2\alpha_k \|\mathbf{x} - \boldsymbol{\xi}_k\|_2^2 - 2 - d \right) \quad (25)$$

Now, we need to compute the gradient of the cost function C , which is expressed as follows:

$$\nabla C_f = \nabla \left(\int_{\mathbb{R}^d} \|\kappa_{\mu f} - \kappa_{\mu f_i}\|_2^2 d\mathbf{x} \right) \quad (26)$$

Given the definition of the partial derivatives of κ , the complete cost function can be expressed as:

$$\nabla C_{fkj} = \frac{8}{d} \int_{\mathbb{R}^d} (\kappa_{\mu f} - \kappa_{\mu f_i}) \alpha_k^2 g_k(\mathbf{x}) (x_j - \xi_{kj}) \left(2\alpha_k \|\mathbf{x} - \boldsymbol{\xi}_k\|_2^2 - 2 - d \right) d\mathbf{x} \quad (27)$$

Martin Červenka is a Computer Science doctoral student at West Bohemia University, Pilsen, Czech Republic. He earned his Master's in Computer Science from West Bohemia University, Pilsen, Czech Republic, in 2019. During this period, his research focused on muscle modelling approaches, radial basis function interpolation, and approximation techniques. He is affiliated with West Bohemia University and continues to be deeply engaged in applying radial basis functions in muscle modelling.

Josef Kohout is an Associate Professor in Computer Science at West Bohemia University, Pilsen, Czech Republic. His expertise lies in the field of computer graphics and medical informatics. He possesses a skill set encompassing mesh processing, data visualisation, scientific visualisation, medical image processing, simulation, and more. Currently, his research is focused on diverse muscle modelling methodologies.

Bogdan Lipuš is an assistant professor at the Faculty of Electrical Engineering and Computer Sciences, University of Maribor. Currently, he is a member of the Laboratory for Geospatial Modelling, Multimedia and Artificial Intelligence. His research interests include lidar data processing, remote sensing, data compression, computer graphics, computer-aided geometric design, and image reconstruction.

Conclusion, Summary & Future work

10

This dissertation provides a comprehensive overview of the evolution and current methodologies in muscle modelling. It traces the journey from the earliest Hill-type muscle models, which were rudimentary yet foundational, through various stages of evolution, including simplistic straight-line approximations, polyline constructs, and models incorporating lines wrapped around obstacles or bones. This progression culminates in more sophisticated, higher-dimensional frameworks such as mass-spring systems, position-based dynamics, and finite element methods. A recurring theme in this evolution is the balance between model accuracy and computational efficiency. A notable trend is observed: models demand more parameters to be accurately determined and configured as they become more complex.

A pivotal contribution of this dissertation is the detailed exploration of existing muscle modelling techniques and the introduction of a novel mathematical model utilizing radial basis function implicit surfaces. This innovative approach marks a significant step forward, offering a memory-efficient model to generate infinitely smooth surface representations. Such a model can potentially revolutionize the field of muscle modelling by providing a more refined and scalable tool for simulating muscle morphology and dynamics.

However, the proposed model is not without limitations. It does not address certain practical aspects crucial in real-world applications, such as collision handling and preserving volume in the modelled entities. These areas are identified as avenues for future research and development. The insights and methodologies presented in this dissertation lay a solid groundwork for tackling these challenges. Building upon the foundation laid by this research, future work is anticipated to advance the field further, enhancing the realism and applicability of muscle models in various scientific and medical applications.

This dissertation, covering a broad range of topics in muscle modelling, opens up several avenues for future research and development. One of the critical areas for

expansion is the proposed Radial Basis Function (RBF) mathematical model. Currently, the model primarily focuses on maintaining the initial shape of the muscle during deformation, akin to the Position-Based Dynamics (PBD) approach. However, it does not yet incorporate other crucial aspects of muscle behaviour. Notably, the interaction between muscles modelled with RBF and bones, especially regarding collision handling, remains unaddressed. Drawing inspiration from existing literature, such as the work by Cani [99] on the collision handling between two implicit surfaces, could be beneficial. Although Cani's method is based on implicit surfaces generated by skeletons rather than RBF sets, the underlying principles could provide valuable insights for developing a similar mechanism for RBF muscles. Additionally, formulating a volume constraint for the RBF model is another crucial aspect that requires attention, ensuring that the muscle volume remains consistent during deformations.

The dissertation also delves into the complexities of the PBD approach. Despite significant advancements, challenges like muscle penetration through bones and muscle tissue getting forced into tight spaces persist. Addressing these issues involves solving intricate problems like determining the side of the bone where most muscle is located post-deformation and devising strategies for muscle tissue to escape from increasingly tight spaces. These challenges highlight the need for more sophisticated algorithms and problem-solving techniques in muscle modelling.

Furthermore, integrating the As-Rigid-As-Possible (ARAP) approach with the existing PBD framework has proven more challenging than initially anticipated. The attempt to intertwine these two methodologies resulted in a rough surface texture, as the algorithms worked at cross purposes, pushing vertices in different directions. This outcome suggests that a more intricate and harmonized cooperation between ARAP and PBD algorithms is essential to achieve the desired smoothness and accuracy in muscle modelling.

In summary, while this dissertation lays a strong foundation in muscle modelling, it also clearly outlines the need for further research in several key areas. These include developing collision handling mechanisms for RBF muscles, volume preservation techniques, resolving issues related to muscle-bone interactions in the PBD approach, and refining the integration of ARAP with PBD for smoother and more accurate muscle deformations. These challenges present exciting opportunities for future research, promising significant advancements in muscle modelling.

Bibliography

1. HAJKOVA, Jana; KOHOUT, Josef. Human Body Model Movement Support: Automatic Muscle Control Curves Computation. In: 2014, pp. 196–211. ISBN 978-3-319-07147-3. Available from DOI: 10.1007/978-3-319-07148-0_18.
2. JANAK, Tomas. *Fast soft-body models for musculoskeletal modelling*. 2012. Tech. rep. University of West Bohemia, Faculty of Applied Sciences.
3. CERVENKA, M.; SMOLIK, M.; SKALA, V. A New Strategy for Scattered Data Approximation Using Radial Basis Functions Representing Points of Inflection. *Computational Science and Its Application, ICSSA 2019 proceedings, Part I, LNCS 11619*. 2019, pp. 322–226. ISBN 978-3-030-24288-6. ISSN 0302-9743. Available from DOI: https://doi.org/10.1007/978-3-030-24289-3_24. UT WoS: 000661318700024, EID: 2-s2.0-85069157052, OBD: 43926678.
4. CARR, J. C. et al. Reconstruction and representation of 3D objects with radial basis functions. In: *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*. New York, NY, USA: Association for Computing Machinery, 2001, pp. 67–76. SIGGRAPH '01. ISBN 158113374X. Available from DOI: 10.1145/383259.383266.
5. SHARAWARDI, Nur; CHOO, Yun-Huoy; CHONG, Shin-Horng; MOHAMAD, Nur Ikhwan. Isotonic Muscle Fatigue Prediction for Sport Training Using Artificial Neural Network Modelling. In: 2018, pp. 582–591. ISBN 978-3-319-60617-0. Available from DOI: 10.1007/978-3-319-60618-7_57.
6. CURRELI, Cristina; DI PUCCIO, Francesca; DAVICO, Giorgio; MODENESE, Luca; VICECONTI, Marco. Using Musculoskeletal Models to Estimate in vivo Total Knee Replacement Kinematics and Loads: Effect of Differences Between Models. *Frontiers in Bioengineering and Biotechnology*. 2021, vol. 9. Available from DOI: 10.3389/fbioe.2021.703508.
7. KAINZ, Hans et al. Reliability of functional and predictive methods to estimate the hip joint centre in human motion analysis in healthy adults. *Gait & Posture*. 2017, vol. 53. Available from DOI: 10.1016/j.gaitpost.2017.01.023.

8. SORKINE, Olga; ALEXA, Marc. As-Rigid-As-Possible Surface Modeling. In: BELYAEV, Alexander; GARLAND, Michael (eds.). *Geometry Processing*. The Eurographics Association, 2007. ISBN 978-3-905673-46-3. ISSN 1727-8384. Available from DOI: 10.2312/SGP/SGP07/109-116.
9. OTAKE, Yoshito et al. Patient-Specific Skeletal Muscle Fiber Modeling from Structure Tensor Field of Clinical CT Images. In: DESCOTEAUX, Maxime et al. (eds.). *Medical Image Computing and Computer Assisted Intervention – MICCAI 2017*. Cham: Springer International Publishing, 2017, pp. 656–663. ISBN 978-3-319-66182-7.
10. KAPTEIN, B.L.; VAN DER HELM, F.C.T. Estimating muscle attachment contours by transforming geometrical bone models. *Journal of Biomechanics*. 2004, vol. 37, no. 3, pp. 263–273. ISSN 0021-9290. Available from DOI: <https://doi.org/10.1016/j.jbiomech.2003.08.005>.
11. PELLIKAAN, P. et al. Evaluation of a morphing based method to estimate muscle attachment sites of the lower extremity. *Journal of Biomechanics*. 2013, vol. 47. Available from DOI: 10.1016/j.jbiomech.2013.12.010.
12. CERVENKA, Martin. *Muscle Fibres Deformation using Particle System*. 2019. MA thesis. University of West Bohemia. Supervisor: Kohout, J.
13. KOHOUT, J.; CERVENKA, M. Nonplanar Surface Shape Reconstruction from a Point Cloud in the Context of Muscles Attachments Estimation. *Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*. 2022, pp. 236–243. ISBN 978-989-758-555-5. Available from DOI: <https://doi.org/10.5220/0010869600003124>. UT WoS: 000774795400024, OBD: 43936004.
14. MEDICINE, National Library of. *Visible Human Project*. [N.d.]. Available also from: https://www.nlm.nih.gov/research/visible/visible_human.html.
15. LEE, Dongwoon et al. A three-dimensional approach to pennation angle estimation for human skeletal muscle. *Computer methods in biomechanics and biomedical engineering*. 2014, vol. 18, pp. 1–11. Available from DOI: 10.1080/10255842.2014.917294.
16. FUKUDA, Norio et al. Estimation of attachment regions of hip muscles in CT image using muscle attachment probabilistic atlas constructed from measurements in eight cadavers. *International Journal of Computer Assisted Radiology and Surgery*. 2017, vol. 12. Available from DOI: 10.1007/s11548-016-1519-8.
17. SPITZER, Victor; ACKERMAN, Michael; SCHERZINGER, Ann; WHITLOCK, DG. The Visible Human Male: A Technical Report. *Journal of the American Medical Informatics Association: JAMIA*. 1996, vol. 3, pp. 118–30. Available from DOI: 10.1136/jamia.1996.96236280.

18. CARBONE, Vincenzo et al. TLEM 2.0-A Comprehensive Musculoskeletal Geometry Dataset For Subject-Specific Modeling Of Lower Extremity. *Journal of biomechanics*. 2015, vol. 48. Available from DOI: 10.1016/j.jbiomech.2014.12.034.
19. KALC, Milos et al. Motor Unit Identification in the M Waves Recorded by High-Density Electromyogram. *IEEE Transactions on Biomedical Engineering*. 2022, vol. PP, pp. 1–11. Available from DOI: 10.1109/TBME.2022.3224962.
20. ZHAO, Youbing et al. Laplacian Musculoskeletal Deformation for Patient-Specific Simulation and Visualisation. In: 2013, pp. 505–510. Available from DOI: 10.1109/IV.2013.67.
21. LI, Hao; SUMNER, Robert; PAULY, Mark. Global Correspondence Optimization for Non-Rigid Registration of Depth Scans. *Computer Graphics Forum*. 2008, vol. 27. Available from DOI: 10.1111/j.1467-8659.2008.01282.x.
22. ARGYRIOU, V.; RINCON, J. Martinez Del; VILLARINI, B.; ROCHE, A. Medical Image Registration Measures. In: *Image, Video & 3D Data Registration*. John Wiley & Sons, Ltd, 2015, chap. 7, pp. 162–200. ISBN 9781118702451. Available from DOI: <https://doi.org/10.1002/9781118702451.ch7>.
23. DELP, Scott; BLEMKER, Silvia. Three-Dimensional Representation of Complex Muscle Architectures and Geometries. *Annals of biomedical engineering*. 2005, vol. 33, pp. 661–73. Available from DOI: 10.1007/s10439-005-1433-7.
24. KOHOUT, Josef; KUKACKA, M. Real-Time Modelling of Fibrous Muscle. *Computer Graphics Forum*. 2014, vol. 33. Available from DOI: 10.1111/cgf.12354.
25. KOHOUT, Josef; CHOLT, David. Automatic Reconstruction of the Muscle Architecture from the Superficial Layer Fibres Data. *Computer Methods and Programs in Biomedicine*. 2017, vol. 150. Available from DOI: 10.1016/j.cmpb.2017.08.002.
26. UHLIR, Karel; SKALA, Vaclav. Reconstruction of damaged images using Radial Basis Functions. *13th European Signal Processing Conference, EUSIPCO 2005*. 2005.
27. HAYKIN, Simon. *Neural Networks: A Comprehensive Foundation (2nd Edition)* *Neural Networks: A Comprehensive Foundation*. 1998. ISBN 0132733501.
28. PAN, Rongjiang; SKALA, Vaclav. Continuous Global Optimization in Surface Reconstruction from an Oriented Point Cloud. *Computer-Aided Design*. 2011, vol. 43, pp. 896–901. Available from DOI: 10.1016/j.cad.2011.03.005.

29. HARDY, Rolland. Multiquadric Equations of Topography and Other Irregular Surfaces. *Journal of Geophysical Research*. 1971, vol. 76, pp. 1905–1915. Available from DOI: 10.1029/JB076i008p01905.
30. CATMULL, Edwin; ROM, Raphael. A Class of Local Interpolating Splines. *Computer Aided Geometric Design - CAGD*. 1974, vol. 74. ISBN 9780120790500. Available from DOI: 10.1016/B978-0-12-079050-0.50020-5.
31. YUKSEL, Cem; SCHAEFER, Scott; KEYSER, John. On the Parameterization of Catmull-Rom Curves. In: *2009 SIAM/ACM Joint Conference on Geometric and Physical Modeling*. San Francisco, California: ACM, 2009, pp. 47–53. ISBN 978-1-60558-711-0. Available from DOI: 10.1145/1629255.1629262.
32. DERESHGI, Hamid; SERBEST, Kasim; SAHIN, Sema; BALIK, Busra. Skeletal Muscle Mechanics from Hill-Based Muscle Model to Computer Applications: State of the Art Review. 2021, vol. 2, pp. 27–39.
33. LEE, Dongwoon; GLUECK, Michael; KHAN, Azam; FIUME, Eugene; JACKSON, Kenneth. Modeling and Simulation of Skeletal Muscle for Computer Graphics: A Survey. *Foundations and Trends in Computer Graphics and Vision*. 2012, vol. 7, p. 229. Available from DOI: 10.1561/06000000036.
34. HILL, AV. The heat of shortening and the dynamic constants of muscle. *Proc. R. Soc. Lond. B*. 1938, vol. 126, pp. 612–745.
35. ZAJAC, F.E. Muscle and tendon: Properties, models, scaling, and application to biomechanics and motor control. *Critical reviews in biomedical engineering*. 1989, vol. 17, pp. 359–411.
36. JIN, Zhongmin; LI, Junyan; CHEN, Zhenxian. *Computational Modelling of Biomechanics and Biotribology in the Musculoskeletal System*. 2020.
37. ARSLAN, Yunus Ziya; KARABULUT, D.; ORTES, F.; POPOVIC, M. Exoskeletons, Exomusculatures, Exosuits: Dynamic Modeling and Simulation. In: 2019, pp. 305–331. ISBN 9780128129395. Available from DOI: 10.1016/B978-0-12-812939-5.00011-2.
38. MODENESE, Luca; KOHOUT, Josef. Automated Generation of Three-Dimensional Complex Muscle Geometries for Use in Personalised Musculoskeletal Models. *Annals of Biomedical Engineering*. 2020, vol. 48. Available from DOI: 10.1007/s10439-020-02490-4.
39. MARTINS, J. A. C.; PATO, M. P. M.; PIRES, E. B. A finite element model of skeletal muscles. *Virtual and Physical Prototyping*. 2006, vol. 1, no. 3, pp. 159–170. Available from DOI: 10.1080/17452750601040626.

40. VALENTE, Giordano; MARTELLI, Saulo; TADDEI, Fulvia; FARINELLA, Giovanna; VICECONTI, Marco. Muscle discretization affects the loading transferred to bones in lower-limb musculoskeletal models. *Proceedings of the Institution of Mechanical Engineers. Part H, Journal of engineering in medicine*. 2012, vol. 226, pp. 161–9. Available from DOI: 10.1177/0954411911425863.
41. GARNER, Brian; PANDY, Marcus. Estimation of Musculotendon Properties in the Human Upper Limb. *Annals of biomedical engineering*. 2003, vol. 31, pp. 207–20. Available from DOI: 10.1114/1.1540105.
42. KOHOUT, J. et al. Patient-specific fibre-based models of muscle wrapping. *Interface Focus*. 2013, vol. 3, no. 2, p. 20120062. Available from DOI: 10.1098/rsfs.2012.0062.
43. LLOYD, John E.; ROEWER-DESPRES, Francois; STAVNESS, Ian. Muscle Path Wrapping on Arbitrary Surfaces. *IEEE Transactions on Biomedical Engineering*. 2021, vol. 68, no. 2, pp. 628–638. Available from DOI: 10.1109/TBME.2020.3009922.
44. GEORGE-GHIOCEL, Ojoc; BABUT, Cornel; UNGUREANU, Nicolae; DELEANU, Lorena. FEM analysis of Storz coupling. 2021, vol. 6, pp. 249–258.
45. BLAHETA, Radim. *Matematicke modelovani a metoda konecných prvku*. 2012. In czech language only.
46. CERVENKA, Martin. *Computer Muscle Modelling. Technical report number DCSE/TR-2022-03*. University of West Bohemia, 2022.
47. MARTINS, J.A.C.; PIRES, E.B.; SALVADO, R.; DINIS, P.B. A numerical model of passive and active behavior of skeletal muscles. *Computer Methods in Applied Mechanics and Engineering*. 1998, vol. 151, no. 3, pp. 419–433. ISSN 0045-7825. Available from DOI: [https://doi.org/10.1016/S0045-7825\(97\)00162-X](https://doi.org/10.1016/S0045-7825(97)00162-X). Containing papers presented at the Symposium on Advances in Computational Mechanics.
48. BOUBAKER, Bader; PATO, Matilde; PIRES, Eduardo. A finite element model of skeletal muscle. *Virtual and Physical Prototyping*. 2006, vol. 1, pp. 159–170. Available from DOI: 10.1080/17452750601040626.
49. OBERHOFER, Katja; MITHRARATNE, Kumar; STOTT, Ngaire; ANDERSON, Iain. Anatomically-based musculoskeletal modeling: Prediction and validation of muscle deformation during walking. *The Visual Computer*. 2009, vol. 25, pp. 843–851. Available from DOI: 10.1007/s00371-009-0314-8.

50. DIFFO KAZE, Arnaud; MAAS, Stefan; ARNOUX, Pierre-Jean; WOLF, Claude; PAPE, Dietrich. A finite element model of the lower limb during stance phase of gait cycle including the muscle forces. *BioMedical Engineering OnLine*. 2017, vol. 16, p. 138. Available from DOI: 10.1186/s12938-017-0428-6.
51. WEI, Yuyang; ZOU, Zhenmin; WEI, Guowu; REN, Lei; QIAN, Zhihui. Subject-Specific Finite Element Modelling of the Human Hand Complex: Muscle-Driven Simulations and Experimental Validation. *Annals of Biomedical Engineering*. 2019, vol. 48. Available from DOI: 10.1007/s10439-019-02439-2.
52. NOLAN, David; GOWER, Artur; DESTRADE, Michel; OGDEN, Ray; MCGARRY, P. *A robust anisotropic hyperelastic formulation for the modelling of soft tissue*. 2020.
53. FOUGERON, Nolwenn; ROHAN, Pierre-Yves; ROSE, Jean-Loic; BONNET, Xavier; PILLET, Helene. Finite element analysis of the stump-ischial containment socket interaction: a technical note. *Medical Engineering & Physics*. 2022, vol. 105, p. 103829. Available from DOI: 10.1016/j.medengphy.2022.103829.
54. VILA POUCA, Maria et al. Modeling Permanent Deformation during Low-Cycle Fatigue: Application to the Pelvic Floor Muscles during Labor. *Journal of the Mechanics and Physics of Solids*. 2022, vol. 164, p. 104908. Available from DOI: 10.1016/j.jmps.2022.104908.
55. SUN, Xiaobang et al. A Statistical Model of Spine Shape and Material for Population-Oriented Biomechanical Simulation. *IEEE Access*. 2021, vol. PP, pp. 1–1. Available from DOI: 10.1109/ACCESS.2021.3129097.
56. ROMEO, M.; MONTEAGUDO, C.; SANCHEZ-QUIROS, D. Muscle Simulation with Extended Position Based Dynamics. In: GARCIA-FERNANDEZ, Ignacio; URENA, Carlos (eds.). *Spanish Computer Graphics Conference (CEIG)*. The Eurographics Association, 2018. ISBN 978-3-03868-067-3. Available from DOI: 10.2312/ceig.20181146.
57. GEORGII, Joachim; WESTERMANN, Rudiger. Mass-spring systems on the GPU. *Simulation Modelling Practice and Theory*. 2005, vol. 13, pp. 693–702. Available from DOI: 10.1016/j.simpat.2005.08.004.
58. AUBEL, Amaury; THALMANN, Daniel. Interactive modeling of the human musculature. In: 2001, pp. 167–255. ISBN 0-7803-7237-9. Available from DOI: 10.1109/CA.2001.982390.
59. JANAK., Tomas; KOHOUT., Josef. Deformable Muscle Models for Motion Simulation. In: *Proceedings of the 9th International Conference on Computer Graphics Theory and Applications - GRAPP, (VISIGRAPP 2014)*. SciTePress, INSTICC, 2014, pp. 301–311. ISBN 978-989-758-002-4. ISSN 2184-4321. Available from DOI: 10.5220/0004678903010311.

60. HONG, Min; JUNG, Sunhwa; CHOI, Min-Hyung; WELCH, Samuel W.J. Fast Volume Preservation for a Mass-Spring System. *IEEE Computer Graphics and Applications*. 2006, vol. 26, no. 5, pp. 83–91. Available from DOI: 10.1109/MCG.2006.104.
61. KELLNHOFER, Petr; KOHOUT, Josef. Time-convenient Deformation of Musculoskeletal System. In: 2012.
62. FASSER, M. -. et al. Subject-Specific Alignment and Mass Distribution in Musculoskeletal Models of the Lumbar Spine. *Frontiers in Bioengineering and Biotechnology*. 2021, vol. 9. Available also from: www.scopus.com. Cited By :2.
63. WANG, Bohan; MATCUK, George; BARBIČ, Jernej. Modeling of Personalized Anatomy Using Plastic Strains. *ACM Trans. Graph.* 2021, vol. 40, no. 2. ISSN 0730-0301. Available from DOI: 10.1145/3443703.
64. AUBEL, Amaury; THALMANN, Daniel. Efficient Muscle Shape Deformation. 2002. Available from DOI: 10.1007/978-0-306-47002-8_12.
65. CERVENKA, M.; KOHOUT, J. Fast and Realistic Approach to Virtual Muscle Deformation. in *Proceedings of the 14th International Joint Conference on Biomedical Engineering Systems and Technologies - Volume 5: HEALTHINF*. 2020, pp. 217–227. ISBN 978-989-758-398-8. Available from DOI: <https://doi.org/10.5220/0009129302170227>. UT WoS: 000571479400020, EID: 2-s2.0-85083710925, OBD: 43929104.
66. SEYLAN, Caglar; SAHILLIOGLU, Yusuf. 3D Shape Deformation Using Stick Figures. *Computer-Aided Design*. 2022, vol. 151, p. 103352. ISSN 0010-4485. Available from DOI: <https://doi.org/10.1016/j.cad.2022.103352>.
67. DVORAK, Jan; KACEREKOVA, Zuzana; VANECEK, Petr; HRUDA, Lukas; VASA, Libor. As-rigid-as-possible volume tracking for time-varying surfaces. *Computers & Graphics*. 2022, vol. 102, pp. 329–338. ISSN 0097-8493. Available from DOI: <https://doi.org/10.1016/j.cag.2021.10.015>.
68. MULLER, Matthias; HEIDELBERGER, Bruno; HENNIX, Marcus; RATCLIFF, John. Position based dynamics. *Journal of Visual Communication and Image Representation*. 2007, vol. 18, no. 2, pp. 109–118. ISSN 1047-3203. Available from DOI: <https://doi.org/10.1016/j.jvcir.2007.01.005>.
69. MACKLIN, Miles; MULLER, Matthias; CHENTANEZ, Nuttapong. XPBD: Position-Based Simulation of Compliant Constrained Dynamics. In: 2016. Available from DOI: 10.1145/2994258.2994272.
70. ANGLES, Baptiste et al. VIPER: Volume Invariant Position-Based Elastic Rods. *Proc. ACM Comput. Graph. Interact. Tech.* 2019, vol. 2, no. 2. Available from DOI: 10.1145/3340260.

71. KOHOUT, J.; CERVENKA, M. Muscle Deformation Using Position Based Dynamics. *Ye X. et al. (eds) Biomedical Engineering Systems and Technologies. BIOSTEC 2020. Communications in Computer and Information Science.* 2021, vol. 1400. Available from DOI: https://doi.org/10.1007/978-3-030-72379-8_24. EID: 2-s2.0-85107281398, OBD: 43932927.
72. HAVLICEK, O.; CERVENKA, M.; KOHOUT, J. Collision detection and response approaches for computer muscle modelling. *Informatics 2022, IEEE proceedings.* 2022, pp. 120–125. Available from DOI: <https://doi.org/10.1109/Informatics57926.2022.10083500>. EID: 2-s2.0-85153333554, OBD: 43937869.
73. HAVLICEK, Ondrej. *Fast Collision Detection in the Context of Muscle Deformation by a Position Based Dynamics Method* [Bachelor's Thesis]. University of West Bohemia, Faculty of Applied Sciences, 2021. Supervisor: Kohout, J.
74. BURZYNSKI, S.; SABIK, Aa; WITKOWSKI, W.; LUCZKIEWICZ, P. Influence of the femoral offset on the muscles passive resistance in total hip arthroplasty. *PLOS ONE.* 2021, vol. 16, no. 5, pp. 1–12. Available from DOI: <https://doi.org/10.1371/journal.pone.0250397>.
75. OLIVER, M. A.; WEBSTER, R. Kriging: a method of interpolation for geographical information systems. *International journal of geographical information systems.* 1990, vol. 4, no. 3, pp. 313–332. Available from DOI: [10.1080/02693799008941549](https://doi.org/10.1080/02693799008941549).
76. KAYMAZ, Irfan. Application of kriging method to structural reliability problems. *Structural Safety.* 2005, vol. 27, no. 2, pp. 133–151. ISSN 0167-4730. Available from DOI: <https://doi.org/10.1016/j.strusafe.2004.09.001>.
77. SAKATA, S.; ASHIDA, F.; ZAKO, M. An efficient algorithm for Kriging approximation and optimization with large-scale sampling data. *Computer Methods in Applied Mechanics and Engineering.* 2004, vol. 193, no. 3, pp. 385–404. ISSN 0045-7825. Available from DOI: <https://doi.org/10.1016/j.cma.2003.10.006>.
78. JOSEPH, V. Roshan; HUNG, Ying; SUDJIANTO, Agus. Blind Kriging: A New Method for Developing Metamodels. *Journal of Mech. Design.* 2008, vol. 130, no. 3, p. 031102. ISSN 1050-0472. Available from DOI: [10.1115/1.2829873](https://doi.org/10.1115/1.2829873).
79. HARDY, Rolland L. Multiquadric equations of topography and other irregular surfaces. *Journal of Geophysical Research.* 1971, vol. 76, no. 8, pp. 1905–1915. Available from DOI: <https://doi.org/10.1029/JB076i008p01905>.

80. HARDY, R.L. Theory and applications of the multiquadric-biharmonic method 20 years of discovery 1968-1988. *Computers & Mathematics with Applications*. 1990, vol. 19, no. 8, pp. 163–208. ISSN 0898-1221. Available from DOI: [https://doi.org/10.1016/0898-1221\(90\)90272-L](https://doi.org/10.1016/0898-1221(90)90272-L).
81. MAJDISOVA, Zuzana; SKALA, Vaclav. Radial basis function approximations: comparison and applications. *Applied Mathematical Modelling*. 2017, vol. 51, pp. 728–743. ISSN 0307-904X. Available from DOI: <https://doi.org/10.1016/j.apm.2017.07.033>.
82. WANG, J.G.; LIU, G.R. On the optimal shape parameters of radial basis functions used for 2-D meshless methods. *Computer Methods in Applied Mechanics and Engineering*. 2002, vol. 191, no. 23, pp. 2611–2630. ISSN 0045-7825. Available from DOI: [https://doi.org/10.1016/S0045-7825\(01\)00419-4](https://doi.org/10.1016/S0045-7825(01)00419-4).
83. AFIATDOUST, F.; ESMAEILBEIGI, M. Optimal variable shape parameters using genetic algorithm for radial basis function approximation. *Ain Shams Engineering Journal*. 2015, vol. 6, no. 2, pp. 639–647. ISSN 2090-4479. Available from DOI: <https://doi.org/10.1016/j.asej.2014.10.019>.
84. COHEN-STEINER, David; ALLIEZ, Pierre; DESBRUN, Mathieu. Variational Shape Approximation. *ACM Trans. Graph.* 2004, vol. 23, no. 3, pp. 905–914. ISSN 0730-0301. Available from DOI: [10.1145/1015706.1015817](https://doi.org/10.1145/1015706.1015817).
85. SARRA, Scott A.; STURGILL, Derek. A random variable shape parameter strategy for radial basis function approximation methods. *Engineering Analysis with Boundary Elements*. 2009, vol. 33, no. 11, pp. 1239–1245. ISSN 0955-7997. Available from DOI: <https://doi.org/10.1016/j.enganabound.2009.07.003>.
86. SKALA, Vaclav; KARIM, Samsul Ariffin Abdul; ZABRAN, Marek. Radial Basis Function Approximation Optimal Shape Parameters Estimation. In: *Computational Science – ICCS 2020*. Cham: Springer International Publishing, 2020, pp. 309–317. ISBN 978-3-030-50433-5.
87. SKALA, Vaclav; KANSA, Edward. Why Is the Least Square Error Method Dangerous? *Computacion y Sistemas*. 2021, vol. 25. Available from DOI: [10.13053/cys-25-1-3473](https://doi.org/10.13053/cys-25-1-3473).
88. SKALA, V.; KARIM, S.; CERVENKA, M. Finding Points of Importance for Radial Basis Function Approximation of Large Scattered Data. *Computational Science - ICCS 2020, Part VI, LNCS 12142*. 2020, pp. 239–250. Available from DOI: https://doi.org/10.1007/978-3-030-50433-5_19. OBD: 43932925, UT WoS: 000841676000019, EID: 2-s2.0-85087274721.

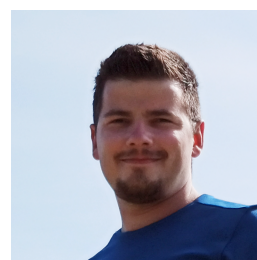
89. CERVENKA, M.; SKALA, V. Behavioral Study of Various Radial Basis Functions for Approximation and Interpolation Purposes. *IEEE 18th World Symposium on Applied Machine Intelligence and Informatics, SAMI 2020*. 2020, pp. 135–140. ISBN 978-1-7281-3149-8. Available from DOI: <https://doi.org/10.1109/SAMI48414.2020.9108712>. UT WoS: 000589772600026, EID: 2-s2.0-85087093548, OBD: 43929006.
90. LEWIS, J.P.; PIGHIN, Frederic; ANJYO, Ken. Scattered data interpolation for computer graphics. *ACM SIGGRAPH 2014 Courses, SIGGRAPH 2014*. 2010. Available from DOI: [10.1145/1900520.1900522](https://doi.org/10.1145/1900520.1900522).
91. CERVENKA, M.; SKALA, V. Conditionality Analysis of the Radial Basis Function Matrix. *ICCSA 2020 proceedings, part II, LNCS*. 2020, pp. 30–43. Available from DOI: https://doi.org/10.1007/978-3-030-58802-1_3. UT WoS: 000719685200003, EID: 2-s2.0-85093112881, OBD: 43932697.
92. HALTON, J. Algorithm 247: Radical-inverse quasi-random point sequence. *Commun. ACM*. 1964, vol. 7, pp. 701–702. Available from DOI: [10.1145/355588.365104](https://doi.org/10.1145/355588.365104).
93. SKALA, V.; CERVENKA, M. Novel RBF Approximation Method Based on Geometrical Properties for Signal Processing with a New RBF Function: Experimental Comparison. *Informatics 2019, IEEE proceedings*. 2019, pp. 357–362. ISBN 978-1-7281-3178-8. Available from DOI: <https://doi.org/10.1109/Informatics47936.2019.9119276>. UT WoS: 000610452900074, EID: 2-s2.0-85087090327, OBD: 43929007.
94. VASTA, J.; SKALA, V.; SMOLIK, M.; CERVENKA, M. Modified Radial Basis Functions Approximation Respecting Data Local Features. *Informatics 2019, IEEE proceedings*. 2019, pp. 445–449. ISBN 978-1-7281-3178-8. Available from DOI: <https://doi.org/10.1109/Informatics47936.2019.9119330>. UT WoS: 000610452900015, EID: 2-s2.0-8508762067, OBD: 43928987.
95. CERVENKA, Martin; KOHOUT, Josef; LIPUS, Bogdan. A mathematical model for smooth Radial Basis Function implicit surface model for the purpose of muscle modelling. *INFORMATICA*. 2024, submitted.
96. ZHOU, Tiancheng et al. Reducing the metabolic energy of walking and running using an unpowered hip exoskeleton. *Journal of NeuroEngineering and Rehabilitation*. 2021, vol. 18, p. 95. ISSN 1743-0003. Available from DOI: [10.1186/s12984-021-00893-5](https://doi.org/10.1186/s12984-021-00893-5).
97. CERVENKA, M.: Geometry Algebra and Gauss Elimination method for solving a linear system of equations without division. *Informatics 2022, IEEE proceedings*. 2022, pp. 55–59. Available from DOI: <https://doi.org/10.1109/>

- Informatics57926 . 2022 . 10083445. OBD: 43937872, EID: 2-s2.0-851533-55665.
98. CERVENKA, M.; HAVLICEK, O.; KOHOUT, J.; VASA, L. Computer muscle modelling. *Computerised muscle modelling and simulation for interactive applications, Proceedings of the 18th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, VISIGRAPP 2023, Volume 1: GRAPP*. 2023, pp. 214–221. ISBN 978-989-758-634-7. Available from DOI: <https://doi.org/10.5220/0011688000003417>. UT WoS: 001066254400019, OBD: 43940148.
99. CANI-GASCUEL, M.; DESBRUN, M. Animation of deformable models using implicit surfaces. *IEEE Transactions on Visualization and Computer Graphics*. 1997, vol. 3, no. 1, pp. 39–50. Available from DOI: 10.1109/2945.582343.

About the author

Personal information

Title, name, surname: Ing. Martin Červenka
Date of birth, place: 24th of March, 1995, Pilsen, CZE
Address: Loza 85, 331 52, Czech Republic
E-mail: cervemar@seznam.cz
Webpages: cervemar.cz
Other contacts: github.com/cervenkam,
m.me/cervemar,
t.me/cervemar



Research identifiers

E-mail: cervemar@kiv.zcu.cz
ORCID: 0000-0001-9625-1872
Researcher ID: AAJ-8878-2020
Scopus ID: 57209970186
ResearchGate: Martin-Cervenka-2

Education

since 2019: **Univerzity of West Bohemia, Pilsen, CZE,**
Faculty of Applied Sciences, Doctoral study
Informatics and computer technology
2017-2019: **Univerzity of West Bohemia,**
Faculty of Applied Sciences, Master study
Medical informatics
2014-2017: **Univerzity of West Bohemia,**
Faculty of Applied Sciences, Bachelor study
Informatics
2010-2014: **The Sec. Industrial School of Electrical Engineering Pilsen,**
Information technologies

Project activities

- 2023: **Project: GACR No. 22-04622L**
Data compression paradigm based on omitting self-evident information – COMPROMISE
Czech Science Foundation
The employment contract
10th of January 2023 – recent
Project: NPO ZCU MSMT-16584/2022
National renewal plan for the area of higher educational institutes for the years 2022 – 2024
The employment contract
3rd of June 2023 – recent
The Czech Ministry of Education, Youth and Sports
- 2022 **Project: IDEG-2021-027**
Signal Classification and interpretation
1st of January 2022 to 31st of December 2023
The employment contract
- 2019: **Project: GACR No. 17-05534S,**
Meshless methods for large scattered spatiotemporal vector data vis.
Czech Science Foundation
15th of September 2019 to 31st of December 2019
The employment contract

Lectures

- 2024 **Tutorial lecturer: KIV/VI**
Information Visualization
8 students, one lecture a week
- 2023 **Tutorial lecturer: KIV/PT-E**
Programming Techniques in English, Winter semester
5 students, one lecture a week
Tutorial lecturer: KIV/PT
Programming Techniques, Winter semester
70 students, four lectures a week
Tutorial lecturer: KIV/UPG
Introduction to Computer Graphics, Summer semester
93 students, four lectures a week (last year in the 1st year of study)
- 2022: **Tutorial lecturer: KIV/PT-E**
1 student, one consultation a week
Tutorial lecturer: KIV/PT
44 students, two lectures a week
Tutorial lecturer: KIV/UPG
90 students, four lectures a week
- 2021: **Tutorial lecturer: KIV/PT-E**
5 students, one lecture a week
Tutorial lecturer: KIV/PT
28 students, two lectures a week
Tutorial lecturer: KIV/UPG
69 students, one lecture a week (online due to COVID-19)
- 2020: **Tutorial lecturer: KIV/PT**
30 students, two lectures a week
Tutorial lecturer: KIV/UPG
56 students, three lectures a week
- 2019 **Tutorial lecturer - KIV/UPG**
22 students, one lecture a week

Lecture's feedback

Throughout the years, we received much feedback on our work in our lectures. Many things happen; sometimes we were successful, sometimes we weren't, but the lecturing was quite successful. The success can be highlighted by the students' responses, from which I selected all of them which were targeted personally. The translation was from Czech, so remember that the following is not a direct quotation.

- I have to mention the professional approach of our lecturer, Mr Martin Červenka, to the solution of the semester work; without his interpretation, help and consultation, I probably would not have been able to get the credit.
- The lectures were interesting, and the friendliness and organization of Mr. Kohout and Mr. Červenka (he took care of communication via Discord) should be an example to other teachers.
- I have to give a lot of praise to Mr. Červenka, who founded the Discord server, which helped us a lot in our curriculum and which, in my opinion, should be the basis of every other technically oriented subject.
- I appreciate Mr. Červenka's approach, who conscientiously helped all students on Discord. He responded almost immediately and went out of his way to help with every problem, even problems in other subjects!
- I want to highlight the willingness and determination to help and generally communicate with students of Ing. Červenka.
- I want to thank my colleague Mr. Červenka for his clear advice during the exercises and the SP.
- Mr Červenka is the salvation of this subject, and it is only thanks to him that I take away helpful information from the subject.

The personal feedback during lectures and those written drove me to improve the lectures each year. I have remembered all the feedback and will never forget it. I am thankful to help and collaborate with each student throughout the years.

Publications

1. Cervenka, M., Kohout, J., Lipus, B.: **A mathematical model for smooth Radial Basis Function implicit surface model for the purpose of muscle modelling**, INFORMATICA 2024 (submitted)
2. Cervenka, M., Havlicek, O., Kohout, J., Vasa, L.: **Computerised muscle modelling and simulation for interactive applications**. Proceedings of the 18th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, VISIGRAPP 2023, Volume 1: GRAPP. 2023, pp. 214–221. ISBN 978-989-758-634-7. <https://doi.org/10.5220/0011688000003417>
3. Kohout, J., Cervenka, M.: **Non-planar Surface Shape Reconstruction from a Point Cloud in the Context of Muscles Attachments Estimation**. in Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, ISBN 978-989-758-555-5, pages 236-243. (2022) <https://doi.org/10.5220/0010869600003124>
4. Cervenka, M.: **Geometry Algebra and Gauss Elimination method for solving a linear system of equations without division**. Informatics 2022, IEEE proceedings. 2022, pp. 55–59. <https://doi.org/10.1109/Informatics57926.2022.10083445>
5. Havlicek, O., Cervenka, M., Kohout, J.: **Collision detection and response approaches for computer muscle modelling**. Informatics 2022, IEEE proceedings. 2022, pp. 120–125. 2022. <https://doi.org/10.1109/Informatics57926>
6. Kohout, J., Cervenka, M.: **Muscle Deformation Using Position Based Dynamics** In: Ye X. et al. (eds) Biomedical Engineering Systems and Technologies. BIOSTEC 2020. Communications in Computer and Information Science 1400 (2021). EID: 2-s2.0-85107281398, OBD: 43932927. https://doi.org/https://doi.org/10.1007/978-3-030-72379-8_24
7. Cervenka, M., Kohout, J.: **Fast and Realistic Approach to Virtual Muscle Deformation**. in Proceedings of the 13th International Joint Conference on Biomedical Engineering Systems and Technologies - Volume 5: HEALTHINF, ISBN 978-989-758-398-8, pages 217-227. (2020) <https://doi.org/10.5220/0009129302170227>
8. Skala, V., Karim, S., Cervenka, M.: **Finding Points of Importance for Radial Basis Function Approximation of Large Scattered Data**. Computational Science - ICCS 2020, Part VI, LNCS 12142, pp. 239-250, Springer, (2020) https://doi.org/10.1007/978-3-030-50433-5_19
9. Cervenka, M., Skala, V.: **Behavioral Study of Various Radial Basis Functions for Approximation and Interpolation Purposes**. IEEE 18th World Symposium on Applied Machine Intelligence and Informatics, SAMI 2020, pp.135-140, ISBN 978-1-7281-314, Slovakia, (2020) (Scopus) <https://doi.org/10.1109/SAMI48414.2020.9108712>
10. Cervenka, M., Skala, V.: **Conditionality Analysis of the Radial Basis Function Matrix**. ICCSA 2020 proceedings, part II, LNCS, pp. 30-43, Springer, (2020) https://doi.org/10.1007/978-3-030-58802-1_3
11. Skala, V., Cervenka, M.: **Novel RBF Approximation Method Based on Geometrical Properties for Signal Processing with a New RBF Function: Experimental Comparison**. Informatics 2019, IEEE proceedings, pp.357-362, ISBN 978-1-7281-3178-8, Poprad, Slovakia, (2019) <https://doi.org/10.1109/Informatics47936.2019.9119276>
12. Vasta, J., Skala, V., Smolik, M., Cervenka, M.: **Modified Radial Basis Functions Approximation Respecting Data Local Features**. Informatics 2019, IEEE proceedings, pp.445-449, ISBN 978-1-7281-3178-8, Poprad, Slovakia, (2019) <https://doi.org/10.1109/Informatics47936.2019.9119330>
13. Cervenka, M., Smolik, M., Skala, V.: **A New Strategy for Scattered Data Approximation Using Radial Basis Functions Representing Points of Inflection**. Computational Science and Its Application, ICSSA 2019 proceedings, Part I, LNCS 11619, pp.322-226, ISSN 0302-9743, ISBN 978-3-030-24288-6, Springer, (2019) https://doi.org/10.1007/978-3-030-24289-3_24

101011000011100010 1100001
1010110001 10001



11010011101101001 10101
01100001 10101
111000101011 101

