

University of Bath



DOCTOR OF ENGINEERING (ENGD)

E-StopMotion: Reconstructing and Enhancing 3D Animation of Stop Motion Characters by Reverse Engineering Plasticine Deformation

Ciucanu, Anamaria

Award date:
2019

Awarding institution:
University of Bath

[Link to publication](#)

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal ?

Take down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Citation for published version:

Ciucanu, A 2018, 'E-StopMotion: Reconstructing and Enhancing 3D Animation of Stop Motion Characters by Reverse Engineering Plasticine Deformation', Doctor of Engineering (EngD), University of Bath.

Publication date:

2018

Document Version

Publisher's PDF, also known as Version of record

[Link to publication](#)

University of Bath

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

E-StopMotion: Reconstructing and Enhancing 3D Animation of Stop Motion Characters by Reverse Engineering Plasticine Deformation

submitted by

Anamaria Ciucanu

for the degree of Doctor of Engineering

of the

University of Bath

Department of Computer Science

September 2018

COPYRIGHT

Attention is drawn to the fact that copyright of this thesis rests with the author. A copy of this thesis has been supplied on condition that anyone who consults it is understood to recognise that its copyright rests with the author and that they must not copy it or use material from it except as permitted by law or with the consent of the author.

This thesis may be made available for consultation
within the University Library and may be
photocopied or lent to other libraries for the purposes
of consultation with effect from.....(date)

Signed on behalf of the Faculty of Science

Contents

1	Introduction	6
1.1	Motivation	6
1.2	Task	8
1.2.1	Task Analysis	9
1.3	Aims and Objectives	12
1.3.1	Context	12
1.3.2	Research Questions	14
1.4	Chapter Organization	17
2	Stop Motion Animation Character Classification	19
2.1	A Short History of Stop Motion Animation	20
2.1.1	Stop Motion and Film	20
2.1.2	Stop Motion and Games	22
2.1.3	Handmade Characters	23
2.1.4	Fat Pebble Animation	24
2.2	Classification of Clay Jam Characters	25
2.2.1	Method	26
2.2.2	Experiments and Results	33
2.2.3	Discussion	39
3	E-StopMotion: Stop Motion Animation Reconstruction Pipeline	43
3.1	Animation Reconstruction	44
3.2	E-StopMotion: Full Pipeline Method	47
3.2.1	Acquisition	50
3.2.2	Cleanup	53
3.2.3	Analysis	53
3.2.4	Helper Addition	55
3.2.5	Deformation	56

3.2.6	Non-rigid Registration	57
3.2.7	Interpolation	57
3.2.8	Applications	62
3.3	E-StopMotion: Simplified Pipeline Implementation	62
3.3.1	Overview	62
3.3.2	Experiments and Results	64
3.3.3	Applications	68
3.4	Discussion	69
4	Representation and Deformation of Virtual Clay Models	74
4.1	Analysis of Clay Properties	75
4.1.1	Volume Preservation	75
4.1.2	Plasticity	78
4.1.3	Part Based Modelling	87
4.2	Representation of Virtual Clay Models	88
4.2.1	Volumetric Models	88
4.2.2	Surface Models	96
4.2.3	Link Between Volumetric and Surface Models	100
4.3	Deformation of Virtual Clay Models	102
4.3.1	Surface Based Deformations	102
4.3.2	Space Based Deformations	111
4.3.3	Stiffness of Virtual Clay Models	112
4.4	E-StopMotion: Deformation Component	114
4.4.1	Method	115
4.4.2	Experiments and Results	130
4.4.3	Discussion	144
5	Registration of Virtual Clay Models	146
5.1	Non-rigid Registration	147
5.1.1	Scales of Deformation	149
5.1.2	Correspondence Problem	153
5.1.3	Trajectory Problem	155
5.2	E-StopMotion: Non-rigid Registration Component	156
5.2.1	Method	156
5.2.2	Experiments and Results	175
5.2.3	Discussion	179
6	Conclusions	182

6.1 Acknowledgements	184
A Clay Jam Characters	185
B Acronyms	189

“When in doubt, put it in a sphere.” (John Benardis)

Summary

Stop motion animation is a popular creative medium with applications across film, games, education and health industries. Traditionally this type of animation has been known as a two dimensional (2D) art form, where the artist iteratively deforms a character and then takes photographs of it. The artist’s task can be overwhelming as he has to reshape a character into hundreds of poses to obtain just a few seconds of animation. Moreover, features that took a lot of effort to create remain unseen due to the preferred 2D method of visualization.

The current project was a collaboration between Fat Pebble Games Studio and the Centre for Digital Entertainment (CDE) from the University of Bath. The aim was to create a novel pipeline for reconstructing and enhancing stop motion animation from three dimensional (3D) character scans, obtained from multi-view images. Deformation, non-rigid registration and interpolation techniques were used to fulfil this aim. These procedures were aided by information about the material a character is made from and the character’s structure. The underling inquiry of the project was to see whether reverse engineering the artist’s plasticine modelling process can result in physically plausible deformations between scans and more accurate non-rigid registrations.

Message: A specialized pipeline for animation reconstruction and enhancement of handmade, plasticine characters can be created by combining deformation, non-rigid registration and interpolation. These techniques can be adapted to imitate the physical properties of plasticine, by including information about the material and structure of the original models.

Key words: plasticine or modelling clay, stiffness, volume preservation, deformation, non-rigid registration, interpolation, nonisometry, animation reconstruction, handmade characters.

Chapter 1

Introduction

1.1 Motivation

Piti (2006) reveals in her article on the increasing popularity of stop motion animation that this art form carries a nostalgia which is hard to achieve with computer generated (CG) animation. The handmade look a plasticine character has does not strive for perfection, but rather for uniqueness. The fact that someone has manually placed the puppet in its current pose, the organic feel of the materials it's made from and the visual illusion of being able to reach out and touch the puppet all contribute to a style of animation worth preserving.

The photorealism of handmade characters in stop motion productions is a quality CG animation has always striven for. The latter, however, has less tedious procedures of obtaining pleasing animations, due to technological advancements. A combination of the two could alleviate issues such as the Uncanny Valley effect for CG artists and long shooting hours for stop motion artists.

Stop motion is often seen in film productions and occasionally in games (examples in section 2.1). Either way, the models in the animation appear as 2D images and don't have the full potential of a 3D model. It would be a novel experience for users to interact with 3D characters that look and move like the original handmade models. Moreover, the artist's workload could be simplified if in-between character poses are generated automatically, given a few key pose character scans. These poses can then be modified or used directly in the animation pipeline.

In order to achieve these goals, this thesis looked at creating a semi-automatic ani-

mation reconstruction and enhancement pipeline for Fat Pebble Games Studio (Pebble (2013)). The focus was on the game Clay Jam (Pebble (2014)), which contains a varied set of animated plasticine characters. These characters were brought into a 3D virtual environment through scanning, non-rigid registration, interpolation and deformation techniques.

Apart from the practical contributions such an application can bring to the games industry, academic contributions added up to making the endeavour worthwhile. Seeing that plasticine models can undergo irreversible, large transformations, the fields of nonisometric deformation and non-rigid registration were explored. Moreover, a deeper understanding of the physics of plasticine enabled the development of a novel deformation model that imitates the behaviour of clay.

Contributions

The first contribution is a heuristic for classifying Clay Jam characters by the animation type they display (chapter 2). For the specific Fat Pebble animation style, it was found that characters are composed of parts, animated either by shape or by skeleton. Depending on the character structure and the scale of deformation, three classes of characters were established. This classification is linked to the categories of shape mapping algorithms (Li & Iyengar (2014)) most suitable for a particular type of character animation.

The second contribution is a stop motion animation reconstruction and enhancement pipeline (chapter 3). After understanding the types of characters available from Clay Jam, the steps for bringing them into the 3D virtual world were identified. The complex methodology is described, but only a simplified version of the pipeline was implemented. The results were published in the Proceedings of Motion Interaction in Games 2018 (Ciucanu et al. (2018)).

The next two chapters (chapters 4 and 5) are in depth improvements of two steps from the implemented pipeline in chapter 3. Chapter 4 contributes with a state-of-the-art report on methods of representing and deforming virtual clay. The thin shell deformation method proposed by Fröhlich & Botsch (2011) was also modified to imitate plasticine deformations via local stiffness and volume preservation.

Chapter 5 contributes with an investigation into how geometric and physics properties of the characters can enhance the non-rigid registration technique proposed by Amberg et al. (2007). This was done to show how virtual clay deformation can be used to reverse engineer and improve the match between a source and a target character scan.

1.2 Task

This project is a collaboration between the University of Bath, Center for Digital Entertainment and Fat Pebble Games Studio, authors of games like Clay Jam and Play-Doh Jam. The style of the company is defined by handmade stop motion animation, with characters and environments crafted from materials like clay and play-doh.

The task revolves around Clay Jam (figure 1-1), a game composed of photographs of real world plasticine characters and backgrounds. The lighting used on the original models gives the illusion of a three dimensional universe, even if the animations seen in the game are quick sequences of 2D sprites. The user can only move in a 2D plane, having a top-down view of the world and cannot rotate around the characters. The question of whether the Clay Jam world can be brought to 3D, without altering the handmade look, naturally occurred in the early stages of the project.



Figure 1-1: Screenshots from the game Clay Jam by Fat Pebble. Notice the characters jumping, running and squashing.

From an academic point of view, this task is one of animation reconstruction, combining the challenges presented in the literature on deformation and non-rigid registration. More specifically, since the Clay Jam models are made from plasticine, which undergo large, irreversible changes, nonisometric deformations and registrations become of particular interest.

From a practical point of view, the task is one of integrating the handmade animation pipeline into the computer generated (CG) animation pipeline. The organic look and feel of real world models are distinguishing features of stop motion animation. As mentioned in the motivation, the CG industry could benefit from the aesthetic of stop motion and the latter would benefit from the efficiency of CG. Next, the component parts of the given task are discussed.

1.2.1 Task Analysis

The given task is thus one of animation reconstruction, which has been studied in various contexts in the graphics literature (Wand et al. (2009), Li et al. (2009)). This field lies between shape deformation and non-rigid registration and knowledge of both is needed in order to provide a suitable solution.

Before starting the scientific inquiries, the complexity of the problem needs to be analyzed by taking a closer look at the given models and their deformation. In the following sections the features that describe Clay Jam characters are stated. A classification of these characters by their shape and movement is given in chapter 2. Also a deep exploration of the physical properties of plasticine and ways of representing it in the virtual world are described in chapter 4.

Static Scans

The available plasticine characters are static and can be scanned fully in different poses. This favours a template based registration approach, since the complete geometry is available. Given a character, the artist can generate a small set of 3D scans of it in different poses. The first scan can then be used as a template for registration against the remaining scans.

For smooth results, the first scan should be cleaned up and prepared for the animation reconstruction procedure. Ideally, the artists' efforts should be minimal for the scanning and cleaning steps. Chapter 3 contains information on the acquisition and cleanup needed for reconstructing the animation of Clay Jam characters.

Plasticine

Plasticine is a material that deforms irreversibly due to its preponderantly plastic quality. The elastic properties of this material are close to negligible. Objects made out of plasticine preserve their volume throughout a deformation. Continuity is also kept during a deformation up to a breaking point. Dents, bulges and sharp indents may appear from one shape to the next, due to the artist’s modelling technique.

Table 1.1 shows the main properties of modelling clay or plasticine. These were taken from material physics, personal observations while working at Fat Pebble and the properties suggested by Dewaele & Cani (2004), who proposed a technique for modelling virtual clay. Plasticity, stiffness and volume preservation were deemed the most relevant for the current project. These properties and their significance are described in chapter 4.

Table 1.1: Modelling clay or plasticine properties.

Physics	Observations	Virtual clay Dewaele & Cani (2004)
Plasticity	Irreversible changes	Plasticity
Stiffness	Localized stiffness	Surface tension
Volume preservation	Dents and bulges	Mass conservation

Since the results will be used in a real-time game engine, rendering speed and storage space are crucial. A surface shape that acts like a volumetric model would be preferred in this case. Thus a triangular mesh undergoing volume preserving deformations is preferable to a physically accurate tetrahedron mesh. Chapter 4 offers a thorough comparison between volumetric and surface deformation methods.

Partial Animation

Characters are created out of two or more parts which move independently, while the simple models consist of a single block of plasticine. It is common for a character’s component parts to exhibit different scales and types of animation. For example, in figure 1-2, the bear on the left has the limbs bending from side to side, while the head and body have almost rigid movements.

It becomes intuitive that a type of segmentation is required to deal with the separately moving parts. The segmentation can be geometric or by the animation displayed in different regions of the character. Chapter 2 explores classifying Clay Jam characters by analyzing the animation of their component parts.



Figure 1-2: Bear character from Clay Jam (left) and a simple plasticine model (right) as 2D sprites.

Scale

As previously mentioned, parts of the character move at various scales. A way to define scale is necessary in order to choose the subsequent registration algorithms needed (chapter 5). One way to define it is to measure the amount of overlap between consecutive frames of animation for each character.

Since creating a 3D scan for every character, for every frame of animation is time-consuming, the available 2D sprite database could be made use of. Chapter 2 explores one method of determining scale by visual observation.

Character Animation Classification

A character can belong to more than one class, depending on the movement exhibited by the component parts. The pipeline should be able to choose the animation class a character belongs to and assign the appropriate helpers to each of its components.

A **helper** can be considered a prior that allows deformation and registration procedures to imitate the properties of the original models. Chapter 2 describes how helpers are divided into model and animation specific layers. The former links to information about the material (eg. stiffness, volume preservation), while the latter is linked to the animation class a character belongs to. Clay Jam characters are classified into characters with *skeletal* or *shape* animations. The former refers to articulated models, while the latter encompasses more free-form behaviours.

1.3 Aims and Objectives

The aim of this research is to use animation reconstruction techniques to obtain 3D character key pose scans and physically plausible in-between poses, with the aesthetic of the original plasticine models.

The resulting animations should run in real time inside a game engine such as Unity. The starting point for the project is Clay Jam and its available characters, but other plasticine models can be used for experimentation. Artists should be able to keep a handmade workflow and use the resulting piece of software to reconstruct and improve the original character animation.

In other words, the resulting character animations should not need any additional modelling, skinning or other general deformations. The amount of effort necessary from the artist should be minimal and related only to the acquisition, cleanup and helper stages, as it will be described in chapter 3. Also, since the characters are very different from each other, the desired technique should be generalizable to as many character animation types as possible.

Both academic and practical objectives are needed to accomplish this aim. The following subsections tackle these objectives, by starting off with the context of the problem, the research questions that come with it and the practical steps of the pipeline. Since the project was ambitious, some of the objectives were achieved, while others are left for future work.

1.3.1 Context

The objectives are prioritized by the observations given in the task analysis and the properties of the available input. Since the input consists of plasticine characters and, more specifically, their 3D scans, deformation and non-rigid registration techniques that can aid the animation reconstruction process need to be considered.

3D scans of a character in different poses can be obtained relatively easily due to current advancements in model reconstruction techniques such as photogrammetry. The question is then whether these scans can be matched to each other. This would help to obtain a template surface which can be interpolated between all the available poses of a character. Ideally the number of scans should be minimal, thus a non-rigid registration technique that allows large deformations is required.

Xu et al. (2005) mention that in order to find the interpolation between two surfaces, two problems need to be solved, the correspondence and the trajectory problems. Interpolation in the context of discrete surfaces (meshes) with different connectivities can be understood as non-rigid registration.

Li (2010) notes that the more correspondences detected between two or more surfaces, the easier it is to generate an intuitive deformation between them. Automatically detecting correspondences based on salient features, as it is common in literature (van Kaick et al. (2011)), would imply that static details remain the same for each surface being analyzed. This is not always the case for plasticine characters as dents and bulges can appear and disappear between frames.

Automatically detecting correspondences between two surfaces can also fail due to the shapes having major differences (figure 1-3). The most obvious solution is to ask the user to provide manual landmark information. However, since the artist should not be burdened with manually selecting correspondences, a sparse set of landmarks should be sufficient for identifying the major changes in the surface. Dense correspondences could then be generated based on a combination of geometric and physics-based priors.

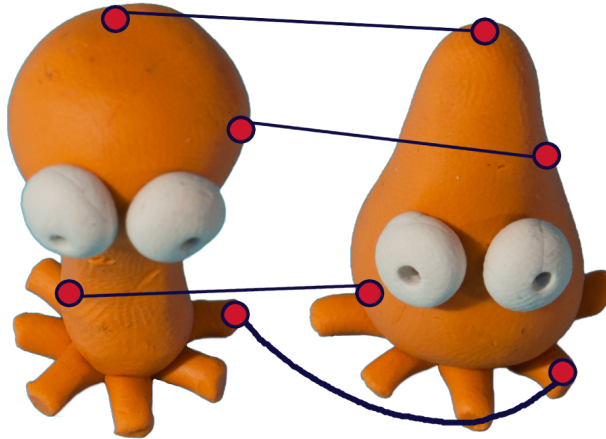


Figure 1-3: Sparse correspondences between two consecutive poses of the Tick character from Clay Jam.

Regardless of the number of correspondences found, the path they follow from the source surface to the target surface is the one that determines what the deformation looks like. The trajectory of a set of points from the source surface to the target surface is usually guided by a form of regularization. Regularization methods are commonly employed to preserve the shape (eg. as-rigid-as-possible techniques proposed by Sorkine & Alexa (2007)), the volume (Eisenberger et al. (2018)) or the smoothness (Amberg

et al. (2007)) of the surface.

This thesis explores whether nonrigid registration can be enhanced by imitating the original plasticine deformation as closely as possible. In order to achieve this, regularization methods which preserve the properties displayed by plasticine characters (eg. volume preservation), while also keeping a smooth appearance of the surface need to be studied.

The state-of-the-art in non-rigid registration and deformation allows for automatically detecting salient features and preserving the shape of models as rigidly as possible. This might fail, however, in the case of plasticine models, since their shape can undergo unexpected, large changes from frame to frame. With limited automatically detected geometric priors, this work turns towards how the physics of the original models can aid the deformation and non-rigid registration.

In the context of Clay Jam, many of the characters move as rigidly as possible so the former mentioned techniques can be applied. The interest, however, is in the character movements where these techniques fail due to their reliance on the geometry rather than on the physical behaviour of the material.

1.3.2 Research Questions

Hypothesis 1 (H1): *By using information about the material the characters are made from and the structure given by their component parts, the character shape match can be enhanced and physically plausible in-between poses can be obtained in the animation reconstruction process.*

This hypothesis is the main driving factor behind the current animation reconstruction work. The belief is that reverse engineering the deformation and understanding how a model acquired a particular shape are crucial to obtaining realistic deformations and accurate shape matches.

The input for the animation reconstruction process are stop motion characters and, more specifically, characters from the game Clay Jam. The argument is that if material behaviour and character structure are truly understood, this work steps closer to discovering a generalized reconstruction method for more types of animation.

To conclude the material presented so far, in order to reach the desired aim, an animation reconstruction technique is to be employed. This technique combines deformation

and non-rigid registration with geometric and physics-based priors. The following paragraphs further break down the aim into a set of research questions.

Research Question 1 (RQ1): *Is there a deformation technique that can replicate plasticine deformation?*

Looking at how characters are created and moved by an artist, the most common action, when it comes to plasticine, is modelling. The artist’s fingers push and pull the material into the desired place. If the contact points of the real world plasticine model are associated with the virtual handle points for any deformation algorithm, suitable deformation methods for imitating the plasticine movement can be studied.

To reiterate an important fact from the task analysis, surface scans and building an application with use in a real-time game engine are desired. Deformation methods that work with surfaces, rather than with solids are thus preferred. Chapter 4 delves into RQ 1 more in depth and the following research question results naturally from it.

Research Question 2 (RQ2): *What deformation techniques would aid a surface to reliably behave like a solid?*

This question can be understood as how to make surface deformations physically plausible. In the case of plasticine, volume preservation and mass conservation (Dewaele & Cani (2004)) are two of the most common features. Having a surface preserve its volume and smoothness under bulging and denting effects would be required.

Once a deformation type is chosen for imitating plasticine deformation, it needs to be integrated with the non-rigid registration algorithm. The reason for this is to use the advantages given by an intuitive deformation to find accurate correspondences between the source and target surfaces.

Many helpers are available for aiding the non-rigid registration. These include a template source surface, the physical properties of plasticine and a sparse set of landmarks. The question is then of finding the minimal set of helpers that would allow an accurate registration. The latter is equivalent to finding a match between a source and a target surface with a infinitesimal Hausdorff distance between them. Chapter 4 also explores this research question.

Research Question 3 (RQ3): *What is the minimum set of helpers necessary to match*

any Clay Jam character?

As a small Clay Jam database of plasticine characters is available, it is reasonable to start the investigation here. Firstly, the character animations within the Clay Jam context need to be analyzed. The goal is to find a reliable classification of these characters based on their geometric and animation types. The classification will then lead to determining the suitable helpers for each character class.

Simplified models can also be employed, to better understand the essence of plastic deformations. This also links in with the scale of deformation mentioned in the task analysis. If a match works at a small scale, it would be interesting to determine the minimal set of helpers that can extend the match to medium and large scale deformations.

To reiterate the meaning of helpers, they are priors for the deformation and non-rigid registration steps. Chapter 2 attempts a classification of handmade, plasticine models from the game Clay Jam. The resulting classification aids in choosing the necessary helpers that can aid a subsequent match or non-rigid registration.

Research Question 4 (RQ4): *What is the most suitable non-rigid registration technique, that can give accurate matches for any Clay Jam character, considering the available helpers?*

Out of the state-of-the art non-rigid registration algorithms, the ones that tackle Clay Jam character deformation types are of interest. As mentioned in the character animation classification section from the task analysis (section 1.2.1), characters can be composed of differently moving parts. Depending on the types of animation a character displays, the suitable registration technique for that model needs to be identified.

A desirable outcome for the project is an algorithm that can choose between a set of registration methods, depending on the detected or received helpers of the input models. This outcome proved to be too ambitious, however, and was only partially studied. Chapter 5 explores non-rigid registration techniques and how they can benefit from material and structure specific helpers.

Research Question 5 (RQ5): *Can the animation reconstruction algorithm (deformation, non-rigid registration and interpolation) be generalized to any type of character from the real world?*

In other words, given the character classification available from the Clay Jam data-set, can it be used to cover more types of characters outside of this set? Is the resulting classification thorough enough to aid in coming closer to a general non-rigid registration algorithm?

Given any character, with a set of priors from the user, like material type, a small set of landmarks and a template scan, it would be ideal for the algorithm to assign it to a class, depending on the type of animation it displays. More specific helpers can then be automatically added, like a skeleton or segmentation into component parts. Lastly, the suitable registration algorithm for that class of characters can then be used to get the most accurate match possible.

Chapter 3 proposes a general pipeline for reconstructing and refining stop motion animation for plasticine characters. Although only partially answering RQ 5, the method proposed in this chapter brings the literature one step closer to a generalized animation reconstruction technique.

1.4 Chapter Organization

The research questions arose at the beginning of the project. As it evolved, the research questions were answered in a different order. This can be seen as presenting an overview of the entire animation reconstruction pipeline (answering RQ3, RQ5) first and then examining the crucial component parts (answering RQ1, RQ2, RQ4) at a deeper level. The following chapters are organized as follows.

Chapter 2 first takes a look at stop motion animation and handmade characters from films and games. An overview of the Fat Pebble animation style is also presented. Clay Jam characters are then analyzed and a heuristic for character classification is proposed. Note that this chapter is light on technical contribution and literature review, as it is mostly based on empirical observations from the company.

Chapter 3 overviews available methods of animation reconstruction. A pipeline for reconstructing and refining stop motion animation is also presented. The focus is on plasticine characters from the game Clay Jam. The simplified version of the pipeline is then implemented, with its applications for the film and games industries.

The following two chapters, are in depth analyses of two of the pipeline components, virtual clay representation and deformation (chapter 4) and non-rigid registration (chapter

5). Chapter 4 is composed of an analysis of the physical properties of plasticine, a detailed literature review of how this material was represented in computer generated mediums and a proposed deformation method for virtual clay modelling. Two of the highlights of the chapter are a definition of localized stiffness and an analysis of how surface and volumetric deformation are connected.

Chapter 5 starts with a detailed literature review of non-rigid registration. Deformation scales and appropriate registration methods are explored. The current literature on the correspondence and trajectory problems is then tackled. Lastly a few techniques for enhancing a popular non-rigid registration method are presented. The goal is to show how reverse engineering plasticine deformation can help improve matches.

Chapter 6 is the conclusion where the project contributions and future work are summarized.

Chapter 2

Stop Motion Animation Character Classification

Stop motion animation is characterized by the unique look and feel of handmade materials. Characters are as versatile as the materials they are made from and do not shy away from imperfections like tears, fingerprints, scratches etc. These are rather signatures of the artist's style, rather than artefacts to be smoothed out like in computer generated models. It is virtually impossible to enumerate all stop motion animation character types. Thus the task of reconstructing the animations of such characters is one of high complexity.

In this chapter, research questions 3 and 5 are explored. In order to achieve a reliable animation reconstruction that can be generalized to as many physical characters as possible, a method of classifying these characters into a small set of categories is needed. These categories can then be linked to helpers (eg. landmarks, skeletons) that can aid the animation reconstruction process.

The first step of this chapter is to present what stop motion is and its uses in films and games. Next, handmade characters and the Fat Pebble animation style are described. The first contribution is then presented, which consists of a heuristic scoring system for classifying thirty Clay Jam characters.

Clay Jam characters are made out of plasticine and are composed of parts that display either an animation by skeleton or by shape. The proposed heuristic classified Clay Jam characters starting from these two types of animation. Three classes of characters were determined, based on the animation type and scale of their component parts.

The work in this chapter was based on empirical observations while working with the company and is light on technical contribution and literature review. Nevertheless, the resulting classification influenced the research on the animation reconstruction pipeline, deformation and non-rigid registration steps that followed.

2.1 A Short History of Stop Motion Animation

Stop motion animation is the traditional craft of giving life to handmade models. The unique look and feel of this art form is hard to reproduce with 3D computer generated techniques. This is due to the unexpected details that appear from frame to frame and to the sometimes choppy appearance of the character movement. Stop motion involves creating characters by hand and manually changing their pose for each frame of animation. Traditionally a photo is taken after each change is done on the models in the scene. These photos are then placed in a sequence to create the illusion of life.

2.1.1 Stop Motion and Film

Given its tedious nature, traditional stop motion has fewer film titles than hybrid and computer generated (CG) animation, but memorable and irreproducible pieces nevertheless. To mention only a few of the animation pioneers in this field: Ladislav Starevich, Ray Harryhausen, Willis O'Brien, Nick Park, Norman McLaren, Suzie Templeton who played a key role in establishing stop motion as a form of entertainment in its own right. Also, film directors that have an affinity for the art form include Wes Anderson with *Fantastic Mr. Fox* (2009), *Isle of Dogs* (2018), Tim Burton with *Corpse Bride* (2005), *Frankenweenie* (2012), Henry Selick with *The Nightmare Before Christmas* (1993), *Coraline* (2009).

The main qualities of stop motion animation are its photorealism and uniqueness, which means that each character pose, once created, is undoable. Depending on the materials the characters are made from, this can result in choppy transitions between frames, with details appearing and disappearing from the image. When plasticine or modelling clay was the preferred material, unexpected artefacts became part of the animation style. Titles include Aardman Studios' early work *Morph* (1977), *Wallace and Gromit: A Grand Day Out* (1989), *Wallace and Gromit: The Wrong Trousers* (1993), Otmar Gutmann's *Pingu* (1986 - 2006), Adam Elliot's *Mary and Max* (2009).



Figure 2-1: Stop motion productions that used plasticine characters: *Morph* (1977) (left) and *Wallace and Gromit: The Wrong Trousers* (1993) (middle) by Aardman Studios, *Pingu* (1986) (right) by Otmar Gutmann.

Hybrid approaches that combine stop motion and CG have been adopted to reduce artefacts and to allow faster production times, while still maintaining the desired handmade look. Some examples include productions by Laika Studios with *Coraline* (2009), *Para-Norman* (2012), *The Boxtrolls* (2014), *Kubo and the Two Strings* (2016). Aardman studios also incorporate CG elements in their work when manually creating the models is not feasible. One example is the sea in *The Pirates! In an Adventure with Scientists!* (2010), which was a computer generated backdrop for the physical puppets.



Figure 2-2: Hybrid stop motion productions that used puppets and CG: *Coraline* (2009) (left) and *Kubo and the Two Strings* (2016) (middle) by Laika Studios, *The Pirates! In an Adventure with Scientists!* (2010) (right) by Aardman Studios.

Computer generated productions that imitate the stop motion style were also made popular. Aardman and Dreamworks's collaboration created *Flushed Away* (2006), a 3D animation that imitates the look of plasticine models. Details like dents, fingerprints, bobbing heads, the design of the characters gave the production an authentic look. The smooth animation, however, reduced some of the effect expected with purely handmade work. Mindbenders' *The Duplicators* (2009) also created characters that mimic the look of modelling clay. Similarly to the first example, the smooth animation, due to the advanced interpolation techniques available in the CG industry, lessens the quality of the unexpected details.

Nevertheless, computer generated stop motion has its own niche, which has been made

popular in recent years by Warner Brothers’ *The Lego Movie* (2014) and the *Lego Batman Movie* (2017). The choppy aesthetic specific to stop motion is incorporated in these films to create humour. Characters are lego models, which makes their appearance and movement easier to manage.

These examples show that there is a demand for expanding stop motion further into the CG animation industry. The quality and attention to detail of handmade models is captivating for an audience. At the same time, the directability and efficiency of computer generated work would offer the former a less strenuous path for their craft.

2.1.2 Stop Motion and Games

Adding stop motion to the games industry results in an even narrower field of entertainment. Out of the few studios who use this art form in their games, none of them, to the extent of our knowledge, have full 3D character sequences acquired from the real world models. Photography is generally used to get the 2D visual content. Some of these games include claymation work like The Neverhood’s *The Neverhood* (1996), *Skullmonkeys* (1998), Trikster Games’ *Tanita: A Plasticine Dream* (2011), Pencil Test Studios’ *Armikrog* (2015). Other game styles include combining materials like clay, cardboard and paper in names like The Sleeping Machine’s *The Dream Machine* (2010), State of Play’s *Lumino City* (2014).

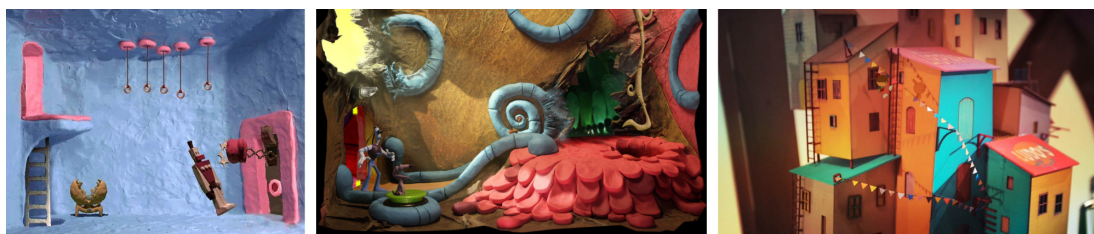


Figure 2-3: Stop motion animation games: *The Neverhood* (1996) (left) by The Neverhood, *Armikrog* (2015) (middle) by Pencil Test Studios, *Lumino City* (2014) (right) by State of Play.

In addition, photogrammetry is sometimes used to reproduce real world, 3D models on the computer. The resulting meshes, however, are usually skinned and animated by artists using a graphics tool such as Maya (Autodesk (2018b)) or 3DS Max (Autodesk (2018a)). Scanning human characters and adding motion capture movement to the resulting avatars is also popular. Although this preserves the original shape and textures, the qualities of motion are often lost or diminished. Some examples include: The Astronauts’ *The Vanishing of Ethan Carter* (2014), The Farm 51’s *Get Even* (2015).

2.1.3 Handmade Characters

Stephen Mottram, world renowned puppeteer, talks about what makes his craft compelling to an audience. The idea of giving life to an inanimate object creates a great conflict in the human mind, which compels it to investigate the phenomenon even further. To make a truly believable character, the audience need to feel the puppet in front of them has a mind of its own. In order to achieve this, puppeteers need to master the movement code specific to a particular type of character. This code is defined by the weight, size and speed of the component parts of the character and, in high quality animation, are unique to the personality of that character. In his series of movement analysis work, *Animata* (1990), Stephen Mottram shows how even ping pong balls can create a fascinating character, when the motion it conveys can stir emotion in the viewer.

Mastering creating emotion from motion of otherwise inanimate objects or drawings is the basis for memorable animated characters and storytelling. Disney were the pioneers of the 12 principles of animation that ensure the creation of such characters. The work of Disney's Nine Old Men rippled through the history of handmade and computer generated (CG) animation and produced entire manuals on what creates good character animation (Thomas & Johnston (1981), Williams (2009)). These principles apply to the world of stop motion as well, but are by no means the only guiding principles when it comes to this versatile art form. The way a character moves comes down to the skill of the artist and the model's structure.

When it comes to building stop motion characters or puppets, artists need to consider how the design and materials can influence the animation. Claymation models are amongst the most popular, since the modelling clay used in the process is affordable and easy to handle. Characters like Morph, Wallace and Gromit from Aardman Studios, Rich Webber's Purple and Brown or Otmar Gutmann's Pingu were preponderantly made of plasticine when first created. After productions became larger and harder to manage, more control was needed for multiple characters and settings. Soft plastics, cast in silicone moulds started to be used instead of clay to create puppets.

Georgina Hayns, the Head of Puppetry at Laika Studios, gives some insight into the puppet making process (Originals (2016)), which is becoming the standard in industry. Her work includes *Coraline* (2009), *ParaNorman* (2012), *The Boxtrolls* (2014). After a character is designed, it is sculpted in the sculpting department. The resulting model is then taken to the moulding department to create the silicone mould for different parts of the character's body. Separating a model into parts is common, since replacements

are needed in case a limb breaks during production. The moulds are then taken to the armature department which creates metal skeletons for the puppets. Joints should be flexible to allow the desired movement for the model.

The casting department then collects the armature and moulds to cast the character components into soft plastics and attach them to the skeleton. Faces are created separately and come in a variety of expressions. Hundreds of face expression models can be swapped during the making of a film. Laika's *ParaNorman* (2012) 3D printed the characters' facial expressions. Faces can also be sculpted, moulded and cast separately like in Wes Anderson's *The Isle of Dogs* (2017). Lastly, once the puppet is built, the art department paints, adds hair and costumes to their models. A team as large as 30 people can work on the creation of a single puppet (Originals (2016)).

2.1.4 Fat Pebble Animation

Fat Pebble, the company involved in the current research project, are specialized in stop motion animation games. In their most popular game, *Clay Jam*, characters are made out of plasticine or modelling clay and move on the screen as 2D sprites oriented towards the camera. The characters' shading and texturing give the illusion of 3D, although users do not have access to more than a planar view of the environment. The animation is stylized and kept at a few frames per second for each model. Characters are vividly coloured and come in a variety of shapes ranging from worms, tentacles, bipeds, quadrupeds, multiple arms, noses with legs to big, bulky and bearded characters. Lighting is kept soft and details like scratches, fingerprints, dust are part of the magic of the *Clay Jam* world (figure 2-4).

Next, a few details are given about *Clay Jam* characters and their movement as seen in the game. Most of the characters are made of differently coloured parts like eyes, eyelids, mouth, teeth, spots, noses, hands, fingers, feet, moustaches, hats, hairs, nails, spheres, propellers. The preferred modelling clay colours are orange, blue and red. Eyes are usually white and round with a small, black iris and occasional red or blue sockets around them. When a mouth exist, opening it usually reveals a sinusoidal tongue. Limb numbers range from one to around eight and it is common for limbs to have a smooth, sinusoidal aspect and motion.

Many of the character movements have a harmonic quality as they go back and forth. The frame rate is usually small, making it easier to load a large number of 2D sprites in the game. Limbs and tongues commonly deform in a swiveling, sinusoidal manner



Figure 2-4: Three plasticine characters from Clay Jam, Party Onion (left), Noodle (middle) and Bingo Winged Bat (right). Notice the choices of colour (orange, blue, red), soft shading and details, the eye style and general smooth, blobby appearance.

over a small number of frames. Some characters change their shape significantly during animation, due to inflating or deflating effects. Table 2.1 gives an overall view of the variety of character shapes and activities of Clay Jam characters.

Table 2.1: Clay Jam character shapes and activities.

Character Themes	Character Activities
pebble, sausage roll, hamburger, octopus, leeches, tube, washing machine, chicken, bearded elephant (no trunk), worms, nose, gingerbread man, bulldog, helicopter, star, insect, dragon	running, skiing, somersault, wobble from side to side, wagging tail, waving arms, advancing and retracting legs, shaking head, rolling, flexing muscles, open mouth, moving tongue

Given an overview of stop motion animation and, more specifically the claymation style of Fat Pebbles' game, Clay Jam, the next steps are to analyze and classify the corresponding characters. The belief behind the following sections is that finding a simple classification of the varied characters from the game, the reconstruction of their shape and movement can be simplified. This is because each class of characters is connected to a set of helpers that can guide the animation reconstruction (deformation, non-rigid registration and interpolation) procedure.

2.2 Classification of Clay Jam Characters

At the beginning of the project, a set of 30 Clay Jam characters were available. The small data set came in the form of 2D images as most of the physical models had been

lost. This was due to models being damaged or destroyed after remaining unused for a long period of time. The full spectrum of Clay Jam animation could not be covered by only focusing on the small number of physical models available, since the range of shapes and activities varied for each character (table 2.1).

A method was needed to simplify the character types to facilitate the animation reconstruction process. By finding a small set of classes that include all 30 Clay Jam characters, generalization of animation reconstruction becomes possible for more types of models. The heuristic that emerged from personal observation at the company, made use of the 2D images available. The heuristic is a scoring system for classifying all thirty Clay Jam characters by animation types and shape mapping techniques. The classification by animation type aided in identifying the helpers required to achieve suitable shape mappings, which can be understood as non-rigid registrations for this work.

2.2.1 Method

The available set of thirty Clay Jam character images was analyzed. A classification soon emerged after observing their geometric and animation types. The geometric classification of the characters is versatile, ranging from simple shapes (eg. sphere, cylinder, torus) to limbed shapes (figure 2-5). The animation based classification is much more concise (figure 2-6), and thus preferred for defining the heuristic from this chapter.

Characters seem to be guided either by a skeletal motion or a shape based deformation (eg. squash and stretch). The *Animation by Skeleton* class includes movement that can be guided by a *Discrete Skeleton*, where limbs are formed of piecewise rigid shapes or a *Spline Skeleton*, where skeletal deformation is continuous. The *Animation by Shape* class includes models that are *Almost Rigid* during the animation and *Non-rigid* models with a free-form deformation style.

The animation classes detected are linked to potential helpers that can aid the deformation and registration of the corresponding character surface scans. The *Animation by Skeleton* characters can be guided by skeletons extracted from the surfaces (Au et al. (2008), Tagliasacchi et al. (2016)) or placed manually by the user. Segmentation can also be used to simplify the registration by matching piecewise rigid regions (Chang & Zwicker (2008)).

The *Animation by Shape* class can be guided by correspondences detected from salient

features of the surface (Gal & Cohen-Or (2006), Ovsjanikov et al. (2012)) or placed manually, as landmarks, by the user. It is well known that the quality of the registration between surfaces is directly linked to the number of reliable correspondences (Li (2010)).

Salient features are known in literature as groups of descriptors that define a shape locally (Gal & Cohen-Or (2006)). These descriptors heavily rely on surface curvature and its changes. Gaussian curvature and mean curvature are usually approximated on a discrete surface (mesh) by utilizing techniques like the Laplace-Beltrami operator (Ovsjanikov et al. (2012)).

A character can be composed of parts that belong to different animation classes. For example, Bingo Winged Bat from figure 2-4 has an almost rigid head movement, discrete skeleton arms and non-rigid wings. The classification by animation type is thus necessary, but not sufficient to completely define the requirements for the animation reconstruction of Clay Jam characters. The scale of deformation and the properties of the material the characters are made from are also needed.

Given a source and a target 3D mesh scan, the deformation scale between them is measured in terms of their overlap. Considering the source and target globally aligned, the overlap is defined as the ratio between the volume of intersection for the two meshes and the source mesh volume. One implementation for calculating the overlap would be to represent the available scans as implicit surfaces. The implicit representation of the source and target meshes could then allow intersection operations and volume calculations (Wyvill et al. (1999), Stanculescu et al. (2011)).

The scale of deformation is thus allocated as follows. Small deformation is given by a 80-100% overlap, an intermediate deformation by a 40-80% overlap and a large deformation by a 0-40% overlap of the intersection and source volumes. Since the available data mostly consisted of 2D images of the physical models, a visual estimation of the overlapping volumes was adopted for all thirty Clay Jam characters. This approach, although imprecise, was deemed sufficient for a heuristic scoring system. The goal was to determine tendencies of Clay Jam character animation, rather than a precise classification.

The material properties of a model are linked to the amount of stretch its surface undergoes from one frame to the next. An exploration of the connection between surface deformation and the corresponding solid deformation is needed in order to define the plasticity of a model. The physics of the material, which in the Clay Jam case is plasticine, and ways of representing it on the computer are described in detail in chapter 4.

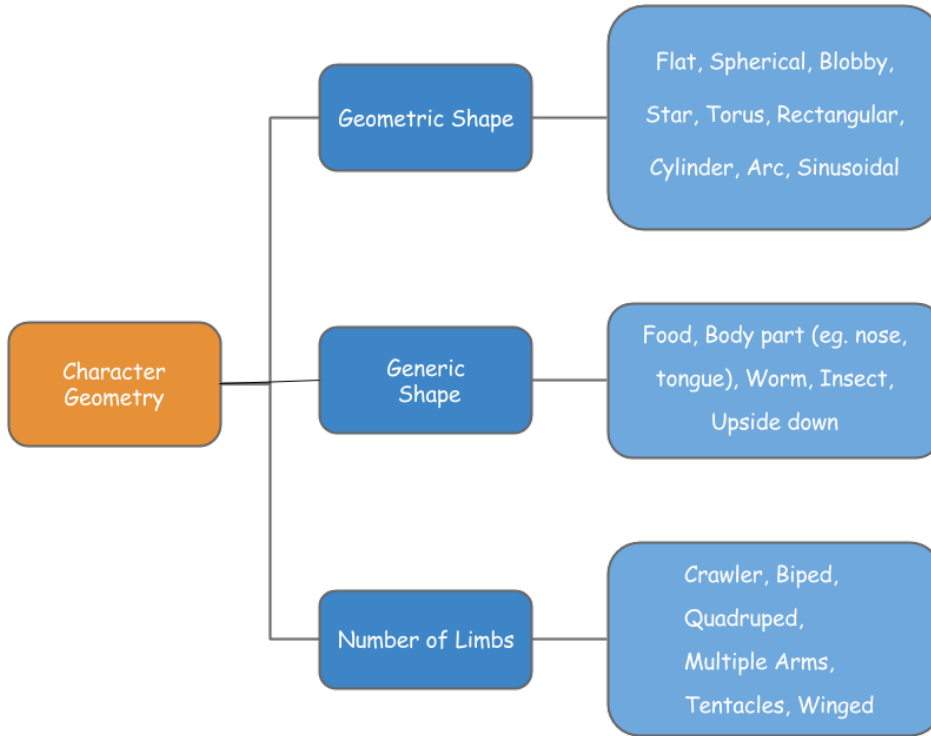


Figure 2-5: Classification of Clay Jam characters by their geometric shapes.

After detecting the animation classes (figure 2-6), scales of deformation and material properties for each character, a heuristic scoring system was created to classify each Clay Jam character. The classification is performed against the shape mapping classes: *Globally Uniform*, *Globally Nonuniform*, *Extrinsic Metric*, *Intrinsic Metric* proposed by Li & Iyengar (2014). This was done in order to determine which types of shape mapping or, in our case, registration algorithms would best suit each animation class. A thorough review of non-rigid registration methods can be seen in chapter 5.

Globally Uniform and *Globally Nonuniform*, or simply uniform and nonuniform, shape mapping methods are defined by the number of degrees of freedom when considering the deformation between a source and a target mesh. Uniform shape mapping requires a small, constant number of global transformations to align the source to the target. Non-uniform shape mapping implies a dense set of transformations, that depend on the structure of the meshes (Li & Iyengar (2014)).

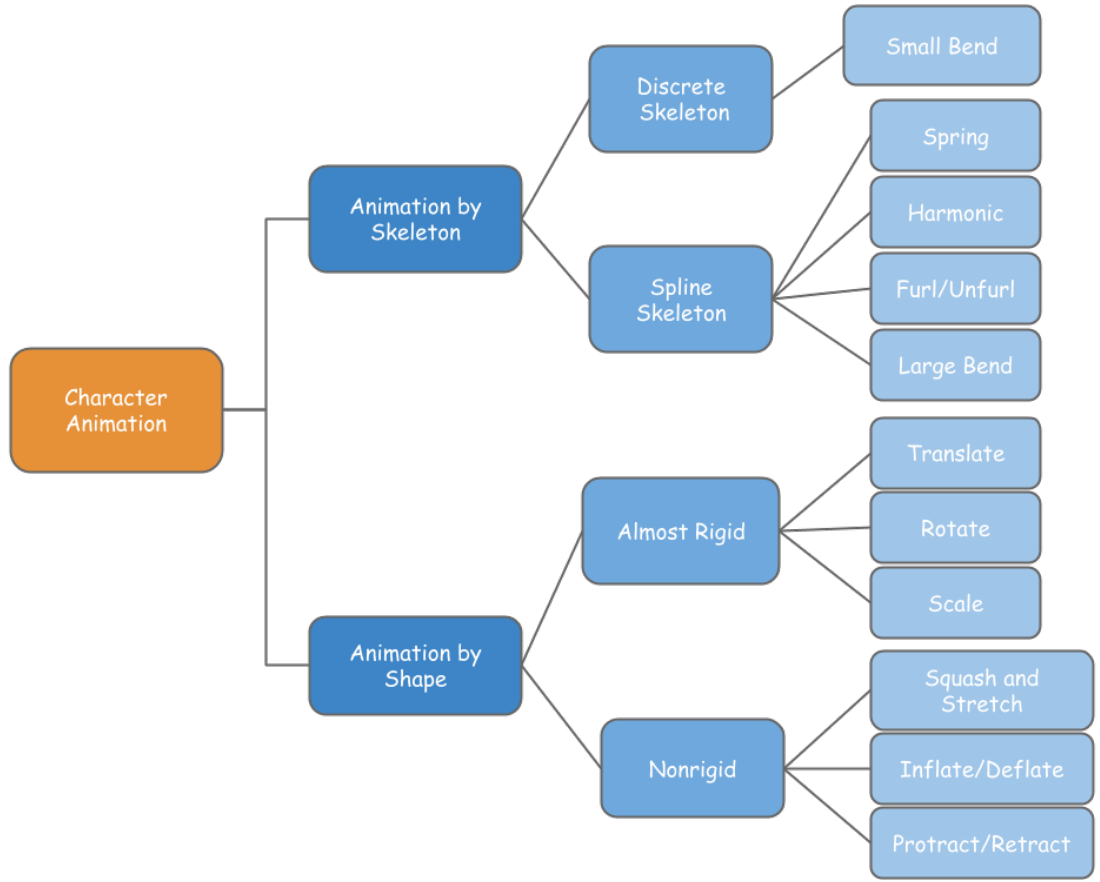


Figure 2-6: Classification of Clay Jam characters by their animation type.

Extrinsic Metric, or simply extrinsic, shape mapping usually involves a deformation vector field between the source and the target mesh (Wand et al. (2009)). The mesh properties are minimally used, normally to choose correspondences or to minimize stretching and bending energies of the surface. Algorithms from this category thrive with small, physically plausible deformations.

Intrinsic Metric, or simply intrinsic, shape mapping, delves into larger deformations. The shape of the mesh is taken into account and properties like surface curvature (Ovsjanikov et al. (2012)) are made use of for registration. Also, techniques like parametrization and embeddings (Tevs et al. (2012)) are common to simplify complex shapes into more manageable ones.

Heuristic Rules

The rules for the proposed heuristic are split into two sections: generic mesh and

individual character rules. Given an animation class (by skeleton or by shape), the generic mesh rules consider each shape mapping class (uniform, nonuniform, extrinsic, intrinsic) in turn. For each shape mapping class, the rules evaluate how well the corresponding algorithms could register meshes undergoing the given animation type.

For example, consider the animation class to be *Animation by Skeleton* and, more specifically, *Discrete Skeleton* animation, and the shape mapping class to be *Globally Uniform*. The first generic mesh rule (see below) questions whether uniform shape mapping algorithms are sufficient for small, intermediate or large skeletal deformations. Since uniform methods do not cover large deformations, a point is subtracted from the starting score.

The starting score before applying the generic mesh rules is 10. Each rule induces a penalty of -1 or -2 points to this value, depending on its possible consequences to the final animation reconstruction process. Table 2.2 contains the generic mesh scores for all the combinations of animation and shape mapping classes. The individual character rules can then be used to estimate a score for each character, given that the latter is composed of a set of generic meshes.

A Clay Jam character can thus be divided into a set of generic mesh segments, with each one occupying a proportion of the whole volume. The segmentation attempted is one by motion and is estimated visually, as are the volume ratios of these segments with respect to the whole mesh. These ratios are used as weights in a weighted average sum of the character's score (equation 2.1).

Segmentation by motion was chosen over geometric segmentation, due to the deformation displayed by the models. Characters are created out of segments that move differently, but that are not independent from the whole body. Segmentation by motion can be defined as the delimitation of parts that have different behaviours from their adjacent parts. Figure 2-7 shows the difference between segmentation by geometry and segmentation by motion.

The score per character (table 2.5) reveals how each shape mapping class could aid in the registration of that particular character. The two sets of rules and the heuristic equation are presented in the next paragraphs. The resulting Clay Jam character scores and the subsequent classification that emerged from them can be found in section 2.2.2.

Generic Mesh Rules

If shape mapping method:

- Is not useful for one scale of deformation (small, intermediate, large) implied by

animation class \Rightarrow -1 point/scale type

- Has problems with detail preservation \Rightarrow -1 point
- Has problems with volume preservation \Rightarrow -1 point
- Is nonlinear \Rightarrow -2 point
- When animation class is *Animation by Skeleton*
 - Shape mapping class can benefit from skeleton \Rightarrow +1 point
 - Shape mapping class is almost invariant under skeleton constraints \Rightarrow -1 point

Table 2.2: Scores after applying Generic Mesh Rules per shape mapping class, for each animation class.

Shape Mapping (Objective Function)				
Animation class	Globally Uniform	Globally Nonuniform	Extrinsic Metric	Intrinsic Metric
Animation by Skeleton				
Discrete skeleton \rightarrow Small bend	7	8	7	8
Spline skeleton \rightarrow Spring	6	8	7	8
Spline skeleton \rightarrow Harmonic (sine wave)	6	8	7	8
Spline skeleton \rightarrow Furl/Unfurl	6	8	7	7
Spline skeleton \rightarrow Large Bend	6	8	7	7
Animation by Shape				
Almost Rigid \rightarrow Rotate	9	7	9	8
Almost Rigid \rightarrow Translate	9	7	9	8
Almost Rigid \rightarrow Scale	9	7	9	8
Non-rigid Squash and Stretch	6	7	7	7
Non-rigid Inflate/Deflate	7	7	7	8
Non-rigid Protract/Retract	6	7	7	7

Individual Character Rules

For each mesh segment:

- Calculate how frequent the deformation scale (small, intermediate, large) not covered by shape mapping class appears in the transitions between frames (eg. intermediate appears 2 out of 9 frame transitions and is not covered $\Rightarrow -2/9 = -0.2$ points)
- When shape mapping class can benefit from salient feature constraints:
 - Mesh segment does not have at least 3 salient points $\Rightarrow -1$ point
 - Mesh segment has ≥ 3 salient points $\Rightarrow +1$ point
- Score is weighted by the volume ratio of mesh segment with respect to the whole mesh
- Mesh segment score is added to total character score using equation (2.1).

The final score for each Clay Jam character is calculated using the weighted average sum

$$\mathbb{S}_{char} = \sum_{\#segs} w_{vol} * (\mathbb{S}_{generic} - w_{SODef} + \mathbb{K}_{features}) \quad (2.1)$$

where w_{vol} is the volume ratio of a mesh segment, whose deformation belongs to a certain animation class, $\mathbb{S}_{generic}$ is the score obtained for that class from the generic mesh set of rules, w_{SODef} is the scale of deformation ratio (table 2.4) and $\mathbb{K}_{features}$ is a constant related to whether the mesh can benefit from salient feature constraints. This heuristic is meant to reveal the tendencies of Clay Jam characters and is not to be understood as an accurate mathematical measurement.

In order to allow an accurate animation reconstruction (deformation, non-rigid registration, interpolation) of the Clay Jam characters, it is useful to understand their intuitive segmentation by motion and the scale of their deformation. This is useful for the helper addition stage (section 3.2.4) and for the subsequent animation reconstruction process. The latter is aided by the output of the experimental work from the next section. The categories resulting from the heuristic classification are, in fact, a simplification of Clay Jam characters into three categories, which are defined by the deformation of their component segments.

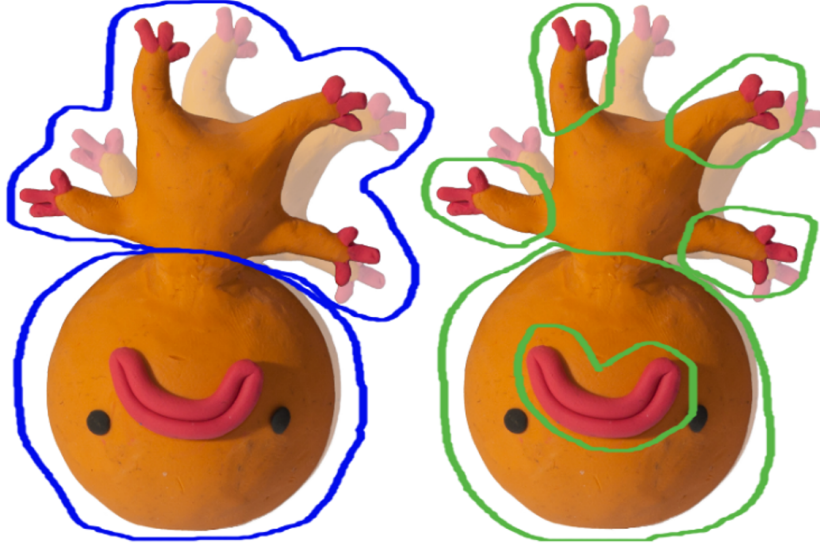


Figure 2-7: Animation segments (blue) - the head moves rigidly, while the rest of the body moves non-rigidly and geometric segments (green) - character is split into its distinct geometric shapes, with the boundary usually representing minimal mesh curvature.

2.2.2 Experiments and Results

In this section the classification heuristic was applied to the available Clay Jam characters. Party Onion (figures 2-8, 2-9, 2-10) was chosen as an example for the steps taken in the classification. Party Onion is a character whose body undergoes non-rigid (squash and stretch) deformation, its tentacle has a spline skeleton deformation and the eyes have an almost rigid transformation. The animation spans over a set of 6 frames, which implies 5 transitions between frames.

Table 2.3: Animation class per mesh segment and the volume ratio of that segment with respect to the whole character mesh volume.

Character Animation Class	Mesh Segment	Volume Ratios
By Shape → Non-rigid → Squash and Stretch	Body	0.6
By Skeleton → Spline Skeleton → Furl/Unfurl	Tentacle	0.3
By Shape → Almost Rigid → Translate	Eyes	0.1

Table 2.3 shows the animation class per mesh segment and the volume ratio of that segment, with respect to the whole character volume. Table 2.4 deals with the scale of deformation ratios for each mesh segment. The latter reveals how frequently a

Table 2.4: Scale of deformation per mesh segment and the frequency with which that deformation type appears in the set of available frames gives the scale of deformation ratio.

Scale of Deformation	Mesh Segment	SODef Ratios
Small	Body	$\frac{3}{5} = 0.6$
	Tentacle	$\frac{0}{5} = 0$
	Eyes	$\frac{3}{5} = 0.6$
Intermediate	Body	$\frac{2}{5} = 0.4$
	Tentacle	$\frac{2}{5} = 0.4$
	Eyes	$\frac{1}{5} = 0.2$
Large	Body	$\frac{0}{5} = 0$
	Tentacle	$\frac{3}{5} = 0.6$
	Eyes	$\frac{1}{5} = 0.2$

Table 2.5: After applying Individual Character Rules per mesh segment and adding up all the scores, the character scores per shape mapping category are obtained. Here are 5 examples of character scores.

Character	Globally Uniform	Globally Nonuniform	Extrinsic Metric	Intrinsic Metric
Party Onion	4,92	6,5	6,2	6,3
Plipa	6,33	7,3	6,51	6,9
Tick	5,72	6,4	7	6,7
Blob Fish	7,1	7,6	7,52	8,6
Hellidropter	9,75	8	10	9

part of the character undergoes small, intermediate or large deformations during the transitions between frames. Figures 2-9, 2-8 and 2-10 show Party Onion in a few transitions between frames, with different scales of deformation for each mesh segment.

The score for the Party Onion character is calculated using the weighted average sum from equation (2.1) and the values from the aforementioned tables. The generic mesh score for each animation class are taken from table 2.2. Its values were calculated beforehand, using the first set of rules. The volume and scale of deformation ratios are taken from tables 2.3 and 2.4, while the salient feature constraint value is -1 per



Figure 2-8: Large deformation of tentacle, small deformation of body over consecutive frames.



Figure 2-9: Intermediate deformation of tentacle, small deformation of body over consecutive frames.



Figure 2-10: Large deformation of tentacle, intermediate deformation of body between consecutive frames.

segment, since there are less than 3 salient points on each one. Table 2.5 shows the resulting individual character score for Party Onion, along with four other examples.

After repeating this scoring procedure for each of the 30 Clay Jam characters, their tendencies emerged. Figures 2-11 and 2-12 are examples of 15 characters and their scores in the *Globally Uniform* versus *Nonuniform* and *Extrinsic* versus *Intrinsic Metric* shape mapping classes respectively. Party Onion scored higher in the *Globally Nonuniform* methods, since its normalized score is higher than the *Globally Uniform* one.

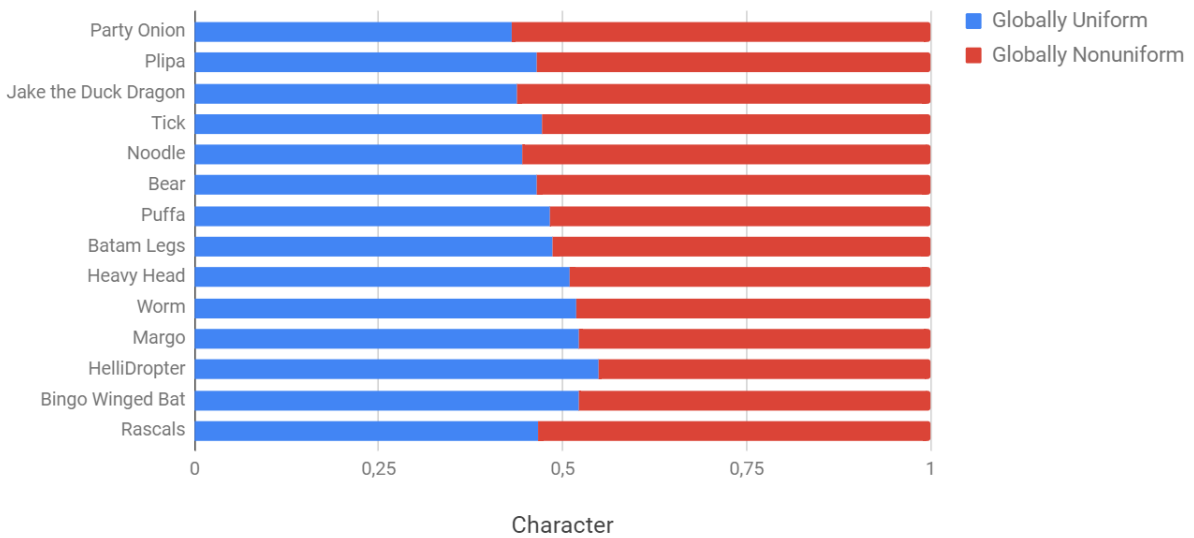


Figure 2-11: *Globally Uniform* versus *Globally Nonuniform* character scores for 15 characters.

The two highest scores from the four shape mapping classes were selected for each character. These scores were then connected to the character’s dominant animation class. An animation class is called *dominant* if it is displayed by a mesh segment of volume ratio higher than 0.5. Characters were finally clustered via their dominant animation classes, with the highest shape mapping scores attached (figures 2-13 and 2-14).

Three possible combinations (C1, C2, C3) of shape mapping classes emerged from figures 2-13 and 2-14. They are presented bellow as bullet points, followed by a more detailed description. Figures 2-15, 2-16 and 2-17 offer the visual representation of these three categories of Clay Jam characters.

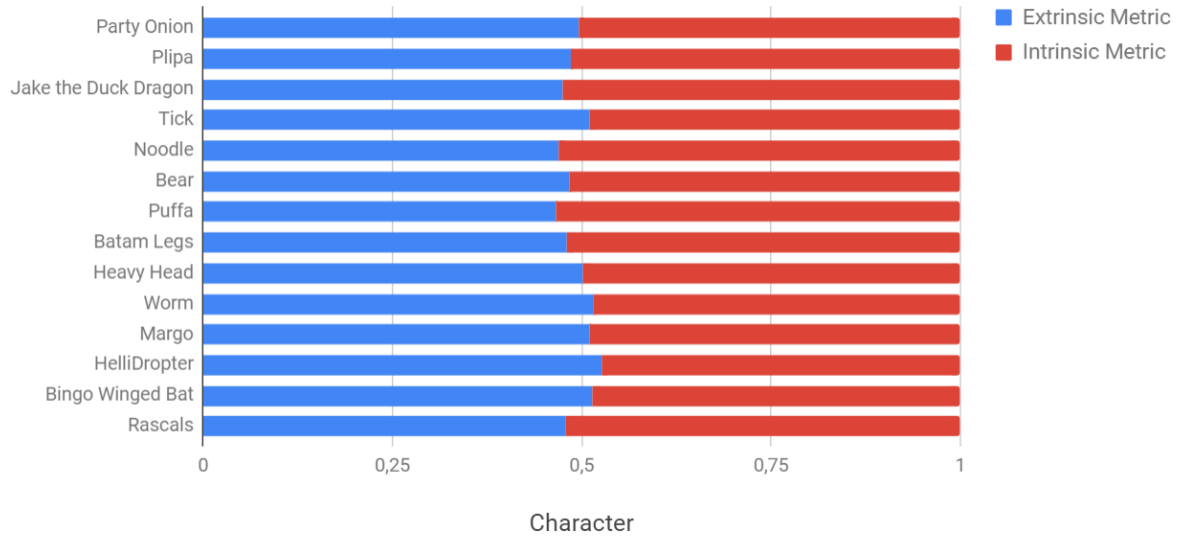


Figure 2-12: *Extrinsic Metric* versus *Intrinsic Metric* character scores for 15 characters.

Character	Main Animation Category	First High Score - Shape Mapping Category	Value	Second High Score - Shape Mapping Category	Value
HelliDropter	Almost Rigid	Extrinsic Metric	10	Globally Uniform	9,75
RotOcopus	Almost Rigid	Extrinsic Metric	10	Globally Uniform	9
Washing Machine	Almost Rigid	Extrinsic Metric	9,8	Globally Uniform	9,7
Monster Perilous	Almost Rigid	Extrinsic Metric	9,6	Globally Uniform	9,502
Margo	Almost Rigid	Extrinsic Metric	9,4	Globally Uniform	9,1
Sausage Roll	Almost Rigid	Extrinsic Metric	9,2	Intrinsic Metric	8,908
Ski Leech	Almost Rigid	Extrinsic Metric	8,95	Intrinsic Metric	8,78
Big Jigger	Almost Rigid	Extrinsic Metric	8,8	Globally Uniform	8,55
Worm	Almost Rigid	Extrinsic Metric	8,789	Intrinsic Metric	8,25
Bung it Billy	Almost Rigid	Extrinsic Metric	8,5	Globally Uniform	8,24
Puffa	Nonrigid	Intrinsic Metric	8,6	Globally Nonuniform	7,6
Bingo Winged Bat	Nonrigid	Extrinsic Metric	8,542	Intrinsic Metric	8,1
Nobody Nose	Nonrigid (*)	Extrinsic Metric	8,4	Intrinsic Metric	7,8
Poop Tube	Nonrigid	Globally Nonuniform	8,15	Intrinsic Metric	8,15
Ringo	Nonrigid	Intrinsic Metric	7,2	Globally Nonuniform	7,1
Tick	Nonrigid	Extrinsic Metric	7	Intrinsic Metric	6,7
Ground Mouth	Nonrigid	Globally Nonuniform	6,4	Intrinsic Metric	6,4

Figure 2-13: Rigid and non-rigid animation classes and the two highest scoring shape mapping classes.

- **C1: Globally Nonuniform - Intrinsic Metric**
 - Animation by Shape → Non-rigid.
 - Animation by Skeleton → Discrete/Spline Skeleton.
- **C2: Globally Uniform - Extrinsic Metric**
 - Animation by Shape → Almost Rigid
- **C3: Extrinsic Metric - Intrinsic Metric**

Character	Main Animation Category	First High Score - Shape Mapping Category	Value	Second High Score - Shape Mapping Category	Value
Heavy Head	Discrete Skeleton	Extrinsic Metric	9.03	Intrinsic Metric	9
Wobble Chicken	Discrete Skeleton	Extrinsic Metric	8.8	Intrinsic Metric	8.8
Wonder Beard	Discrete Skeleton	Globally Nonuniform	8.6	Intrinsic Metric	8.6
Bear	Discrete Skeleton	Intrinsic Metric	8.25	Globally Nonuniform	8.1
Batam Legs	Discrete Skeleton	Intrinsic Metric	7.8	Globally Nonuniform	7.4
Rascals	Discrete Skeleton	Intrinsic Metric	7.4	Globally Nonuniform	7.2
Slinky	Discrete Skeleton	Intrinsic Metric	7.3	Globally Nonuniform	7.15
Wibbly Dabbly	Spline Skeleton (*)	Extrinsic Metric	8.4	Intrinsic Metric	8.1
Jake the Duck Dragon	Spline Skeleton	Intrinsic Metric	7.4	Globally Nonuniform	7.15
Psych Sid	Spline Skeleton	Intrinsic Metric	7.4	Globally Nonuniform	7.1
Plipa	Spline Skeleton	Globally Nonuniform	7.3	Intrinsic Metric	6.9
Noodle	Spline Skeleton	Intrinsic Metric	7	Globally Nonuniform	6.7
Party Onion	Spline Skeleton	Globally Nonuniform	6.5	Intrinsic Metric	6.3

Figure 2-14: Discrete and Spline skeleton animation classes and the two highest scoring shape mapping classes.

- Animation by Shape → Almost Rigid/Non-rigid
- Animation by Skeleton → Discrete/Spline Skeleton

The **C1: Globally Nonuniform - Intrinsic Metric** category contains 14 characters out of 30 (46%) that undergo larger deformations. This is reflected in the animation classes associated with both the *Globally Nonuniform* and the *Intrinsic Metric* shape mapping classes. The majority of characters in C1 display non-rigid deformation (eg. squash and stretch) or skeletal deformation (discrete or spline skeleton). This implies that intrinsic information about the character mesh (eg. curvature, skeleton, feature descriptors) might be required to allow a good animation reconstruction of such models.

The **C2: Globally Uniform - Extrinsic Metric** category contains 7 characters out of 30 (23%) with small deformations. The almost rigid animation class appeared for all the *Globally Uniform* and *Extrinsic Metric* combinations. Since the number of degrees of freedom is small, models from this category are assumed to register well given a sparse set of landmarks.

The **C3: Extrinsic Metric - Intrinsic Metric** category contains 9 characters out of 30 (31%) with half of the mesh displaying small deformations, while the other half undergoing large deformations. Heavy Head from figure 2-7 is a good example from C3. Its large head is almost still, while the rest of the body sways from side to side with the help of discrete skeleton animation.

The allocation of characters into the C1, C2, C3 categories simplifies the types of deformation that can be expected from any Clay Jam model. It also reveals that a character can benefit from more than one type of shape mapping class, as it can be composed of mesh segments with different animation types. Characters can gravitate towards more

rigid deformations if, for example, their dominant parts are in the almost rigid class, or non-rigid deformations if the dominant parts display large skeletal deformations.

This character analysis was created in order to better understand and simplify the animation displayed by Clay Jam characters. The resulting heuristic does not give an accurate classification, as more testing is required to determine its robustness. Nevertheless, the results reduce the directions of investigation when considering the animation reconstruction of such characters.

2.2.3 Discussion

A heuristic was presented for classifying characters based on the type of animation their component parts display. Once the class has been established for a character, it can be assigned the corresponding helpers for the registration step, which is essential in the animation reconstruction pipeline.

The **C1: Globally Nonuniform - Intrinsic Metric** class has character animations with predominantly large deformations. Guidance from a skeleton and feature points might be required for a good registration between consecutive scans. The **C2: Globally Uniform - Extrinsic Metric** has characters with almost rigid deformations. It is expected that a sparse set of landmarks would be sufficient to guide the registration. Lastly, the **C3: Extrinsic Metric - Intrinsic Metric** present a combination of large and small deformations. A segmentation procedure can delimit the different character parts that display these types of animation.

The heuristic can be used to classify any type of character from the game Clay Jam and establish a set of helpers that can guide the subsequent animation reconstruction. These results partially answer research questions 3. However, a more robust method of testing the heuristic is needed in order to see whether it can be adapted to any type of stop motion character (RQ5). More investigation is needed on the available methods of classifying animation characters to be able to produce a pertinent classification. The following chapter presents a pipeline for reconstructing a character shapes and their deformations, considering the obtained classification.

Nonrigid



Discrete Skeleton



Spline Skeleton

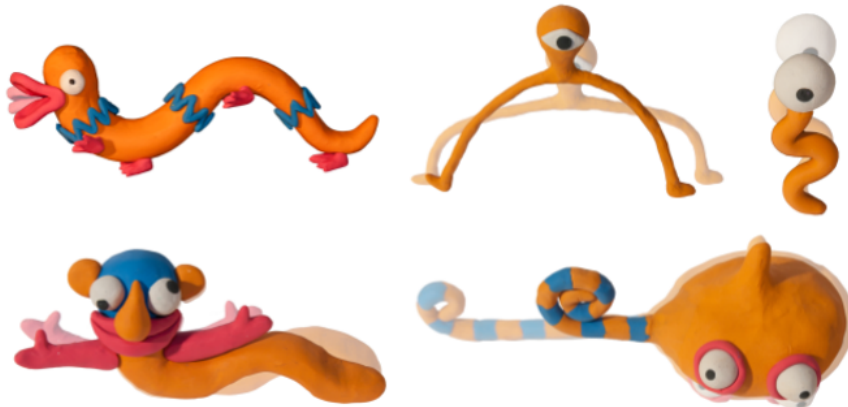


Figure 2-15: C1: Globally Nonuniform - Intrinsic Metric (46%) characters from non-rigid, discrete and spline skeleton animation classes.

Almost Rigid

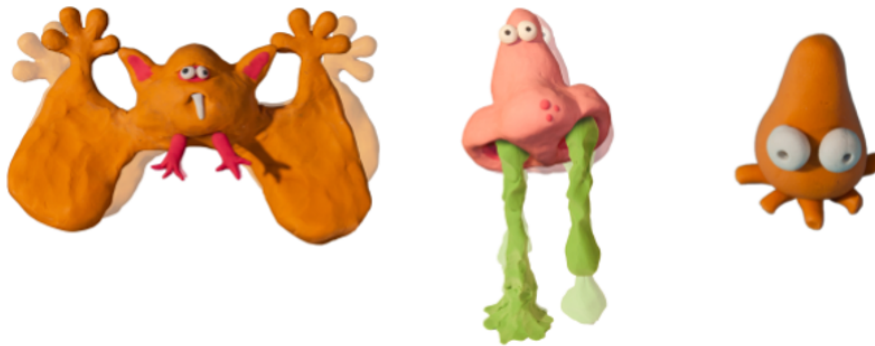


Figure 2-16: C2: Globally Uniform - Extrinsic Metric (23%) characters from the almost rigid animation class.

Almost Rigid



Nonrigid



Discrete and Spline Skeleton



Figure 2-17: C3: Extrinsic Metric - Intrinsic Metric (31%) characters from almost rigid, non-rigid, discrete and spline skeleton animation classes.

Chapter 3

E-StopMotion: Stop Motion Animation Reconstruction Pipeline

In the previous chapter a heuristic was presented for classifying Clay Jam characters. After understanding that characters are composed of mesh segments which can deform by shape or by the guidance of a skeleton, reconstructing some of these characters in various *key poses* is the next step.

In a traditional stop motion animation setting the artist’s task can be overwhelming as he has to reshape a character into hundreds of poses to obtain just a few seconds of animation. Similarly, physical models of the character in different shapes or facial expressions need to be individually modeled by the artist, which can take very long periods of time. The results of the animation are usually applied in 2D mediums like films or platform games. Character features that took a lot of effort to create thus remain unseen.

In this chapter the focus is on answering research question 5 on whether a generalized animation reconstruction algorithm can be designed so that any type of physical character can benefit from it. The literature on animation reconstruction is first described, then the E-StopMotion pipeline method is presented, followed by the technical details, experiments and applications of a simplified implementation of it. Chapters 4 and 5 then provide enhancements of two components of the pipeline that can be added to the implementation in the future.

The contribution of this chapter is E-StopMotion, a stop motion animation digitization pipeline. Its role is to take a small set of scans of a plasticine character in different *key poses* and return the clean meshes for those poses, with the same connectivity, and a sequence of physically plausible in-between shapes. To the extent of the author’s knowledge there are no pipelines in the graphics industry for digitizing 3D stop motion animation. Handmade characters are usually scanned and rigged or only the 2D version of the animation is digitized and manipulated by the artist.

The focus is on plasticine characters from the game Clay Jam, but it is shown that applications are for both the games and film industries. The idea is to reconstruct the original character animation and also allow tools for enhancing it. If the artist desires to change the animation or add more in-between poses for the character, it is tedious to do so with the original handmade model. By digitizing a few *key poses* and applying deformation, registration and interpolation techniques we construct a pipeline for bringing a stop motion character into the virtual world and extending its original animation in 3D.



Figure 3-1: Blob Fish *key poses* used in the Clay Jam animations. No other in-between poses are used in the game. Notice that the difference between shapes can differ significantly from frame to frame.

3.1 Animation Reconstruction

Animation reconstruction aims at capturing and constructing animated content from moving objects in the physical world (Li (2010)). This involves capturing data from a deforming model as it evolves over time. The available data for these scenarios is usually dense and the deformations between frames are small, making it easier to find reliable correspondences.

Common approaches make use of a template shape, which is fitted to the point clouds at each frame of animation (Süßmuth et al. (2008), Li et al. (2009)). Although this

approach offers a significant prior, it can restrict the deformations allowed and impede topological changes as argued in Tevs et al. (2012).

Alternative approaches have been proposed to reconstruct animation without a template shape. Tevs et al. (2012) propose a cartography inspired method of reconstructing a shape and its movement from partial scans and by imposing isometric deformations. This restriction would be an impediment in the current case, since plasticine can deform non-isometrically and irreversibly.

Wand et al. (2009) present an efficient pipeline for animation reconstruction, which involves the minimization of an energy term considering distances between data points, acceleration and velocity, volume change and surface smoothness. Their method is complex to implement, however, and only works accurately for small deformations between consecutive frames.

Sharf et al. (2008) reconstruct motion by making use of a 3D grid that embodies the whole character. Although volume preservation is a desirable feature for plasticine models, the use of a volumetric model for reconstructing animation is time consuming and needs an expensive rendering mechanism for visualization.

When considering stop motion animation, there isn't a viable procedure of continuously recording the character animation. The artist needs to intervene at every frame and change the shape or pose of the character. When the frame rate is small, these shapes can sometimes differ significantly (figure 3-1). The animation reconstruction then becomes more a problem of non-rigid registration and interpolation between scans. Non-rigid registration techniques are covered in chapter 5.

Interpolation is used when two or more shapes of a model are available. When using meshes, the connectivity between those shapes needs to be identical. In case the number of vertices and faces differs between the shapes, it is common to use cross-parametrization techniques to ensure the same connectivity (Kraevoy & Sheffer (2004)). In either case a one-to-one correspondence between all the mesh vertices is established, turning interpolation into a trajectory problem (Xu et al. (2005)).

The constraints imposed on the deformation path of the mesh can give a variety of in-between shapes. Since the physical models from Clay Jam are made of modelling clay, it is expected for their deformation to flow in a physically plausible manner. More information on representing and deforming plasticine in the virtual environment, also known as manipulating virtual clay, can be found in chapter 4.

Interpolation comes in many forms, from linear shape interpolation of vertices, to

Poisson based interpolation of the gradient field (Xu et al. (2005)), to mesh inverse kinematics (Sumner et al. (2005)), example based mesh deformations or morph target deformations (Fröhlich & Botsch (2011), Wampler (2016)) and data driven interpolation (Allen et al. (2003), Gao et al. (2016)). Shape preserving (Alexa et al. (2000)) and volume preserving (Eisenberger et al. (2018)) interpolation methods were considered, since they connect to the physical properties of plasticine. The experiments in this chapter were related to the former property by adopting an as-rigid-as-possible interpolation technique (Alexa et al. (2000), Sorkine & Alexa (2007)). The idea of considering individual elements of the surface rigid can also link to the surface tension property that plasticine displays (Dewaele & Cani (2004)).

Alexa et al. (2000) propose an algorithm that allows the interpolation between two sets of meshes using triangles in 2D or tetrahedra in 3D by minimizing a quadratic error function. They account for translation by fixing a vertex using linear interpolation, which, depending on the vertex chosen, can give different results. Building on the work of Alexa et al. (2000), Liu et al. (2011) address this problem by generating a set of surface tetrahedra on the fly as opposed to needing a fixed initial and end volumetric mesh. No vertex needs to be fixed, as translation is taken into account in their minimization function.

Baxter et al. (2008) provides an improvement on several algorithms for interpolating between 2D shapes by incorporating normal equations. They provide a mathematically equivalent solution to minimising the quadratic error. A vertex does not need to be fixed and more control over the vertex trajectories is enabled by using Lagrange multipliers. Their solution does not extend into 3D, and for it to be mathematically equivalent to the method of Alexa et al. (2000), it would require the use of volumetric tetrahedral meshes.

In the proposed approach the works of Baxter et al. (2008) and Liu et al. (2011) are extended. Normal equations and dynamically generated tetrahedra are used to create as-rigid-as-possible deformations between shapes. This interpolation step is the penultimate component of E-StopMotion, the stop motion animation reconstruction and enhancement pipeline. This pipeline is suitable when only sparse *key poses* of characters are available. It combines third party tools for scanning and cleaning the physical characters with deformation, non-rigid registration and interpolation techniques that take the properties of plasticine into account. The full theoretical E-StopMotion pipeline is first described and then the simplified version is implemented.

3.2 E-StopMotion: Full Pipeline Method

In chapter 2 it was mentioned that stop motion animation characters are posed by hand for each frame (figure 3-2). It is also common for these poses to be modeled separately, when a particular shape is desired per frame. The results are usually a sequence of images, which are combined further to create a 2D animation (Purves (2016)). Compared to digital animation based on computer graphics (CG) techniques, the stop motion style is more photorealistic, since frames are obtained by taking photographs of real world characters.



Figure 3-2: Chris Roe from Fat Pebble (Pebble (2013)) working on characters for the game Clay Jam (Pebble (2014)).

However, the whole process is time consuming and solely depends on the skills and efforts of the stop motion artist. The resulting animations often exhibit non-smooth transitions between frames due to the limited number of poses created. This effect is sometimes desired, given the unique signature look of the stop motion art form.

Nevertheless, when enhancements of the animation are required, it is very hard to edit an existing stop motion animation sequence. This is due to capturing the character poses as 2D images, without having the corresponding 3D geometry for making further refinements. Moreover, it is not trivial to bring the handmade character animation into a 3D environment, like a game. Thus qualities of the real model like texture, volume, custom animation style, may be lost when 2D mediums are the preferred method of visualization.

To address the above limitations, this thesis proposes a novel system that allows the

artist to reconstruct and refine handmade animation on the computer. The method chiefly benefits scenarios where multiple character poses need to be modeled individually, as it reduces the artist’s workload significantly. Even when a single physical character is used to shape all the poses in an animation, the proposed digitization pipeline can help bring it to life in a 3D environment.

Figure 3-3 presents the steps of the proposed E-StopMotion pipeline method. The main idea is to digitize a few *key poses* of a physical character and use non-rigid registration and interpolation to generate plausible in-between poses with the same connectivity. Optional helpers and deformations can be used to initialize the registration for better matches. Also, the digitization enables flexible post-production editing of the animation based on existing CG techniques, such as shape deformation, image and shape composition etc. Note that it is not the intention of the author to diminish the virtues of stop motion animation, but rather make it more versatile and accessible to the virtual worlds of animation and 3D games.

In practice, the challenges that arise are linked to the sparse number of scans and large deformations between them. Since the available *key poses* modelled by the artist can be sparse, the deformation between them is often large. This is different from continuous capture of deformable shapes like humans, where the significant amount of available data makes it possible to reconstruct the shape and movement reliably (Süßmuth et al. (2008)). Thus it becomes more difficult to identify correspondences between the respective scans. Generating plausible intermediate poses is also not straightforward, as common techniques like linear interpolation may result in artefacts.

The steps of the full E-StopMotion pipeline are presented next, together with some experiments, results and applications of its simplified version. The latter includes the *Acquisition & Cleanup*, *Non-rigid Registration*, *As-Rigid-as-Possible (ARAP) Interpolation* and *Applications* steps from figure 3-3. The bullet points were not included in the simplified implementation. *Analysis* and *Helper Addition* are linked to the character classification from chapter 2. *Deformation* refers to using a surface deformation method that imitates clay to improve the non-rigid registration step. An extensive exploration of virtual clay deformation methods can be seen in chapter 4. Chapter 5 makes use of deformation and other helpers to improve non-rigid registration.

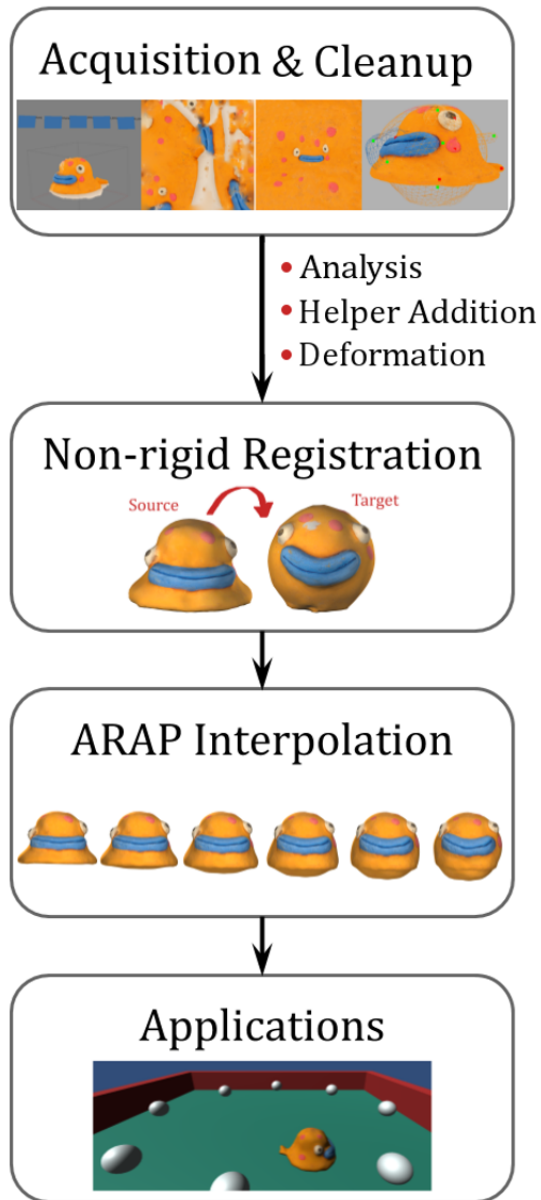


Figure 3-3: E-StopMotion pipeline components: *Acquisition and Cleanup* - involves scanning and cleaning the plasticine model scans, *Non-rigid Registration* - finds the correspondences and transformations between the clean meshes, *ARAP Interpolation* - interpolates between the registered shapes as-rigidly-as-possible, *Applications* - highlights the uses of the pipeline in films and games. The red bullet points are optional and were not implemented in the simplified version of E-StopMotion.

3.2.1 Acquisition

Acquisition refers to the action of capturing 3D data from the real world characters. For this work, photogrammetry, commonly associated with multi-view 3D reconstruction, was employed to generate the poses of each character at a few frames of animation. Given a set of photographs of a physical model from different angles, photogrammetry is the technique of finding the 3D positions of the corresponding 2D points sampled from these images. The result is generally a 3D point cloud that approximates the shape of the model. Other forms of representation, such as triangular meshes, can then be derived from these point clouds. High definition textures and UV maps can also be extracted from the photographs.



Figure 3-4: Photo of Rob character, who was used with the initial, rudimentary setup.

The project started with a rudimentary setup, made up of a wooden table, 2-3 lights and a Nikon D90 camera rig that could be moved. A biped clay model called Rob (figure 3-4) was the first character to join in the experiments. Although the beginning was exciting, a few issues emerged from the available setup. The lighting was too bright in front of the character and insufficient at the back. Also, the angles at which the photographs were taken could not be controlled intuitively (figure 3-5).

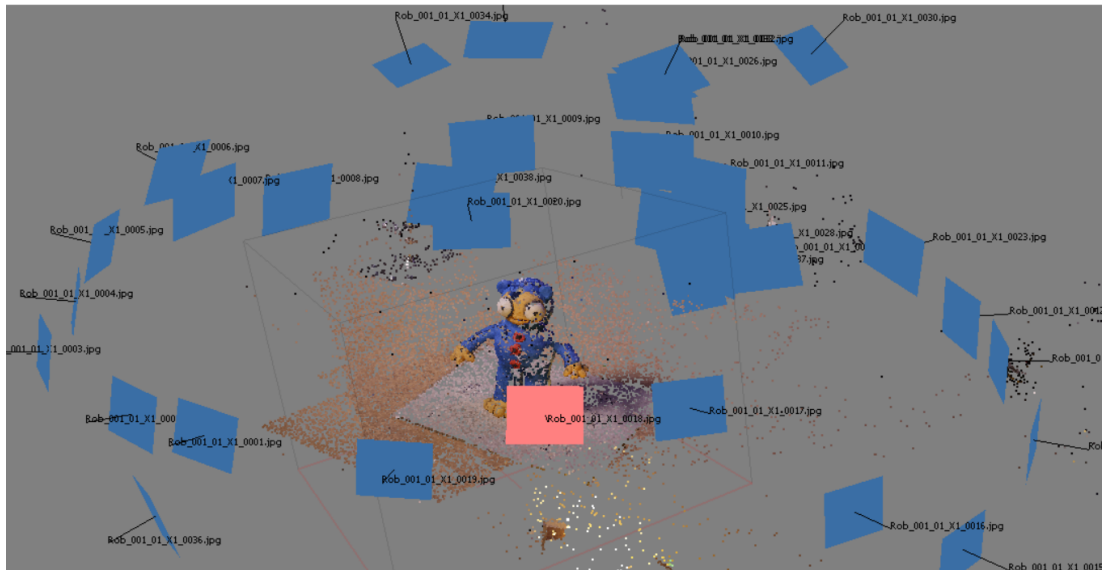


Figure 3-5: Screen capture from PhotoScan of Rob's point cloud and the photographs acquired with the rudimentary setup. Notice how the photographs are dispersed chaotically in 3D space. The red square represents figure 3-4.

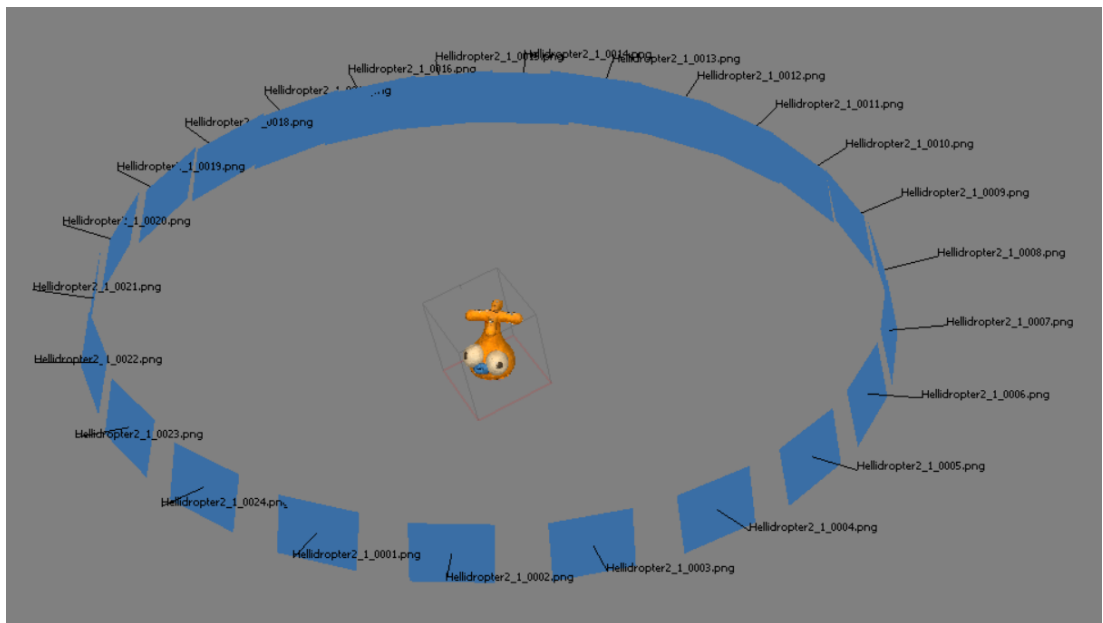


Figure 3-6: Screen capture from PhotoScan of Helldropter's clean point cloud and the images acquired with Alphashot. Notice how the images are spread uniformly around the character.

Agisoft PhotoScan Standard Edition (Agisoft (2018)) was the third party software used to generate the 3D point cloud, mesh and texture for each plasticine model, from a set of

photographs. PhotoScan is a multi-view 3D reconstruction software, commonly used in the film, games and cultural heritage industries. It uses state-of-the-art methods developed by Agisoft, that can capture details of up to 1mm.

A typical procedure in PhotoScan, for a character like Hellidropter (figure 3-6), started with uploading and aligning 26 images. These images were taken all around the character, in order to capture as much information as possible, from a relatively small number of angles. The desired mesh was kept to a maximum resolution of 10000 polygons, due to tablet and phone based requirements for Fat Pebble games. For an average number of 3000 polygons, the 3D mesh would be obtained in up to 10 seconds, with a dense cloud of 217,621 points, at medium quality.

The results with the initial, rudimentary setup (figure 3-5) were cluttered triangular meshes with fragmented UV maps and textures. A smoother technique was necessary to obtain more accurate results. After the Orbitvu Alphashot 360 (OrbitVU (2018)) photogrammetry machine was purchased by Fat Pebble, the acquisition performance improved significantly. This was mostly due to the full control over lighting and photographic angles (figure 3-6).

The Alphashot 360 (OrbitVU (2018)) is a compact photographic studio with 65x63x88cm measurements and 49kg in weight. It is ideal for scanning small objects of up to 30x30x25cm in size and 3kg in weight. The machine also contains a turntable that can rotate at custom degrees, allowing the user to capture as many views of an object as possible. The lighting is uniform and the interior of the Alphashot is white. This allows quick calculations of alpha channels for transparency purposes.

Alphashot can obtain the photographs of a character, at dozens of angles, in a manner of seconds. With the previous rudimentary setup, hours were sometimes required to get all the necessary images. Nikon and Canon Digital Single Lense Relfex (DSLR) cameras can be attached to the Alphashot's flexible rig, which has a 0-90° rotation. The machine also comes with a software where photographs can be viewed in real time.

Nevertheless, the triangular meshes which resulted from the Alphashot scanner and PhotoScan software still presented artefacts due to noise or missing information from hidden parts (eg. under the arm). A cleanup process was needed to make these meshes uniform and thus easy to work with.

3.2.2 Cleanup

Cleanup was thus the next required step, since the geometry obtained from the acquisition step was non-manifold in most cases. Artifacts like intersecting triangles and holes were common, especially in less visible areas (eg. underarms, feet, between fingers). For each character, one of the obtained meshes served as a template or a source mesh in the non-rigid registration part of the pipeline. This usually required a cleanup, which boiled down to transforming the scanned surface into a genus zero, 2-manifold, isotropic mesh. The reason for having an isotropic mesh is linked to the advantages of quasi-uniform meshes in volume preserving deformations over ordinary meshes, as can be seen in Stanculescu et al. (2011). This step is necessary for matching the chosen source to the target mesh, but it is not sufficient for guaranteeing good results.

Another use for the cleanup step was to simplify meshes, especially in the experimental stages, when results needed to be obtained faster. Most of the meshes used for the experiments had 1000-3000 faces (figure 3-7). Higher resolution meshes were only used to check whether the results varied significantly between medium and high resolutions. Autodesk Maya (Autodesk (2018b)) and OpenFlipper (Visual Computing Institute (2018)) were used for the mesh cleanup and simplification.

OpenFlipper (Visual Computing Institute (2018)) is an open-source geometry processing software. Hole filling and isotropic remeshing were performed in it. Although Maya could have sufficed in the geometry processing stages, OpenFlipper was found to be more intuitive for the simple operations needed. Also, due to the cluttered UV maps, meshes took longer times to load in Maya than in OpenFlipper.

Lastly, texture cleanup was also done for the template mesh. The same texture was then mapped onto the deformed or registered meshes, depending on the component of the pipeline considered. This was done for visual comparison between the deformation achieved by the current method and the original plasticine deformation, as given by the available target scans. Maya was used to transfer the texture maps from the high resolution scans to the lower resolution meshes (figure 3-8). The open source Gimp software (Team (2018)) was then used to edit the remapped texture images.

3.2.3 Analysis

In order to find a matching procedure that would suit most of the Clay Jam characters, a closer look at the character movement involved was needed. The goal was to find a

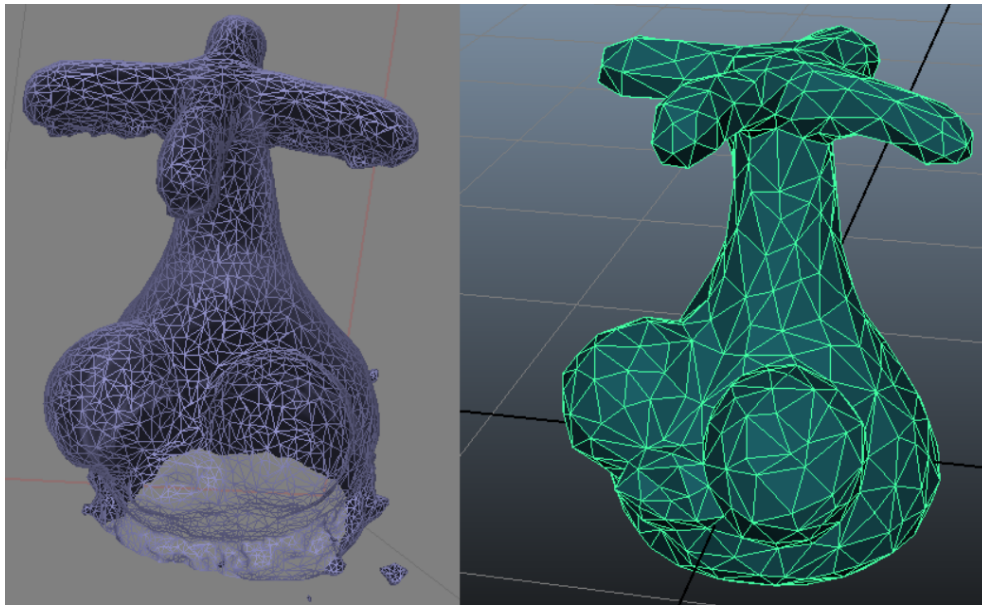


Figure 3-7: Hellidropter initial scan result from PhotoScan (left) and medium resolution mesh after cleanup (right) in Maya (Autodesk (2018*b*)) and OpenFlipper (Visual Computing Institute (2018)).

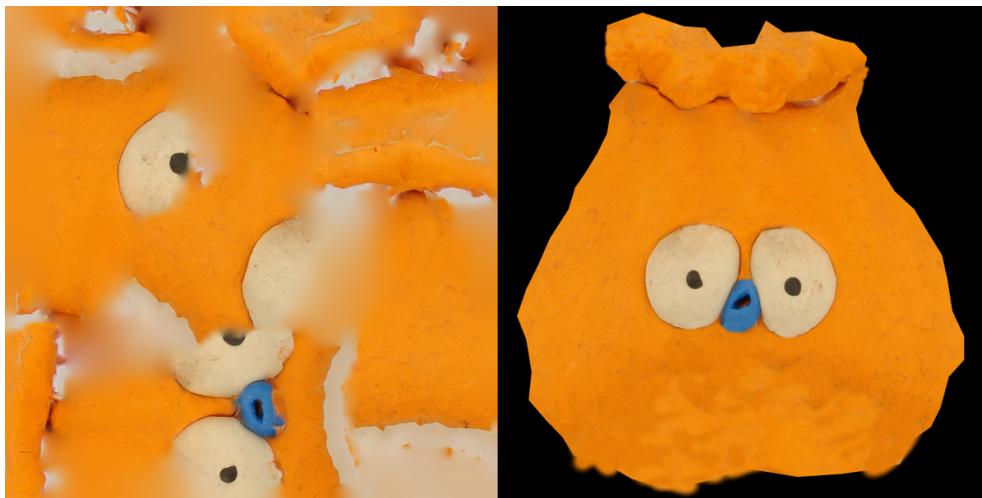


Figure 3-8: Hellidropter initial texture result from PhotoScan (left) and clean version after remapping (right) in Maya and editing in Gimp.

classification for the character animation types, so that any character from a particular class could benefit from a predefined processing strategy. This would also give artists the freedom of generating any type of artwork, leaving the decision making to the software.

The heuristic presented in chapter 2 could be used to classify a given character by its animation type. The classification could then link that character to corresponding helpers that can guide the registration. This step was not included in the simple pipeline implementation, but was left for future work. The animation class considered for most of the Clay Jam characters was *Animation by Shape*, which usually required landmark correspondences as helpers.

3.2.4 Helper Addition

Helper is a general term used to signify the preparation a clean mesh undergoes, before it can be used for deformation or registration. This includes identifying the priors available for aiding the registration process. Some examples of priors are feature points, material properties and the use of a template mesh. By observing the appearance and behaviour of a plasticine character, as it is deformed by the artist, the idea of reverse engineering the animation started to emerge.

Two main layers of helpers were identified for Clay Jam characters, a *Model Layer* and an *Animation Layer* (figure 3-9). The *Model Layer* refers to the properties attributed to the mesh before deformation, while the *Animation Layer* refers to the properties during the deformation. These layers are meant to aid the deformation and registration steps.

When considering deformation algorithms (chapter 4), they are split into either a space based or a surface based deformation category (Botsch & Sorkine (2008)). These deformation methods, together with regularization methods (eg. as-rigid-as-possible, as-conformal-as-possible or as-smooth-as-possible) are separated from the initial helpers. Regardless of the method used, however, the more helpers available, the more accurate the deformation and registration become.

Figure 3-9 gives more insight into what the helper layers entail. The *Model Layer* encompasses attributes common to most Clay Jam characters, like the material they are made from and the way they are structured. Modelling clay or plasticine is the preferred material, which is incompressible and displays plastic deformations with negligible elastic effects. Concerning the structure of the characters, the most common technique of the company is stitching character parts together. Skeletons are occasionally used to add rigidity to the models.

The *Animation Layer* considers the animation type of each character component. Limbed characters can benefit from a skeleton guiding the mesh, while a blobby shape

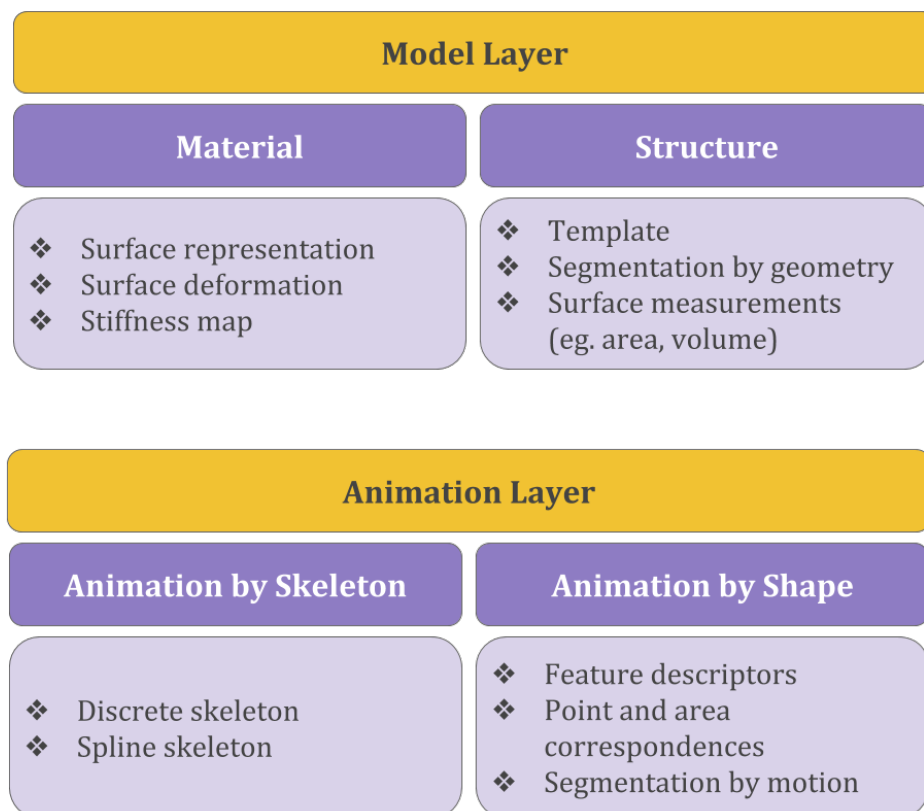


Figure 3-9: Helper Layers to consider when analyzing what a character needs for a good animation reconstruction. This involves physically plausible deformation, accurate non-rigid registration and as-rigid-as-possible interpolation.

may benefit more from landmark or handle points. The scale of deformation can also influence the number of steps needed for achieving an accurate registration and the most suitable regularization techniques.

3.2.5 Deformation

The deformation step is used as a technique to improve the initialization for the non-rigid registration, as presented in chapter 5. This produces not only better matches between meshes, but also provides a few physically plausible in-between shapes.

Chapter 4 gives an extensive report on ways of representing and deforming virtual clay. It also provides a description of a proposed nonlinear deformation method, which imitates plasticine. The algorithm builds on the work of Fröhlich & Botsch (2011)

by adding localized stiffness per vertex and volume preserving properties per surface tetrahedron, to better imitate clay properties.

The deformation step was not included in the simplified pipeline implementation, but was explored separately. It can easily be added to the current implementation in the future. Chapter 5, however, experiments with having deformation as an initialization step for the non-rigid registration method. This was done in order to explore whether physically plausible shapes can be obtained in this manner and if the latter can initialize more accurate registrations.

3.2.6 Non-rigid Registration

Once the helpers (eg. landmarks, stiffness) are set for the source mesh, it is prepared to register against the target mesh. The non-rigid registration method can benefit from the deformation step presented above. The idea is to reverse engineer the deformation the clay went through during the animation process, to improve the match between available scans.

Registration between mesh scans consists of finding a mapping between a source mesh and one or more target meshes. After a source is registered to the target mesh, the result consists of two similar meshes, with the same connectivity. The Non-rigid Iterative Closest Point Algorithm (NRICP), as described by Amberg et al. (2007), was adapted for the proposed pipeline. The reason for choosing this algorithm was because it is relatively fast, easy to implement and allows free-form deformations of vertices. Chapter 5 gives a thorough explanation of this method and the helpers and deformation enhancements experimented with.

3.2.7 Interpolation

Once one or more target meshes are obtained, with the same connectivity as the source mesh, a sequence of in-between poses can be generated using interpolation. As-rigid-as-possible (ARAP) interpolation is used, by extending on the work of Liu et al. (2011) and Baxter et al. (2008). The work presented in this section was the contribution of co-author Naval Bhandari on the E-StopMotion paper, published at Motion, Interaction in Games (Ciucanu et al. (2018)).

The proposed ARAP interpolation method employs a local-global optimization strategy to compute the topologically consistent intermediate meshes between a source and a

target mesh. Locally it computes the affine transformation for each local shape element from the source to the target and obtains an expected intermediate interpolation first. A local shape can consist of a triangle or a tetrahedron, the latter being the preferred option in this implementation. The algorithm then globally solves for the intermediate mesh by minimizing the sum of differences between all expected local transformations and the corresponding actual transformations.

Unlike the original as-rigid-as-possible method (Alexa et al. (2000)), consistent tetrahedral meshes are not necessary since surface tetrahedra can be calculated from the triangle meshes (Liu et al. (2011)). Consistent tetrahedral meshes are difficult to create, while dynamic, surface based tetrahedra can be calculated on the fly. The method of creating a tetrahedron from a triangle consists of adding a fourth vertex along the normal (Liu et al. (2011)). The application point is at the centroid of the triangle, given by the average of the three component vertex positions. Given the triangle vertices, v_1 , v_2 and v_3 , the position of the fourth vertex v_4 is calculated using the equation proposed by Liu et al. (2011),

$$v_4 = \frac{v_1 + v_2 + v_3}{3} + \frac{(v_2 - v_1) \times (v_3 - v_1)}{\sqrt{(v_2 - v_1) \times (v_3 - v_1) \cdot (v_2 - v_1) \times (v_3 - v_1)}} \quad (3.1)$$

The global minimization problem is formulated using normal equations, addressing the issues which are carried over from 2D shape interpolation into 3D. This gives an elegant solution to symmetric interpolation and adds constraints and control to the interpolated results. It also allows for easy correction of rotational inconsistencies.

Energy Minimization

The mesh is constructed of several tetrahedra, many of them sharing vertices. An interpolated affine transformation for one tetrahedron will not necessarily be the same affine transformation for another tetrahedron with a common vertex. The goal is to minimise the global error between every expected affine transformation $A_T(t)$ at time t , and the actual transformation B_T . Let $V_T(t)$ be the set of unknown interpolated vertex positions at time t and V_T the vertex positions of the source tetrahedron defined as

$$V_T = \begin{bmatrix} v_i^x & v_i^y & v_i^z \\ v_j^x & v_j^y & v_j^z \\ v_k^x & v_k^y & v_k^z \\ v_l^x & v_l^y & v_l^z \end{bmatrix}$$

where $v_i^x, v_i^y, v_i^z, v_j^x, v_j^y, v_j^z$ etc. are the x, y, z coordinates of the source tetrahedron vertices, indexed by i, j, k and l .

Given matrix

$$D = \begin{bmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & -1 & -1 \end{bmatrix}$$

then the inverse source tetrahedron matrix V_T^* can be defined as $V_T^* = (DV_T)^{-1}D$. The relationship between $V_T(t)$ and V_T^* gives the actual, unknown affine transformation matrix, $B_T = V_T^*V_T(t)$.

The matrix $A_T(t)$ represents the expected affine transformation at time t . Similarly to the source tetrahedron V_T , the target tetrahedron U_T is defined as

$$U_T = \begin{bmatrix} u_i^x & u_i^y & u_i^z \\ u_j^x & u_j^y & u_j^z \\ u_k^x & u_k^y & u_k^z \\ u_l^x & u_l^y & u_l^z \end{bmatrix}$$

where $u_i^x, u_i^y, u_i^z, u_j^x, u_j^y, u_j^z$ etc. are the x, y, z coordinates of the target tetrahedron vertices, indexed by i, j, k and l . An affine transformation from source to target tetrahedrons can then be defined as $A_T = V_T^*U_T$ as proposed by Baxter et al. (2008). The value of $A_T(t)$ is obtained by a linear interpolation between the identity matrix and A_T .

The definitions of B_T and $A_T(t)$ result in the global minimisation equation

$$E(t) = \sum_{T=1}^M a_T \|B_T - A_T(t)\|_F^2, \quad (3.2)$$

where a_T is the area of the original triangle of tetrahedron V_T , $A_T(t)$ is the expected affine transformation at time t , B_T is the actual affine transformation and $\|\cdot\|_F$ is the

Frobenius norm. The area is used to weigh the importance of a particular triangle, as it allows for large differences in tessellation of a mesh, whilst maintaining the same interpolation results.

If all tetrahedron transformations were to be calculated by an interpolation of the affine transformation A_T , the resulting mesh would be disconnected. To ensure smoothness of the surface, the global energy from equation (3.2) needs to be minimized. The actual transformation B_T can also be found in this way, together with the tetrahedron vertex positions $V_T(t)$. In order to simplify the problem matrix notation of the energy function was introduced.

Matrix Notation

The energy function (3.2) can be expressed in matrix form by grouping the quadratic terms in a matrix $-H$ and the linear terms in $G(t)$. These matrices are defined per tetrahedron as

$$-H_T = a_T(V_T^*)^T V_T^*, \quad (3.3)$$

$$G_T(t) = a_T(V_T^*) A_T(t), \quad (3.4)$$

where the values of $-H_T$ and $G_T(t)$ can be placed into $-H$ and $G(t)$, at the position of the index of each vertex. $-H$ is a sparse $(N + M) \times (N + M)$ matrix, where the values of $-H_T$ are summed at corresponding indices in both directions of $-H$. $G(t)$ is an $(N + M) \times 3$ matrix, where the values of $G_T(t)$ are summed in rows corresponding to the vertex indices. N and M are the number of the vertices and triangles, respectively.

It is important to note that $-H$ is independent of t , and thus only needs to be calculated once, before interpolation. As $G(t)$ relies on each $A_T(t)$, it must be calculated at every time step. Using the approach proposed by Alexa et al. (2000), $V(t)$ is found by solving the following normal equation.

$$V(t) = -H^{-1}G(t). \quad (3.5)$$

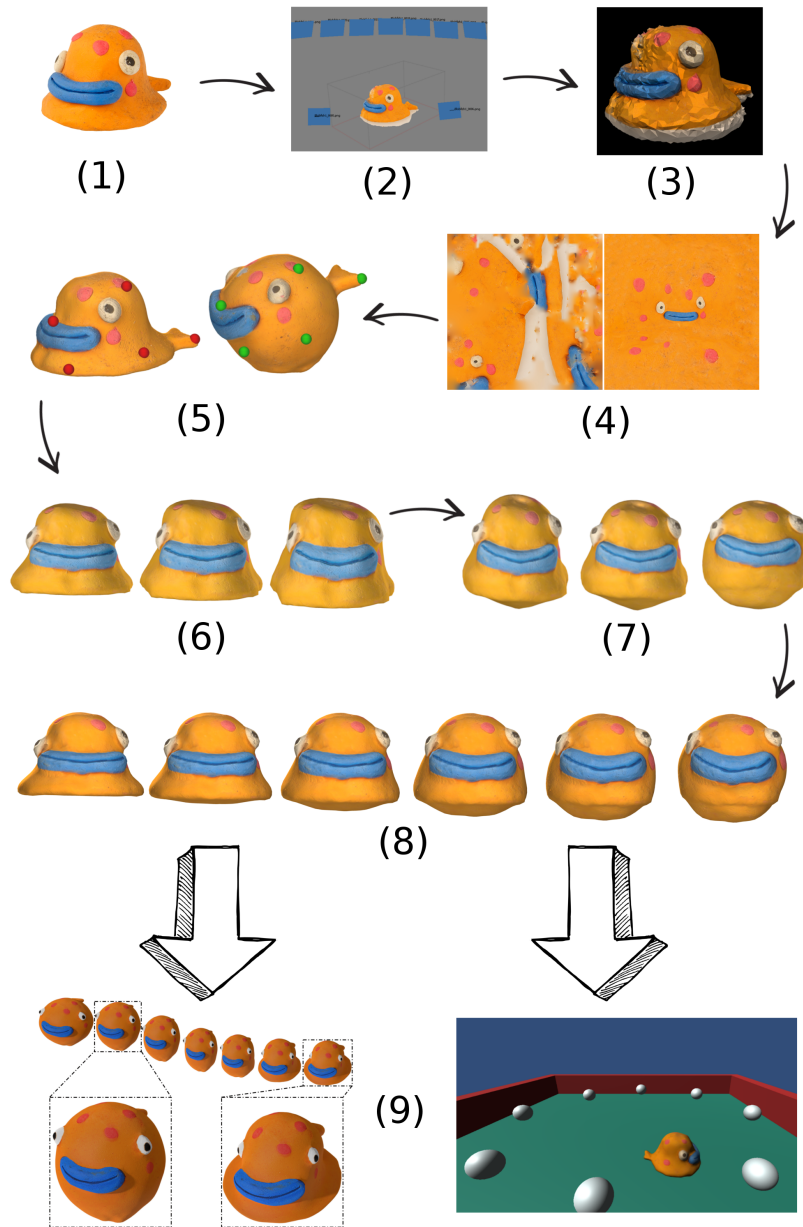


Figure 3-10: An example of the full pipeline with Blob Fish as an example character. Analysis hasn't been added, but Blob Fish is a C1: Globally Nonuniform - Intrinsic Metric character. (1) Physical model, (2) Photogrammetry in PhotoScan, (3) Mesh cleanup in OpenFlipper, (4) Texture cleanup in Maya, (5) Adding helpers (landmarks) between source and target, (6) Deformation of source towards landmarks, (7) NRICP between deformed source and target, (8) ARAP interpolation between source and target, (9) Applications in films (left) and games (right).

3.2.8 Applications

The applications of the current method range from enhancing the original stop motion animation in a film to having 3D versions of plasticine characters in games. We can not only generate multiple in-between shapes, but also modify them with third party tools like Blender to enhance the animation. More details on the applications are given in the next section.

3.3 E-StopMotion: Simplified Pipeline Implementation

3.3.1 Overview

Given a limited number of *key poses* for a physical character made of plasticine, the goal of the proposed system is to digitize these shapes and create plausible in-between poses for animation and game applications. It consists of four components (figure 3-3): *Acquisition & Cleanup* is based on an existing multi-view reconstruction approach, two major algorithmic steps, *Non-rigid Registration* and *As-rigid-as-possible (ARAP) Interpolation*, and *Applications* that find novel uses for the reconstructed and interpolated shapes in stop motion animation and games.

The *Acquisition & Cleanup* step is responsible for capturing and reconstructing the geometry and texture of the *key poses* of a plasticine character created by an artist (figure 3-11). The multi-view 3D reconstruction approach, as implemented in Agisoft PhotoScan (Agisoft (2018)), is employed to generate the digital shapes represented by triangular meshes. The images used in the reconstruction process are captured with an Alphashot 360 machine (OrbitVU (2018)). The resulting meshes display most of the character’s details in their *key poses*, depending on the desired resolution. Artefacts like holes and overlapping triangles also make an appearance. The clean up procedure makes the meshes watertight, 2-manifolds. Texture remapping is also used to redefine the UV and texture maps.

The following two algorithmic steps, *Non-rigid Registration* and *As-rigid-as-possible (ARAP) Interpolation*, are devised to register the clean, scanned *key poses* and generate plausible in-between shapes through interpolation. This can help reduce the number of manually manipulated poses, which facilitates the traditional stop motion creation process and benefits subsequent animation and game applications. Non-rigid registration (Amberg et al. (2007)) was used, with local affine transformations to align

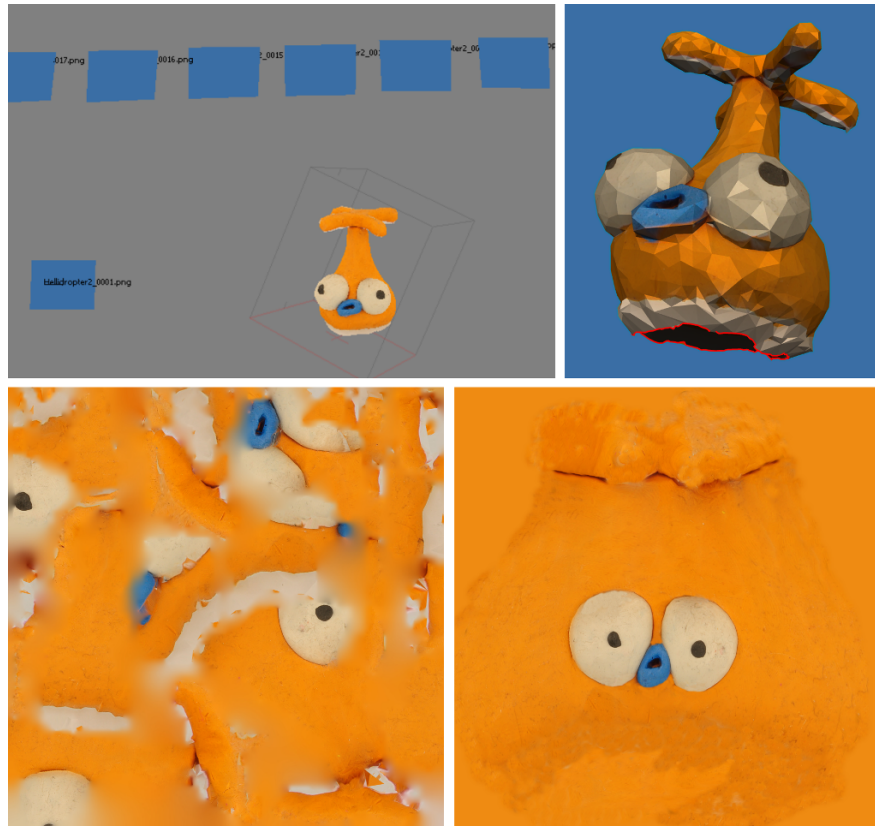


Figure 3-11: *Acquisition & Cleanup* step for Helldropter consists of photogrammetry in PhotoScan (top left), mesh cleanup (top right) and texture remapping in Maya (bottom).

the key pose scans. Since the deformations between the meshes vary from small to large, the user needed to pick a sparse set of landmarks between the source and the target, to guide the registration.

Based on the resulting surface correspondences, the as-rigid-as-possible interpolation method was used to obtain pleasing intermediate shapes. The proposed approach to interpolation extends an existing normal equation based shape interpolation technique from 2D to 3D (Liu et al. (2011), Baxter et al. (2008)). Also it does not require consistent tetrahedral meshing between *key poses*, thus can save computational time while achieving plausible results.

The scanned *key poses* and interpolated intermediate poses can next be imported into existing animation and game development software for applications. To demonstrate the use of the system, the digitized poses are imported into a popular animation software and a game engine (Blender and Unity respectively) to generate enhanced stop

motion animation and stop motion-like games.

3.3.2 Experiments and Results

The simplified pipeline was applied to a set of characters from the game Clay Jam (Pebble (2014)) created by Fat Pebble (Pebble (2013)). One of the models was even recreated by hand, since the original character had been lost (figure 3-20). The scale of deformation between frames was small or medium, with large amounts of overlap. In some cases, nonisometries made the matches difficult, but it was found that landmark correspondences corrected the artefacts to some extent. A minimal set of handpicked landmarks (10-15) was utilized, however, since it can be a tedious process for the artist.

Figures 3-17 - 3-21 show the in-between poses obtained from the original source and registered source scans of a few plasticine characters. The exception is figure 3-19, which has the target shape generated by rigging and deforming the source shape. This was done to exemplify the future use of skeleton helpers for registration and the difference between linear and as-rigid-as-possible interpolation when rotation produces nonlinear deformations. These pairwise results were extended to more *key poses* to create varied animation sequences.

Figures 3-17, 3-18 and 3-19 display a comparison between linear and ARAP interpolation. Figures 3-17 and 3-18 do not display significant differences in shape, due to the deformations being approximated well by both linear and ARAP interpolation. Figure 3-12 shows the small shape difference for the Blob Fish character. The difference between the two types of interpolation methods becomes visible when rotation produces nonlinearities that cannot be approximated well with the linear approach.

Figure 3-19 shows how linear interpolation shrinks the volume of the Party Onion character when a large rotation is introduced, while ARAP interpolation manages to preserve the shape better. This example was created for clarifying the difference between linear and ARAP interpolation. Skeletal helpers that can guide the deformation and registration could be used in future implementations of the pipeline.

The current skinning experiment was done manually for interpolation performance purposes. When skeletonization becomes an automated feature for the E-StopMotion pipeline, it would be worth comparing the ARAP interpolation result to the rigged character. This would clarify whether structure specific helpers can guide the non-rigid registration and, thus the animation reconstruction process better than a shape preserving interpolation technique.

Tables 3.1 and 3.2 show the average area and volume distortions and standard deviations for characters in both types of interpolation. The distortion of area or volume is to be understood as the relative deviation from the original corresponding values. The distortions are calculated per interpolation step over a set of 10 iterations. The method for measuring area or volume distortion for a surface is by finding the average of all the corresponding triangle distortions. The area distortion of a triangle, for example, is calculated with equation

$$d_A = \frac{A_c}{A_0} - 1.0 \quad (3.6)$$

where d_A is the relative area distortion for the triangle in cause, A_c is its current area and A_0 is the initial area. By subtracting 1.0 from the relative area value, one can estimate if the triangle is shrinking ($d_A < 0$) or expanding ($d_A > 0$).

The resulting average area distortions and standard deviations are generally smaller for the ARAP interpolation, compared to the linear one (tables 3.1 and 3.2). The average volume distortions and standard deviations are close in values, with ARAP displaying smaller distortion than linear interpolation when nonlinear deformations are more common. The difference is especially noticeable in the case of Party Onion from figure 3-19. While the linear interpolation shrinks the area and volume of the model, the ARAP interpolation distortion is significantly smaller.



Figure 3-12: The contours of three iterations of the Blob Fish interpolations. Linear interpolation is shown in grey, while ARAP interpolation is red. Notice the small difference in shape between the two types of interpolation in this case.

Figure 3-13 shows a normal distribution of the area and volume distortions for Tick (figure 3-20) and Party Onion (figure 3-18). In case of the Tick (top part of the image), although the average and standard deviation of the area and volume distortions are smaller for the ARAP interpolation, we see the frequency of smaller distortions is higher for the linear interpolation (red), than for the ARAP interpolation (blue). The ARAP

Table 3.1: Average relative area distortions for linear and as-rigid-as-possible (ARAP) interpolations and the corresponding standard deviations. Negative values signify that the mesh area is preponderantly shrinking during the interpolation, while positive values signify the area is growing. The meshes considered can be seen in figures 3-17 to 3-21 respectively. The landmarks specified are used in the non-rigid registration step.

Character	Faces	Lmrks	Linear Distortion	Linear σ	ARAP Distortion	ARAP σ
Blob Fish	3482	12	-0.0537	0.0245	-0.0208	0.0157
Party Onion	3600	13	0.0224	0.0185	4.6793 E-04	0.0020
Party Onion Rigged	3600	N/A	-0.0299	0.0208	9.2850E-05	1.3018E-04
Tick	2750	13	0.0835	0.0568	-0.0329	0.0235
Hellidropter	1454	10	0.0202	0.0196	-0.0206	0.0142

Table 3.2: Average relative volume distortions for linear and as-rigid-as-possible (ARAP) interpolations and the corresponding standard deviations. Negative values signify that the mesh volume is preponderantly shrinking during the interpolation, while positive values signify the volume is growing. The meshes considered can be seen in figures 3-17 to 3-21 respectively. The landmarks specified are used in the non-rigid registration step.

Character	Faces	Lmrks	Linear Distortion	Linear σ	ARAP Distortion	ARAP σ
Blob Fish	3482	12	0.1025	0.0759	0.1663	0.0822
Party Onion	3600	13	-0.0334	0.0207	0.0357	0.0222
Party Onion Rigged	3600	N/A	-0.0134	0.0077	8.7708E-04	5.6068E-04
Tick	2750	13	0.0674	0.0454	-0.0542	0.0402
Hellidropter	1454	10	0.0229	0.0195	-0.0254	0.0182

distortion, however, seems more uniformly distributed than the skewed linear one. This signifies that distortion evolves more smoothly in the former case. The Party Onion distortion distributions (bottom part of the image) are close in shape, probably due to most of the mesh having small deformations, while only the tentacle rotates.

Performance

All the experimental results were generated on a laptop with Intel Core i7-4710HQ CPU (2.50 GHz) and 16 GB memory. The code was written in MATLAB. For a typical digitized model (as the example shown in figure 3-17) with 3490 triangles it took 267.634 seconds to register a source mesh to a target mesh and 120.065 seconds to obtain the interpolated shapes between them. The Hausdorff distance between the registered source and target meshes from figure 3-14 is 0.0621, given that the meshes

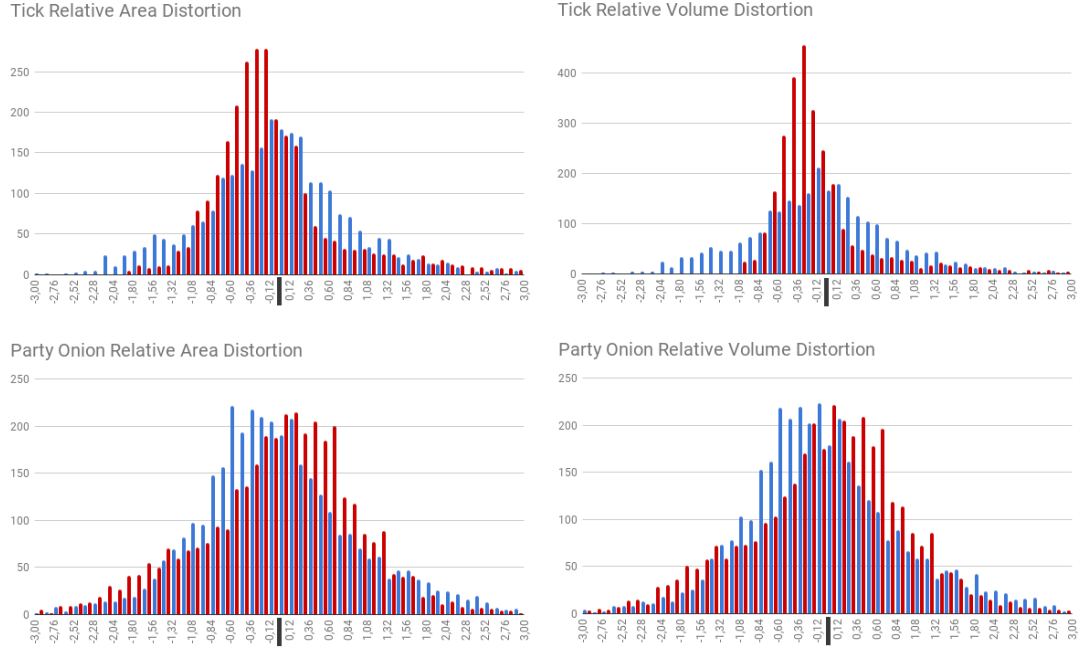


Figure 3-13: Normalized relative area and volume distortions for Tick (figure 3-20) in the top part and Party Onion (figure 3-18) in the bottom part of the image. The ARAP interpolation distortions are shown in blue, while the linear distortions are red. This analysis was made on all the faces of each mesh, for one iteration of the interpolation.

are normalized within a $1 \times 1 \times 1$ bounding box. In the future, the pipeline could be reimplemented in C++ for faster times.

Nevertheless, even with the current performance, together with the needed time for scanning and cleaning a few character poses, the artist’s workload is reduced considerably. The time required for scanning and cleaning a character can take 0.5 - 2 hours, depending on the skill of the artist and the complexity of the character. In the case of digitizing stop motion for 3D games, at least, the artist needs only a small set of *key poses* to be cleaned up. The intermediate poses are then generated automatically with the proposed pipeline. In the case of an interpolation step $t = 0.1$, the needed time for an artist to obtain a 3D sequence of poses between two *key poses* is approximately 10 times faster than doing everything manually.

3.3.3 Applications

Enhanced Stop Motion Animation

Based on the reconstructed key poses and interpolated intermediate poses, the existing 2D stop motion animation frames can be easily enhanced using traditional CG animation and composition techniques. For example, as shown in figure 3-14, the input stop motion animation only contains two 2D frames, corresponding to two *key poses* created by the artist.

The corresponding 3D meshes were imported into Blender, an open source animation software, together with the in-between shapes obtained from the interpolation step. To combine the available 3D shapes with the 2D frames, the virtual camera parameters for projecting 3D shapes to 2D are estimated by manually aligning 3D reconstructed meshes with the existing stop motion animation frames. These camera parameters can be used to render new 2D frames from the 3D shapes to enhance the input animation frames (only two in this case).

Moreover, since the 3D geometry of the key pose is available, existing deformation tools (e.g. Free Form Deformation in Blender) can be used to easily create new poses which are challenging to achieve otherwise, as all the resources needs to be physically replicable afterwards, including human resources (artist), and natural resources (plasticine, camera, etc.).

Stop Motion-like Games

The resulting interpolated meshes can bring stop motion character animations in Unity games. Meshes were imported as FBX files in a modified Unity demo game. Since the MATLAB code exported results as OBJs, an extra step of importing models into Maya and exporting them as FBX files was necessary. This could be easily automated in the future with a python script.

An empty game object was created, with two C# scripts attached to it. One script controlled the stop motion animation effect and the other was in charge of the gameplay. The first script had a public array of game objects named **frames**, where the meshes were loaded. Another public field was **fps** (frames per second), which coincides with how fast the frames appear and disappear. A sampling variable was also included, so that the user can choose which meshes should be displayed. The resulting game scene is shown in figure 3-15.

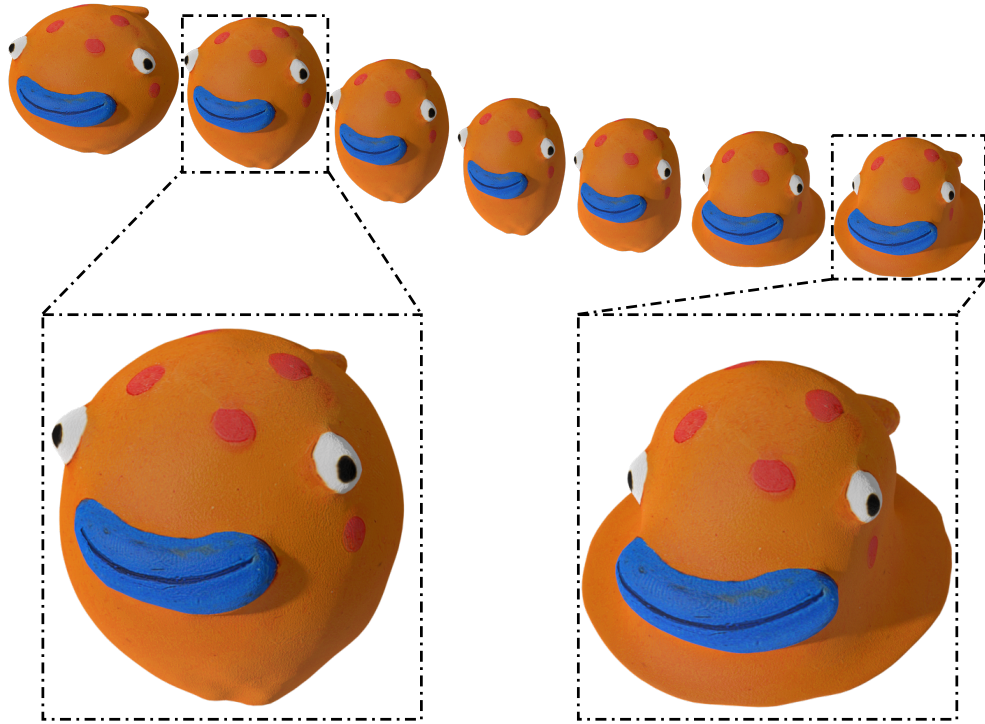


Figure 3-14: Enhancing existing stop motion animation with additional frames using the proposed approach. In the example figure, the two highlighted frames are the source and target scans, with the same connectivity after registration. The in-between poses are modified versions of the interpolated shapes obtained through the ARAP interpolation. The added modifications create a pleasing squash and stretch effect for the resulting animation.

3.4 Discussion

A novel pipeline for reconstructing and refining stop motion animation for use in 3D games and films was presented. By combining the handmade craft of stop motion with 3D scanning, non-rigid registration and interpolation, artists are now able to bring their creations onto the screen in three dimensions.

Artists can scan and clean a small set of *key poses* for their character and then obtain plausible in-between shapes generated automatically. This process makes the animation reconstruction technique accessible to any games company. Users can then interact with a digital version of a handmade character in a game.

Moreover, both 3D and 2D stop motion animation can benefit from the proposed pipeline. The in-between poses can be easily aligned to match the original orientation of a character. Instead of photographing every frame, automatically generated in-

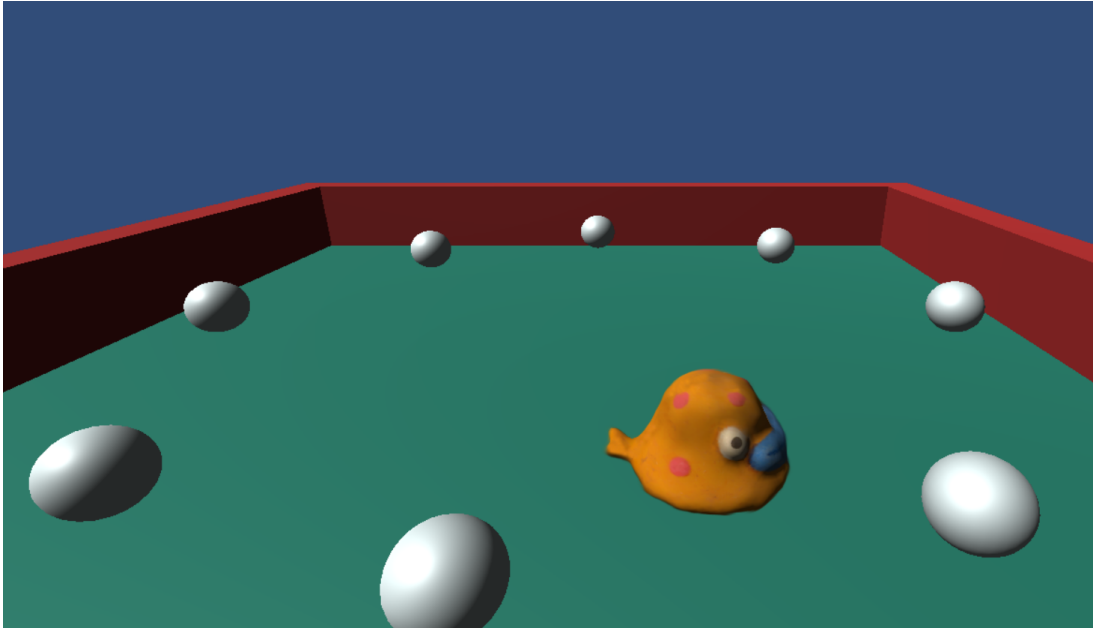


Figure 3-15: Application of E-StopMotion pipeline for video games. In this example, the Blob Fish animation is imported into an example Unity (Technologies (2018)) game.

between poses can be used to reduce the workload or refine an initial animation.

Research question 5 has been partially answered in this chapter, but experiments on more types of characters are needed to ensure that the pipeline is extensible. In summary, the simplified E-StopMotion pipeline implementation makes two main contributions: i) A novel system that digitizes stop motion based on robust 3D registration and interpolation, and ii) Two major applications for enhanced stop motion animation and games.

Limitations and Future Work

Figure 3-16 shows an example of a character pose that did not match the target shape as expected. This is due to rotation induced nonlinearities that cannot be approximated well through affine transformations. A guiding skeleton could have been beneficial in such a scenario. Figure 3-19 shows an example of how a skeleton might guide the deformation before registration is employed.

The interpolation process was satisfactory, but only produced results as good as the registration process allowed it to. This limitation needs to be addressed in the future by adopting a state-of-the-art non-rigid registration method that allows nonisometry between the source and target models.

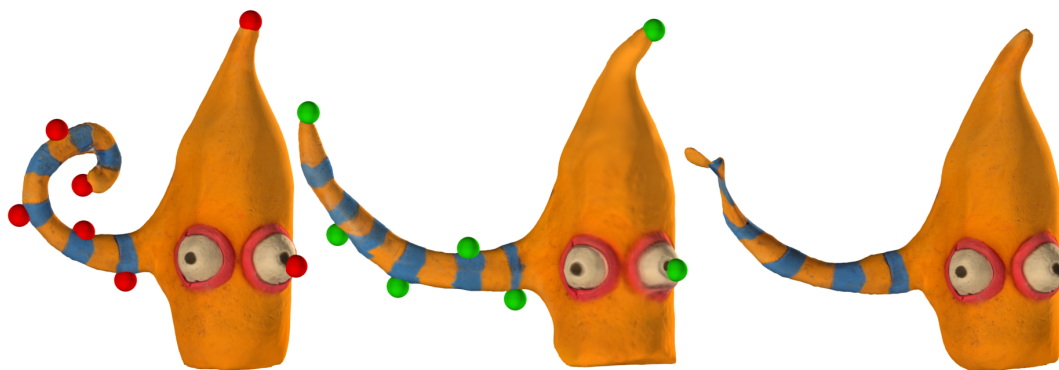


Figure 3-16: This is an example where the NRICP fails, due to the nonlinearity of deformation. Party Onion source (left) and target (middle) meshes with landmarks and the registered mesh (right). Notice the collapse in the tentacle of the registered mesh.

The correspondence problem can be better addressed by identifying nearly-isometric regions like the eyes and mouth and extending correspondences from there based on the smoothness of deformation (Vestner et al. (2017)). The trajectory problem can be addressed by using coarser structures that guide the large deformation. Deformation graphs (Li et al. (2008)) or skeletons extracted from the mesh (Tagliasacchi et al. (2016)) could significantly improve the deformation. Regarding the look and feel of stop motion animation, it would be interesting to conduct qualitative experiments on the aesthetics differences between linear and ARAP interpolation.

The next chapter describes virtual clay and proposes a nonlinear surface based deformation method that imitates plasticine. By understanding the physics of plasticine, the author believes it is possible to reverse engineer the simplest deformation path to get from the source shape to the target shape. In chapter 5 this deformation method is used to initialize and enhance the NRICP algorithm.

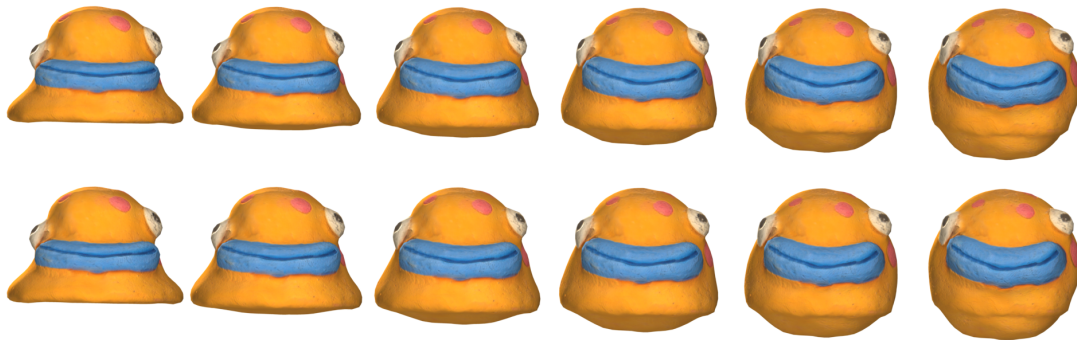


Figure 3-17: Example linear interpolation (top) versus ARAP interpolation (bottom) for Blob Fish character animation, between two clean scans. Although the results seem similar visually, table 3.1 reveals how the area distortion is smaller for the latter.

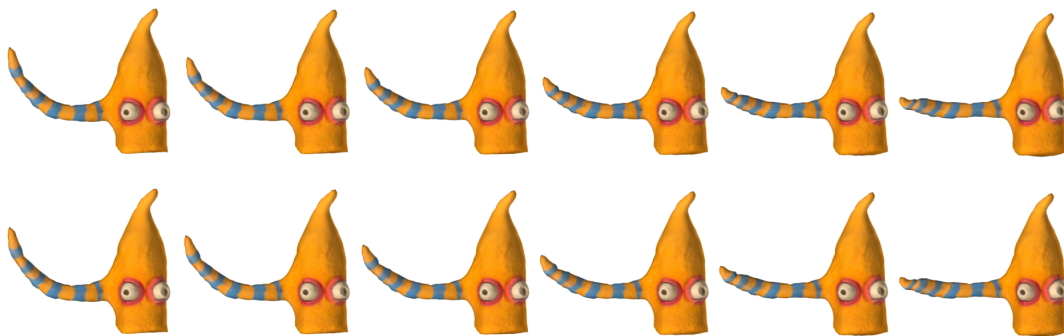


Figure 3-18: Example linear interpolation (top) versus ARAP interpolation (bottom) for Party Onion character animation, between two clean scans. Although the results seem similar visually, table 3.1 reveals how the area distortion is smaller for the latter.



Figure 3-19: Example linear interpolation (top) versus ARAP interpolation (bottom) for Party Onion character animation, between two clean scans. The target shape in this scenario was created by rigging the tentacle of the source shape and rotating the joints.

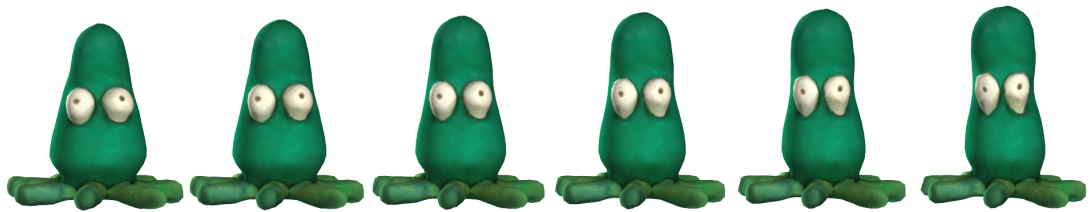


Figure 3-20: Example of ARAP interpolation for Tick character animation between two clean scans.

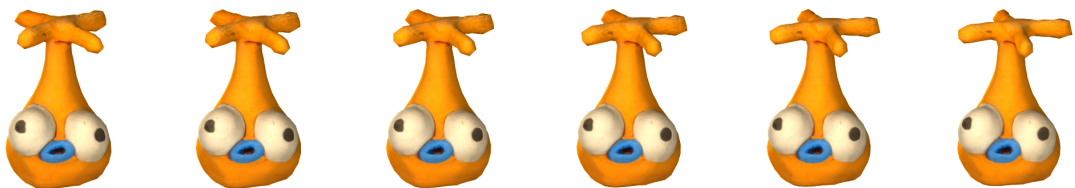


Figure 3-21: Example of ARAP interpolation for Hellidropter character animation between two clean scans. The animation here is subtle, notice how the propeller rotates slightly.

Chapter 4

Representation and Deformation of Virtual Clay Models

In the previous chapter a pipeline was presented to allow the reconstruction and enhancement of stop motion animation for use in 3D games and films. The following two chapters investigate techniques for improving the *Non-rigid Registration* component of the pipeline. One way this is achieved is by initializing the registration with a deformed version of the source model.

Since the characters being digitized are made out of plasticine, it became apparent that using the properties of this type of material could aid the reconstruction process. To be more specific, the registration step involved in the reconstruction could benefit from material related regularization. By guiding the deformation trajectory through plasticine specific properties like volume preservation and localized stiffness, better registrations and also physically plausible in-between poses can be obtained, as it is demonstrated in this chapter.

Research questions 1 and 2 are also explored. Sections 4.2 and 4.3 investigate available techniques for representing and deforming plasticine models on the computer, thus touching on RQ 1. Next, RQ 2 was considered by adopting surface based deformations to imitate the solid modelling clay properties. For this the link between volumetric and surface representations and deformations was studied (sections 4.2.3 and 4.3.3). The investigation is prefixed with a thorough analysis of the physical properties of plasticine, to identify what characteristics describe this material's movement reliably (section 4.1).

The author’s contribution is the addition of localized stiffness to the thin shell deformation method proposed by Fröhlich & Botsch (2011). This chapter shows how stiffness can aid the surface deformation to better mimic volume preservation and localized shape changes specific to plasticine. In chapter 5 the proposed method is used as the optional *Deformation* step of the pipeline (figure 3-3), to improve non-rigid registration.

The concept of virtual clay was made popular by Dewaele & Cani (2004), who propose a technique for mimicking the properties of plasticine (modelling clay), rather than simulating them. Extending on this idea and on corresponding literature, the focus of this chapter became surface based techniques that imitate the physics of plasticine. The reasoning ranged from efficiency, directability and rendering advantages that surface based representation and deformation methods have over volumetric or simulation based techniques.

4.1 Analysis of Clay Properties

Modelling clay or plasticine is a material frequently used in stop motion animation, due to the simplicity of its use. The main characteristics observed when handling plasticine models were volume preservation, plasticity and part based modelling. These properties are described in the following sections. The links with the properties described by Dewaele & Cani (2004) are also mentioned, who represent virtual clay by a discrete scalar field grid. They underline that mass conservation, large scale displacements and surface tension are the main qualities of clay (table 1.1).

4.1.1 Volume Preservation

Volume preservation of a source model is linked to mass conservation and surface tension. This is due to the tendency of plasticine to redistribute mass locally, around a force application point. The material flows from the aforementioned point to the closest resting position, according to plastic flow principles. In doing so, the model’s volume is preserved to a certain extent.

To better understand the process, an inquiry was made into the mathematical entities that describe how volume is contained. When an area is compressed or stretched, the material compensates by stretching or compressing a neighboring area respectively. This property manifests itself as the combination of dents and bumps in the surface

of a model. The ratio of these two complementary deformations is given by Poisson's ratio ν , described as

$$\nu = \frac{-\varepsilon_t}{\varepsilon_l} \quad (4.1)$$

where ε_l we denote the longitudinal strain, while ε_t is the transverse strain. The strains considered are only permanent strains, since elastic strains in the case of plasticine are negligible.

Uniaxial strain is defined as a relative change in length

$$\varepsilon = \frac{\Delta L}{L} \quad (4.2)$$

where L is the initial length of the source model. The change in length occurs under the influence of a stress

$$\sigma = \frac{F}{A} \quad (4.3)$$

where F is the applied force at the original area A .

Figure 4-1 illustrates the deformation of a source cube model, under the pull of a longitudinal force \vec{F}_l and the counteracting push of transverse forces \vec{F}_{t_1} and \vec{F}_{t_2} .

If the deformation in figure 4-1 is considered in a right handed Cartesian coordinate system, \vec{F}_l will exert an elongation of ΔL_l and thus a longitudinal strain

$$\varepsilon_l = \frac{\Delta L_l}{L} \quad (4.4)$$

along the x axis. In response to the stretching effect, the sides of the cube will contract towards the interior, along the y and z axes, to compensate for the change of volume. The corresponding forces are \vec{F}_{t_1} and \vec{F}_{t_2} , which induce contractions $-\Delta L_{t_1}$, $-\Delta L_{t_2}$ and transverse strains

$$\varepsilon_{t_1} = \frac{-\Delta L_{t_1}}{L}, \quad \varepsilon_{t_2} = \frac{-\Delta L_{t_2}}{L} \quad (4.5)$$

Considering the initial length $L = 1$, initial volume $V = L^3 = 1$ and $\varepsilon_{t_1} = \varepsilon_{t_2}$. After the deformation, the volume becomes $V_d = (L + \varepsilon_l) * (L - \varepsilon_{t_1})^2$. Volume is preserved if $V_d = V = 1 \Leftrightarrow (1 + \varepsilon_l) * (1 - \varepsilon_{t_1})^2 = 1$. In Prager & Hodge (1951)'s book, *Theory of Perfectly Plastic Solids*, they explain that when higher powers of these small strains are neglected, they obtain $1 + \varepsilon_l - 2\varepsilon_{t_1} = 1 \Leftrightarrow \varepsilon_{t_1} = \frac{\varepsilon_l}{2}$. By linking this result to equation (4.1), it can be concluded that volume is preserved when $\nu = 0.5$.

Intuitively this means that the amount of elongation should be equal to the amount of contraction. A Poisson ratio of 0.5 signifies an incompressible (volume preserving) material and ideal plasticity. Modelling clay or plasticine tends towards this value consistently in experiments, with Crandall et al. (1971) indicating an average of $\nu = 0.434$. The engineering value of the Poisson ratio is smaller than the ideal 0.5 value. Nevertheless, it offers a significant direction towards the incompressible nature of plasticine. Ideal plasticity is assumed in the experiments of this thesis, thus approximating the Poisson ratio at 0.5.

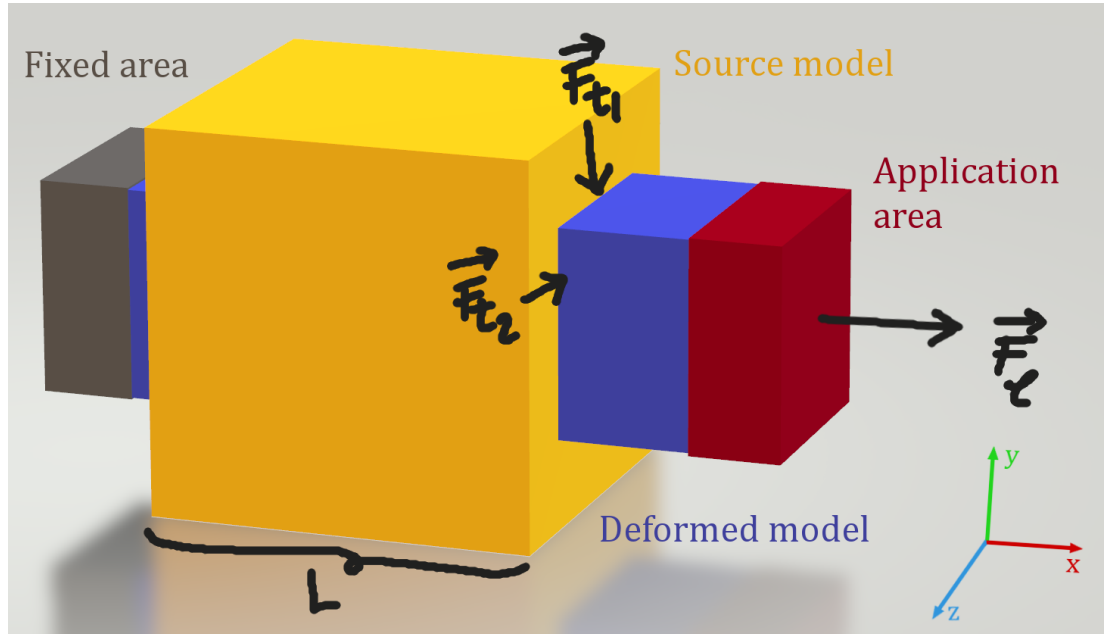


Figure 4-1: Longitudinal and transverse forces acting on a source cube model and the effect of the deformation. As a side note, the fixed and application areas, in this ideal case, are just the left and right faces of the cube, but are represented here as larger regions for visualization purposes.

It is known from material science that most objects undergoing deformation will resist a change in volume more than a change in shape. The change in volume is given by the bulk modulus B , while the change in shape is given by the shear modulus G .

Plasticine and other incompressible materials have $\frac{B}{G} \gg 1$ and $\nu \rightarrow 0.5$ (Greaves et al. (2011)). The link between Poisson's ration and the elastic moduli B , G and E (Young's modulus) is as follows.

$$\nu = \frac{3B - 2G}{6B + 2G}, \quad \nu = \frac{1}{2} - \frac{E}{6B}, \quad \nu = \frac{E}{2G} - 1 \quad (4.6)$$

The elastic moduli B , G and E are also known as the stiffness of the material. These are more suitable in the elastic domain (figure 4-2), as nonlinearities occur in the plastic domain. The average Poisson's ratio for modelling clay, however, remains approximately constant at $\nu = 0.434 \simeq 0.5$, regardless of the deformation domain (Crandall et al. (1971)). Thus it is assumed that the Clay Jam models will undergo as-volume-preserving-as-possible deformations.

4.1.2 Plasticity

Plasticine or modelling clay is a material between a fluid and a solid, with negligible elastic properties (Dewaele & Cani (2004)). When folded gently, layers, similar to skin fat, can appear inside the fold. Larger folds distribute the material locally, around the rotation point. The deformation is not propagated throughout the whole model, but rather seems to have a falloff region around the application points.

Plasticity is defined by Wikipedia as the “propensity of a material to undergo permanent deformation under load when compressed”. Materials with plastic properties undergo irreversible deformations when the applied force surpasses a yielding force. This force, also referred to as yielding stress, lies between the elastic and plastic regions on the stress-strain curve of a material (figure 4-2). The tangent to this curve can be seen as the stiffness of the material (Chakrabarty (2009)).

As the name implies, plasticine belongs mostly to the plastic domain of the aforementioned curve. This means that once a model changes shape, the deformation is irreversible. When the applied force is equal to the breaking point force, the model divides into two parts and the stress-strain curve ends. The resulting areas at the break point are coarse and frizzy.

Deformations that occur up to the breaking point depend on a series of factors like modelling tools, force and temperature. Plasticine takes the form of the tools it interacts with. If these are the artist's fingers, dents and bumps will appear in all shapes and

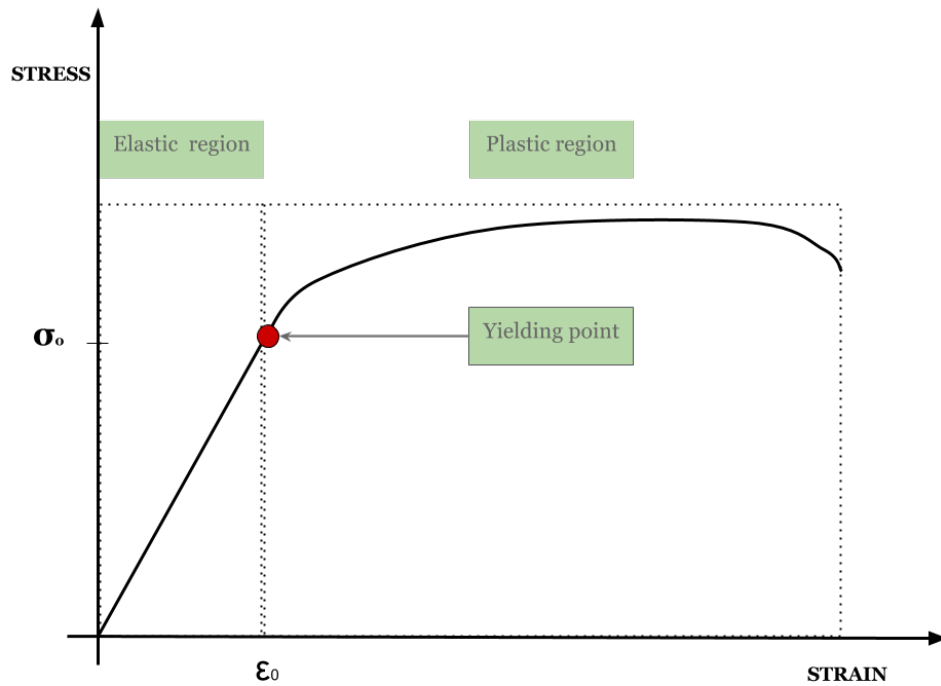


Figure 4-2: Stress-strain curve of a generic model, with its yielding point (σ_0, ϵ_0) , elastic and plastic regions.

sizes, including fingerprints. The force applied by the artist affects the object inversely proportional to the local volume of the material. The larger the quantity of material being tackled, the harder it will be to deform.

Temperature also plays a significant role in the amount of deformation a character can undergo. The propensity to yield under pressure grows with temperature. This is to do with the oily, partly fluid nature of the material which relaxes under heat. Focused specular highlights can also be observed when the plasticine is warm, while matte highlights cover it when cold.

Sharp seams are part of the look clay has when multiple chunks are clumped together. Also, sharp margins appear around dents. If the modelling tools are sharp, the resulting dents have sharp margins. Nevertheless, plasticine has a smooth appearance overall.

Stress and Strain

The deformations performed on a plasticine model can be described by stress and strain. Stress measures the amount of force applied to an area, while strain is the relative elongation that results in the process. If the case of uniaxial deformation is

considered, stress and strain are defined respectively as

$$\sigma = \frac{F}{A_0}, \quad \varepsilon = \frac{\Delta L}{L_0} \quad (4.7)$$

where F is the applied force, A_0 is the original area of application, L_0 is the initial length of the model, $\Delta L = L - L_0$ is the elongation after the deformation and L is the current length of the model.

These versions of the stress and strain are the engineering values, which approximate the true stress and strain (Chakrabarty (2009)). The true stress and strain are defined respectively as

$$\bar{\sigma} = \frac{F}{A}, \quad \bar{\varepsilon} = \ln\left(\frac{L}{L_0}\right) \quad (4.8)$$

where A is the current area of cross section and L is the current length of the model being deformed. Either type of stress and strain can be considered for the remainder of this section. The engineering version was considered during the method implementation from section 3.2.5.

Figure 4-2 shows the relationship between stress and strain. Any material has an elastic region and a plastic region of deformation. When the stress applied is within the elastic region, the model regains its initial shape when the force ceases to be active.

Once stress surpasses a yielding point σ_0 , deformation enters the plastic domain. The effects of the force load become irreversible and shape usually changes. In the case of plasticine, the elastic domain is negligible as changes become irreversible from very small stress values.

Stress in 3D

When multiple dimensions are considered, the stress at a point P becomes a tensor which can be decomposed in normal stresses $\sigma_x, \sigma_y, \sigma_z$ and tangential or shear stresses $\tau_{xy}, \tau_{yx}, \tau_{yz}, \tau_{zy}, \tau_{xz}, \tau_{zx}$ as shown in figure 4-3. Material equilibrium at point P simplifies the list due to the symmetry of the stress tensor. Hence $\tau_{xy} = \tau_{yx}$, $\tau_{yz} = \tau_{zy}$, $\tau_{xz} = \tau_{zx}$.

The six stress components $\sigma_x, \sigma_y, \sigma_z, \tau_{xy}, \tau_{yz}, \tau_{zx}$ completely define the state of stress

at a point P in 3D space. The corresponding matrix for this stress tensor at point P is

$$\sigma_P = \begin{bmatrix} \sigma_x & \tau_{xy} & \tau_{xz} \\ \tau_{xy} & \sigma_y & \tau_{yz} \\ \tau_{xz} & \tau_{yz} & \sigma_z \end{bmatrix}$$

The current description is in the xyz Cartesian coordinate system, but any choice of axes can be made. When three orthogonal axes are chosen, with the tangential stresses $\tau_{xy} = \tau_{yz} = \tau_{zx} = 0$, they become principal axes. The corresponding stresses $\sigma_1, \sigma_2, \sigma_3$ are called the principal stresses.

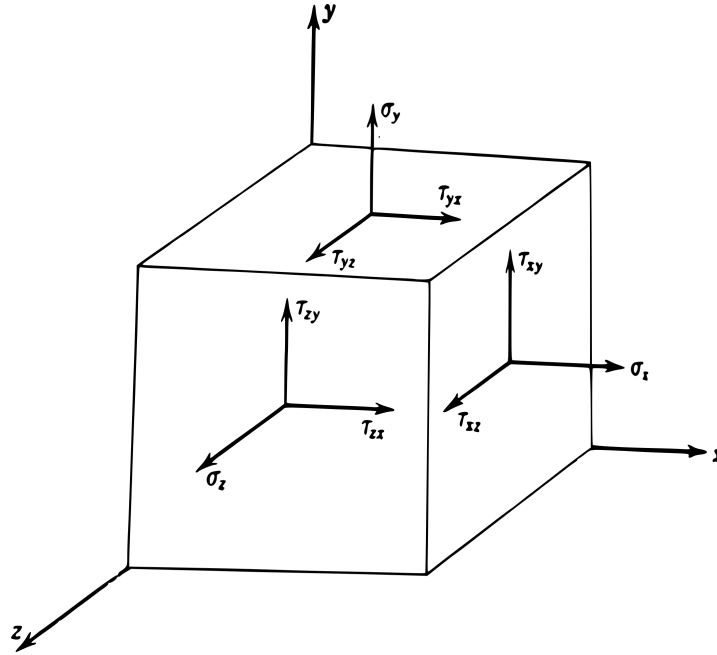


Figure 4-3: Components of the stress tensor at a point in 3D space. Image taken from Prager & Hodge (1951).

It is important to note that the sum of the three normal stresses for a point P is independent of the choice of orthogonal axes used to define the stress tensor. Also, the mean normal stress

$$s = \frac{1}{3}(\sigma_x + \sigma_y + \sigma_z) \quad (4.9)$$

is an invariant of the choice of stress tensor. Thus, as it is implied by Prager & Hodge (1951),

$$s = \frac{1}{3}(\sigma_x + \sigma_y + \sigma_z) = \frac{1}{3}(\sigma_1 + \sigma_2 + \sigma_3). \quad (4.10)$$

Strain in 3D

Following a similar line of thought, the strains when considering multiple dimensions are now analyzed. The normal strains corresponding to the normal stresses $\sigma_x, \sigma_y, \sigma_z$, as defined in Prager & Hodge (1951), are

$$\epsilon_x = \frac{\partial u_x}{\partial x}, \quad \epsilon_y = \frac{\partial u_y}{\partial y}, \quad \epsilon_z = \frac{\partial u_z}{\partial z} \quad (4.11)$$

where u_x, u_y, u_z are the displacements of point P in the x, y, z directions respectively.

Similarly, the tangential or shear strains are defined as

$$\gamma_{xy} = \frac{\partial u_y}{\partial x} + \frac{\partial u_x}{\partial y}, \quad \gamma_{yz} = \frac{\partial u_z}{\partial y} + \frac{\partial u_y}{\partial z}, \quad \gamma_{zx} = \frac{\partial u_x}{\partial z} + \frac{\partial u_z}{\partial x} \quad (4.12)$$

To intuitively understand the tangential strain, imagine two perpendicular lines overlapping the x and y axes respectively. The deviation from the originally right angle between these two lines is the γ_{xy} tangential strain.

The symmetric strain tensor at a point P is obtained by combining the normal and tangential strains in tensor

$$\varepsilon_P = \frac{1}{2} \begin{bmatrix} 2\epsilon_x & \gamma_{xy} & \gamma_{xz} \\ \gamma_{xy} & 2\epsilon_y & \gamma_{yz} \\ \gamma_{xz} & \gamma_{yz} & 2\epsilon_z \end{bmatrix}$$

The choice of axes for which the tangential strains are zero, corresponds to the principal strains $\epsilon_1, \epsilon_2, \epsilon_3$. Similarly to the stress tensor, the mean normal strain e is invariant to the choice of Cartesian coordinate system. Thus the following equalities stand

$$e = \frac{1}{3}(\epsilon_x + \epsilon_y + \epsilon_z) = \frac{1}{3}(\epsilon_1 + \epsilon_2 + \epsilon_3) \quad (4.13)$$

where $\epsilon_x, \epsilon_y, \epsilon_z$ are the normal strains and $\epsilon_1, \epsilon_2, \epsilon_3$ are the principal strains.

Stress-strain Relations

When a model undergoes deformation there are two sets of forces acting on it, the internal and external forces. The external forces are applied by the artist when pulling

and pushing the material, while the internal forces are the corresponding reaction of the volume and surface of the model.

Consider the volume as a set of unit volume finite elements. The normal and tangential stresses discussed so far are examples of internal forces, which appear as a response to an external force $F(X, Y, Z)$ per unit volume. The relation between the two types of forces reflects how the model changes. When an object tends to preserve its initial shape, the following equations of equilibrium are in full effect.

$$\begin{aligned}\frac{\partial \sigma_x}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} + \frac{\partial \tau_{zx}}{\partial z} + X &= 0 \\ \frac{\partial \tau_{xy}}{\partial x} + \frac{\partial \sigma_y}{\partial y} + \frac{\partial \tau_{yz}}{\partial z} + Y &= 0 \\ \frac{\partial \tau_{zx}}{\partial x} + \frac{\partial \tau_{yz}}{\partial y} + \frac{\partial \sigma_z}{\partial z} + Z &= 0\end{aligned}\tag{4.14}$$

The zeroing out of the sum of internal and external forces can be intuitively understood by Newton's third law of motion. This states that "For every action, there is an equal and opposite reaction". By identifying the strains a model undergoes from equation (4.14) and equations (4.15), as suggested by Prager & Hodge (1951), it becomes intuitive to obtain the new positions of the component finite elements.

When the model is perfectly elastic, Hooke's laws for the shear modulus G and bulk modulus B (section 4.1.1) are adopted. Since this work investigates perfectly plastic objects, the equations describing the motion of a viscous fluid as stated by Prager & Hodge (1951) were of more interest. These are

$$\begin{aligned}s_x &= 2\mu\dot{e}_x, \quad s_y = 2\mu\dot{e}_y, \quad s_z = 2\mu\dot{e}_z \\ \tau_{xy} &= \mu\dot{\gamma}_{xy}, \quad \tau_{yz} = \mu\dot{\gamma}_{yz}, \quad \tau_{zx} = \mu\dot{\gamma}_{zx}\end{aligned}\tag{4.15}$$

where $s_x = \sigma_x - s$, $s_y = \sigma_y - s$, $s_z = \sigma_z - s$ are the normal components of the stress deviation tensor, τ_{xy} , τ_{yz} , τ_{zx} are the original tangential stresses, $\dot{e}_x, \dot{e}_y, \dot{e}_z$ are the rates of change of the strain deviation tensor components $e_x = \epsilon_x - e$, $e_y = \epsilon_y - e$,

$e_z = \epsilon_z - e$, $\dot{\tau}_{xy}, \dot{\tau}_{yz}, \dot{\tau}_{zx}$ are the rates of change of the original tangential strains and μ is the coefficient of viscosity.

The important idea reflected by equations (4.15) is that in a plastic material, the stress deviation from the mean stress is directly proportional to the rate of change of the strain deviation from the mean strain. In an elastic material, the stress deviation would be proportional to the strain deviation itself, since the relationship between the two components is linear. Equations (4.15) thus explain the nonlinearity seen in the plastic domain (figure 4-2).

Lastly, an equation which is relevant to the tendency of plasticine to preserve its volume is the incompressibility equation (Prager & Hodge (1951)). If v_x, v_y, v_z are the velocity components of a vector \mathbf{v} , applied at point P , then the sum of strain rates is

$$\dot{\epsilon}_x + \dot{\epsilon}_y + \dot{\epsilon}_z = 0 \quad (4.16)$$

where

$$\dot{\epsilon}_x = \frac{\partial v_x}{\partial x}, \quad \dot{\epsilon}_y = \frac{\partial v_y}{\partial y}, \quad \dot{\epsilon}_z = \frac{\partial v_z}{\partial z} \quad (4.17)$$

Equation (4.16) is known as the condition of incompressibility. Section 4.2.3 delves more into how this equation links to solenoidal vector fields, which produce volume preserving deformations. It is linked with Poisson's ratio (equation 4.1) in the sense that whatever strain occurs in one direction is counterbalanced by the inverted strains in the remaining orthogonal directions.

Stiffness

The stress and strain relations described in the previous section showed the nonlinear nature of plasticity and the volume preserving properties of plastic models. The underlying characteristic that connects stress and strain and creates such material specific effects is the model's stiffness. Stiffness is the propensity of a material to yield under an external force. In its simplest form, uniaxial stiffness at a point P can be approximated by

$$k = \frac{\sigma}{\epsilon} \quad (4.18)$$

where σ and ε are the uniaxial stress and strain respectively.

By linking equation (4.18) to figure 4-2, the stiffness at a point P on the curve, for a given moment in time, is the equivalent to the slope of the stress-strain curve at that point. In the elastic domain equation (4.18) describes the elastic modulus E , also known as Young's modulus. In the plastic domain, stiffness is given by the rates of change of the stress and strain. The corresponding plastic modulus

$$H = \frac{d\sigma}{d\varepsilon} \quad (4.19)$$

represents the slope of the curve in the plastic domain.

The elastic and plastic moduli come together in a tangent modulus T , which represents the slope at any point on the stress-strain curve. The relationship between the elastic, plastic and tangential moduli (Chakrabarty (2009)) is given by

$$\frac{1}{T} = \frac{1}{E} + \frac{1}{H} \quad (4.20)$$

Another way to look at the stress-strain relationship and, implicitly, at stiffness is through the Ramberg-Osgood formula (Ramberg & Osgood (1943))

$$\varepsilon = \frac{\sigma}{E} + \alpha \frac{\sigma_R}{E} \left(\frac{\sigma}{\sigma_R} \right)^m \quad (4.21)$$

where E is Young's modulus, $\sigma_R \geq \sigma_0$ is the reference or yielding stress and α, m are constants dependent on the material. Equation (4.21) represents the total strain ε of an area when it is stretched or compressed uniaxially. It is composed of the sum of the elastic strain ε_e and plastic strain ε_p , where

$$\varepsilon_e = \frac{\sigma}{E}, \quad \varepsilon_p = \alpha \frac{\sigma_R}{E} \left(\frac{\sigma}{\sigma_R} \right)^m \quad (4.22)$$

$$\varepsilon = \varepsilon_e + \varepsilon_p \quad (4.23)$$

In Chakrabarty (2009) the author simplifies the stress-strain relationship even further by adopting the Ludwik curve formula

$$\sigma = \begin{cases} E\varepsilon & \text{if } \varepsilon \leq \frac{\sigma_0}{E} \\ \sigma_0 \left(\frac{E\varepsilon}{\sigma_0} \right)^{\frac{1}{m}} & \text{if } \varepsilon \geq \frac{\sigma_0}{E} \end{cases} \quad (4.24)$$

where $\sigma_0 \leq \sigma_R$ is the initial yielding stress. A material's work hardening property usually increases this value to σ_R . Figure 4-4 shows the difference between the Ramberg-Osgood and Ludwik curves. Notice how different values of m and $n = \frac{1}{m}$ increase or decrease the slope of the curve in the plastic domain and, consequently, the plastic modulus or stiffness.

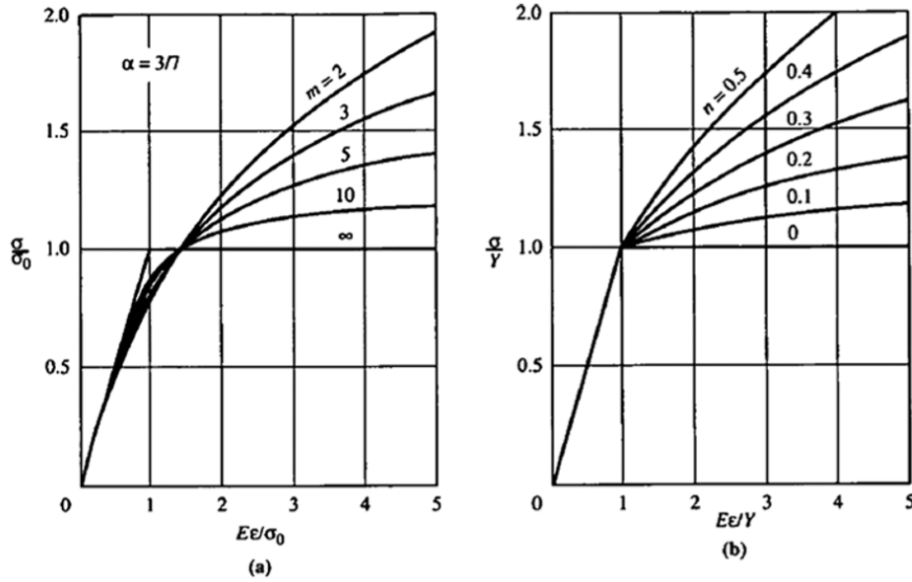


Figure 4-4: Stress-strain curves for material that display both elastic and plastic properties. (a) Ramberg-Osgood curve; (b) Ludwik curve. The image is a copy from Chakrabarty (2009).

In section 4.4.1 the Ludwik equations (4.24) were used to define stiffness locally on a surface mesh. The aim was to imitate the material and structural properties of a deforming plasticine model. If a point undergoes larger strains, smaller local stiffnesses were adopted to allow such deformations.

4.1.3 Part Based Modelling

Although the focus so far has been on material properties of plasticine, the structure of the model plays a significant role in its flexibility. A large block of modelling clay is less likely to deform than a small one. Thus it became necessary to study the component parts of Clay Jam characters.

Plasticine models are irregular in both shape and symmetry. Complex models are usually composed of several parts, of different colours, to make them stand out. This may also cause colour leaking from different chunks of material stuck together. Stretching of character limbs is usually moderate, in order to keep a uniform texture. Too much stretching can introduce streaks and ruptures and they are usually avoided.

Character rigs can be used for models with limbs undergoing medium or large deformations or to constrain the movement of the models. Skeletal rigs are sometimes made out of foam, which reduces plasticity and mobility, especially in the torso area. However, characters become lighter and easier to pick up and move around. If not rigged, limbed models have the mobility of a spline rig, rather than a rigid set of bones.

The flexibility of a character depends on its volume and mass. Some characters present an animation of their shape (eg. squash and stretch, inflate and deflate), while others have animations which can be mimicked by skeletal motions. Body parts can change in shape and size from frame to frame (eg. eyes, tongue). Some parts are animated separately from their body. Others appear out of a surface over a small set of frames. The surface from which they protrude may break or have ripple effects.

Lastly, each animation frame introduces texture details on the character (eg. fingerprints, dents, bumps, scratches). Smoothing is sometimes done when the plasticine breaks in order to hide the coarse effect. Some characters aren't smoothed at all and ruptured plasticine can be seen where limbs were bent too much. As an observation, although effects like fingerprints, dents and bumps are a crucial part of the effect clay animation has, they are at a level that can be imitated with bump or displacement maps from the corresponding textures.

The following two sections can be considered as a state-of-the-art report on ways of representing and deforming virtual clay. From visco-plastic simulations to surface and space based deformation, virtual clay is a popular topic in the CG field.

4.2 Representation of Virtual Clay Models

After examining the physical properties of plasticine, the next step was to find a suitable representation for models made out of this material. Representation in this case means the graphical reconstruction of such models, suitable for subsequent deformation and rendering. The aim was to discover the simplest modelling technique that displays plasticine behaviour, as observed in the previous sections (eg. volume preservation, localized stiffness), and rendering in real time.

Since the output of the animation reconstruction pipeline was intended for games that run on tablets and phones, a surface based modelling method was preferred over a realistic physics simulation. Nevertheless, for the purpose of understanding the advantages and disadvantages of multiple techniques used to represent virtual clay, a thorough investigation was made. The classification from figure 4-5 gives an overview of different representation methods for sculpting or simulating virtual clay. All types of representations are described, in order to find useful links between them.

4.2.1 Volumetric Models

Meshless and Finite Element Methods

Meshless systems were introduced by Terzopoulos et al. (1989) and Tonnesen (1991) and were used for representing viscous material by elastic particle-based models. Terzopoulos et al. (1989) propose a mass spring system, governed by Lagrangian equations of motion, while Tonnesen (1991) explore a molecular dynamics model, based on Newtonian equations of motion.

Stiffness of the material is simulated through the heat equation, with warm, soft regions being more malleable than cold, stiff regions. Volume preservation is not considered, but surface tension is acknowledged in the use of long range attraction and short range repulsion Lennard-Jones equations (Tonnesen (1991)). Visualization is achieved through non-real-time rendering of implicit surfaces. Cani-Gascuel & Desbrun (1997) extend this form of visualization with collision detection and a distance based form of material stiffness.

Behaviour similar to plastic flow was obtained by Desbrun & Gascuel (1994) and Desbrun & Gascuel (1995), who use hybrid models of particles and implicit surfaces to imitate plastic and elastic deformations respectively. Volume preservation is considered by defining and controlling a local volume around each particle (Desbrun & Gas-

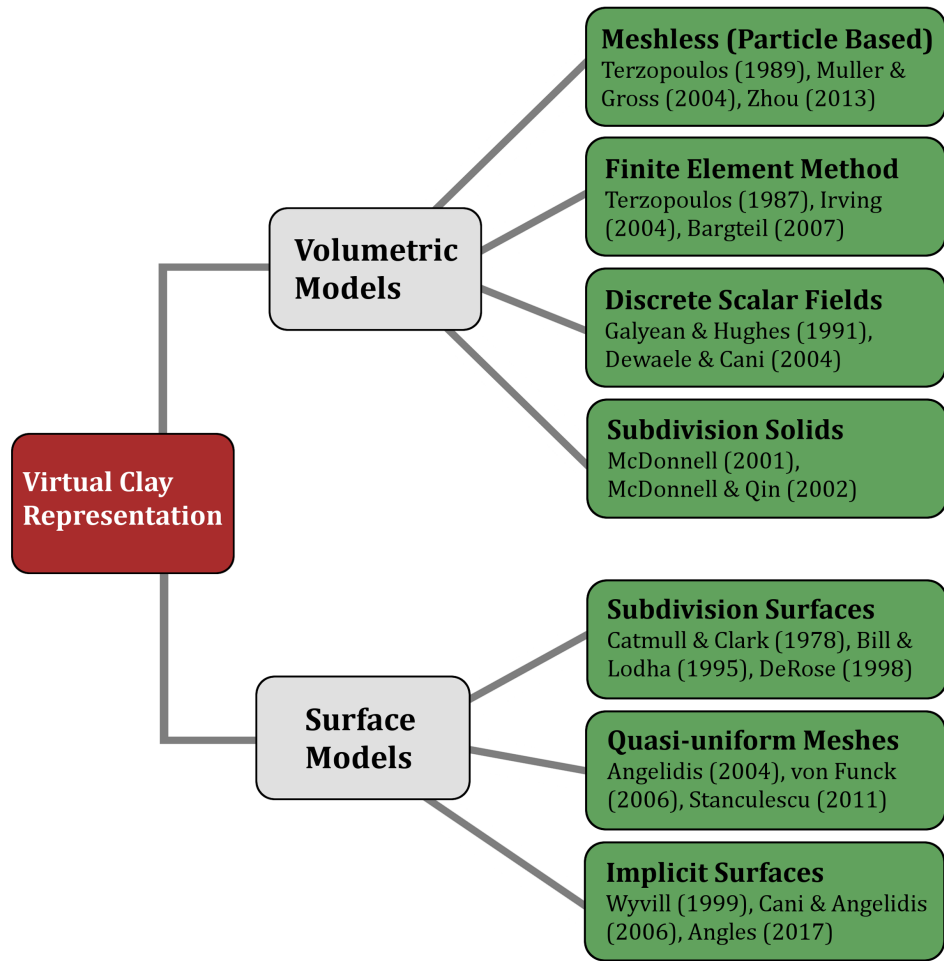


Figure 4-5: Classification of virtual clay representation methods.

cuel (1995)). Local stiffness is also defined as the norm of the field function gradient and has elastic qualities. Plasticity is simulated by introducing an adaptable damping coefficient for the forces acting on the model (Desbrun & Gascuel (1994)).

Physically-based solid models were introduced to the graphics and vision communities to simulate elastic (Terzopoulos et al. (1987)) and plastic (Terzopoulos & Fleiseher (1988)) deformations. The geometry of their models is discretized from continuous surfaces and solids and is attached to mass-spring meshes. One main advantage of this model representation, also known as the finite element representation, is the concept of an objects rest shape. This gives an object the ability to locally absorb the deformations that exceed a given threshold. Material properties such as creeping, yielding and work hardening are simulated through a combination of elastic and plastic springs. Volume

preservation is not considered, however and stiffness is set as a geometric rather than a physical property.

The finite element method (FEM) was extended with an enhanced plasticity model by O'Brien (2002). The authors adopted a simple additive method of plasticity, which preserved volume more realistically than previous techniques. Elastic strain and plastic strain are added to give the total strain of deformation, hence the name for this method. One disadvantage of finite element methods, which has been tackled in subsequent research, is the numerical instability issue. Large elastic or plastic deformations may produce artefacts in the finite element representation and thus give erroneous renders.

Bargteil et al. (2007) corrected the numerical instability by remeshing the simulation domain whenever the basis functions become ill-conditioned during deformations. A new high-quality finite-element mesh is constructed, which preserves boundary details, while improving the quality of the tetrahedral elements. Moreover, their model can handle plastic deformations which preserve volume and include work hardening or softening over time. Large plastic deformations, as well as topology changes are permitted and effects like squashing, dents and flow effects can also be obtained. Lastly, connectivity between elements speeds up the simulations as nearest neighbor searches are not needed, unlike particle-based approaches.

Nevertheless, frequent remeshing for larger deformations is time consuming and can introduce numerical errors and artefacts, due to resampling and smoothing of the physical properties of the material (Dewaele & Cani (2004), Bargteil et al. (2007)). Dewaele & Cani (2004) also show that finite element methods do not allow the simulation of very large deformations, including relative matter displacement inside the shape, carving, fusion and separation. Enabling topological changes, such as separation of matter into several pieces, appears incompatible with real-time performance, even in more recent models (Bargteil et al. (2007)).

This is a consequence of the necessity of exploiting the materials internal structure for pre-inverting matrices or pre-computing hierarchies when deformations take place. Moreover, the high number of polygons needed for smooth results, brings the issue of longer, non-real time processing periods (Bargteil et al. (2007)). Wicke et al. (2010) proposed a local remeshing algorithm to limit these issues, without completely removing them.

Eulerian fluid simulations were also used to represent clay-like materials by Stam (1999) and Stam (2002), with real-time performance. Issues emerged as the fluid was occupying the entire space, in contrast to clay, which is bounded by surface tension properties.

Goktekin et al. (2004) pioneered Eulerian methods for an elastic forces based fluid simulator. The measure of deformation was obtained by integrating the velocity gradient across time. Although useful for animating materials that behave like fluids, no direct control of the shape is available in their model and the shape can drift over time. This is a consequence of the underlying volumetric representation, which is not explicit, leading to irreversible deformations.

Müller & Gross (2004) first proposed a unified particle-based approach to model all of elastic, plastic and melting behavior of objects. The authors use the Moving Least Squares (MLS) algorithm to derive a deformation field from particle positions between the resting state and deformed state. Their approach is similar to the Eulerian methods applied by Goktekin et al. (2004).

The main difference is that the Müller & Gross (2004) model directly computes deformation during each timestep, via a simple additive technique, while the Goktekin et al. (2004) one integrates deformation rates across time. The former authors use both explicit and implicit integration schemes in their formulation. Stability issues existent in the simpler explicit numerical integrator are thus reduced by using implicit numerical integration. Although this allows for both small and large plastic flow deformations to occur, errors can still be introduced over time, leading to the incapacity of calculating the total deformation of an object.

The meshless method described by Müller & Gross (2004) inspired the research of Irving et al. (2004), who developed similar algorithms on finite element elasto-plastic simulations. The authors proposed a multiplicative model of plasticity such that larger plastic deformations can be controlled. Their simulation still suffers from instability when the basis matrices of the elements are ill-conditioned. Further improvements to the Irving et al. (2004) model were made by Bargteil et al. (2007) and Wojtan & Turk (2008).

Extensions to the Müller & Gross (2004) model also emerged in the works of Keiser et al. (2005) and Pauly et al. (2005). The former introduced the Navier-Stokes equation to the unified meshless method, allowing the transition from elasto-plastic materials to fluids. Pauly et al. (2005) on the other hand, used the meshless plasticity method for modelling fracturing materials. Changes in topology influence the updates more than changes in plastic flow in their work. They also introduced the concept of updating the shape functions for each simulation particle.

Gerszewski et al. (2009) introduce a multiplicative and deformation gradient model, which, unlike the additive model presented by Müller & Gross (2004), can handle large

plastic deformations. Explicit numerical integration, however, causes stability issues when the timesteps are not sufficiently small.

Zhou et al. (2013) extended Irving et al. (2004), Bargteil et al. (2007), Gerszewski et al. (2009) to particle-based simulation for elasto-plastic materials with implicit numerical integration. Similarly to Gerszewski et al. (2009), the authors use a multiplicative plasticity and deformation gradient formulation. The original finite elements method is substituted with a particle-based method, inspired by the meshless models surveyed in Gross & Pfister (2007) work.

One of the main contributions of Zhou et al. (2013) is limiting the deformation computation to a sparse Laplacian linear system, which can be efficiently solved, especially on the graphics hardware. Another feature of their method is that the plastic deformations are postponed until the end of a time step, after all the elastic deformations have been computed. This permits stable elastic and plastic simulations even for large time steps. Drifting issues, such as the one present in Gerszewski et al. (2009) are no longer part of the simulation.

All in all, the model presented by Zhou et al. (2013) preserves numerical stability even with larger time steps, is memory-efficient (Laplacian linear system is sparse) and greatly speeds up the computation time, particularly with highly plastic or stiff materials. It can handle solids with a much wider range of material properties (stiff to soft) compared to existing techniques. Lastly, large elastic or large plastic deformations provide stable results. However, the deformations are not detail oriented, are difficult to render due to the meshless nature of the technique and cannot be obtained in real time.

In conclusion, both finite element and meshless methods can simulate the physical properties of clay. Topology changes and large scale deformations, although previously known to create issues such as artefacts in the finite element method remeshing technique, have progressed both in quality and speed. However, simulation based methods have not entered the real-time domain and offer limited directable options for users.

Skeletons or other deformation methods would be hard to adapt on such systems. Rendering is also an issue, since particle based methods are mainly represented by point clouds, while finite element meshes change frequently when modifications occur. The history of numerical instability in simulations is also a disadvantage that discouraged the choice of such methods for clay model representation. Physical accuracy will thus be compromised slightly for the sake of a more robust technique that offers satisfactory results from both a clay deformation point of view, as well as a shape matching one.

Discrete Scalar Fields in 3D Grids and Subdivision Solids

Three dimensional grid based methods have been regarded as suitable representations for models which undergo physically plausible clay-like deformations. This is due to the control such methods have over the mass of the material and how it is distributed throughout the object. Plastic flow, volume preservation and localized stiffness are imitated rather than simulated, resulting in physically plausible rather than physically accurate results.

Galyean & Hughes (1991) were the pioneers of the discrete scalar field representation model. Field values were stored on a regular 3D grid called a voxmap, which is analogous to a pixmap in a 2D image. They used tools to add or remove material from each voxel at the intersection point and then render the boundary surface with a marching cubes algorithm. The rendering time was $O(n^3)$ where n is the number of voxels in the solid. This made it difficult to have highly detailed models in a reasonable amount of time.

A similar concept was later presented by Arata et al. (1999) who used cellular automata to perform free-form deformation of clay-like objects. Deformation was achieved via mass transportation of the material from high density cells to low density cells, in the direction of an external force. Rather than simply adding or removing material with the help of a tool, their approach imitates the plastic flow of modelling clay when subjected to stress. Although this simplification of a natural phenomenon gives appealing results, similar rendering time issues make it tedious to use. It is worth noting, however that processing times are lower than physically-based simulation times. This observation has led to extensions done by Dewaele & Cani (2004) and Cani & Angelidis (2006), who further develop physics mimicking clay modelling techniques.

Dewaele & Cani (2004) propose a layered approach consisting of plasticity, mass conservation and surface tension, adopted to allow global and local deformations. This brings coherence to the results, helping them imitate real clay in a believable manner. Properties like extreme plasticity, topological changes, constant volume, surface tension are possible with this technique. Moreover, plasticine effects like bending, holes, folds, prints appearing in the model can be obtained.

Similarly to discrete scalar fields, subdivision solids have the appearance of a grid, with the field function being a volumetric model like a trivariate B-spline (McDonnell et al. (2001)). They are an extension of the subdivision surfaces proposed by Catmull & Clark (1978), who divided a surface recursively into smooth bicubic B-spline patches. In the case of subdivision solids, masses and spring networks were attached to the

subdivided solid points. Elastic and plastic deformations were made possible through varying the point masses and spring stiffnesses.

McDonnell et al. (2001) and McDonnell & Qin (2002) use subdivision solids as a physically-based modelling technique, underlining how a single control lattice for a subdivision solid is sufficient for representing a topologically complex, smooth, deformable object. Stiffness is painted onto the model by the user, which allows varied material properties. Plasticity is assumed to be a natural extension of elasticity, through changing spring stiffnesses and adding damping forces. Volume preservation is not considered, since the tool is intended for more free-form deformations.

The main features of their model include topological changes, fine level details, via a local, adaptive subdivision algorithm, sharp features, such as corners and creases, stiff areas, inflation and deflation operations for effects like bumps and tapering respectively. They are ideal for level-of-detail (LOD) control since a model can be subdivided until the desired amount of detail is achieved. However, a user interactive tagging system is required for attributes to be assigned to the appropriate areas of the model. For example, topological changes, stiff areas and sharp features all need the user to tag which cells should undergo such transformations (McDonnell & Qin (2002)).

Both discrete scalar field grids and subdivision solids are visualised by rendering the surfaces enclosing the manipulated volumes. While relatively fast rendering can be achieved for subdivision surfaces, iso-surfaces of volumetric grids are notorious for slow performances. Galyean & Hughes (1991) voxel data visualization is achieved with an iso-surface using the marching cubes algorithm. Optimizations were made in order to only revisualize the area where modifications appeared. Dewaele & Cani (2004) note, however, how marching cubes is computationally expensive, as grid connectivity can change during sculpting, especially with large deformations.

Cani & Angelidis (2006) extended the work of Dewaele & Cani (2004), allowing interactive frame rates for all shape creation and editing methods. This was partly achieved by multi-resolution sculpting, which enables real-time interaction with a sculpted object at any modelling scale. However, rendering artefacts may still appear between adjacent cells of different sizes.

Ray casting is an alternative rendering technique for implicit surfaces. It can also be expensive, because triangles, extracted from the iso-surface of the 3D grid, need to be recomputed each time a part of the model moves. This can cause details to be blurred by the successive re-samplings of the grid. Furthermore, the triangles used for display are not persistent during the editing process, making operations like texturing difficult

(Stanculescu et al. (2011)).

Real-time sculpting and rendering started to appear after multi-resolution based storage optimisation methods were proposed (Bærentzen (1998), Raviv & Elber (2000), Perry & Frisken (2001), Ferley et al. (2001)). In the Dewaele & Cani (2004) method, triangles only update where density changes, allowing for real-time performances. Cani & Angelidis (2006) introduced multi-resolution implicit sculpting, which allows storing the scalar field in a hierarchical grid that dynamically subdivides or un-divides itself according to the size of the tool being used. This mechanism gives control over the number of displayed cells at each frame and dynamically selects a level-of-detail dependent on the distance to the projection plane. Multi-resolution sculpting enables real-time interaction with a sculpted object at any modelling scale.

On the subdivision solids side of things, a local, adaptive subdivision algorithm has already been mentioned as a contribution from McDonnell & Qin (2002) for creating fine details. Both an explicit and an implicit solver were also implemented in their system. The explicit numerical solver is founded upon the energy-based Lagrangian equation of motion, is efficient and easy to implement, while the implicit solver offers numerical stability to calculations at every time step.

In conclusion, both discrete scalar field and subdivision solid representations offer a relatively accurate physically-based representation of clay, with real-time performances. They were both enhanced with multi-resolution techniques, allowing for localised, detailed deformations of the models. Both methods were utilised mostly in user interactive sculpting applications. Subdivision solids exhibit a better performance in rendering and texturing as the resulting surfaces are polygon based, with a smooth, consistent topology. Iso-surfaces covering scalar grids are bound to suffer from artefacts when the object is deformed, due to resampling the discrete scalar grid at render time.

The grid based representation is closer to a plastic model, while the subdivision solid representation is closer to an elastic model. The former allows properties such as extreme plasticity, topological changes, constant volume, surface tension and effects like bending, holes, folds, prints, without the need for the users input on the model. The latter allows topological changes, localized details, smoothness, sharp features, stiff areas, inflation and deflation, when corresponding user input is given in the model tagging system.

Far from being an exhaustive survey of virtual clay represented by volumetric or subdivision-solid systems, the goal was to underline the earlier history behind such methods and their use. However, virtual clay structures remain popular mostly in the

early 2000s, which is why references mostly revolve around this period. It is worth noticing, that even from this small list of authors, a combination of the volumetric properties of discrete scalar grids and the rendering properties of polygonal surfaces would seem like a suitable combination.

4.2.2 Surface Models

Volumetric models and physics simulations can thus represent the nature of plasticine in a physically plausible manner. The characteristic issues, however, are slow rendering times, inconsistent mesh connectivity, numerical instabilities due to remeshing and lack of directability. These obstacles are difficult to overcome when the ultimate goal is to aid the animation reconstruction pipeline with a virtual clay deformation technique. Preserving the mesh connectivity and allowing the user to direct the deformation are compulsory for accomplishing this aim.

Considering the qualities of volumetric models that allow plasticine properties to emerge in the deformation, similar properties were sought after in surface based representations. The literature on surface based virtual clay representations brought insight into how they link to the physics of plasticine. The question was whether a surface based representation is sufficient to imitate plasticine or clay properties when undergoing deformation.

Section 4.2.3 discusses the link between volumetric and surface based representations more in depth, while section 4.3 gives the background for deformation algorithms. The focus is now of surface based representations of virtual clay that may be used for subsequent sculpting or deformation algorithms.

Surface Representation Overview

Surfaces in computer graphics are orientable, continuous 2-manifolds which are mathematically described by parametric or implicit representations in \mathbb{R}^3 (Botsch et al. (2010)). A *parametric surface* is defined by a function $f : \Omega \rightarrow S$ that maps a two dimensional parameter domain $\Omega \subset \mathbb{R}^2$ to the surface $S \subset \mathbb{R}^3$ embedded in Euclidean space. It is also said that surface S is defined explicitly by function f .

Discretizations of these surfaces have appeared as piecewise polynomial surface representations (Botsch et al. (2010)). Different levels of the original surface smoothness can be approximated, depending on the degree of the polynomial. Examples range from spline surfaces (NURBS) with C^{n-1} smoothness to polygonal meshes of C^0 smoothness.

Although continuity is preserved better than fairness in the case of polygonal meshes, they are the preferred representation in graphics, due to their high rendering speeds.

A polygonal mesh M contains vertices V , edges E and faces F . Faces are commonly quads for modelling and animation purposes and triangles for rendering purposes. The link between these parameters is the Euler characteristic $\chi = V - E + F$, which reveals the genus of a mesh. The genus, or the number of holes in a mesh, is defined as

$$g = \frac{2 - \chi}{2} \quad (4.25)$$

and is zero for meshes that are topologically equivalent to a sphere.

An *implicit surface*, on the other hand, is defined as the zero set of a scalar valued function $F : \mathbb{R}^3 \rightarrow \mathbb{R}$. The coordinates of the surface points p are not known beforehand, but they can be found implicitly by solving the equation $F(p) = 0$. Hence the implicit surface can be defined as $S = \{p \in \mathbb{R}^3 | F(p) = 0\}$. The implicit surface has a direct link to volumetric models, since they are the boundary of a continuous scalar field, similar to the discrete scalar field grids in section 4.2.1.

Subdivision Surface Meshes

Subdivision surfaces are surface representations that lie between spline surfaces and polygonal meshes. They are considered a generalization of spline surfaces since they can represent multiple approximations of a surface, depending on the level of subdivision. The subdivision rules were introduced by Catmull & Clark (1978), but the surfaces themselves started to become more popular in computer animation (DeRose et al. (1998)). An advantage subdivision surfaces have over NURBS is their inherent continuity, whereas spline patches need additional geometric constraints to ensure continuity (Botsch et al. (2010)).

On the down side of subdivision surfaces is that each refinement step creates a deeper level in a hierarchical structure containing that surface. Hence the more subdivision steps a surface undergoes, the larger the hierarchy becomes. Also, the whole surface is subdivided in a similar manner, regardless of the amount of detail it displays in different regions. If a region is flat, for example, more polygons would not smoothen it better.

Bill & Lodha (1995) address the latter issue by presenting an adaptive subdivision and smoothing system, which allows the user to add details or flatten out artefacts in a specified region. This has the advantage of allowing mesh complexity in an efficient

manner, as polygons are only added when needed. Although the authors mention physical properties of a mesh as being a famous question in computer graphics, their proposed sculpting technique is purely geometric.

As Dewaele & Cani (2004) remark, topological changes and constant volume deformations are complicated to adapt to models like Bill & Lodha (1995) as they require user intervention. While their winged-edge object representation method is easy to implement and render, it can require a large storage size and higher processing times, especially for smooth or detailed objects. Geometric continuity is much more difficult to control due to the discrete nature of the method proposed by Bill & Lodha (1995).

Quasi-uniform Meshes

Quasi-uniform meshes are triangular meshes where vertices are distributed uniformly across the surface (Stanculescu et al. (2011)). The edge lengths in a quasi-uniform mesh are kept within a threshold set by the user. Given a maximum distance d_{detail} , each edge will be smaller than this value and, whenever possible, larger than $\frac{d_{detail}}{2}$.

Quasi-uniform meshes are a combination of space deformations and grid based methods. Volume preserving space deformation methods are based on the sweepers presented by Angelidis & Wyvill (2004) and von Funck et al. (2006), while Galyean & Hughes (1991) and Dewaele & Cani (2004) provide inspiration with their topologically dynamic grid based structures.

Topological changes may be required when parts of the character are removed or when limbs bend to touch the body. Although less frequent, this property would be a useful feature, especially when working with real plasticine characters. Stanculescu et al. (2011) observe that grid based methods allow topological changes, but with compromises in mesh coherence, texturing and detail preservation due to successive resampling when the mesh moves.

The quasi-uniform mesh sculpting system can handle intuitive topological changes, detail and volume preservation. Moreover, their surfaces are always closed, do not suffer from self-intersections and permit texturing and fast rendering, due to the triangular approach (Stanculescu et al. (2011)).

Topological genus modification is one of the advantages of quasi-uniform meshes over adaptive meshes (Bill & Lodha (1995)). Similar to grid based methods (Dewaele & Cani (2004)), topological changes are enabled by a given resolution-based threshold. A predefined distance between vertices allows splits and merges of the mesh via Laplacian

editing. By modifying the distance threshold, physical properties of the mesh can be imitated.

The proposed algorithm provides a uniform, time-coherent, quality triangulation of any given mesh provided as input. The level-of-detail is defined by the sampling resolution, without losing the adaptive topology characteristic. A disadvantage of this approach, however, is that objects have a similar polygon count in both complex and simple areas. Also, a large number of triangles is needed to approximate smooth surfaces, while sharp features are hard to maintain due to the merge/split threshold.

Despite these observations, real-time performance is made possible by several optimizations. Mesh connectivity and vertex positions are stored directly on the GPU, while modifications are made only in deformed regions. Connectivity and sampling of the vertices evolve dynamically when needed. Parallelization on the GPU is more difficult than with multi-resolution models, however. Collision tests for intersection prevention are also time consuming, although a spatial subdivision method is used for speed enhancements (Stanculescu et al. (2011)).

All in all, the quasi-uniform mesh seems to be a good match for a virtual clay representation that can preserve the physical properties of the material and render in a reasonable amount of time. This is a helper worth considering in the *Model Layer* helpers (section 3.2.4), in order to allow better registrations.

Implicit Surfaces

Implicit surfaces became popular for modelling organic, clay-like objects, due to their smooth appearance (Cani & Angelidis (2006)). Blobby or metaball surfaces (Blinn (1982)) were introduced to create object approximations through a summation of Gaussian functions. Soon after, hierarchical structures of implicit surfaces emerged under the name of Blob Trees (Wyvill et al. (1999)).

The main principle behind modelling with implicit surfaces is using combinations of primitives called skeletons, which contain scalar fields around them (Desbrun & Gascuel (1994, 1995)). Visualizing such models is done by extracting the iso-surface around the scalar field, similarly to the discrete grid renders from section 4.2.1.

Some of the advantages of sculpting with implicit surfaces are handling topological changes and self-intersections easily (Noble et al. (2004), Cani & Angelidis (2006)). Dents and bulges can be created interactively with tools whose influence depends on the distance from the surface (Noble et al. (2004), Dewaele & Cani (2004)). Also blends between various shapes can be done smoothly, due to the field functions being

differentiable (Angles et al. (2017)).

The disadvantages of implicit surfaces are mostly to do with the level-of-detail captured and rendering. A large number of primitives is required to capture all the details of a high resolution model. When these primitives need to be rendered, a marching cubes algorithm is popularly used to polygonize the surfaces for rendering purposes. Ray tracing can also be used, but its cost is even more demanding, especially if a model contains a high number of primitives. Remeshing can cause details to be lost at render time, especially when the mesh deforms (Stanculescu et al. (2011)).

4.2.3 Link Between Volumetric and Surface Models

Plasticity theory acknowledges two types of forces acting on a model. Body forces act on all particles of the model and surface forces which act only on the surface particles (Prager & Hodge (1951)). Notice that these are both internal forces, acting as a response to external forces (section 4.1.2). Various branches of calculus have attempted to find a relation between body and surface forces.

Gauss's theorem shows a powerful link between the integral over the volume of an object and the integral over its boundary surface. This is the three dimensional equivalent of **the fundamental theorem of calculus** as presented by Lubliner (2008)

$$\int_R \phi_i \, dV = \int_{\delta R} n_i \phi \, dS \quad (4.26)$$

where ϕ is any differentiable field, R is the region inside the object, while δR is the region's boundary surface, dV and dS are the changes in volume and surface area respectively. One can speculate that this link between the volume and surface reveal properties of the material like heat conductivity and stiffness, which can be expressed as differentiable functions on the surface.

One more common case is for ϕ to be replaced by v_i , the i th vector of a vector field \mathbf{v} . This is known as the **divergence theorem** and can be expressed as

$$\int_R (\nabla \cdot v_i) \, dV = \int_{\delta R} v_i \, dS \quad (4.27)$$

where $\nabla \cdot v_i = \text{div}(v_i)$ is the divergence of vector v_i . The divergence of a vector field signifies the rate at which material density is displaced from a point in space.

When $\nabla \cdot \mathbf{v} = 0$, the vector field is divergence-free or solenoidal. In other words matter that is displaced from one point in space is replaced by matter from another point. It is a well known fact that solenoidal vector fields are volume preserving (Davis & Snider (1995)). von Funck et al. (2006) create a divergence-free 3D vector field $\mathbf{v}(x, y, z) \in \mathbb{R}^3$ from the gradients of two scalar fields $p(x, y, z), q(x, y, z) \in \mathbb{R}$. They then use this vector field to produce volume preserving deformations for their models. The following equation defines \mathbf{v} and is based on the theory presented by Davis & Snider (1995).

$$\mathbf{v}(x, y, z) = \nabla p(x, y, z) \times \nabla q(x, y, z) \quad (4.28)$$

The fundamental theorem of surfaces (do Carmo (1976)), as interpreted by Terzopoulos et al. (1987), states that two surfaces have the same shape if their metric tensors G and curvature tensors B are identical functions for every point. These values are identical to the first (I) and second (II) fundamental forms of the surface (section 4.3.1). Terzopoulos's choice of letters links the two terms to the shear (G) and bulk (B) moduli presented in section 4.1.1.

The first fundamental form can be seen as measuring the stretching of a surface, while the second fundamental form measures bending. These values are often approximated with thin shell energy minimization techniques (Fröhlich & Botsch (2011)). In such scenarios surface meshes are treated as hollow objects, attempting to represent the boundary surface of a solid object. Often volume preserving terms need to be added to the deformation to ensure a physically plausible deformation.

The conditions under which surfaces may behave like solids when deforming were of great interest for this thesis. Intuitively, the way a solid deforms determines the shape of its boundary surface. The endeavour was towards finding methods that provide a reverse correlation. Volume preservation was the most sought after property for such deformation methods, since plasticine is an incompressible material. It has been shown in the previous paragraphs that solenoidal vector fields can deform a model without changing its volume.

When a divergence-free vector field cannot be intuitively determined, is there an intuitive way to preserve volume using surface based deformations? To answer this question, space and surface based methods are discussed in the next section, to determine if they allow physically plausible, volume preserving clay-like deformations.

4.3 Deformation of Virtual Clay Models

After having seen different ways of representing virtual clay, the conclusion is that the advantages of imitating the physical properties of this material outweigh the disadvantages. Also, a surface deformation technique that allows fast rendering and imitates volumetric deformation characteristics is preferred. All the choices for the representation of a virtual clay model are backed up by the necessity of fast animation times required by a game engine and the resulting tablet or phone games.

Surface deformation methods that imitate virtual clay (figure 4-6) will now be classified and studied, in order to determine the method that mostly resembles plasticine deformation. This classification comprises of surface based and space based deformation methods. Space based methods can be applied to solids as well as surfaces, as they do not depend on the intrinsic structure of the model being deformed. The focus will be on surfaces, however, and how they change shape under the influence of an external vector field. Surface based methods depend on the structure of the model and usually produce physically plausible results by minimizing stretching and bending qualities of the deforming surface.

Most of the literature focuses on elastic deformations of discrete surfaces (triangle meshes), due to the linearity of stiffness in the elastic domain (figure 4-2). The contribution of this chapter is to create a type of plastic deformation for triangle meshes by enhancing the nonlinear surface deformation method proposed by Fröhlich & Botsch (2011). The meshes used are quasi-uniform whenever possible, as their advantages over general triangle meshes have been underlined in section 4.2.2.

Let M_S represent a source triangle mesh with vertices $V_S \in \mathbb{R}^3$, faces $F_S \in \mathbb{N}^3$ and edges $E_S \in \mathbb{N}^2$. This mesh undergoes deformation when a deformation vector field, $\delta : M_S \rightarrow \mathbb{R}^3$, is applied to it. The properties of δ determine the quality of deformation. A smooth and solenoidal vector field is desired, so that there are no artefacts and volume is preserved for the resulting mesh. The following algorithms provide methods for determining a deformation vector field δ , subject to whether the intrinsic or extrinsic properties of the source mesh are considered.

4.3.1 Surface Based Deformations

Surface based deformations have become popular for the abundance of linear mesh editing techniques from recent years. The main characteristic of such methods are solving

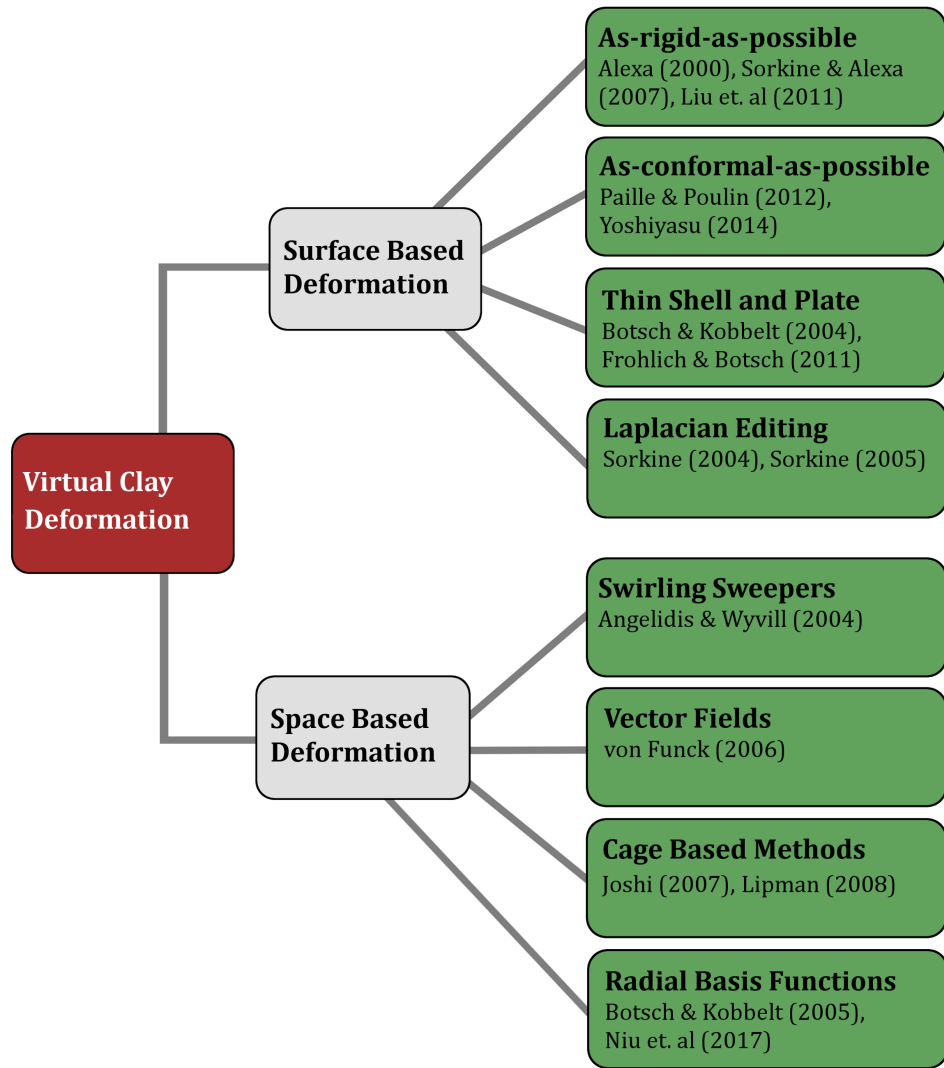


Figure 4-6: Classification of virtual clay deformation methods.

quadratic energy minimization problems, using fast linear system solvers. The careful choice of boundary conditions for such methods ensure a unique global minimum for these energies. The deformations obtained are usually smooth and physically plausible, but not physically accurate (Botsch & Kobbelt (2004), Botsch & Sorkine (2008)). Approximations of nonlinear local rotation might lead to volume changes, but the overall shape tends to be preserved.

When considering triangular meshes M_S , surface based deformations generally describe the displacement of the mesh vertices, subject to preserving geometrically intrinsic

(parametrization independent) features of the surface like lengths, areas, curvature etc. Given a fixed region \mathcal{F} and a handle region \mathcal{H} of vertices, the resulting deformation field δ describes how vertices move in the unknown region $M_S \setminus \{\mathcal{F} \cup \mathcal{H}\}$.

The desired movement preserves the afore mentioned intrinsic features as closely as possible through some form of regularization. Usually the name of the regularization comes from the name of the features being preserved. For example as-conformal-as-possible is a form of regularization that preserves triangle angles when deforming the mesh.

In order to preserve intrinsic mesh features, one must find a method of extracting them from the surface. Differential geometry offers tools to do so through the first (*I*) and second (*II*) fundamental forms (do Carmo (1976)). These are metric tensors associated with the surface stretching and bending respectively (Terzopoulos et al. (1987)).

A return to smooth or at least C^2 surfaces $S \subset \mathbb{R}^3$ will be made, to first explain how *I* and *II* encode intrinsic surface information. Afterwards, it will be revealed how elements of these fundamental forms can be discretized for triangular meshes.

First Fundamental Form

Let $p : [a, b] \times [c, d] = \Omega \rightarrow \mathbb{R}^3$ be a smooth parametrization of the surface S ,

$$p(u, v) = \begin{pmatrix} x(u, v) \\ y(u, v) \\ z(u, v) \end{pmatrix}$$

with $u \in [a, b], v \in [c, d], \Omega \subset \mathbb{R}^2$. The tangent plane $T_p(S)$ at point $p(x, y, z) \in S$ is spanned by $p_u, p_v \in T_p(S)$, corresponding to the partial derivatives of point p , with respect to u and v . These orthogonal tangent vectors form a local basis, which can be used to define more coplanar vectors. They also describe the Jacobian matrix at point $p(u, v) = p(x(u, v), y(u, v), z(u, v))$ (Botsch et al. (2010)), defined as

$$J_p = \begin{bmatrix} p_u(u, v) & p_v(u, v) \end{bmatrix} = \begin{bmatrix} \frac{\partial x(u, v)}{\partial u} & \frac{\partial x(u, v)}{\partial v} \\ \frac{\partial y(u, v)}{\partial u} & \frac{\partial y(u, v)}{\partial v} \\ \frac{\partial z(u, v)}{\partial u} & \frac{\partial z(u, v)}{\partial v} \end{bmatrix} \quad (4.29)$$

where the Jacobian J_p or simply J is a linear map that transforms a vector $\bar{w} \in \Omega$ from parameter space to a tangent vector $w \in T_p(S)$ in the surface tangent plane around

point p . In other words $w = J\bar{w}$ (Botsch et al. (2010)).

Let $w_1, w_2 \in T_p(S)$, with $w_1 = ap_u + bp_v$ and $w_2 = cp_u + dp_v$. The inner product

$$I_p(w_1, w_2) = \langle w_1, w_2 \rangle \quad (4.30)$$

of the two tangent vectors w_1 and w_2 , with $I_p : T_p(S) \rightarrow \mathbb{R}$ is called the *First Fundamental Form* at point p . I_p or simply I is a metric tensor which expresses how a surface S inherits the inner product of \mathbb{R}^3 (do Carmo (1976)). This can be expanded as

$$I_p(w_1, w_2) = \langle ap_u + bp_v, cp_u + dp_v \rangle = ac\langle p_u, p_u \rangle + (ad + bc)\langle p_u, p_v \rangle + bd\langle p_v, p_v \rangle \quad (4.31)$$

It is common in differential geometry to use the following notation: $E = \langle p_u, p_u \rangle$, $F = \langle p_u, p_v \rangle$, $G = \langle p_v, p_v \rangle$, where E, F, G are known as the *coefficients of the first fundamental form*. Equation (4.30) rewritten in coefficient form is

$$I_p(w_1, w_2) = Eac + F(ad + bc) + Gbd \quad (4.32)$$

or in tensor form as suggested in Botsch et al. (2010)

$$\mathbf{I} = J_p^T J_p = \begin{bmatrix} E & F \\ F & G \end{bmatrix} = \begin{bmatrix} p_u^T p_u & p_u^T p_v \\ p_u^T p_v & p_v^T p_v \end{bmatrix} \quad (4.33)$$

where the first fundamental form is approximated by the squared Jacobian. This is a consequence of the relationship $w = J\bar{w}$. If $w_1 = J\bar{w}_1$ and $w_2 = J\bar{w}_2$ are vectors that stem from point p , then their inner product is

$$I_p = \langle w_1, w_2 \rangle = w_1^T w_2 = (J\bar{w}_1)^T (J\bar{w}_2) = \bar{w}_1^T (J^T J) \bar{w}_2 = \bar{w}_1^T \mathbf{I} \bar{w}_2 \quad (4.34)$$

The first fundamental form can be used to measure angles, distances and areas on a surface. For example, the area of surface S is calculated as

$$A = \int \int_{\Omega} \sqrt{\det(\mathbf{I})} du dv = \int \int_{\Omega} \sqrt{EG - F^2} du dv \quad (4.35)$$

Second Fundamental Form

Similarly to *The First Fundamental Form* the *Second Fundamental Form* of a surface is a symmetric bilinear form on the tangent space $II_p : T_p(S) \rightarrow \mathbb{R}$ and is commonly used to measure curvature of a surface. The second fundamental form at a point p , with two tangent vectors w_1 and w_2 defined as before, can be written as

$$II_p(w_1, w_2) = \langle p_{uu}, \mathbf{n} \rangle ac + \langle p_{uv}, \mathbf{n} \rangle (ad + bc) + \langle p_{vv}, \mathbf{n} \rangle bd \quad (4.36)$$

or in coefficient form

$$II_p(w_1, w_2) = eac + f(ad + bc) + gbd \quad (4.37)$$

or in tensor form, as suggested in Botsch et al. (2010)

$$\mathbf{II} = \begin{bmatrix} e & f \\ f & g \end{bmatrix} = \begin{bmatrix} p_{uu}^T \mathbf{n} & p_{uv}^T \mathbf{n} \\ p_{uv}^T \mathbf{n} & p_{vv}^T \mathbf{n} \end{bmatrix} \quad (4.38)$$

where \mathbf{n} is the surface normal at point $p(u, v) \in S$, e , f and g are the inner products between the normal and the second order derivatives defined as

$$\begin{aligned} p_{uu}(u, v) &= \frac{\partial^2 p(u, v)}{\partial u^2} \\ p_{uv}(u, v) &= \frac{\partial^2 p(u, v)}{\partial u \partial v} \\ p_{vv} &= \frac{\partial^2 p(u, v)}{\partial v^2} \end{aligned}$$

The link between I_p and II_p is given by the shape operator at point $p(u, v)$, which is effectively the normal curvature $\kappa_n(p)$. Let $w_1(p) = u_1 p_u + v_1 p_v \in T_p(S)$ be a vector in the tangent plane of $p(u, v)$, represented as $\bar{w}_1 = (u_1, v_1) \in \Omega$ in the parameter space. The normal curvature of point $p(u, v) \in S$ in direction \bar{w}_1 can then be calculated as Botsch et al. (2010) suggest,

$$\kappa_n(p) = \frac{\bar{w}_1^T \mathbf{II} \bar{w}_1}{\bar{w}_1^T \mathbf{I} \bar{w}_1} = \frac{eu_1^2 + 2fu_1v_1 + gv_1^2}{Eu_1^2 + 2Fu_1v_1 + Gv_1^2} \quad (4.39)$$

Although the normal curvature is an extrinsic metric of a surface, it can be linked to the *Gauss Curvature* which is an intrinsic metric, i.e. invariant to the parametrization and

only depends on the first fundamental form (do Carmo (1976)). Euler discovered that the normal curvature is a combination of the principal curvatures of a surface (Botsch et al. (2010)). They are the paths of maximum (κ_1) and minimum (κ_2) curvature on a surface. The Euler equation is

$$\kappa_n(\bar{w}_1) = \kappa_1 \cos^2 \phi + \kappa_2 \sin^2 \phi \quad (4.40)$$

where ϕ is the angle between the principal directions corresponding to κ_1 and κ_2 . The Gauss curvature can then be found as

$$\kappa = \kappa_1 \kappa_2 \quad (4.41)$$

These surface metrics are used to preserve curvature, angles, lengths in thin shell deformation methods. The methods are usually nonlinear, but benefit from numerical stability when performing local rotations. Linear techniques are faster, but less accurate than the nonlinear ones. They also tend to increase the volume of the models, which is not desirable for incompressible plasticine models.

Thin Shell and Plate Deformations

If $S \subset \mathbb{R}^3$ is the smooth source surface, parametrized by $p(u, v) : \Omega \rightarrow \mathbb{R}^3$, let $S' \subset \mathbb{R}^3$ be the corresponding deformed surface, parametrized by $p'(u, v) : \Omega \rightarrow \mathbb{R}^3$. The deformation between them is given by a deformation field $\delta(u, v) : \Omega \rightarrow \mathbb{R}^3$, with $p'(u, v) = p(u, v) + \delta(u, v)$. Given a set of handle points which can be manipulated by the user, the deformed point positions are found by minimizing stretching and bending of surface S as the handle points are placed in the new locations. This technique is common for obtaining surface deformations that imitate a stretching and bending elastic sheet (Botsch & Kobbelt (2004), Sorkine & Alexa (2007)) and less common for plastic effects.

The first (I) and second (II) fundamental forms can be used to define a nonlinear thin shell energy (Botsch & Sorkine (2008)) that measures the difference in stretching and bending between S and S' . This energy is invariant to rigid transformations such as global rotations and translations, and is defined as

$$E_{shell}(S') = \int \int_{\Omega} k_s \|I'(u, v) - I(u, v)\|_F^2 + k_b \|II'(u, v) - II(u, v)\|_F^2 \quad (4.42)$$

where k_s and k_b are stiffness parameters that assign the resistance to stretching and bending respectively and F is a weighted Frobenius norm.

Energy (4.42) can also be seen as the potential energy that an object needs to preserve its shape under external forces. The energy function is obtained from the Lagrange equations of motion as adapted from Terzopoulos et al. (1987)

$$\frac{\partial}{\partial t} \left(\mu \frac{\partial p(u, v)}{\partial t} \right) + \gamma \frac{\partial p(u, v)}{\partial t} + \frac{\partial E_{shell}(S')}{\partial p(u, v)} = f(p(u, v), t) \quad (4.43)$$

where $p(u, v)$ is the position of a point on the surface S at time t as it deforms towards S' , $\mu(u, v)$ is the mass density of that point, $\gamma(u, v)$ is the damping density and $f(p(u, v), t)$ is the sum of external forces acting on the original surface. In a state of equilibrium the energy is zero, meaning that the external forces are zeroed out by the internal forces of the surface, given by the first two terms of equation (4.43). The model is said to remain in its rest pose when $E_{shell}(S') = 0$.

In the (4.42) formulation of the energy term, the minimizer is the deformed surface itself, S' . Due to the computational overload of solving a nonlinear optimization problem every time the user moves the handle points, the first and second fundamental forms are approximated by the first and second partial derivatives of the deformation field δ (Botsch & Sorkine (2008)).

In effect this approximates the thin shell energy locally by its partial derivative $\frac{\partial E_{shell}(S')}{\partial p(u, v)}$ from equation (4.43). It has been observed that this variational derivative not only approximates the energy locally, but also behaves well outside the local region (Terzopoulos et al. (1987), Celniker & Gossard (1991)). The form for the potential energy (Botsch et al. (2010)) thus becomes

$$E_{shell}(\delta) = \int \int_{\Omega} k_s \left(\|\delta_u(u, v)\|^2 + \|\delta_v(u, v)\|^2 \right) + k_b \left(\|\delta_{uu}(u, v)\|^2 + 2\|\delta_{uv}(u, v)\|^2 + \|\delta_{vv}(u, v)\|^2 \right) dudv \quad (4.44)$$

where $\delta_u = \frac{\partial \delta}{\partial u}$, $\delta_{uv} = \frac{\partial^2 \delta}{\partial u \partial v}$. The minimizer for (4.44) is the solution to the Euler-Lagrange partial differential equation

$$-k_s \Delta \delta + k_b \Delta^2 \delta = 0 \quad (4.45)$$

where $\Delta\delta = \text{div } \nabla \delta = \delta_{uu} + \delta_{vv}$ is the Laplacian (or the divergence of the gradient) operator of the deformation field at point $p(u, v)$ and $\Delta^2\delta = \Delta(\Delta\delta) = \delta_{uuuu} + 2\delta_{uuvv} + \delta_{vvvv}$ is the bi-Laplacian (Botsch & Sorkine (2008)). This equation solves the problem of minimizing changes in area and bending and is linked to the linearized membrane and thin plate energies (Moreton & Séquin (1992), Botsch et al. (2010)).

The latter energies minimize the area and bending in a fairing, rather than a deformation procedure with energy terms

$$E_{\text{memb}}(p(u, v)) = \int \int_{\Omega} \left(\|p_u(u, v)\|^2 + \|p_v(u, v)\|^2 \right) dudv \quad (4.46)$$

$$E_{\text{plate}}(p(u, v)) = \int \int_{\Omega} \left(\|p_{uu}(u, v)\|^2 + 2\|p_{uv}(u, v)\|^2 + \|p_{vv}(u, v)\|^2 \right) dudv \quad (4.47)$$

Notice that the partial derivatives are applied to the surface points in (4.46) and (4.47), rather than the deformation fields. The corresponding Euler-Lagrange partial differential equations are $-\Delta p(u, v) = 0$ and $\Delta^2 p(u, v) = 0$ respectively. Generally the Euler-Lagrange equation has the form $(-1)^k \Delta^k \delta$. The exponent k corresponds to a maximum C^{k-1} continuity for the resulting surface after applying the obtained deformation field δ (Botsch & Kobbelt (2004)).

Both nonlinear and linear approximations of the thin shell energy (4.42) have been implemented in literature. Discrete surfaces or triangular meshes are usually the preferred medium for such deformations. Fröhlich & Botsch (2011) present a nonlinear discrete shell deformation technique based on the work of Grinspun et al. (2003). Fröhlich & Botsch (2011) minimize the stretch of edges and the bending of dihedral angles between triangles. Physically plausible behaviour is also achieved by Botsch et al. (2006) who combine thin shell methods with a layer of volumetric prisms to ensure local shape preservation. Linear methods became popular after the introduction of laplacian editing (Sorkine et al. (2004)) and differential coordinates (Sorkine (2005)) as described below.

Laplacian Editing Deformations

Sorkine et al. (2004) use surface based deformations such as Laplacian editing to deform a mesh depending on its intrinsic geometry. The preservation of Laplacian or differential coordinates (Sorkine (2005)) in a triangular mesh during deformation permits

details to remain almost intact. This method inspired as-rigid-as-possible deformations (Sorkine & Alexa (2007)), where cells of triangles around a vertex undergo rigid transformations. The common idea was to preserve fine details of a model even under large scale manipulations of the mesh.

Laplacian deformation is inspired by editing using differential representations in the image domain. The parametrization of the gradient in two dimensions allows for the modelling tasks to be solved with discrete Poisson equations. Texture manipulation during mesh editing is also facilitated during the editing procedure.

The Laplacian coordinate for a vertex on a mesh is encoded as the difference between its position and the weighted average of its neighbours (equation (4.48)). The conversion between intrinsic and absolute representations is made possible by solving a sparse linear system that involves these differential coordinates and boundary conditions.

Consider a set of handle points \mathcal{H} controlled by the user, on the surface mesh M_S . A fixed set of vertices \mathcal{F} is also added as a boundary condition to ensure a stable solution. After manipulating the handle points, the deformed mesh becomes $M_{S'} = \{p(v_i) + \delta(p(v_i)) \mid p(v_i) \in M_S\}$, where v_i is the i th vertex of mesh M_S and $p(v_i)$ is the vertex position in \mathbb{R}^3 .

In the discrete setting the Laplace operator, used to calculate the differential coordinates, is replaced with the discrete Laplace-Beltrami operator (Botsch & Sorkine (2008)). Given a piecewise linear scalar function on the initial mesh $f : M_S \rightarrow \mathbb{R}$, the Laplace-Beltrami operator at vertex v_i is

$$\Delta_S f(v_i) = w_i \sum_{v_j \in \mathcal{N}_1(v_i)} w_{ij} (f(v_j) - f(v_i)) \quad (4.48)$$

where $v_j \in \mathcal{N}_1(v_i)$ are the 1-ring neighbours of v_i , $w_i = \frac{1}{A_i}$ are the per-vertex normalization weights, A_i is the Voronoi area around vertex v_i , w_{ij} are edge weights.

The main contributions of Sorkine et al. (2004) are the introduction of rotation and scale invariant Laplacian coordinates, detail preservation during editing, coating or transfer of geometric details between surfaces of equivalent or different topologies. Boundary homeomorphism is sufficient for mapping geometry between two patches of arbitrary geometry.

Deformations in their model are intuitive as simple handles can be used to reshape the

object in a free form fashion. Geometry follows the control points smoothly, giving the illusion of a soft material. This physical imitation is not sufficient, however, for volume preservation, large scale deformations, dents, bulges and other general plasticity properties clay displays (Sorkine et al. (2004), Botsch & Sorkine (2008)). Next, space deformations are studied to see whether they provide a better solution for virtual clay manipulation.

4.3.2 Space Based Deformations

Space deformations are closer to the plasticine modelling process as a new shape emerges through the contribution of external forces. Moreover, the constraint of volume preservation, which can be easily obtained with such techniques, promises to give physically plausible deformations (von Funck et al. (2006), Stanculescu et al. (2011)). Adaptive mesh refinement can also be incorporated to allow topological changes (Stanculescu et al. (2011)).

Angelidis & Wyvill (2004) introduced swirling sweepers, the first space deformation that is both volume-preserving and foldover-free. The basis for their work consists of a geometric tool called sweepers. Together with a motion path, vertices can follow a deformation field, regardless of the geometry of the model. The continuous deformation path is split into smaller steps that can be keyframed to playback the modelling process. Intermediate poses are cached, which allows real-time, high quality rendering. Large distortions (twisting, bending, stretching) and volume preservation are also made possible. A drawback does occur when stretching geometry, as texture suffers from similar strains.

Model sculpting is permitted through dragging movements of vertices through space. More emphasis on this approach was made by von Funck et al. (2006), who used a continuous, divergence-free, time dependent vector field to allow free-form deformations. An implicit tool is used to define the areas of modelling influence on the mesh. Path line integration is used to determine the vertex positions from the resulting vector field.

Due to the zero-divergence vector field in their model, no self-intersections are permitted and volume and C^1 smoothness are preserved. Moreover, sharp features and details are maintained during deformations, as no energy minimization techniques are used to smoothen the results. However, artefacts may occur during large scale deformations, which is why remeshing is adopted. This may slow down the process and insignificant volume changes may appear. Real-time is still attainable for moderate sized meshes,

which lack mesh connectivity information, a characteristic for their approach (von Funck et al. (2006)). Stanculescu et al. (2011) allow volume preservation and topology changes by adapting sweeper techniques to quasi-uniform triangular meshes.

More traditional space based deformation techniques are cage based methods (Joshi et al. (2007), Lipman et al. (2008)). The vertices of these cages act as movable handles, which envelop meshes undergoing deformation. Once handles are placed by the user in a desired position, the resulting vector field is applied to the mesh. Techniques like Harmonic (Joshi et al. (2007)) or Green coordinates (Lipman et al. (2008)) are adopted for smooth results and shape preservation. Cages are usually tedious to create and don't allow detailed and intuitive mesh manipulation.

Lastly radial basis functions (Botsch & Kobbelt (2005), Niu et al. (2017)) are a linear method, used to create smooth deformation vector fields. Their initial use was in surface reconstruction by interpolating through a sparse set of available points. In the case of Botsch & Kobbelt (2005), the deformation field is created from a sparse set of vectors stemming from the handle points. Although a powerful tool, the results depend on the number of handle points and volume control is not intuitive.

Regardless of the type of deformation, weights or, in the case of this work, local stiffnesses, are used to control the flexibility of the surface when modified by the artist. The next section delves into how stiffness is represented in the graphics literature, when considering virtual clay.

4.3.3 Stiffness of Virtual Clay Models

Stiffness takes many forms in the graphics literature, ranging from realistic representations in simulations to a constant approximation in elastic surface deformations. The analysis of this physical property from section 4.1.2 has shown that its values are inherently nonlinear, especially in the plastic domain. Volumetric virtual clay representations remark that stiffness at a point p is a function that depends on its position inside the solid (Cani-Gascuel & Desbrun (1997)). Points that are towards the center of the model are harder to reach and thus tend to be stiffer than boundary points. This shows that the structure of a model can influence its stiffness in deformation.

In mass spring systems structural stiffness can be seen in the linear spring analogy. Edges delimiting faces or tetrahedrons are endowed with elastic properties following Hooke's law. It is common to set an edge stiffness inversely proportional to the edge length (Selim & Koomullil (2016)). One of the reasons for this is to prevent vertices

from colliding with each other. The Elastic modulus and Poisson’s ratio are thus related to geometric properties of the mesh. The elastic modulus is sometimes set equal to the aspect ratio of the tetrahedron component of the volumetric mesh. This results in high stiffnesses in regions with larger volumes, which is similar to the way plasticine behaves. A survey about similar techniques can be found in Selim & Koomullil (2016).

Zhou et al. (2013) use the notion of stiffness to describe Young’s modulus, as a property of the material. A stiff object is rigid or elastic, in the sense that it returns to its original shape after deformation. A soft material deforms plastically and irreversibly. Dey et al. (2015) use the finite element method (FEM) proposed by Irving et al. (2004), who use an elasto-plastic multiplicative FEM to produce larger elastic and plastic deformations. The latter algorithm, however, is known to be numerically unstable. Dey et al. (2015) say, however, that since they have used a coarse mesh, they did not notice any numerical instabilities, even at large time steps.

In surface deformations stiffness describes the tension of the boundary and controls how stretching and bending are distributed. This is also linked with surface fairing. The exponent of the Laplace operator $(-1)^k \Delta^k$ used in the fairing process is linked to how curvature shapes a model. A membrane surface ($k = 1$), thin-plate surface ($k = 2$), minimal curvature variation ($k = 3$) produce models that are stiffer in aspect as the exponent k grows (Botsch & Kobbelt (2004)).

Surface based deformation methods (Botsch & Sorkine (2008)) and space based deformation methods (von Funck et al. (2006)) make use of the stiffness of the material, but often simplify it to a constant, thus favoring elastic behaviour. Another approach is to limit the stiffness or relaxation weights to geometric properties and not make use of the physics of the material. The plastic nature of a material is usually disregarded, resulting in a limited set of possible deformations.

Stiffness is considered locally in Popa et al. (2006) who suggested painting it as a scalar field on the surface of the deformable shape. They show how local control of bending and shearing stiffnesses can yield more intuitive results for a deformation transfer scenario. Later on Fröhlich & Botsch (2011) extended this idea for nonlinear discrete shell deformations, which give results even closer to the physical model. The latter algorithm is used for the deformation step as will be explained in section 4.4.

Local stiffness maps are commonly found from a set of example poses (Popa et al. (2006), Fröhlich & Botsch (2011)) or from images of possible deformations of the source surface (Kanazawa et al. (2016)). In the work of this thesis, the extrinsic and intrinsic information of the source and target shapes is used to estimate a local stiffness map.

An imitation of the physical properties of plasticine at a surface level is preferred to relying on the amount of target shapes available, like in the previous papers.

Sorkine et al. (2004), Sorkine (2006) popularized Laplacian or differential coordinates, which encode the intrinsics of the shape. A vertex is defined in relation to the distance to its neighbours, which is very appealing, considering that the author wishes to define vertex stiffness in relation to its neighbours. This is also linked to how strain is calculated in Zhou et al. (2013) and all the other previous FEM methods.

In nonrigid registration, stiffness is usually a global property representing the rigidity of a deforming object. Amberg et al. (2007) propose iterative stiffness, which decreases as the source converges towards the target shape. Zell & Botsch (2013) use a joint fairing method to smooth out the source and target meshes. This method makes use of the voronoi area around a vertex and the Laplace coordinates. Afterwards, a non-rigid registration procedure is used to match the smooth source to the smooth target, thus finding an accurate correspondence between the initial meshes. Stiffness is mentioned and is used similarly to Amberg et al. (2007), as an iterative global measure of deformation rigidity.

Local stiffnesses in previous work is thus seen more like an influence weight (Dey et al. (2015)) and has no physical meaning. Moreover, most of the deformation and registration algorithms assume an elastic deformation of the object, which usually implies isometries. The plastic domain and non-isometric deformations are very rarely considered.

The endeavour of this thesis is to show that local stiffness can be used to enhance deformation and non-rigid registration of plasticine model scans. Non-isometric, plastic deformations and volume preservation are made possible through the stiffness value. The next section presents the optional *Deformation* step of the E-StopMotion pipeline (figure 3-3) and how local stiffness can influence the deformation results.

4.4 E-StopMotion: Deformation Component

After getting an overview of how virtual clay is represented in literature, from simulations to surface deformations, the next step was to experiment and extend an available nonlinear surface based deformation method, proposed by Fröhlich & Botsch (2011). The investigation was done to understand whether physically plausible surface deformations can imitate real plasticine models, without changing their topology.

The algorithm for calculating and propagating local stiffness on a surface is first described. Next, the nonlinear deformation method proposed by Fröhlich & Botsch (2011) is implemented and modified, to imitate plasticine deformation. Their technique was preferred due to the volume preservation properties and the numerical stability ensured for local rotations. The linear method proposed by Botsch & Kobbelt (2004) was experimented with briefly. Due to the large increase in volume during deformation, however, it was not considered a valid direction for the current work.

4.4.1 Method

A preponderantly elastic surface deformation algorithm for triangular meshes was extended to imitate plasticine deformation. The proposed method is nonlinear, based on Fröhlich & Botsch (2011) and preserves surface smoothness to a certain extent during deformation. The nonlinear method is slower than available linear methods (Botsch & Kobbelt (2004)), but preserves volume and allows the stable addition of localized stiffness values.

The reasoning behind choosing surface based deformations is as follows. As it was shown in the previous section, surface based deformations tend to preserve details and smoothness. They are focused on the intrinsic structure of the shape and are useful for obtaining appealing deformations by preserving surface features like lengths, angles, areas, curvature on the source mesh. Volume preservation, however, is less frequent in these types of deformation.

Space based deformations, on the other hand, act using external forces that sculpt the surface similarly to how an artist sculpts a plasticine model. While this modelling technique is closer to the idea of external forces acting on an object, the deformation fields produced don't usually have a physical significance. Volume preservation, however, can be obtained easily with divergence free vector fields.

It then becomes intuitive to have a combination of space and surface based deformation. The former could act on the model's embedded space with external forces, while the latter could oppose those forces through some feature preserving regularization. If the reverse engineering approach is considered, however, it is the model's physical properties that generates the volume preserving vector field.

This means that when applying an external force of any kind on the source model, its physical properties (section 4.1) should help counteract it. In the case of plasticine, the

model's volume is preserved during the interaction. Moreover, surface tension can be imitated by the surface smoothness obtained through the chosen deformation method.

Lastly, as it can be observed in figure 4-2, in the plastic domain, the propensity of a model to stretch grows with the amount of stretch it has already undergone. In other words the thinning effect of a material renders it more vulnerable to further modelling. Similarly, if one compresses a model, the area of application grows, making it harder to continue the process. This is known in plasticity theory as structural stiffness, where both the structure of a model and its material contribute to how it deforms.

These properties inspired the definition of the local stiffness on a mesh, to control how a shape stretches or compresses locally. The goal is to achieve intuitive effects of sculpting clay and to find the limits for a deformation method that imitates plasticine without changing the connectivity of the triangular mesh it manipulates. The local stiffness is first described, followed by applying it to a modified version of the nonlinear thin shell deformation algorithm proposed by Fröhlich & Botsch (2011).

Local Stiffness

A plasticine model now has a representation and a set of deformation methods. Section 4.1.2 described a few formulas for the stiffness of a material with both elastic and plastic properties. It was also shown how the model's structure can influence the amount of deformation it undergoes when a force is applied to it. A shape is created by locally sculpting the modelling clay it is made from. This can be seen as separate forces acting on local regions of the model. The argument is that the structural stiffness in one region will be different from the stiffness in another region, due to the different amounts of material available in those regions. Hence the need for a local stiffness, which describes the model's structural stiffness per region of influence.

Stiffness Formulation

Let $k = \frac{F}{\Delta d}$ be the uniaxial stiffness in an infinitesimal region of area A . The applied force in that region is F and the resulting elongation is Δd . The corresponding stress is $\sigma = \frac{F}{A}$ and strain $\varepsilon = \frac{\Delta d}{d}$. The uniaxial stiffness can be rewritten as

$$k = \frac{\sigma A}{\Delta d} \quad (4.49)$$

Since the materials used are plastic, σ can be replaced with the plastic stress σ_p of the

Ludwik curve equation (4.24),

$$\sigma = \sigma_p = \sigma_0 \left(\frac{E\varepsilon}{\sigma_0} \right)^n \quad (4.50)$$

which gives the following form for stiffness k

$$k = \frac{\sigma_0 \left(\frac{E\varepsilon}{\sigma_0} \right)^n A}{\Delta d} \quad (4.51)$$

where σ_0 is the initial yielding stress, E is Young's modulus, ε is the total strain (elastic and plastic), A is the infinitesimal area of the application region, Δd is the current elongation between the source and target regions, n is an exponent (see figure 4-4 for values of n).

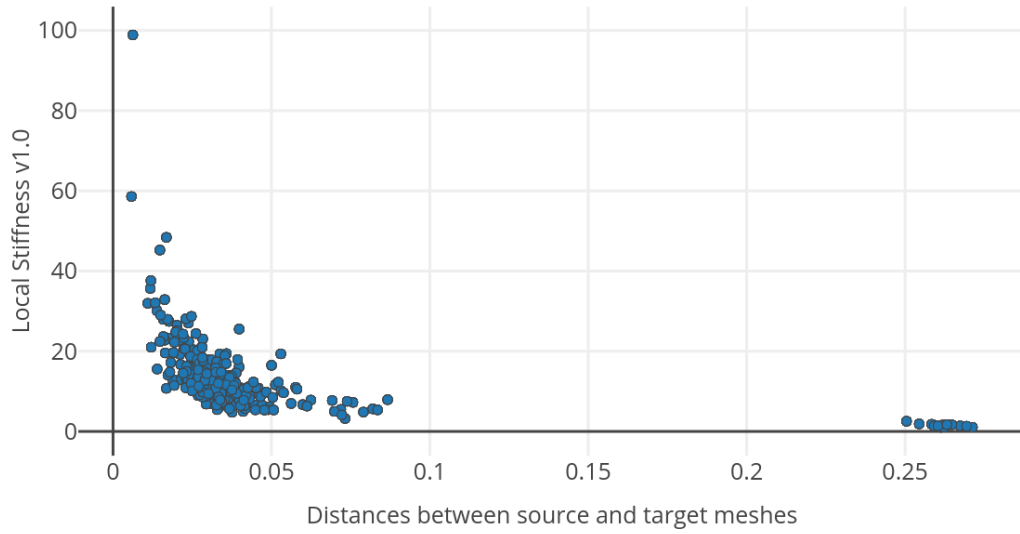


Figure 4-7: Relation between the source and target mesh distances and the stiffnesses calculated per corresponding source vertex using equation (4.51)

This first version of the local stiffness (figure 4-7) is directly proportional to the local area of the region being deformed. It is also inversely proportional to the elongation applied to that region. This shows that the structure of the model being deformed can influence the stiffness value. Given the same force, a region with a large area will be harder to deform, hence stiffer, than a region with smaller area. This is due to the

stress being smaller in the first case and larger in the second case. Similarly, the larger the elongation being obtained, the smaller the stiffness is, since a small stiffness allows for the material to be more malleable and thus undergo larger deformations.

The first implementation of local stiffness was between a source mesh M_S and a target mesh M_T , as it was originally designed for the non-rigid registration method presented in chapter 5. Figure 4-7 shows the relation between the source and target mesh distances and the local stiffness, calculated using equation (4.51). The local regions selected are the source vertices of mesh M_S . For each source vertex, the closest target vertex is found and the distance between them is used in to find the stiffness for the source vertex.

Notice that the distribution of the stiffness values is not smooth in this first version, however, which can cause artefacts during deformation. Apart from smoothness, the goal was to find a stiffness formula that only depends on the strain of the mesh. In most of the experiments that followed strain was approximated by elongation for efficiency reasons. Local area should ideally be kept within a certain threshold, to avoid shape distortions. After going back to plasticity theory the following equation for the relation between the elastic and plastic moduli (Chakrabarty (2009)) was found.

$$\frac{E}{H} = m \left(\frac{\sigma}{\sigma_0} \right)^{m-1} \quad (4.52)$$

This equation can be combined with the plastic strain ε_p from (4.22) as given by the Ramberg-Osgood equation. If work hardening is discounted, $\sigma_0 = \sigma_R$, otherwise, $\sigma_0 \leq \sigma_R$. The former interpretation was chosen as a starting point for finding a new formula for the stress applied to an infinitesimal region on the surface.

$$\varepsilon_p = \alpha \frac{\sigma_0}{E} \left(\frac{\sigma}{\sigma_0} \right)^m \Rightarrow \sigma = \left(\frac{\varepsilon_p E \sigma_0^{m-1}}{\alpha} \right)^{\frac{1}{m}} \quad (4.53)$$

The thought process behind obtaining a second version for the local stiffness (equation (4.56)) is next presented. This formula only depends on the plastic strain and some material constants. From equation (4.52) a new form for the plastic modulus is extracted. This modulus can also be understood as the stiffness at a moment in time, in the plastic domain (section 4-2). The stress σ is then replaced with the formula found in equation (4.53).

$$H = \frac{E}{m} \left(\frac{\sigma_0}{\sigma} \right)^{m-1} \Rightarrow H = \frac{E}{m} \sigma_0^{m-1} \left(\frac{\alpha}{\varepsilon_p E \sigma_0^{m-1}} \right)^{\frac{m-1}{m}} \quad (4.54)$$

where the exponents for E and σ_0 are simplified as follows. In case of E , the exponent becomes $1 - \frac{m-1}{m} = \frac{m-m+1}{m} = \frac{1}{m}$. In the case of σ_0 the exponent becomes

$$m-1 - \frac{(m-1)^2}{m} = \frac{m^2 - m - (m^2 - 2m + 1)}{m} = \frac{m^2 - m - m^2 + 2m - 1}{m} = \frac{m-1}{m}.$$

The plastic modulus then becomes

$$H = \frac{\frac{1}{Em}}{m} (\sigma_0 \alpha)^{\frac{m-1}{m}} \left(\frac{1}{\varepsilon_p} \right)^{\frac{m-1}{m}} \quad (4.55)$$

so that a formula of the form $H = f(C, \varepsilon_p)$ for the stiffness in the plastic domain is now available. C is a constant that depends on the intrinsics of the material. Since plasticine has negligible elasticity, it is assumed that the plastic modulus is a good enough approximation of the overall stiffness of the model. So by assuming $H = k$, the second version of the local stiffness is obtained, which only depends on the local strain (elongation in experiments) performed on a small region of the surface.

$$k = \frac{\frac{1}{Em}}{m} * \sigma_0^\gamma * \frac{\alpha^\gamma}{\varepsilon_p^\gamma} \quad (4.56)$$

where k is the local stiffness, usually taken per vertex when working with a mesh, $\gamma = \frac{m-1}{m}$, E is Young's modulus, α , m are constants related to material intrinsics (figure 4-4), σ_0 is the initial yield stress, ε_p is the plastic strain.

Figure 4-8 illustrates how the stiffness and distances between the source and the target meshes are related using the second stiffness version. An improved continuity of the values can already be observed. The next step is to take a closer look at how the stiffness values propagate across the surface. Also, notice that the interval values for the local stiffness are not consistent. In version one (figure 4-7) stiffness ranges from 0

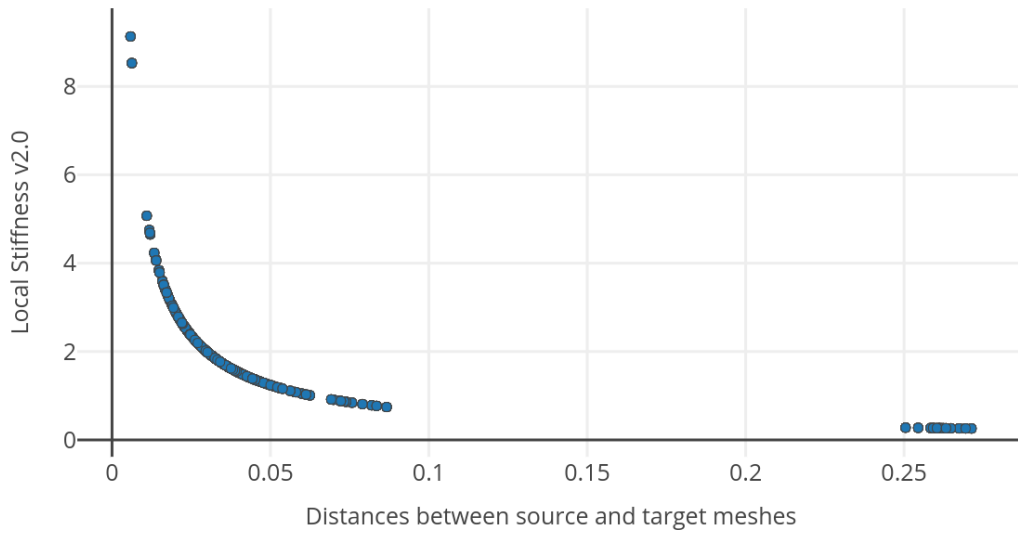


Figure 4-8: Relation between the source and target mesh distances and the stiffnesses calculated per corresponding source vertex using equation (4.56)

to 100, while in version two (figure 4-8) it ranges from 0 to 10. This was amended by mapping the landmark stiffnesses to a min stiffness and max stiffness interval.

If stiffness is too low, then mesh faces stretch too much and if it is too high, there might be no deformation at all. An ideal balance between the two modes is needed for preserving the shape and obtaining clay-like deformations. For most of the experiments the maximum stiffness is 100, while the minimum stiffness is 1. *Stiffness Propagation*

For the two examples shown so far stiffness is calculated per source vertex and then averaged out per edge to ensure some form of continuity. This method of calculating stiffness is a very rough approximation of what might happen with the surface of a plasticine model when being deformed. Artefacts can easily appear when deforming the surface as can be seen in figure 4-9. Nevertheless, this is a starting point to creating localized deformations.

The second technique used to propagate stiffness across a surface was to use the idea of application points. Imagine the handle points or landmarks a user places on a surface as being at the center of a region being pushed or pulled by the artist. The remaining points are then modified depending on the regularization used. If the regularization imposes the same metric preservation on all points, results are usually smooth, but tend to display an increase in volume as errors are distributed globally across the surface. By allowing some regions to be more malleable than others, local stiffness can be used

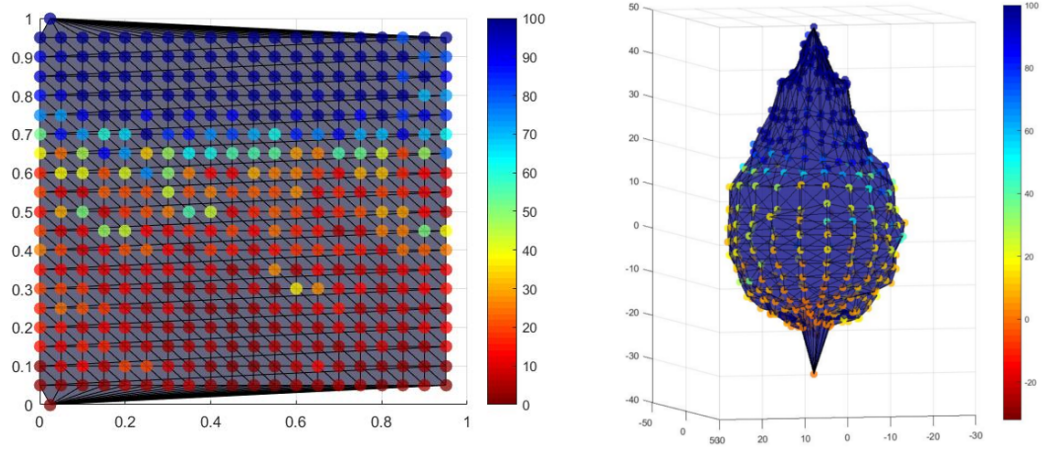


Figure 4-9: Version two of the local stiffness, calculated per vertex for a sphere model. The source mesh is the sphere, while the target mesh is a scaled sphere along the Y axis. For each source vertex, the closest target vertex is found. The distance between them is used directly as the strain in equation (4.56). The source mesh UV map (left) has red vertices where the stiffness is low, meaning that the strain is large. Similarly, blue represents a low stiffness. The deformed sphere (right) presents some characteristics of a locally deformed object, but its appearance is nonsmooth, since the propagation method only averages out stiffnesses per edges and does not have a surface based distribution.

to strengthen the regularization in some regions and weaken it in others.

The local stiffness is first calculated at the handle points and then is propagated along the surface using an inverse distance function with controllable falloff. Shepard's interpolation method (Shepard (1968)) was first used to propagate the stiffness values from the landmarks to the rest of the vertices. Figure 4-11 shows an example of a sphere deformation, where the bottom part is fixed, while the top part is displaced along the Y axis. Blue areas mark high stiffness, while red areas mark low stiffness, which explains why the bottom part of the mesh remains fixed, while the top part deforms towards the desired landmarks, shown in green.

Given an source mesh M_S , the local stiffness k_i of a vertex v_i is then calculated as

$$k_i = \begin{cases} \frac{\sum_{j=1}^N w_j k_j}{\sum_{j=1}^N w_j}, & \text{if } d(p(v_i), p(v_j)) \neq 0 \text{ for all } j \\ k_j, & \text{if } d(p(v_i), p(v_j)) = 0, \text{ for } j == i \end{cases} \quad (4.57)$$

where N is the number of landmarks around $p(v_i)$, $d(p(v_i), p(v_j))$ represents the distance

between vertex points $p(v_i)$ and $p(v_j)$ and the weight

$$w_j = \frac{1}{d(p(v_i), p(v_j))^q} \quad (4.58)$$

is the inverse distance weight as proposed by Shepard (1968). A few experiments made use of Gaussian weights of the form

$$w_j = \frac{1}{\exp(d(p(v_i), p(v_j))^q)} \quad (4.59)$$

An exponent of $q = 2$ was found to work well for most of the experiments. Gaussian weights tend to have fewer discontinuities in values than the inverse distance weights (IDW). This might be explained by the asymptote that appears in the IDW when distances are close to zero (figure 4-10). The distance between two vertices can be either an Euclidean or a geodesic distance, with the latter giving a more refined distribution of stiffness across the surface.

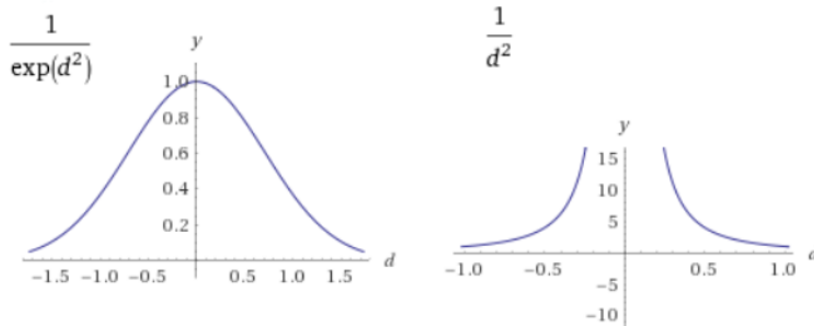


Figure 4-10: The difference between a Gaussian (left) and an Inverse Distance (right) weights. Notice that the Gaussian has no discontinuities for small distance values.

The IDW distribution of the stiffness gave intuitive results for simple models, but started having issues for more complex ones. Empirical studies showed that when landmarks are sparse deformation can become unstable, due to vertices estimating their stiffness based on far away landmarks. It is safer to assume that where landmarks are not present, the region can remain fixed or at least have a maximum stiffness across it.

Landmark are usually placed where displacement is required, but if the displacement value is close to zero, a high stiffness should keep the desired vertices in place. From

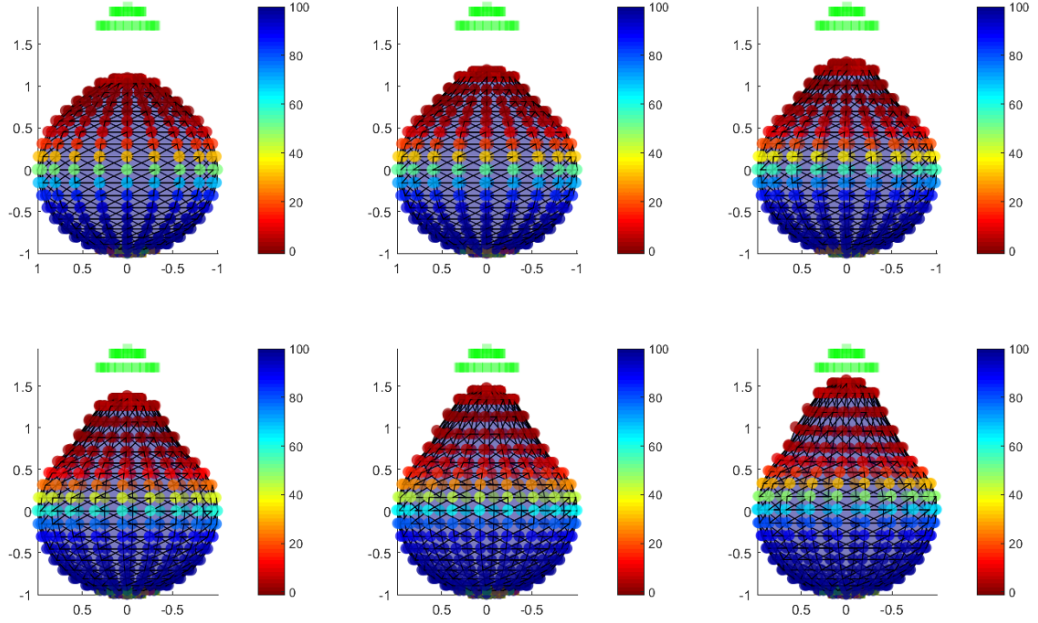


Figure 4-11: Inverse distance weights take all the landmarks into account when propagating stiffness across a sphere model, as its top displaces vertically. Red indicates low stiffness, while blue indicates high stiffness which explains the behaviour during the deformation. The green points represent the target landmarks, which correspond to source landmarks moving towards them.

these observations a new stiffness propagation method emerged. It also uses inverse distance weights, but only considers landmarks within a certain radius of the current vertex. If no landmarks are close by, the maximum stiffness is assumed for that vertex.

Algorithm 1 shows the pseudocode for this method, while figure 4-12 uses it on the same model as in figure 4-11. The differences between the two images are that in the first implementation the falloff of the displacement influence is much slower than in the second implementation. Since the aim is for localized control (figure 4-14), the second implementation is expected to be more suitable.

Nonlinear deformation

Fröhlich & Botsch (2011) present a physically based surface deformation method, which is nonlinear and preserves volume. Their algorithm includes an elegant formulation for thin shell deformation and is the basis of the proposed method in this chapter. They make use of stretching and bending properties of the mesh, which allow physically plausible deformations. Although localized stiffness from a set of example poses is

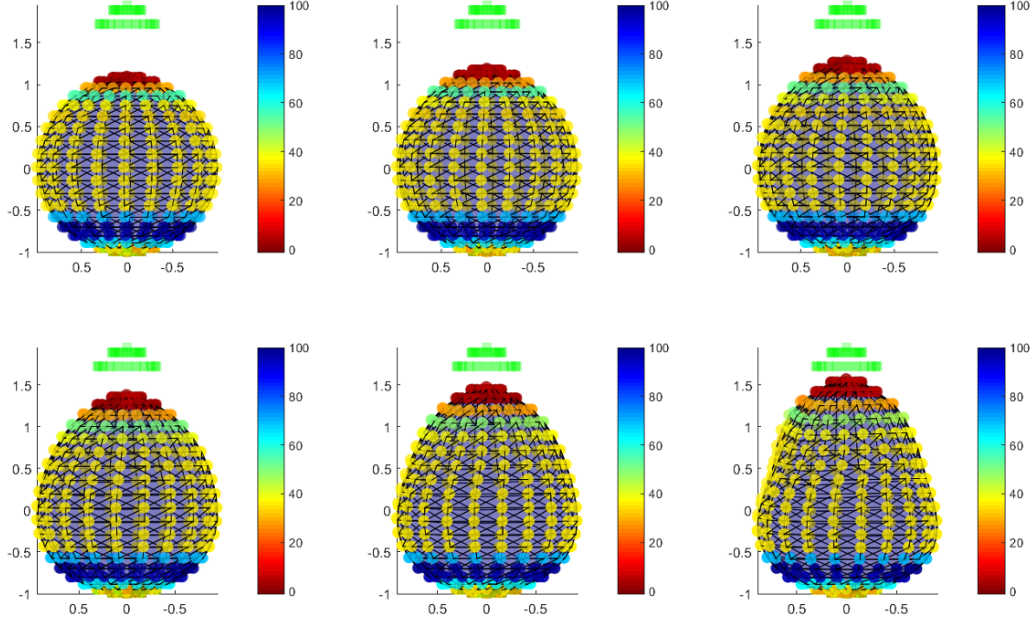


Figure 4-12: Inverse distance weights take only landmarks within a radius into account when propagating stiffness across a sphere model, as its top displaces vertically.

mentioned, the experiments presented in the paper only make use of global stiffness values, which results in elastic deformations.

The thin shell deformation technique is enhanced through the current work by adding localized stretch and bend stiffnesses. These are calculated from the vertex based stiffnesses as presented in the previous section. The volume preservation is also extended to consider a list of face corresponding tetrahedron volumes, rather than a global mesh volume. This is motivated by the shearing artefacts which appear during deformations that use mesh volume as a hard constraint (Huang et al. (2006)).

Since a set of scans is available for each character, a small number of landmarks is chosen between a source and a target scan. The target landmarks act as the desired position for the source landmarks. They can be considered handle points which the artist has modified during the deformation. A series of iteration steps divide the displacement vectors between the source and target handle points, ensuring a relatively smooth deformation.

The intention is not to register the source mesh to the target mesh at this point, since the available displacement vectors are sparse. The obtained shapes could represent

Algorithm 1 Radial propagation of local stiffness

Data: Source landmark stiffnesses $lStiff$ and source shape.

Result: Source shape vertex stiffnesses.

Calculate inverse weight matrix $wMat$

$wThreshIn = 0.4$; $wThreshOut = 0.1$; $expWeight = 2$;

```
for  $i = 1:numberOfSourceVertices$  do
    /* Calculate total weight first */
    totalWeighti = sum(wMat(i, :))
    wMat(i, :) = wMat(i, :) / totalWeighti
    for  $j = 1:numberOfLandmarks$  do
        sourceLandmarkIndex = sourceLandmarkIndices(j, 1)
        if  $i \neq sourceLandmarkIndex$  then
            /* Adding radial weight control */
            if  $wMat(i, j) \geq wThreshIn$  then
                vertexStiffnessMap(i,1) += wMat(i, j)*lStiff(j, 1)
            end
            if  $wMat(i, j) < wThreshIn$  and  $wMat(i, j) > wThreshOut$  then
                wThreshDiff = wThreshIn - wThreshOut
                wThreshToIn = (wThreshIn - wMat(i, j))/wThreshDiff
                wThreshToIn = power(wThreshToIn, expWeight)
                wThreshToIn = wThreshToIn * wMat(i, j)
                wThreshToOut = wMat(i, j) - wThreshToIn
                vertexStiffnessMap(i,1) += wThreshToIn * lStiff(j, 1) + wThreshToOut * maxStiffness
            end
            else if  $wMat(i, j) > wThreshOut$  then
                vertexStiffnessMap(i,1) += wMat(i, j) * maxStiffness;
            end
        else
            vertexStiffnessMap(i,1) = lStiff(j, 1);
        end
    end
end
```

plausible shapes that the artist has sculpted in the process of modelling the target shape. Thus it is said that the deformation is reverse engineering the artistic process.

Energy Minimization

Similarly to Fröhlich & Botsch (2011) an iterative energy minimization problem is solved, in order to find a deformation vector field δ , or the collection of all displacement vectors per iteration step. This is then used to update the source vertex positions as

they are guided towards the target landmarks. Gauss-Newton optimization is used to find the desired deformation vector field at each iteration. Four energy terms are used to control stretching, bending, volume change and landmark matching between source and target. In the original implementation the first three energy terms were weighted by global constants $\lambda = 100$, $\mu = 1$ and $\nu = 1000$ respectively.

Figure 4-13 shows some toy examples of twisting and bending. The twisting example shows the effects of using different combinations of λ and μ . During the experiments, it was noticed that a low value for the stretch constraint elongates the triangles, while a low bend constraint value creates a candy wrap effect. The best balance for λ and μ for the toy experiments was found to be 10 and 100 respectively. These values have the localized deformation effect that plasticine displays (figure 4-14) and also present the least visible triangle distortion.

In the modified version of the deformation algorithm, λ and μ are replaced with per edge stiffnesses, while ν is kept as a global stiffness for all tetrahedra. Tetrahedra correspond to the triangle faces of the mesh. Two techniques were tested to calculate the volume of a tetrahedron, one with respect to the global coordinate system and one considering local coordinate systems around the triangles. It was found that the former produces more stable results and reduces artefacts due to numerical instabilities.

The energy terms used as adapted from Fröhlich & Botsch (2011) are defined as

$$E(\delta) = E_{Stretch} + E_{Bend} + E_{Volume} + E_{Landmarks} \quad (4.60)$$

$$E_{Stretch} = \frac{1}{2} \sum_{e \in E} \frac{K_{Stretch}^e}{L_e^2} (l_e - L_e)^2 \quad (4.61)$$

$$E_{Bend} = \frac{1}{2} \sum_{e \in E} \frac{K_{Bend}^e L_e^2}{A_e} (\theta_e - \Theta_e)^2 \quad (4.62)$$

$$E_{Volume} = \nu \frac{\sum_{f \in F} (v_f - V_f)^2}{2V^2} \quad (4.63)$$

$$E_{Landmarks} = \gamma \frac{1}{2} \sum_{i \in \alpha} (v_{S_i} - v_{T_i})^2 \quad (4.64)$$

where δ is the deformation vector field added to the current source shape to take it

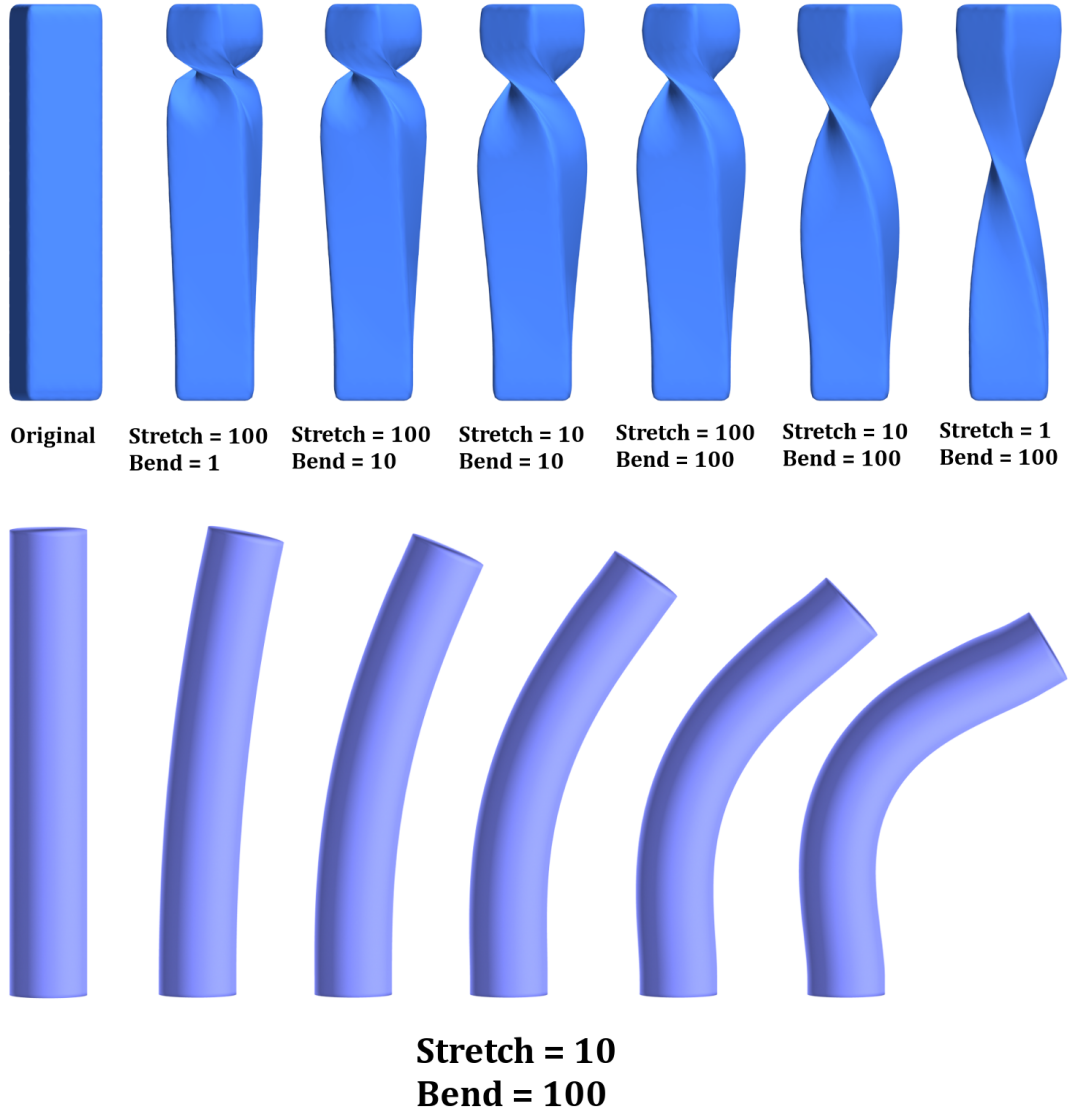


Figure 4-13: Toy example of bar model twisting 67.5° and Botsch & Sorkine (2008) cylinder bending 60° for different settings of stretch (λ) and bend (μ) values. The volume preservation is not used in these cases but was found to not influence the results significantly. These toy examples were used to test the implementation of Fröhlich & Botsch (2011)’s thin shell deformation algorithm.

closer to the target, $e \in E$ is an edge in the set of source edges, $f \in F$ is a face in the set of source faces (triangles), $K_{Stretch}^e$ is the edge stretch stiffness, K_{Bend}^e is the edge bend stiffness, L_e is the initial source edge length, l_e is the current source edge length, Θ_e is the initial source dihedral angle between the 2 incident faces of edge e , θ_e in the

current source dihedral angle, A_e is the initial sum of areas of the incident faces for edge e , V is the initial source volume, v is the current source volume, ν is the volume weight, $i \in \mathcal{H}$ is the landmark i in the set of handle points (landmarks) \mathcal{H} , v_{S_i} is the source landmark i , v_{T_i} is the corresponding target landmark i and γ is the landmark influence weight.

$E_{Stretch}$ is the stretch energy term, which calculates the difference between the current edge lengths and the initial edge lengths. It is weighted per edge by the corresponding edge stretch stiffness, which specifies how much that edge can stretch. The stretch energy links to the first fundamental form (section 4.3.1) and the shear modulus from plasticity theory (Lubliner (2008)).

E_{Bend} is the bend energy term and is weighted per edge by the edge bend stiffness, which describes the amount of change permitted in the dihedral angle between the two faces incident to that edge. The bend energy links to the second fundamental form and the bulk modulus. The relationship between the shear modulus (G) and bulk modulus (B) is

$$G = \frac{3B(1 - 2\nu)}{2(1 + \nu)} \quad (4.65)$$

where in this particular case, ν represents the Poisson ratio (Prager & Hodge (1951)).

E_{Volume} ensures the volume preservation of the mesh. This was modified from the original method to consider a list of tetrahedron volumes, rather than a global volume. Experiments have shown that a list of tetrahedron volumes offers more stability and less shear in the deformation than by using a global volume. Also, volume was considered as a material property, unlike stretching and bending, which can be considered structural properties of the model. A reason for this is that the Poisson ratio, which is linked to the change of local volume, can be approximated with a constant (Crandall et al. (1971)). The ν volume stiffness from the original Fröhlich & Botsch (2011) algorithm can thus remain the same in the current implementation.

$E_{Landmarks}$ is the landmark influence term and is weighted proportional to the stretch and bend stiffnesses. This is to allow a smooth transition between the source and target shapes, while also guaranteeing change for every iteration. The choice of landmarks and their weighting was found to be crucial to the effect of the deformation. A few settings for landmark weights were experimented with, since the original paper by Fröhlich & Botsch (2011) did not specify this term. The results can be seen in the 4.4.2 section.

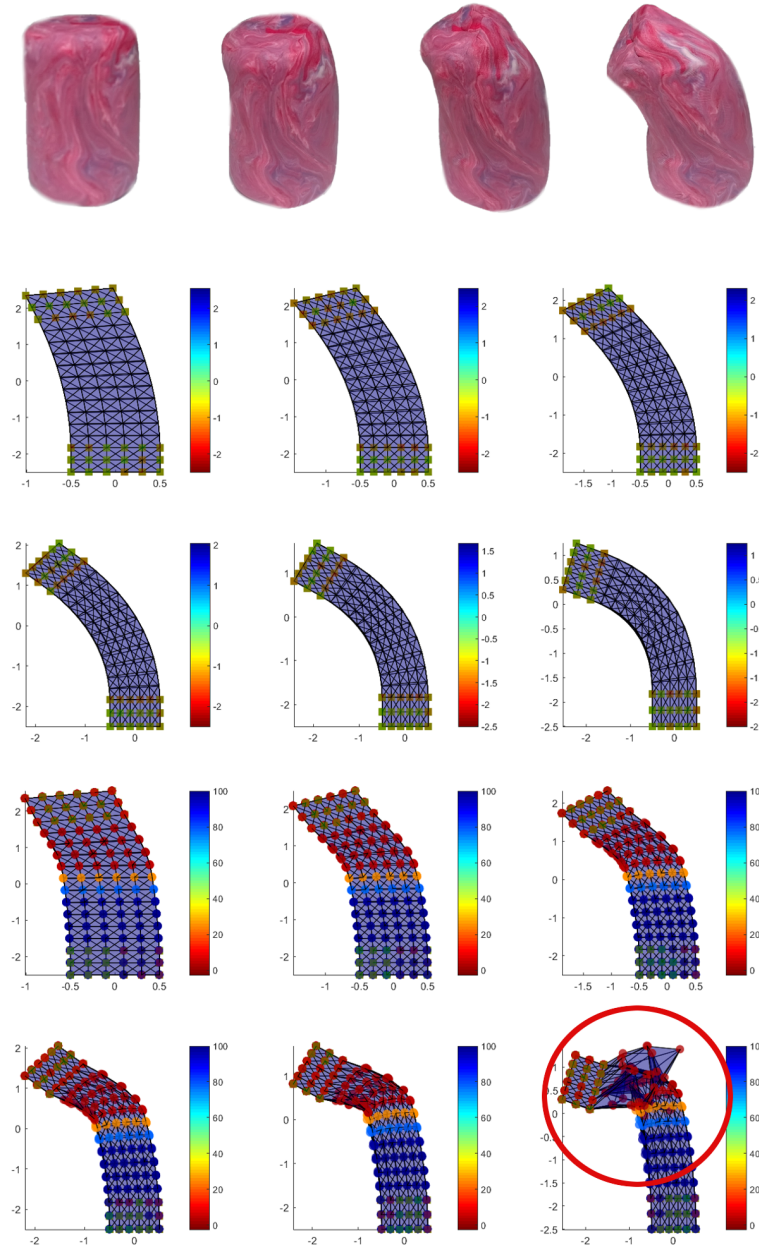


Figure 4-14: The top shows a small bending of a plasticine cylinder. Notice the localized deformation of the tip, where fingers moulded the material. The next six images represent bending a cylinder using the author's implementation of Fröhlich & Botsch (2011). The last six images are the same deformation algorithm, with the added IDW local stiffness. Notice how the proposed improvement makes the bending more similar to the original clay model. Artefacts can occur, however, due to the discontinuities in stiffness values (bottom row).

The most challenging task for the proposed method was to find a link between stretching and bending and how to balance them with the landmark weights. Landmarks needed to be distributed somewhat uniformly along the surface, in order to ensure a fair split of both low (large displacement) and high stiffnesses (low displacement). If either low or high stiffness dominate, then the deformation is inaccurate. This is due to the propensity of landmarks in regions of high displacement to guide the deformation, while landmarks in region of low displacement ensure stability. The latter is similar to the effect of fixed regions in deformation methods.

Implementation

The Fröhlich & Botsch (2011) algorithm was implemented in Matlab and extended to support multiple tetrahedra for volume preservation. The global stretch (λ) and bend (μ) weights were also replaced with per edge localized stiffnesses. Bending and stretching were considered the same per edge for most of the experiments.

At each iteration of the algorithm the landmark stiffnesses are calculated. Their values are then propagated across all the surface vertices, using one of four propagation techniques: inverse distance weights (IDW) + global propagation of landmark influence, IDW + radial propagation, Gauss weights + global propagation or Gauss weights + radial propagation. The distances used for the propagation are geodesics calculated on the mesh.

Note that global propagation of landmark stiffnesses is different from global stiffness. The former signifies that all landmarks are considered when calculating a vertex stiffness. The latter means that all vertices have the same stiffness. The following experimental results are generated on a laptop with Intel Core i7-4710HQ CPU (2.50 GHz) and 16 GB memory.

4.4.2 Experiments and Results

Global Stiffness

The experimentation commences with a case study that investigates configurations for the original Fröhlich & Botsch (2011) algorithm. Figure 4-15 shows the model of choice for this study, Blob Fish from Clay Jam. Figure 4-16 shows the 12 landmarks placed between the source and target mesh scans. The visible mesh is the source, while the wireframe represents the target. For the deformation algorithm, only the target landmarks are of interest, for nonlinear deformation purposes. The resulting



Figure 4-15: Blob Fish source mesh (left) and target mesh (right) from front (top) and side (bottom) views. For the deformation experiments a sparse set of landmarks was considered between the two (figure 4-16) during the nonlinear thin shell deformation.

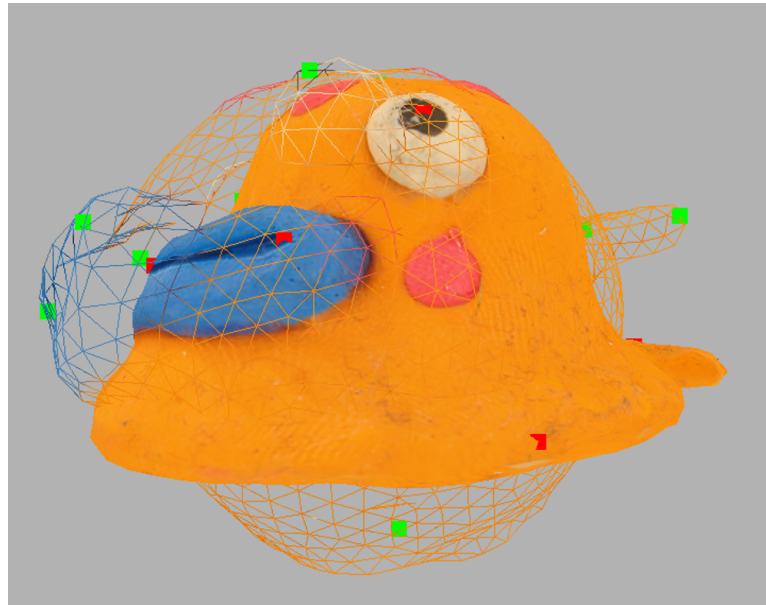


Figure 4-16: A screenshot of the source and target (wireframe) meshes for the Blob Fish character. The target landmarks (green) are the destination points for the source landmarks (red).

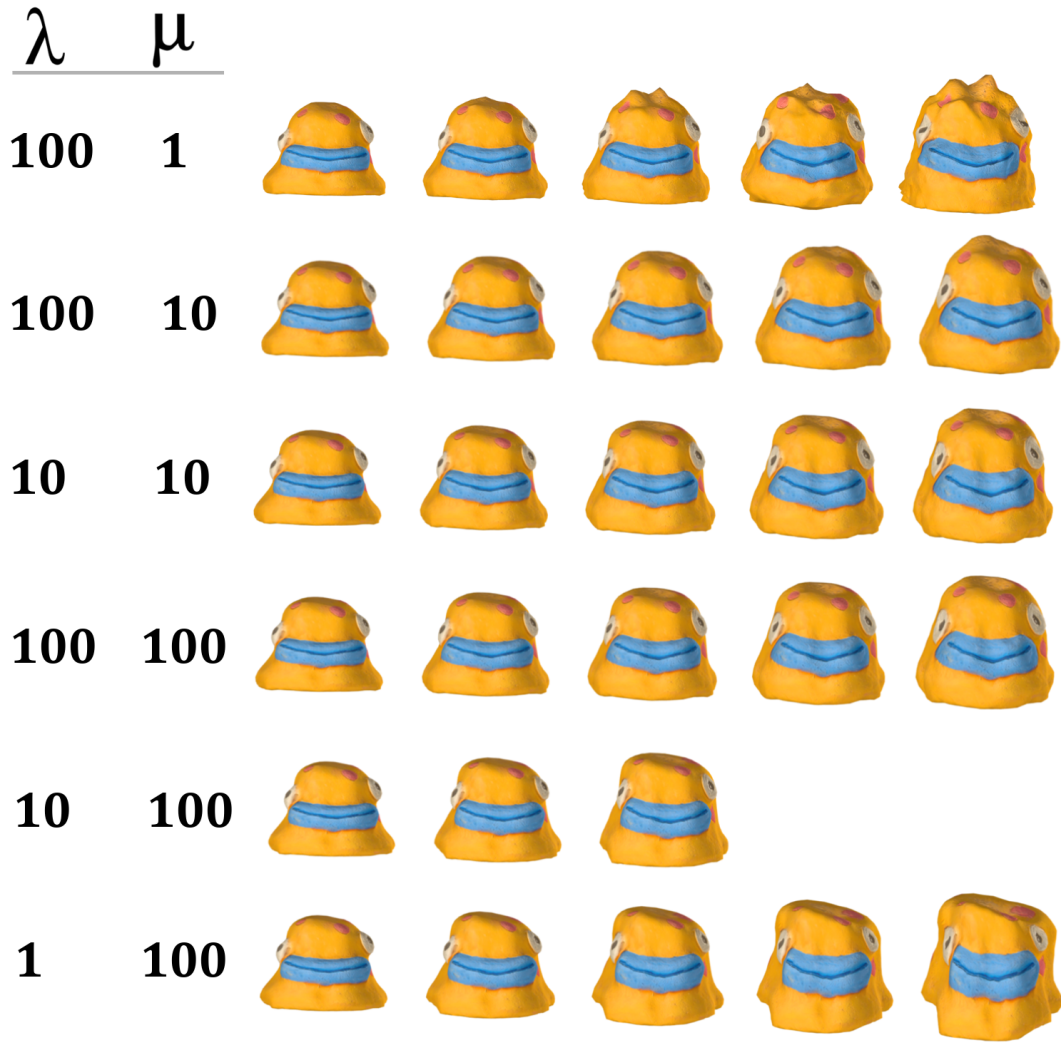


Figure 4-17: Deformations using different values for λ (stretch) and μ (bend) stiffnesses. The volume stiffness is kept constant at $\nu = 1000$. The top sequence was obtained with the Fröhlich & Botsch (2011) settings of $\lambda = 100$, $\mu = 1$, $\nu = 1000$. The following deformations are obtained by changing only the λ (stretch) and μ (bend) global stiffnesses.

deformation can then be used to initialize non-rigid registration, as proposed in chapter 5. The landmark weight is chosen as the product between the maximum stiffness and a constant factor.

```
factor = 1000
maxStiffness = max( $\lambda$ ,  $\mu$ )
```

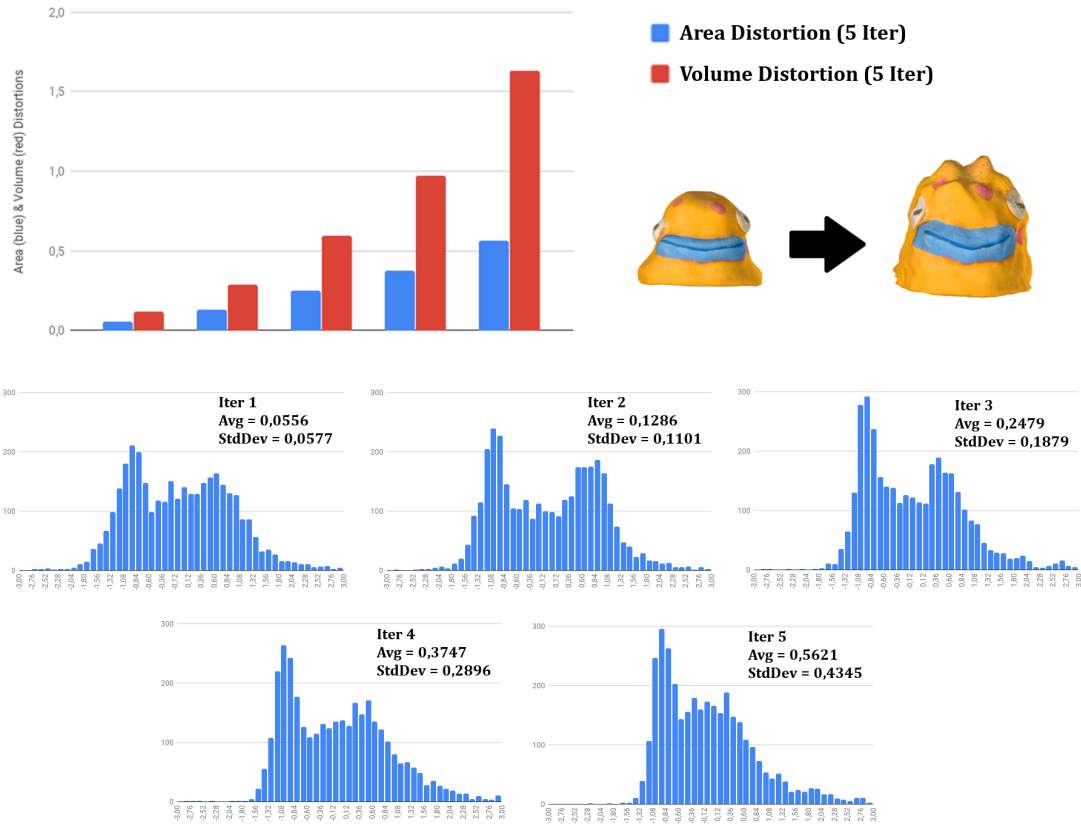


Figure 4-18: Cumulative relative area and volume distortion with respect to the original values (top). The normal distribution of areas over 5 iterations (bottom) of thin shell deformation, using the original method, implemented from Fröhlich & Botsch (2011)

`landmarkWeight = factor * maxStiffness`

The first row of figure 4-17 shows the result of the deformation with the original weights or global stiffnesses from Fröhlich & Botsch (2011), $\lambda = 100$ (stretch), $\mu = 1$ (bend) and $\nu = 1000$ (volume). Notice that due to the small bending stiffness high curvature features appear on the model as it attempts to displace 12 landmarks. Numerical instabilities can appear when the constraints of preserving the surface smoothness are outweighed by the landmark weights.

This can be observed in the cumulative area and volume distortions of Blob Fish in figure 4-18. The graphs at the bottom of the figure show that triangles have a tendency of shrinking towards the end of the 5 iterations. This effect could explain the small curved features that appear on the character at the end of the deformation (top right of figure 4-18).

The following rows in figure 4-17 show the results of varying the stretch and bend weights for the original algorithm. Notice that a large stretch weight (λ) corresponds to a small size growth of the character. Similarly, a large bend weight (μ) corresponds to a small variation in curvature. In the bottom row of figure 4-17 the last pose of Blob Fish almost keeps its original curvature, while the small λ allows the shape to stretch too much.

Landmark Weights

The ideal weighting for landmarks has been found through many iterations as a balance between preserving the mesh smoothness via stretching and bending constraints and allowing deformation. When the landmark weights are too small, the shape does not deform significantly, due to the stretch and bend constraints preserving the shape. When landmark weights are too large, spike effects tend to appear on the surface, as their influence outweigh the strength of the soft constraints. A quick fix for this last issue was to recalculate the landmark vertex position after an iteration as the average position of its neighbours.

The weight combination $\lambda = 10$ and $\mu = 10$ was chosen as it seemed to preserve the shape and volume best in figure 4-17. Experiments then focused on the different weights for the 12 available landmarks. If previously `factor = 1000`, the respective result was compared with the effects of `factor = 100`, `factor = 10` and `factor = 1`. Figure 4-19 shows a visual comparison of each deformation setting on Blob Fish.

Table 4.1 shows the minimum and maximum Hausdorff distances and the time taken to deform the mesh over 5 iterations. Four landmark factor values are used to calculate the respective weights, as presented in the previous section. The Hausdorff distances are between the deformed source mesh and the target mesh, to validate whether the deformation is following the shortest path of deformation towards the target.

Table 4.1: Minimum and maximum Hausdorff distances for each landmark factor. The global stiffnesses considered are $\lambda = 10$, $\mu = 10$, $\nu = 1000$. The total deformation time is also displayed in seconds.

Landmark factor	Min. Hausdorff Distance	Max. Hausdorff Distance	Time (s)
1000	0.0923	0.1249	163.61
100	0.0911	0.1246	182.14
10	0.1065	0.1317	175.12
1	0.1027	0.1328	185.14

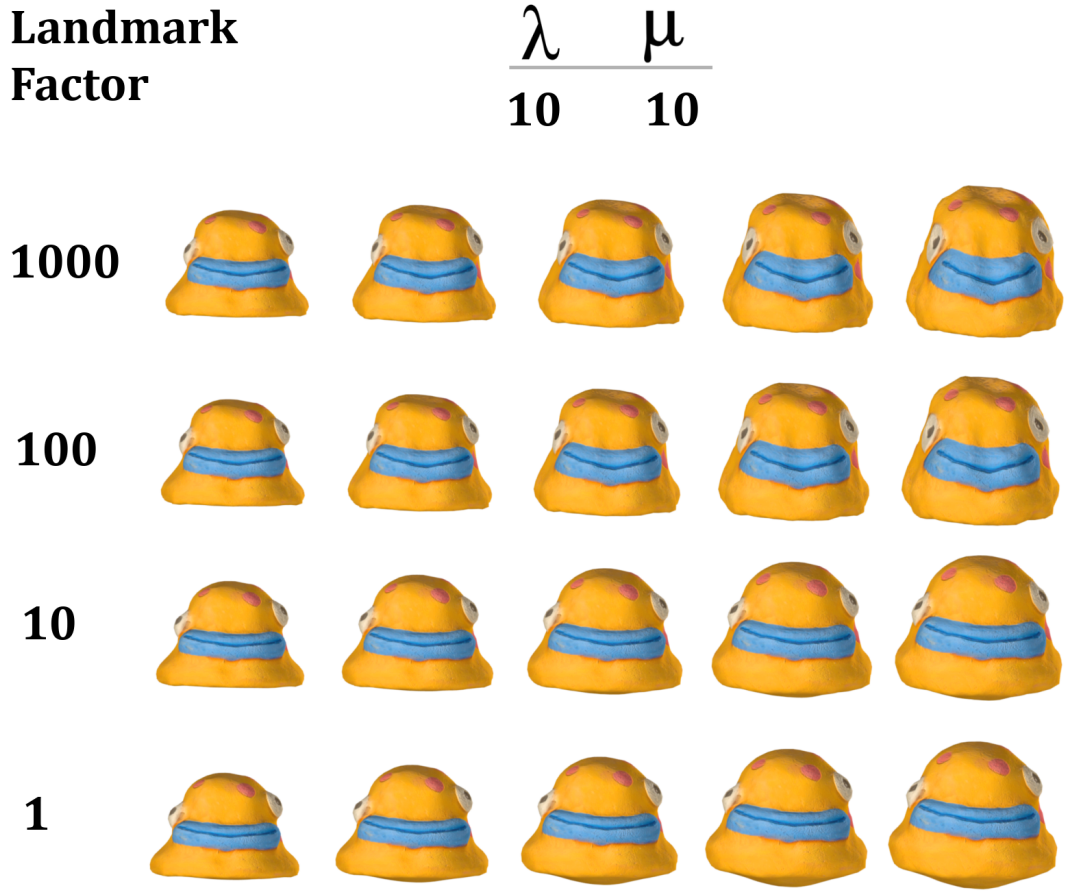


Figure 4-19: Deformations using different values for the landmark factor. The landmark global weight is then calculates as $\text{landmarkWeight} = \text{factor} * \max(\lambda, \mu)$.

Table 4.2: Average relative area and volume distortions during the nonlinear deformation with $\lambda = 10$, $\mu = 10$, $\nu = 1000$, between the source mesh and the target landmarks. Negative values signify the mesh is shrinking, while positive values mean it is expanding. The area and volume are taken relative to the original values of the source mesh.

Landmark factor	Area Dist.	Area σ	Volume Dist.	Volume σ
1000	0.2628	0.1798	0.6639	0.5019
100	0.2584	0.1737	0.6432	0.4769
10	0.2600	0.1637	0.6248	0.4326
1	0.2578	0.1616	0.6177	0.4265

Table 4.2 contains the average relative area and volume distortions for each landmark factor. The area and volume distortions were taken as the average and standard devi-

ation of surface distortions at each of the 5 possible deformation steps (i.e. iterations). Notice from the two tables that there is a trade-off between the proximity of the source mesh to the target mesh and the amount of distortion the shape undergoes. The minimum values are shown in bold in the tables.

Local Stiffness

The local stiffness formulations are then introduced into the nonlinear, thin shell deformation algorithm. The goal was to see whether local stiffness reduces area and volume distortions during deformation. The Hausdorff distance was also measured between the source and target meshes, although only a sparse set of landmarks guides the deformation between the two. This was done to see whether local stiffness allows a quicker deformation path towards the target.

The first scenario considered was to calculate the stiffness per source vertex by using all the landmarks, regardless of the distance to them. Figure 4-25 shows the deformation using inverse distance weights (IDW) local stiffness for Blob Fish, during four iterations. The red areas signify lower stiffness, equivalent to larger displacements needed to reach the target landmarks. Similarly, the blue areas signify higher stiffness, where the mesh has smaller displacements.

The IDW, as defined in figure 4-10, was tested with an exponent of two (tables 4.3 and 4.4). Figure 4-20 shows a comparison regarding the Hausdorff distance values, between the global and IDW local stiffness deformations. Each column represents one of the landmark factors tested. Notice that the local stiffness usually provides smaller distances, which means that the deformed source shape is closer to reference target shape. Similarly, figure 4-21 reveals that the average relative area distortion is smaller when using local IDW stiffness than the original, global one.

Table 4.3: Minimum and maximum Hausdorff distances for each landmark factor. The local stiffness is **propagated globally** from landmark stiffness values via Shepard's method (equation 4.57). The **inverse distance weights** are calculated with an exponent value of 2. The total deformation time is also displayed in seconds.

Landmark factor	Min. Hausdorff Distance	Max. Hausdorff Distance	Time (s)
1000	0.0886	0.1304	331.84
100	0.0885	0.1303	324.01
10	0.0947	0.1280	409.72
1	0.0841	0.1261	255.63

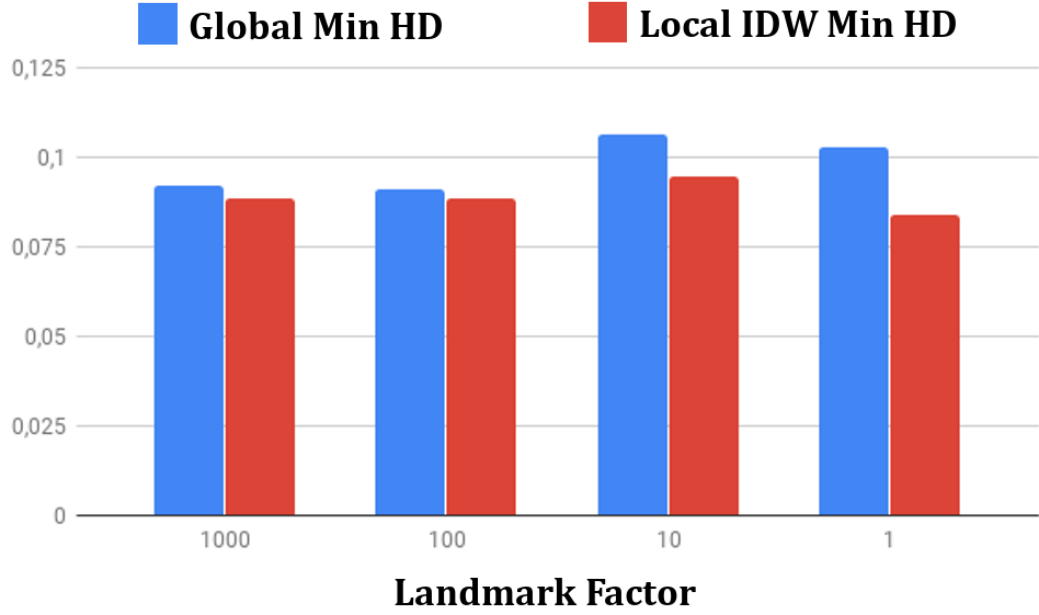


Figure 4-20: Hausdorff distances for each landmark factor during global (blue) and local IDW (red) deformations.

Table 4.4: Average relative area and volume distortions during the nonlinear deformation between the source mesh and the target landmarks. The local stiffness is **propagated globally** from landmark stiffness values via Shepard’s method (equation 4.57). The **inverse distance weights** are calculated with an exponent value of 2. Negative values signify the mesh is shrinking, while positive values mean it is expanding. The area and volume are taken relative to the original values of the source mesh.

Landmark factor	Area Dist.	Area σ	Volume Dist.	Volume σ
1000	0.1958	0.1289	0.4731	0.3374
100	0.1957	0.1287	0.4726	0.3367
10	0.2505	0.1676	0.6174	0.4529
1	0.1465	0.0807	0.3264	0.1925

Gauss weights, as defined in figure 4-10, followed in the experimentation, also with an exponent of 2 (tables 4.5 and 4.6). The observations were similar to the IDW results. The Hausdorff distance, area and volume distortions were generally smaller when using local Gauss stiffness, compared to the global stiffness case. The IDW, however, provided slightly better results than the Gauss stiffness ones.

Next the radial version of the local stiffness was used, as described in algorithm 1. This calculates the stiffness per vertex by only considering landmarks in close proximity.

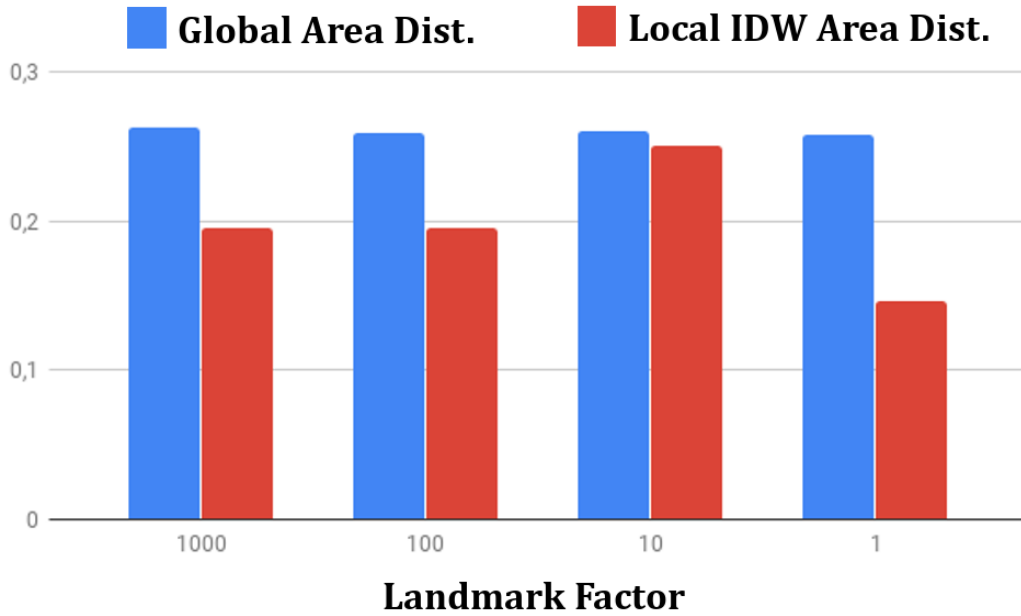


Figure 4-21: Average relative area distortions for each landmark factor during global (blue) and local IDW (red) deformations.

Table 4.5: Minimum and maximum Hausdorff distances for each landmark factor. The local stiffness is **propagated globally** from landmark stiffness values via Shepard's method (equation 4.57). The **Gauss weights** are calculated with an exponent value of 2. The total deformation time is also displayed in seconds.

Landmark factor	Min. Hausdorff Distance	Max. Hausdorff Distance	Time (s)
1000	0.0919	0.1248	410.56
100	0.0891	0.1244	340.77
10	0.1033	0.1301	387.08
1	0.1017	0.1320	394.63

Inside and outside thresholds are used to delimit the influence from landmarks. The first experiments were with inverse distance weights (tables 4.7 and 4.8) and then with Gauss weights (tables 4.9 and 4.10). The visual comparison between global and radial stiffness propagation can be seen in figure 4-22.

The Hausdorff distance, area and volume distortions were overall smaller when using a radial stiffness propagation, compared to the original algorithm. They were, however, larger than the global stiffness propagation deformation results. In other words, the method that provided the best quality of deformation for the Blob Fish character was

Table 4.6: Average relative area and volume distortions during the nonlinear deformation between the source mesh and the target landmarks. The local stiffness is **propagated globally** from landmark stiffness values via Shepard’s method (equation 4.57). The **Gauss weights** are calculated with an exponent value of 2. Negative values signify the mesh is shrinking, while positive values mean it is expanding. The area and volume are taken relative to the original values of the source mesh.

Landmark factor	Area Dist.	Area σ	Volume Dist.	Volume σ
1000	0.2599	0.1772	0.6545	0.4925
100	0.1973	0.1266	0.4711	0.3281
10	0.2553	0.1618	0.6161	0.4299
1	0.2550	0.1596	0.6099	0.4202

Table 4.7: Minimum and maximum Hausdorff distances for each landmark factor. The local stiffness is **propagated radially** from landmark stiffness values via Shepard’s method (equation 4.57). The **inverse distance weights** are calculated with an exponent value of 2. The total deformation time is also displayed in seconds.

Landmark factor	Min. Hausdorff Distance	Max. Hausdorff Distance	Time (s)
1000	0.0910	0.1309	408.12
100	0.0909	0.1309	402.09
10	0.0999	0.1299	399.48
1	0.0946	0.1291	419.07

Table 4.8: Average relative area and volume distortions during the nonlinear deformation between the source mesh and the target landmarks. The local stiffness is **propagated radially** from landmark stiffness values via Shepard’s method (equation 4.57). The **inverse distance weights** are calculated with an exponent value of 2. Negative values signify the mesh is shrinking, while positive values mean it is expanding. The area and volume are taken relative to the original values of the source mesh.

Landmark factor	Area Dist.	Area σ	Volume Dist.	Volume σ
1000	0.2625	0.1780	0.6641	0.4986
100	0.2620	0.1774	0.6614	0.4954
10	0.2541	0.1660	0.6215	0.4465
1	0.2500	0.1569	0.5983	0.4132

the local IDW with global propagation. This proposed method managed to outweigh the original nonlinear algorithm presented by Fröhlich & Botsch (2011) in the shape deformation properties. The times, however, were still smaller with the original method.

Another observation was that landmarks are not reached during deformation. This is most likely due to a large Hausdorff distance threshold of 0.09 and solving the problem in a least squares sense. This threshold was chosen after observing large processing

Table 4.9: Minimum and maximum Hausdorff distances for each landmark factor. The local stiffness is **propagated radially** from landmark stiffness values via Shepard’s method (equation 4.57). The **Gauss weights** are calculated with an exponent value of 2. The total deformation time is also displayed in seconds.

Landmark factor	Min. Hausdorff Distance	Max. Hausdorff Distance	Time (s)
1000	0.0916	0.1247	383.76
100	0.0902	0.1245	405.39
10	0.1006	0.1266	402.48
1	0.0999	0.1322	413.45

Table 4.10: Average relative area and volume distortions during the nonlinear deformation between the source mesh and the target landmarks. The local stiffness is **propagated radially** from landmark stiffness values via Shepard’s method (equation 4.57). The **Gauss weights** are calculated with an exponent value of 2. Negative values signify the mesh is shrinking, while positive values mean it is expanding. The area and volume are taken relative to the original values of the source mesh.

Landmark factor	Area Dist.	Area σ	Volume Dist.	Volume σ
1000	0.2586	0.1759	0.6497	0.4879
100	0.2579	0.1750	0.6463	0.4839
10	0.2518	0.1638	0.6120	0.4371
1	0.2540	0.1592	0.6072	0.4186

times when its value was too small. Lastly, the comparison between global and local stiffness deformation was performed on three more Clay Jam characters, as it can be seen in the next subsection.

Local versus Global Stiffness

The local stiffness setting with the smallest Hausdorff distance and the smallest area and volume distortion was chosen. As seen in the previous section, this was the inverse distance weights (IDW) local stiffness with a global propagation, an exponent of 2 and a landmark factor of 1. Although the results for this chosen setup were better than the deformations using the original algorithm by Fröhlich & Botsch (2011), only three iterations were used to obtain it (bottom of figure 4-24). This was because the threshold Hausdorff distance was reached before the designated 5 iterations per deformation could be completed.

The goal was to see how this setting performed on other types of Clay Jam characters. A comparison was done between the global stiffness setting of $\lambda = 10$, $\mu = 10$ and $\nu = 1000$ and the author’s local IDW stiffness deformation method. Tables 4.11 and

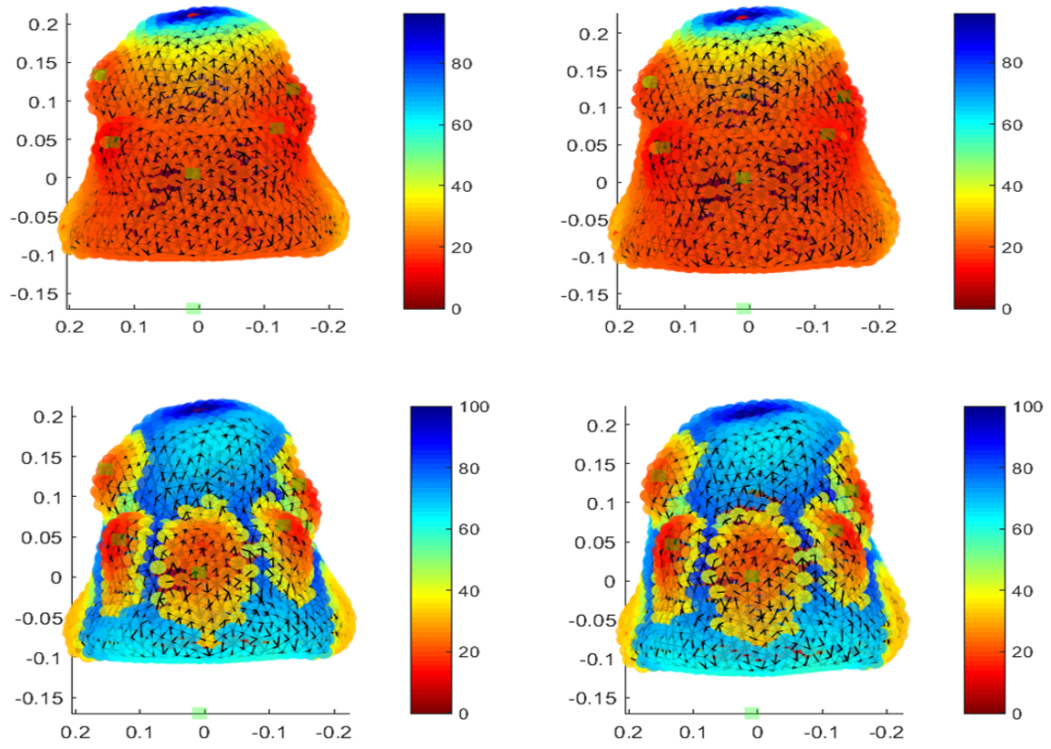


Figure 4-22: Localized stiffness with global (top) and radial (bottom) IDW propagation of landmark stiffnesses.

Table 4.11: Minimum and maximum Hausdorff distances for **global stiffness** nonlinear deformation ($\lambda = 10$, $\mu = 10$, $\nu = 1000$) and landmark factor = 1. The total deformation time is also displayed in seconds.

Character	Faces	Lmrks	Min. Hausdorff Distance	Max. Hausdorff Distance	Time (s)
Blob Fish	3482	12	0.1027	0.1328	158.89
Party Onion	3600	13	0.1799	0.1941	172.19
Tick	2750	13	0.3015	0.4037	91.76
Hellidropter	1454	10	0.0876	N/A	4.70

4.12 show a comparison between the Hausdorff distance values for the global and local IDW stiffness respectively. Similarly tables 4.13 and 4.14 compare the average relative area and volume distortions for the two algorithms. Figure 4-23 shows an example of the local IDW stiffness map during the deformation of Party Onion over 3 iterations.

Notice that with the exception of the maximum Hausdorff distance and volume standard deviation for the Tick character, all other Hausdorff distances, average area and

Table 4.12: Minimum and maximum Hausdorff distances for **local stiffness** nonlinear deformation and landmark factor = 1. Stiffness is **propagated globally** from landmark stiffness values via Shepard’s method (equation 4.57). The **inverse distance weights** are calculated with an exponent value of 2. The total deformation time is also displayed in seconds.

Character	Faces	Lmrks	Min. Hausdorff Distance	Max. Hausdorff Distance	Time (s)
Blob Fish	3482	12	0.0841	0.1261	255.63
Party Onion	3600	13	0.1605	0.1755	440.62
Tick	2750	13	0.2747	0.4053	262.00
Hellidropter	1454	10	0.0863	N/A	12.10

Table 4.13: Average relative area and volume distortions for **global stiffness** nonlinear deformation ($\lambda = 10$, $\mu = 10$, $\nu = 1000$) and landmark factor = 1. Negative values signify the mesh is shrinking, while positive values mean it is expanding.

Character	Faces	Lmrks	Area Distortion	Area σ	Volume Distortion	Volume σ
Blob Fish	3482	12	0.2578	0.1616	0.6177	0.4265
Party Onion	3600	13	0.3264	0.1985	0.7955	0.5384
Tick	2750	13	0.3264	0.1958	0.7956	0.5316
Hellidropter	1454	10	0.1031	N/A	0.2181	N/A

Table 4.14: Average relative area and volume distortions for **local stiffness** nonlinear deformation and landmark factor = 1. Stiffness is **propagated globally** from landmark stiffness values via Shepard’s method (equation 4.57). The **inverse distance weights** are calculated with an exponent value of 2. Negative values signify the mesh is shrinking, while positive values mean it is expanding.

Character	Faces	Lmrks	Area Distortion	Area σ	Volume Distortion	Volume σ
Blob Fish	3482	12	0.1465	0.0807	0.3264	0.1925
Party Onion	3600	13	0.3198	0.1968	0.7802	0.5345
Tick	2750	13	0.3054	0.1869	0.7520	0.5155
Hellidropter	1454	10	0.1019	N/A	0.2154	N/A

volume distortions are smaller when using the proposed localized stiffness deformation. However, although the local stiffness deformation seems to be giving better deforming models, the visual results have rather small differences from the global deformation method (figure 4-24).

Figure 4-24 shows a visual comparison between a global stiffness and a local stiffness deformation. The latter reaches the Hausdorff distance threshold in only 3 iterations, compared to the 5 iterations of the global stiffness deformation. Also, the final shape

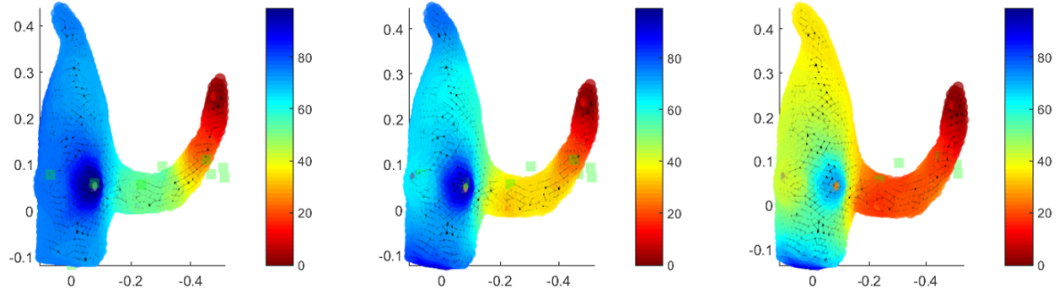


Figure 4-23: Three iterations of Party Onion local stiffness deformation. The inverse distance weights used have an exponent factor of 2. The stiffness is propagated globally from 13 landmark stiffnesses.



Figure 4-24: Original global stiffness deformation (top) with $\lambda = 10$, $\mu = 10$, $\nu = 1000$ versus local stiffness deformation (bottom) with globally propagated inverse distance weights stiffness for Blob Fish.

is slightly smaller in the local stiffness case, signifying that volume is preserved better. The respective values in table 4.14, compared to the ones in table 4.13 confirm this observation.

4.4.3 Discussion

A nonlinear, thin shell deformation technique was proposed, which builds on the work of Fröhlich & Botsch (2011) by adding localized stiffness to guide the deformation. A source and a target mesh are used, with a sparse set of landmarks between them. The deformation considers only the source mesh and the target landmarks. It was shown that the localized stiffness deformation method has smaller area and volume distortions than the original method. Also, the Hausdorff distances between the deformed source and the target shapes are overall smaller when using local stiffness.

Research questions 1 and 2 were partially answered in this chapter. It is possible to replicate the behaviour of solid plasticine using only a surface based deformation model. However, the settings needed to facilitate physically plausible deformation are not yet intuitive to setup. Also, more experimentation is needed to see where the ideal balance is between stretch, bend and landmark stiffnesses for more intuitive deformations.

Moreover, in the stiffness propagation algorithm, geodesic distances between vertices are used. Although geodesics give a better approximation of distances between vertices on a mesh than Euclidean distances, they are sensitive to noise (Panozzo et al. (2013)). This would not allow the current method to be extended easily to non-manifold meshes. Alternatives include high dimensional Euclidean-embedding metrics that approximate the surface based distances. Some examples of such metrics are surveyed by Lipman et al. (2010) and include diffusion distance, commute-time distance and biharmonic distance methods.

The next chapter presents the *Non-rigid Registration* step of the E-StopMotion pipeline in more detail. The proposed nonlinear deformation method will be used as one of the enhancements for the registration. By reverse engineering the deformation a model has undergone, the hope is to obtain more accurate matches and physically plausible in-between shapes.

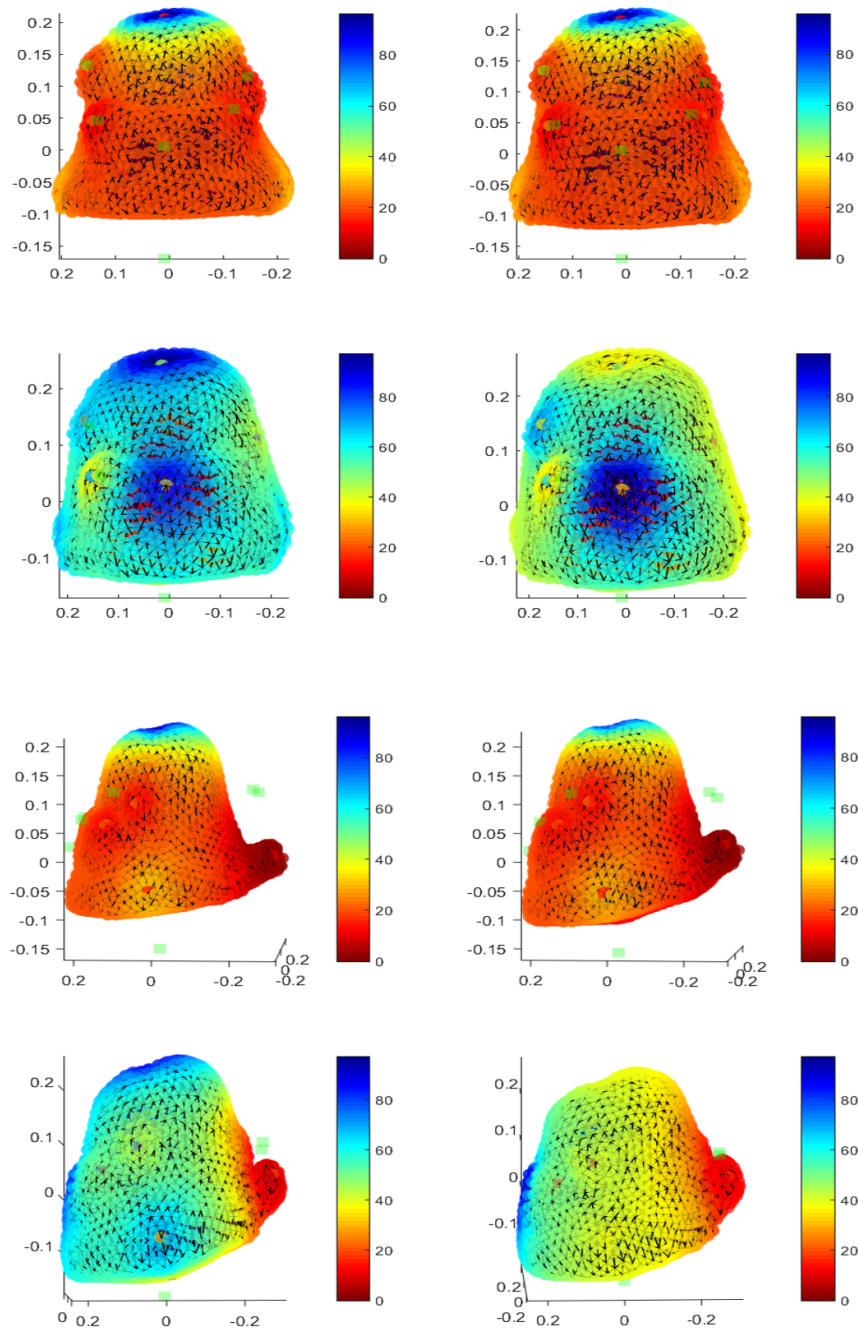


Figure 4-25: Local stiffness map calculated with inverse distance weights (IDW), propagated globally, using Shepard's method. All landmarks are considered when calculating the stiffness per vertex. Landmark factor is 10 (table 4.3). Red signifies low stiffness, blue is high stiffness. Notice how the stiffness increases as the source gets closer to the target landmarks (green).

Chapter 5

Registration of Virtual Clay Models

After seeing the properties of the Clay Jam plasticine characters and the physically plausible deformation methods that can imitate their movement, the *Non-rigid Registration* step of the E-StopMotion animation reconstruction pipeline (figure 3-3) can now be explored in detail.

Research question 4 is tackled in this chapter by studying suitable registration techniques for plasticine model scans. A method that would also allow physically plausible in-between shapes during registration is desirable. These shapes can later be used for refining the original stop motion animation.

An extensive literature review of different types of non-rigid registration is first presented, followed by a description of Amberg et al. (2007)’s non-rigid iterative closest point (NRICP) algorithm. The author’s contribution is to enhance the original method with the deformation technique developed in chapter 4. A set of other enhancements, linked to the *Animation by Shape* helpers described in chapter 2, were also attempted. These include techniques to generate dense correspondences that can guide the registration.

Given a source shape and a desired target shape, the non-rigid registration technique returns a transformation that morphs the former as closely as possible into the latter. This procedure usually requires a correspondences step and a trajectory step. The correspondence step implies finding common features on both the source and target shape. The trajectory step involves the regularization method that controls the deformation

of the source shape into the target shape, given the chosen correspondences.

5.1 Non-rigid Registration

Several surveys discussing shape registration (Tam et al. (2013)), correspondences (van Kaick et al. (2011), Biasotti et al. (2016)) and shape mapping (Li & Iyengar (2014)) have appeared in recent years. Most of the work focuses on isometric or nearly isometric shape deformations, which preserve the local shape as accurately as possible.

Shape registration usually refers to obtaining a transformation between two or more meshes and does not always require a one-to-one mapping between the template and target shapes. An example is Brown & Rusinkiewicz (2007)’s work on aligning almost-rigid 3D scans, that differ through low-frequency deformations, due to noise or general calibration issues.

Shape correspondence deals with mapping a sparse or dense set of points between two or more meshes. Methods from this category work well with meshes that maintain a consistent global intrinsic structure during deformations. In other words, meshes are isometric or nearly isometric to each other. An example is Lipman & Funkhouser (2009) who find correspondences between nearly isometric meshes using Möbius voting.

Shape mapping is described as a bijective map between two or more meshes with nontrivial topologies. Parametrization of shapes can be considered as a particular case of shape mapping, when the meshes involved have a similar topology. van Kaick et al. (2011) also mention the term of embeddings, which can either go from a higher dimension to a lower dimension (3D to 2D parametrization) or the other way around.

Spectral transforms are a popular method of obtaining a higher dimensional representation of a shape. They allow the transformation of intrinsically complex shapes into their simpler extrinsic representations. This reduces the number of degrees of freedom for each deformation and permits solving nonlinear problems as linear ones in the new embedded space. An example would be Jain et al. (2007), who use spectral domain embedding to tackle general non-rigid deformations.

Bronstein et al. (2008) mention that using a parametrization to simplify registration usually introduces errors related to approximating a curved surface with a planar mapping. This is a direct consequence of *Theorema Egregium*, which states that Gauss curvature is an intrinsic property. This means that Gauss curvature shouldn’t change,

regardless of the parametrization. Since a plane’s curvature is zero and a curved surface’s curvature is always higher than zero, it is mathematically impossible to approximate one with the other. This observation led to surface to surface mappings, which allowed more accurate correspondences and matches (Kraevoy & Sheffer (2004), Bronstein et al. (2006), Zell & Botsch (2013)). It introduced constraints, however, like inserting additional vertices, isometry and smoothing out the source and target meshes.

Since plasticine can deform in any direction, with irreversible changes, it becomes intuitive to focus the search on non-isometric registration methods. In other words, the distance between two points on the source shape might be significantly different on the target shape. Non-isometric shape registration is slowly gaining more interest from authors like Ganapathi-Subramanian et al. (2016), who discuss finding correspondences between shape regions. They consider the latter more appropriate than point-to-point correspondences, especially when the models share similarities, but are from different categories. In order for two regions to correspond, however, they need to score a certain rank in feature correspondences. This solution is not sufficient in the case of plasticine models, as features can vanish completely from one frame to the next.

Structural and material properties are considered by Dey et al. (2015), when proposing a finite element structure to correct volume changes during registration. Kanazawa et al. (2016) learn stiffness maps from a set of 2D to 3D correspondences and argue that localized stiffness allows a more intuitive deformation for articulated models. Surface based plastic deformation and registration can also be found in the area of simplified simulations. Morph targets are becoming popular for replacing complex simulations with real time shape deformations Jones et al. (2016), Schulz et al. (2014). The results, however, are as good as the intermediate poses provided, which usually require extra work from the artist designing them.

One of the goals of this thesis is to simplify the artist’s workload as much as possible, by generating the in-between shapes from the deformation. A combination of the deformation methods discussed in chapter 4 and the non-rigid registration algorithm proposed by Amberg et al. (2007) is done in order to obtain a reliable registration and intuitive in-between poses.

By using the non-rigid registration (NRICP) method proposed by Amberg et al. (2007), affine transformations are allowed per vertex. This implies that a vertex can freely rotate, scale, shear and translate in Euclidean space, which bears a close resemblance to the real world phenomenon. Volume preservation might be needed, however, in order to preserve the overall shape and prevent artefacts.

5.1.1 Scales of Deformation

Another way of viewing registration, correspondence and mapping would be from the point of view of deformation scale. Smaller deformations tend to be considered closer to the rigid deformations and can be solved efficiently with extrinsic, global registration methods such as the iterative closest point (ICP) algorithm or its variants (Amberg et al. (2007)). As the differences become larger, more information about the intrinsic structure of the shapes becomes necessary for finding accurate correspondences or mappings. An example of handling nonisometric, large deformations is the work of Kim et al. (2011) who use blended intrinsic maps to morph between surfaces that have different scales of local deformation.

Secondary areas of research related to the previous three are shape retrieval, cross-parametrization, (statistical) shape analysis and processing, soft body deformation, segmentation, feature and skeleton extraction etc. This literature review will mention and analyze a set of papers that might belong to one or more primary or secondary research areas.

Extensive work has been done in the area of matching characters obtained from scanners (Allen et al. (2003), Anguelov et al. (2004, 2005), Gal & Cohen-Or (2006), Mitra et al. (2007), Li et al. (2008, 2009), Chang & Zwicker (2009), Cheng et al. (2010), Papazov & Burschka (2011), Tevs et al. (2012), Zhao et al. (2013), Bonarrigo et al. (2014), Dey et al. (2015)). These characters can be either static or in motion, while the scanner captures their geometry. The current problem is part of the former category, although some work from the latter category, also known as space-time registration, is worth mentioning.

Regardless of the motion of the character or the speed of the scanner, different scales of deformation occur. The general tendency of the literature is to have simpler, global, extrinsic algorithms for smaller deformations (Allen et al. (2003), Mitra et al. (2007), Hontani et al. (2012)), while larger deformations need more information about the local, intrinsic and semantic structure of the model (Huang et al. (2008), Li et al. (2009), Bronstein et al. (2010), Dey et al. (2015)). Bonarrigo et al. (2014) have a similar discussion on small and large deformations, where similar conclusions are drawn.

When deformations are medium or large, many authors tend to divide the problem into simpler, more manageable tasks. An effective strategy is the coarse-to-fine approach (Li et al. (2009), Zhang et al. (2008), Yang et al. (2015)), which can be linked to the resolution of the mesh being deformed. Other terms linked to this method are sparse-

to-dense correspondences (Zell & Botsch (2013), Dey et al. (2015)) and stiff-to-smooth transformations (Amberg et al. (2007), Bonarrigo et al. (2014)). The idea behind this strategy is to first fit the general shapes together and then attend to matching the finer details. Embedding is also related to this simplification area and will be discussed more in the helpers section.

Another important aspect of the available work on shape matching is the use of a template as prior knowledge. Since obtaining meshes is relatively easy with the current setup and the use of a template can lead to robust and fast results (Dey et al. (2015), Yang et al. (2015)), this method was considered suitable. However, clean versions of the scans are used as templates and not universal or parametric models, to avoid the difficulties that may appear during registration (Allen et al. (2003), Hirshberg et al. (2012)), especially since the shapes, sizes and deformations of characters are varied.

Since the Clay Jam character appearances and deformations are so varied, it is difficult to define a general registration method that would satisfy all types of animation. Chapter 2 succinctly classifies these animation types into *Animation by Skeleton* and *Animation by Shape*. Although this classification informs the types of helpers needed for registration, most characters are usually composed of parts that deform in different ways. Thus, segmentation is definitely a helper worth considering, as discussed in chapter 2.

Regardless of the chosen techniques, however, the scale of deformation (discussed in chapter 2) remains the factor that influences the intrinsic information needed (eg. feature points, skeletons, segmentation). A body part can have small, medium or large deformations in the same animation loop and it is difficult to predict what the next pose would be. A few papers are presented next by the scale of deformation covered during registration or matching algorithms.

Small

Sharf et al. (2008) present a volumetric space-time technique for the reconstruction of geometry and motion of characters. An incompressible flow of material is used to preserve density of the solid over time. The output is a set of watertight meshes representing the motion. Although this representation of a mesh is similar to clay representation, it only allows small deformations from frame to frame, is heavy computation-wise and is tedious to render.

Allen et al. (2003) introduced a frequently cited template-based non-rigid registration technique to fit template meshes to high resolution range scans. This allows the creation

of a database of meshes with the same topology, which can then be manipulated to obtain intermediate shapes. The objective function used is based on the minimization of an energy containing three terms, a distance term, a transformation smoothness term and a landmark term. Although the algorithm performs well from similar poses and characters, it can easily get trapped in local minima when larger differences between the template and target mesh occur. Manually placed landmarks are crucial for good results in such situations. However, they can be tedious to place manually and do not allow differences in pose, as the smoothness term in the energy function penalizes that.

In an attempt to improve the quality of the match, Amberg et al. (2007) extend Allen et al. (2003) by combining it with the iterative closest point algorithm (ICP). New correspondences between the template and the target mesh are calculated at every iteration, thus eliminating the black-box effect of the previous algorithm. They also introduce a stiffness term that allows non-rigid local deformations. Although this results in better registrations, even without landmarks, the deformations permitted are still relatively small and can get trapped in local minima when larger deformations are present (Li et al. (2008), Cheng et al. (2010), van Kaick et al. (2011)).

Medium

In a similar spirit to Allen et al. (2003) and Amberg et al. (2007), medium deformations were made possible with the aid of a deformation graph described by Li et al. (2008). The following two papers also make use of an energy functional which combines proximity heuristics with smoothness regularization.

Li et al. (2008) introduced a non-rigid, nonlinear registration technique for partial range scans. Their method automatically combines finding correspondences and confidence weights, detecting non-overlapping areas and using a warping field to align the source and the target meshes. The optimization used maximizes the overlap areas while minimizing the alignment error. A deformation graph is introduced to facilitate both rigid and non-rigid deformations of vertices. The resolution of the graph is directly linked to the resolution of the mesh, however, and can lead to large computational costs. Moreover, the key feature and also the limitation of the algorithm is the use of depth information from the scanner to parametrize the target mesh, thus allowing its use in the fit term of the energy function.

Li et al. (2009) extend Li et al. (2008) with a coarse-to-fine, data driven geometry and motion reconstruction algorithm for shapes that deform non-rigidly. A smooth template is created and fitted to each scan, with the aid of a deformation graph, while a detailed synthesis procedure adds high-frequency details. More materials are considered

in experiments and the characters range from faces, hands, humans to puppets and crumpled paper. The geometry and motion reconstruction are accurate, when the number of frames is large enough, so deformations are small or medium between frames.

Apart from feature points and deformation graphs, as the deformations become larger, more complex constraints seem to appear. Zheng et al. (2010), for example, propose a non-rigid space-time registration technique for point clouds, making use of the novel consensus skeleton. The main idea is of combining global, skeleton driven deformation with local non-rigid deformation in order to attain larger scale deformations. The quality of the match depends, however, on the initially extracted skeleton and also on the amount of overlap skeletons have in each frame.

Chang & Zwicker (2008) use piecewise rigid alignments in an algorithm that morphs between articulated shapes. This is done without prior knowledge, through sampling and clustering vertex motion. A labelling procedure is then adopted to mark similarly moving parts. This method works best with articulated models that undergo isometric deformations. Although it is an attractive method, given the fact that no template or other prior knowledge is used, the calculations are costly, expanding over an order of minutes.

Large

Sumner & Popović (2004) use deformation transfer to morph between differently shaped meshes. They use a set of manually placed landmarks to calculate a dense correspondence between a template in different poses and a target mesh. The meshes involved must have similar isometry and proportions in order for the algorithm to work as expected. The result also depends on the global position of the characters. Also, the algorithm involves having the template mesh in different poses. The deformation of the template can then be transferred to the target mesh. This information is not available in the current case, however, and to obtain it would imply rigging the template mesh.

Anguelov et al. (2004) define a joint probabilistic model for isometric non-rigid deformations. Their method makes use of Markov networks and geodesics to obtain correspondences between articulated models. The model also penalizes twisting and stretching of points and is mostly suitable for characters that are formed of a rigid material. Also, calculating geodesic distances is time consuming and sensitive to topology changes. Moreover, as Li et al. (2008) point out, the main requirement for the algorithm to work is that the data mesh is a subset of the model shape.

Anguelov et al. (2005) next propose to use an initial scan as a template for a data driven reconstruction and deformation system. Markers are used to track the non-rigid deformation of an actor through time. Deformation is split into rigid and non-rigid, which corresponds to skeleton driven deformation and muscle bulges. This algorithm, however, needs a large dataset of poses and is also aimed mostly at people.

Zell & Botsch (2013) present an algorithm for morphing between nonisometric, non-rigid face models. This is done through bringing the template and target shapes to a simpler domain after applying a fairing procedure on each surface. Their paper is the first example that presents a clay character, scanned with different expressions, registered using this procedure. Although the results presented are good, the algorithm fails to match body poses, being limited to heads. Similar examples of shape mapping algorithms are presented by Kim et al. (2011) and Yoshiyasu et al. (2014) who propose a blend of intrinsic maps and as-conformal-as-possible maps respectively, for large scale, nonisometric deformations.

Yang et al. (2015) present a state of the art method similar to Amberg et al. (2007), where the energy term allows sparse, larger deformations and does not penalize them. These deformations are usually at joints and their method works well for articulated characters. Also the coarse-to-fine strategy is used for obtaining robust results. The method is restricted, however, to piecewise smooth characters and can be slow with large meshes.

After seeing the literature tendencies regarding shape matching and non-rigid registration, the helpers used in these techniques are next discussed. As mentioned before, the larger the deformation is the more intrinsic prior knowledge is required for accurate registrations. It is worth noting that although helpers related to the character animation are necessary, helpers related to the nature of the material and structure of the characters might offer a deeper understanding of what is needed to create a universal registration method, at least in the Clay Jam universe.

5.1.2 Correspondence Problem

Feature points are one of the most popular choices when choosing correspondences and excel at matching meshes with small deformations (Allen et al. (2003), Amberg et al. (2007)), parametrization (Tevs et al. (2012)), embedding (Zell & Botsch (2013), Dey et al. (2015)), tracking (Anguelov et al. (2004)) and finding dense correspondences from sparse correspondences (Sumner & Popović (2004)). Finding correspondences

automatically or picking landmarks manually are one of the cornerstones of non-rigid registration. The issues landmarks introduce are related to the amount of stretching that can appear when correspondences are not placed appropriately.

Embeddings into smaller or larger domains are also used to turn a high-dimensional registration problem into a low dimensional one. Some examples include embedding intrinsic measures into extrinsic space (Elad & Kimmel (2003), Bronstein et al. (2007)), in the spectral domain (Jain et al. (2007), Dey et al. (2015)), in the implicit space (Cheng et al. (2010)). If the latter examples were for 3D or higher dimensional transformations, lower dimensional embeddings are also known as parametrizations. Tevs et al. (2012) present a method for shape reconstruction and matching inspired by cartography. Although these techniques simplify the dimension of the problem, they are bound to introduce inaccuracies in geometry due to the approximation of the metric space (Bronstein et al. (2010)).

Global pose changes can be easily obtained with the use of a skeleton (Zheng et al. (2010), Berger & Silva (2012)). This greatly simplifies the matching procedure as it makes use of mesh intrinsic information to align the models extrinsically. Skeleton extraction is also often seen in conjunction with segmentation (Pekelnny & Gotsman (2008)) and their common aim is the simplification of a large scale deformation into smaller manageable deformations. A good example is piecewise deformation, where rigidly deforming parts are identified by clustering them after a labelling step (Chang & Zwicker (2008), Tam et al. (2014)).

Although global descriptors simplify the registration process, care needs to be given to the influence such constraints can have on the mesh, as they may introduce additional artifacts (Zheng et al. (2010)). It is worth noting that a skinning technique will not have the same effect as deforming clay. Nevertheless, a global constraint could give a good initial pose, leaving the refinements for a later stage.

In order for shape descriptors to aid accurate registrations, a good semantic correspondence is desirable (Sumner & Popović (2004), Li et al. (2009), Kin-Chung Au et al. (2010)). Salient features not only have to be similar, like in shape retrieval algorithms (Funkhouser et al. (2003), Bronstein et al. (2007, 2011), Budd et al. (2013)), but they also must correspond topologically and geometrically. As Li et al. (2008) point out, wrongly placed correspondences between shapes can introduce geometric distortions, which renders them unreliable for larger deformations.

Yang et al. (2013) extend Sumner & Popović (2004) with an elasticity model that considers stretchiness of the model. This helps in reducing artefacts when deforming

meshes, but is not real time and uses a mass spring system which is not invertible. Another example of correcting inaccurate correspondences is the use of geodesics on shapes that deform isometrically (Huang et al. (2008)). Geodesics are however sensitive to topological changes, like holes, which is why more modern methods use diffusion distances. The latter calculate an average of distances between two points, rather than choose the shortest path (Bronstein et al. (2010)). Like all calculations on smooth manifolds, however, they are time consuming and can be inaccurate when approximated on a coarser mesh.

All in all, the tendency of adopting helpers in the non-rigid registration and matching literature seems to revolve around isometric and nonisometric deformations, with many examples of the former and considerably fewer of the latter. Also, depending on the scale of deformation, helpers seem to be needed more in larger deformations. They usually reveal intrinsic details about the mesh (eg. skeleton, geodesic, curvature) and play a major role in reducing the degrees of freedom a larger deformation may have. Extrinsic techniques are commonly used with as-rigid-as-possible deformations, while intrinsic techniques are popular with non-rigid deformations (Anguelov et al. (2004), Bronstein et al. (2007), Li & Iyengar (2014)).

Considering that Clay Jam characters have a variety of animation types, the challenge becomes to reduce the degrees of freedom of each animation by just the right amount of helpers. The effort must be minimal for the artist, which is why a minimal set of necessary helpers must be determined for each type of animation. The artist can choose these constraints manually or they can be calculated automatically prior or during the registration. For this work the *Animation by Shape* class was mostly considered, together with its relevant helpers, to guide the registration of a few Clay Jam characters.

5.1.3 Trajectory Problem

As stated in Xu et al. (2005), even if correct correspondences are known, the deformation trajectory can distort the model if not chosen appropriately. When all the correspondences are known, the trajectory problem becomes an interpolation problem. Interpolation has been discussed in section 3.2.7.

When one-to-one correspondences are unreliable, a regularization method needs to ensure that the deformation path is smooth and intuitive. Depending on what surface properties need preserving (eg. angles, lengths) the regularization can be as-rigid-as-possible, as-conformal-as-possible etc. The strength of the constraints can be controlled

by a weight, which also acts as a stiffness of the surface. Regularization weights allow the surface points to move more towards the target points when weights are low. When weights are high, the surface is resistant to deformation.

In the case of the E-StopMotion steps (chapters 3, 4), a volume preserving regularization was required to deform and interpolate plasticine scans. Local stiffness also permitted to have fixed and flexible regions on the source mesh. The resulting shapes can be used to initialize the following *Non-rigid Registration* step of the pipeline.

5.2 E-StopMotion: Non-rigid Registration Component

Non-rigid registration has made great progress in recent years, taking more steps towards matching characters that have undergone non-isometric deformations. The state-of-the-art is, however, is still linked more to elastic or locally shape preserving deformations, leaving room for improvement in the plastic deformation area. When the local and global shape of a character changes significantly from pose to pose, methods that rely on shape analysis or proximity measures fail to give satisfying results. The argument of this thesis is that by using information about the material the models are made from and the general deformation procedure, non-isometric registrations can be improved. Hence, by addressing mainly plasticine characters, the attempt is to reverse engineer the deformations they undergo in the hands of an artist.

The proposed technique combines physical properties (stiffness, volume) with surface geometry properties (geodesics, stretch, bend) to convey a realistic registration. The nonlinear deformation algorithm from chapter 4 is used to better initialize the non-rigid registration technique described by Amberg et al. (2007). The proposed approach can overcome some of the limitations of the original non-rigid registration method (Amberg et al. (2007)) by using material and structure based priors. An analysis is done on how enhancements like dense correspondences and segmentation can aid the registration. Moreover, physically plausible intermediate poses are obtained from the deformation and a subsequent interpolation step, as it was shown in chapter 3. These can be used to enhance the original animation and help the artist simplify their work.

5.2.1 Method

Given a source triangular mesh M_S and a target triangular mesh M_T , the aim of this method is to register the source as closely as possible to the target. The context of

the given problem is still within scans of real world plasticine models. For simplicity, both the source and the target meshes were cleaned using OpenFlipper (Visual Computing Institute (2018)), an open source software, to obtain isotropic triangular meshes.

After examining the non-rigid registration methods available in literature, it was concluded that a flexible transformation model, with affine matrices per vertex, and a large number of correspondences is the best combination to reach the aim. The necessity for affine transformations per vertex comes from the quality of plasticine to deform in an unconstrained manner. For the second part of the conclusion, however, the priority of simplifying the artist’s task needs to be taken into account. In the current implementation the artist is asked to clean the source and target scans and pick up to 15 landmarks. These steps could be automated in future implementations.

Amberg et al. (2007)’s non-rigid iterative closest point (NRICP) algorithm is next described. This was found to work well for small to medium deformations, but failed for large deformations. A few enhancements are then explored, including the nonlinear deformation algorithm proposed in chapter 4. These enhancements are linked to the material and structural helpers of the model. The contribution of this chapter was to show that non-rigid registration can be improved by using such methods and they can aid in obtaining a more generalized registration method.

Non-rigid Iterative Closest Point

The *Non-rigid Registration* step of the pipeline (chapter 3) is adapted from Amberg et al. (2007). The algorithm is based on an iterative procedure similar to the rigid iterative closest point (Besl & McKay (1992)). It also uses the closest point correspondences to match a source mesh M_S as closely as possible to a target mesh, M_T . The difference is that non-rigid registration employs local affine transformations instead of a global rigid-body motion.

Due to the high number of degrees of freedom given by the vertex affine transformations, a smoothness regularization term is involved to make the problem solvable. Each iteration of the registration is solved by finding the optimum affine transformation per vertex, which minimizes the energy functional described in equation (5.1).

A decreasing global stiffness is further used to constrain the amount of movement each vertex can undergo. The closer the source shape gets to the target shape, the higher the probability that the chosen correspondences are accurate. To prevent wrong correspondences from influencing the match, the stiffness starts off as high and decreases

as the distance between the source and target meshes becomes smaller. An example of non-rigid registration is shown in figure 5-1 and the mathematical formulation and registration procedure are presented in the following paragraphs.

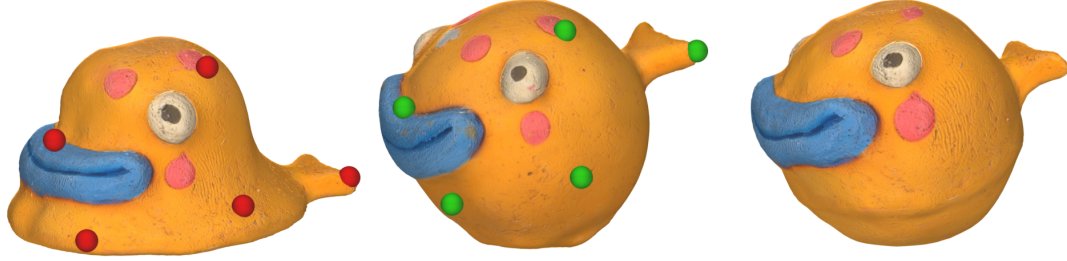


Figure 5-1: Blob Fish source (left) and target (middle) meshes with landmarks and the registered mesh (right).

Energy Minimization in a Single Iteration

$$E(X) = E_d(X) + \alpha E_s(X) + \beta E_l(X) \quad (5.1)$$

$X = [X_1, \dots, X_N]^T$ is the transformation matrix that minimizes the total energy $E(X)$. It contains N 4×4 affine matrices X_i , one per source vertex. E_d is the energy term that measures the sum of squared distances between correspondences, E_s is the smoothness regularization term between neighbouring vertices, and E_l is the landmark energy term which allows the user to pick a few corresponding landmark points to guide the registration. The energy terms are defined as

$$E_d(X) = \sum_{v_i \in M_S, u_i \in M_T} w_i \|X_i v_i - u_i\|^2 \quad (5.2)$$

$$E_s(X) = \sum_{(i,j) \in \mathcal{E}} \|(X_i - X_j) G\|^2 \quad (5.3)$$

$$E_l(X) = \sum_{(v_i, l_i) \in \mathcal{L}} \|X_i v_i - l_i\|^2 \quad (5.4)$$

where the weights w_i from (5.2) are 0 or 1, signifying whether the correspondence pair (v_i, u_i) is reliable or not. A correspondence pair may be filtered out as unreliable when

the angle between their respective normals is greater than 90° .

The smoothness of the transformation between reliable correspondences is ensured by the term in (5.3). The idea is to minimize the differences in transformation between neighbouring vertices. The pair (i, j) represents an edge between vertices v_i and v_j , while \mathcal{E} is the list of edges of the source mesh M_S . Also, matrix G from (5.3) is a diagonal matrix of the form $G = \text{diag}\{1, 1, 1, \gamma\}$, which can influence the scale and rotation contribution of the regularization through the term γ .

The pair $(v_i, l_i) \in \mathcal{L}$ represent a landmark correspondence pair between the source and target meshes and is considered separately from the closest point correspondences. These pairs of vertices are deemed to be essential in guiding the deformation and are usually weighted higher than the global stiffness at the beginning ($\beta > \alpha$). After a certain threshold of α , it is considered that the landmark influence is no longer necessary ($\beta = 0$). This is due to the implication of the source mesh being close enough to the target mesh, which results in enough reliable correspondences.

Landmarks can also be seen as helpers that guide the deformation. They work best when chosen in regions with high displacement between the source and the target meshes. It was found that 10-15 landmarks, distributed noncoplanarly across the surface, are sufficient for guiding the deformation and avoiding artefacts. Coplanar landmarks produce flattening effects and too few or too many landmarks may cause the solution to get stuck in local minima.

Matrix Notation

In practice, to facilitate implementation, it is more convenient to rewrite the energy functional from equation (5.1) using matrix notation. For a full description of the matrix notation please refer to the work of Amberg et al. (2007). The following representation of the energy function in matrix notation is the final form used in the proposed implementation. Note that $\|\cdot\|_F$ indicates the Frobenius norm.

$$\bar{E}(X) = \left\| \begin{bmatrix} \alpha M \otimes G \\ WD \\ \beta D_{\mathcal{L}} \end{bmatrix} X - \begin{bmatrix} 0 \\ WU \\ U_{\mathcal{L}} \end{bmatrix} \right\|_F^2 \quad (5.5)$$

Matrix M is the directed node-arc matrix, representing whether there is an edge between two vertices, α is the global stiffness, that decreases at every outer iteration (algorithm 2). The direction of the respective edge is indicated by ± 1 . $W = \{w_1, \dots, w_N\}$

contains the weights applied to the source to target correspondences stored in matrices D and U respectively. Similarly, $D_{\mathcal{L}}$ and $U_{\mathcal{L}}$ contain the landmark correspondences, weighed by β .

Iterative Registration Procedure

As described in algorithm 2, two loops are involved in an iterative process to incrementally align the source shape to the target. The outer loop decreases the global stiffness for each iteration, allowing the current mesh more freedom of movement. The inner loop corresponding to each iteration has two steps. The first step is to find correspondences between the source and the target meshes. Correspondences are obtained for each source vertex by searching for the closest target vertex. Correspondences are dropped if the line between them intersects the template mesh, or if the angle between the respective normals exceeds an angle threshold. The case in which correspondences are found on a boundary are not considered, although mentioned in the original algorithm.

The second step is to calculate the transformation matrix X that would bring the current source mesh closer to the desired target mesh. This loop finishes when the differences between two consecutive transformation matrices is lower than a threshold ε .

Algorithm 2 Non-rigid Registration

Input: Clean source mesh M_S and clean target mesh M_T with a small set of landmarks \mathcal{L} between them.

$\alpha = MAX_STIFFNESS$

Reduction value $rv = [10, 4, 0.5, 0.1]$ for $\alpha > [10, 2, 0.5, 0]$ respectively.

Output: The source mesh registered to the target mesh, $M_{S \rightarrow T}$.

Align M_S and M_T globally using a Procrustes analysis.

Initialize the transformation matrix X_0 .

while $\alpha > MIN_STIFFNESS$ **do**

while $\|X_i - X_{i-1}\| < \varepsilon$ **do**

 Find correspondences u_i between current source mesh M'_S and target mesh M_T .

 Calculate X_i as the solution to a linear system of the form $AX = B$, where A is the matrix described in the previous section.

end

$\alpha = \alpha - rv$

end

Implementation

The first version of the non-rigid iterative closest point (NRICP) algorithm first de-

scribed by Amberg et al. (2007) was implemented using OpenGL4.4 and C++ in Qt 5.3.0. Libraries such as GLFW3 for the window system, GLEW for OpenGL versioning, Eigen for sparse matrix operations were used. The Matrix class is a header based library, written by Anton Gerdelen and can be replaced with the Eigen matrix class, for consistency, in the future.

The `GLFWContainer` class initializes the window and OpenGL rendering environment, loads the source and target meshes, their shaders and an instance of the NRICP algorithm class. `Mesh` class describes the properties of a mesh and contains vertices, normals, texture coordinates, vertex indices for representing faces and distortion measurements. A k-D Tree partitioning algorithm is employed to reduce the nearest neighbour time from $O(N^2)$ to $O(\log N)$, where N is the number of vertices.

Algorithm 2 is implemented in the NRICP class. A `NRICP_Segment` model was also added to the C++ implementation after the first version of the program was completed. It stored a source and a target segment as private data, similarly to how NRICP stores the full source and target meshes. Early experiments can be seen in section **Enhancement 4: Part Based Deformation**, where two segmentation methods are discussed.

A few numerical inconsistencies that resulted in the source mesh presenting spikes in the landmark regions compelled the author to resort to a more reliable Matlab implementation (see acknowledgements in section 6.1). This implementation was modified to suit a few enhancements which were experimented with in the current work. They are presented in the following sections.

The Need for Enhancements

Non-rigid registration as proposed by Amberg et al. (2007) has the advantages that it requires a linear, global minimization procedure. Also, the use of affine matrices per vertex and a global stiffness that decreases per iteration gives the mesh enough flexibility to deform similarly to clay. On the downside, the global aspect of the stiffness propagates the deformation error throughout the whole model, smoothing out the mesh as a result. Regions which should not displace at all, move in the same manner as regions which have large displacements.

Another drawback of the NRICP is its tendency to fall into local minima, giving sub-optimal transformation solutions. This is due to the dividend matrix from the $AX = B$ equation that emerges from (5.5) not being positive definite, thus corresponding to a non-convex energy. One reason for this is poor correspondences in the initialization stage and subsequently throughout the iterative process. Since the closest point pro-

cedure fails to provide good correspondences in the case of large deformations (Tam et al. (2013)), helpers need to be adopted to guide the process. Handpicked landmarks can only go so far, as they are tedious to pick and introduce small errors, as it is virtually impossible to have a one-to-one mapping of vertices on triangular meshes. Nevertheless, they guide the overall deformation to begin with and can be a sparse set of correspondences which can guide a dense, more reliable set of correspondences.

Rather than disregard the NRICP, the approach was to address some of its issues with geometric or physics based enhancements. The flexibility of the method and the ease of implementation favour NRICP as a good option for registering plasticine character scans. The aim was to show that by using information about the structure of the models and the material they are made from, registrations can be improved. Physically plausible in-between poses are also an outcome of the proposed method. These can be used together with the interpolated poses proposed in chapter 3 to enhance the original stop motion animation for game and film applications.

Enhancement 1: Virtual Clay Deformation

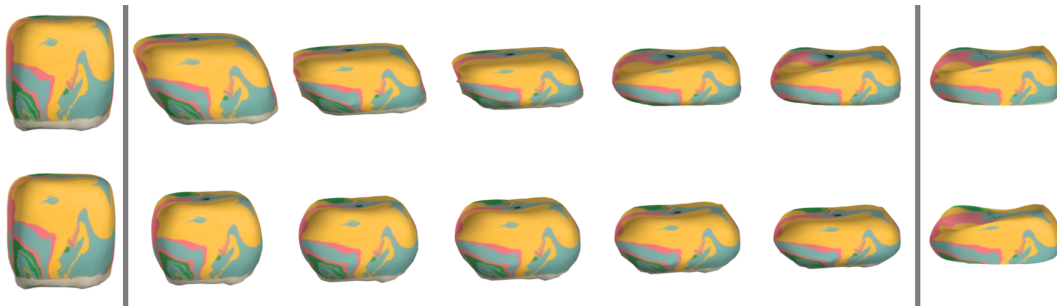


Figure 5-2: Results with original non-rigid registration algorithm (top) versus the proposed method, local stiffness thin shell deformation + non-rigid registration (bottom). Notice that the in-between poses look more plausible with the proposed method. The start and end shapes are the source (left) and target (right) meshes.

Chapter 4 showed how to represent virtual clay using a surface based, nonlinear deformation method. Although slow, it provides a robust set of shapes with different settings for the local or global stiffnesses. Since non-rigid registration is notorious for its propensity to fall in local minima (Li (2010)), the proposed deformation technique was added as an initialization step for improving the registration.

Figure 5-2 shows a comparison between the original NRICP (Amberg et al. (2007)) and the proposed method. The registration was done on the simple model from figure 5-3, which undergoes an uneven compression. Notice how the proposed method generates



Figure 5-3: Source shape (left) and target shape (right) for the registration in figure 5-2

more physically plausible in-between poses than the original NRICP.

Figure 5-4 illustrates the steps of the implemented method, which are described in more detail in algorithm 3. The main idea is to first deform the source mesh using only the target landmarks as end points for the source landmarks. Once a certain threshold has been reached, or the number of iterations exceeds a certain number, the deformation stops and the NRICP registration begins. Experiments showed that by initializing the registration with a deformed version of the source mesh, better registrations are obtained, together with more reliable in-between shapes (figure 5-2).

Since the deformation algorithm is nonlinear and the registration is linear, it would have been difficult to combine them in a single energy function. Nevertheless, the next attempt was to apply local stiffness directly to the registration.

Enhancement 2: Localized Stiffness

When registration algorithms evaluate geometry globally, approximation usually takes place and details tend to be smoothed out. It is much harder to preserve details when all of the geometry is considered with equal weights for the optimization. Larger features tend to dominate the results similarly to how outliers distort average values in a data set. Applying a localized stiffness to the registration process should allow a weighting of geometry influence per region, with some regions being stiffer than others.

Using Stiffness in the Correspondence Problem

Local high stiffness signifies that the vertices in the respective region move less than the vertices in low stiffness regions. Therefore it is safe to assume that high stiffness vertices are more reliable than low stiffness ones. Stiffness values can thus be used to

Algorithm 3 E-StopMotion: Non-rigid Registration Component Algorithm with Virtual Clay Deformation Enhancement

Data: Source and target shapes, a small set of landmarks**Result:** Aligned source shape, correspondences and a set of in-between poses

```
/* Close enough node in Figure 5-4 */
while not closeEnough do
    Calculate vertexStiffnessMap
    Calculate stretchStiffnessMap
    Calculate bendStiffnessMap

    while differenceInDeformation > deformationThreshold do
        | Deform mesh step
    end
    Calculate hausdorffDistance
    Save in-between pose
    if hausdorffDistance < hausdorffDistanceThreshold then
        | closeEnough = true
    end
end
/* Really close node in Figure 5-4 */
while globalStiffness > stiffnessThreshold do
    while distanceInTransform > transformationThreshold do
        | Find correspondences using closest point
        | Filter correspondences
        | Calculate transformation
    end
    Save in-between pose
end
```

filter out erroneous correspondences in the registration procedure. The disadvantage of this method is, however, that reliable correspondences may be filtered out when the stiffness map isn't calculated appropriately. The results were inconclusive with this method.

Using Stiffness in the Trajectory Problem

From the early formulations of local stiffness (section 4.4.1), an interesting property of the energy function became apparent. When using either type of local stiffness during the registration, the energy decreases monotonically (figures 5-5 and 5-6). Although this property does not always imply a good registration (Amberg et al. (2007)), it was a confirmation that local stiffness can decrease the overall error propagation during the registration better than global stiffness. Error from a region does not propagate

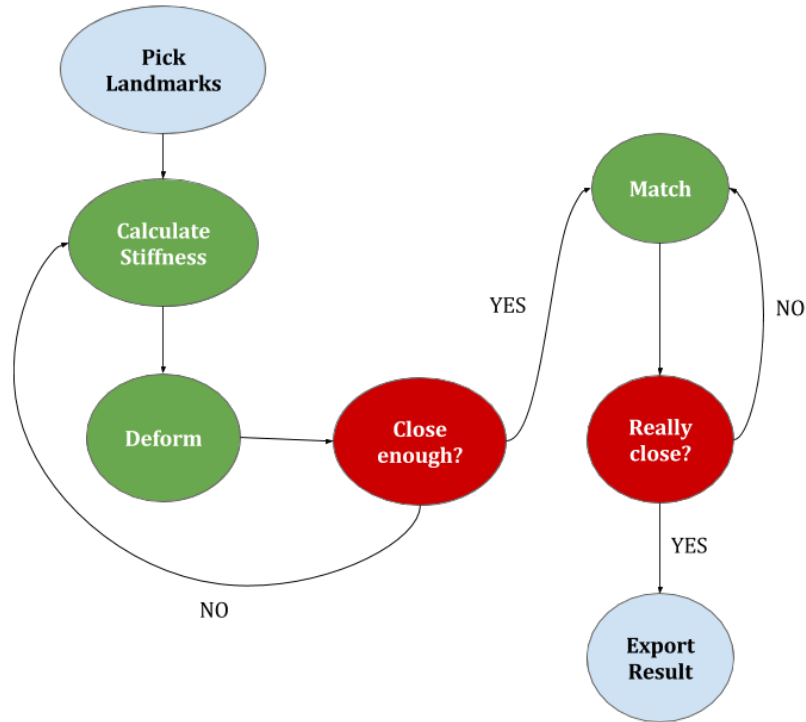


Figure 5-4: Diagram illustrating the steps of the algorithm after adding Enhancement 1: Virtual Clay Deformation.

globally when using local stiffness, but rather affects vertices locally.

The main idea for the trajectory problem was to allow detail preservation in stiff areas, while the rest of the shape deforms as required. The global stiffness α from Amberg et al. (2007) was replaced by a vector of stiffnesses, with each value corresponding to a source mesh vertex. The stiffness map was calculated as presented in section 4.4.1. This method was attempted on a few simple models with little success. Regardless of the type of local stiffness map used in the registration, instabilities appeared often during the tests.

Figure 5-7 shows an example of a simple model being matched to a stretched version of itself. The stiffness map can be seen to the right of the image. Notice that a region of high stiffness prevents the vertices from displacing in that region, creating artefacts. The smoothness regularization method contained by the NRICP is not sufficient to completely prevent these artefacts from occurring in such scenarios. This is due to the linearity of the registration method and also to the difficulty of generating an intuitive stiffness map from a sparse set of landmarks.

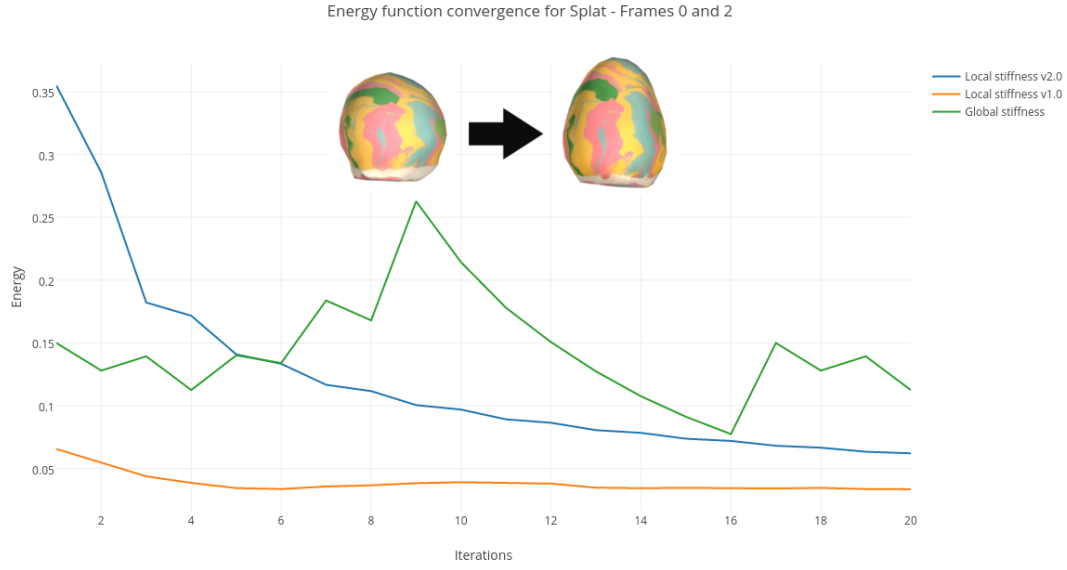


Figure 5-5: Energy graph for the NRICP between source and target 2 from figure 5-8, by using global (green) and version 1 (orange) and 2 (blue) of the local stiffness presented in section 4.4.1).

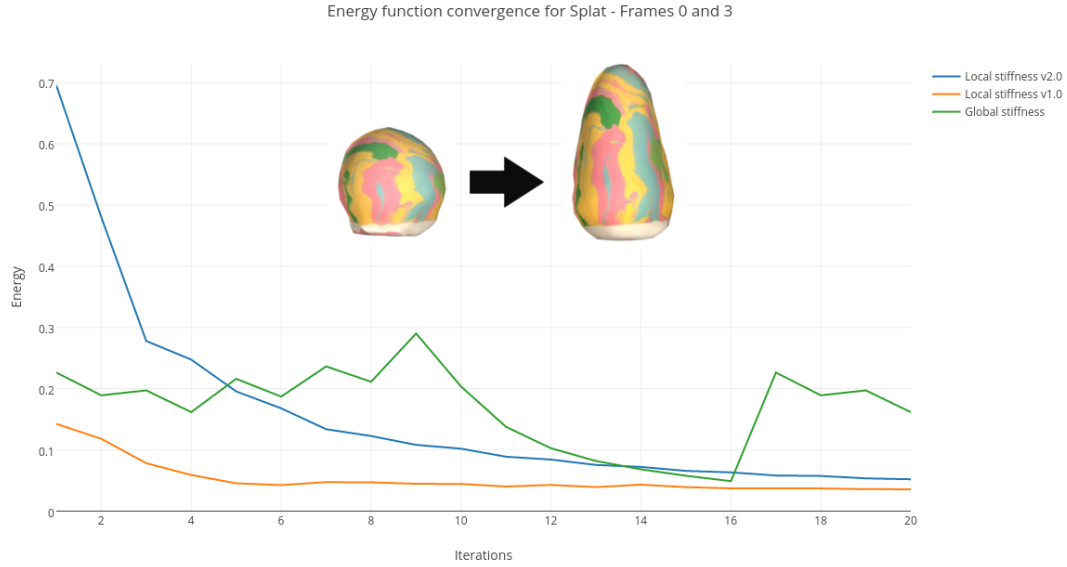


Figure 5-6: Energy graph for the NRICP between source and target 3 from figure 5-8, by using global (green) and version 1 (orange) and 2 (blue) of the local stiffness presented in section 4.4.1).

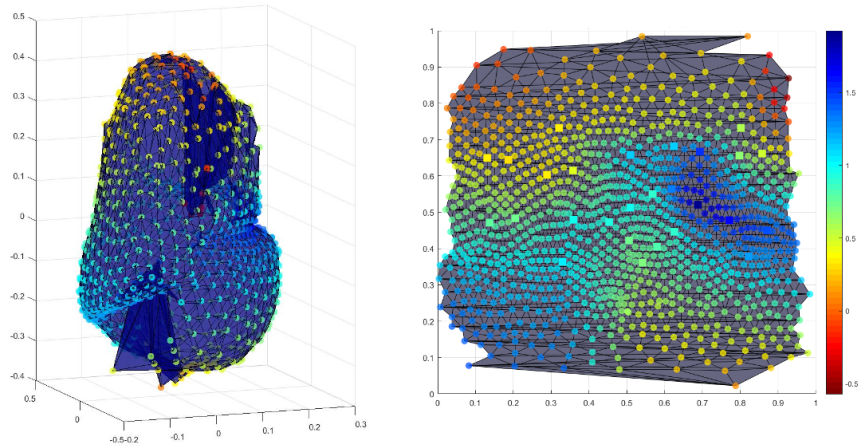


Figure 5-7: Early work on adding local stiffness directly to the NRICP algorithm. The source shape was a spherical chunk of clay, while the target shape is a stretched out version of it, along the Y axis.

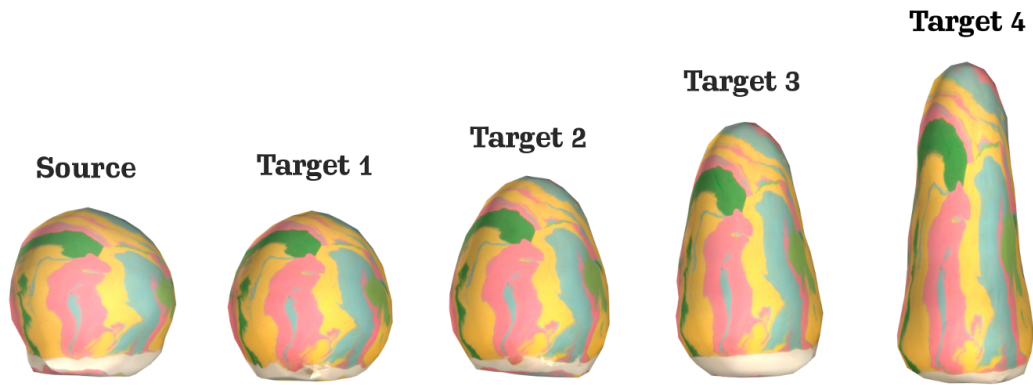


Figure 5-8: Source mesh and four potential target meshes for a simple plasticine model. Notice that each target shape presents a higher level of deformation with respect to the source shape.

Enhancement 3: Dense Correspondences

Two approaches were tested to obtain a dense correspondence map. Firstly, texture information was used to detect salient features using Sift (Lowe (1999)) and Optical Flow (Horn & Schunk (1981)). This was done on the UV texture maps of the source and target meshes for a couple of simple plasticine models like the ones in figure 5-8. Due to the large differences in the UV planar projections, it was hard to establish reliable correspondences using these techniques.

Secondly, experiments were done on the surface by adopting the weighted average algorithm proposed by Panozzo et al. (2013). This allowed dense correspondences to be generated on the surface from a sparse set of handpicked landmarks. Their method calculates the generalized barycentric coordinates of a point inside a patch delimited by landmarks. Once the coordinates for a point on the source mesh are calculated, the corresponding point can be found on the target mesh.

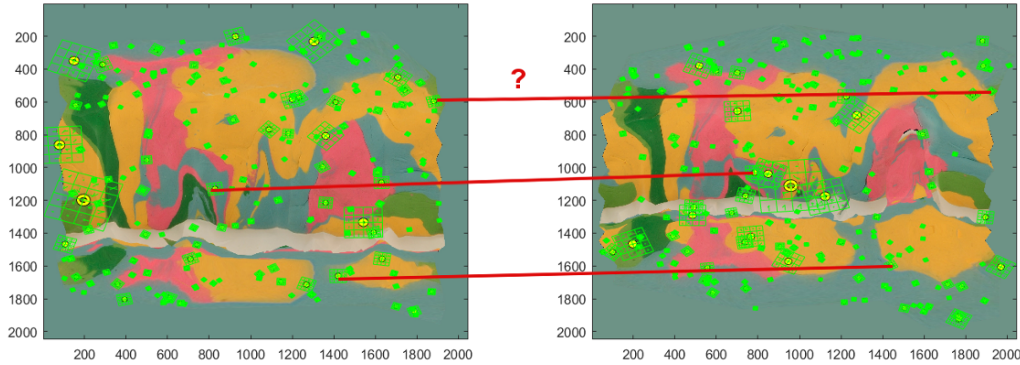


Figure 5-9: Sift features on source and target shapes for a simple plasticine model. Notice that it is hard to establish reliable correspondences between the two UV maps, due to the projections having significant differences.

Texture Landmarks

The initial experiments were done with vividly coloured plasticine models, which display salient features in the texture. A Sift salient feature detector was applied to a source and a target UV texture map. Figure 5-9 shows one of the results for a simple source and a target shape. Due to the distortion specific to planar projections and the sampling rate, it was difficult to obtain reliable 2D correspondences between the two texture maps.

The next strategy was to use optical flow to transform a source UV texture map into the target map. The resulting one-to-one correspondences between the two textures could then be used to identify the vertex correspondences in 3D space. A set of 50 correspondences were sampled from the template texture to test the method.

Even when the warped image is very close to the desired output (figure 5-10), good correspondences are not guaranteed. This is due to the fact that the connectivity of the source and target meshes are different, thus a vertex on the template mesh might correspond to a point on an edge in the target mesh. The target vertex chosen will be the closest possible point to the real correspondence found. If this is done for all



Figure 5-10: Source UV texture map (left), target UV map (middle) of source and target 2 models from figure 5-8 and the image generated by applying an optical flow from the source to the target (right).

correspondences, the global error increases significantly. The error propagation makes it difficult to obtain good matches.

Figure 5-11 shows the result of a NRICP match using the 50 landmarks sampled from the optical flow correspondences. Notice that the larger the deformation between the source and the target meshes, the more unreliable the correspondences and hence the registration. The errors introduced by the 2D embedding of the shape into UV space, together with the inaccurate salient feature detection make this method unsuitable.

The errors emerging from embedding a curved shape into a planar space, are also remarked by Bronstein et al. (2008) who state that it is impossible for the two embeddings to be isometric, due to the differences in Gaussian curvature. As a consequence, the next methods to be investigated were for finding dense correspondences on the mesh, rather than relying on an intermediate parametric space.

Weighted Averages on Surfaces

Panozzo et al. (2013) calculate the generalized weighted averages of points on a surface through the Fréchet mean. The forward problem in \mathbb{R}^k is to find the position of the centroid $\hat{\mathbf{p}}$ by calculating the weighted sum of the n neighbour points p_i ,

$$\hat{\mathbf{p}} = \sum_{i=1}^n w_i p_i \quad (5.6)$$

where $\sum_{i=1}^n w_i = 1$.

Let the surface S be a metric space, on which n landmark points l_i have been chosen,

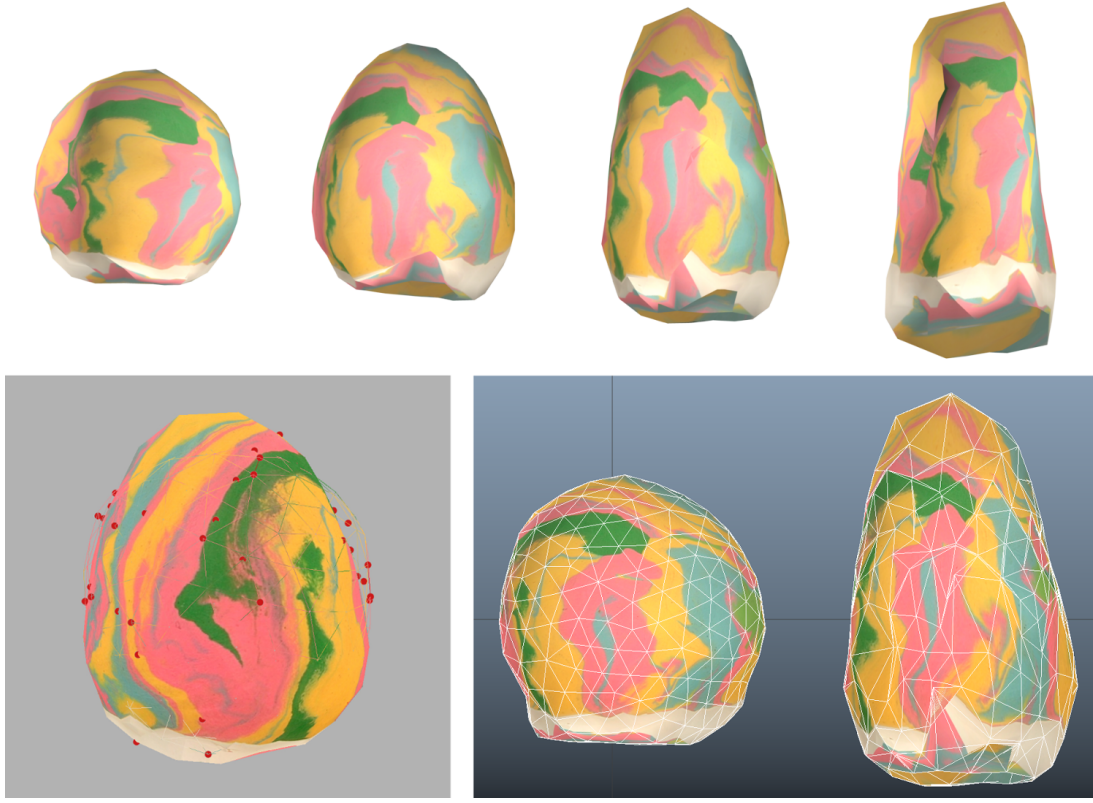


Figure 5-11: The top of the image shows the morphed source shape in an attempt to match the target shapes from figure 5-8. The match was done with NRICP and 50 landmarks generated from the UV texture maps (bottom left) using optical flow. Notice that the triangle distortion of the resulting shapes is high (bottom right), making it an unsuitable method.

together with their corresponding weights w_i . The generalized weighted average $\hat{\mathbf{p}}$ is found as the minimizer of the Fréchet mean

$$\hat{\mathbf{p}} = \operatorname{argmin}_{p \in S} F(p)$$

$$F(p) = \sum_{i=1}^n w_i d(p, l_i)^2 \quad (5.7)$$

where $p \in S$ and d is the distance between two points on the surface. Usually d represents the geodesic distance, but Panozzo et al. (2013) remark its sensitivity to noise and the fact that it is not C^1 continuous when represented on a discrete surface such as a mesh. When d is smooth, the gradient $\nabla F(\hat{\mathbf{p}}) = 0$ and the minimum solution $\hat{\mathbf{p}}$ for the Fréchet mean can be easily found.

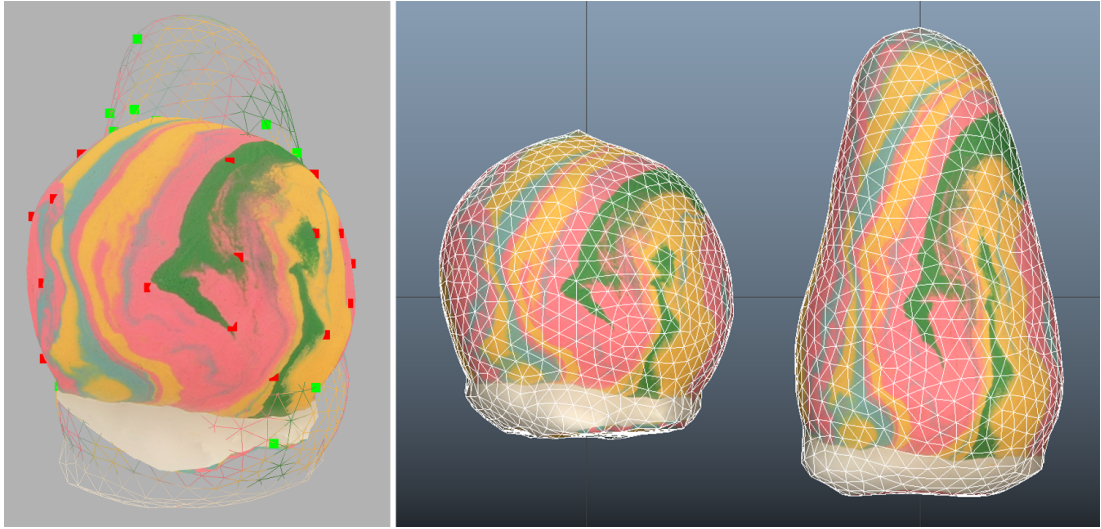


Figure 5-12: Source and target for simple blob mesh (right) and the 27 landmarks between them (left).

In order to ensure smoothness, Panozzo et al. (2013) propose embedding the surface in 8-dimensional Euclidean space. The distances between points are faster to calculate, but only approximate the geodesics. In order for the Fréchet mean to be well defined, landmark points need to be in close proximity, the Gaussian curvature of the region delimited by the landmarks needs to be kept to a minimum and weights need to be positive ($w_i > 0$).

The inverse problem, as described by Panozzo et al. (2013), is finding the weights w_i of the chosen landmarks l_i , given the centroid $\hat{\mathbf{p}}$. In Euclidean space, this problem has been studied under the name of generalized barycentric coordinates (Rustamov (2010)). Panozzo et al. (2013) solve the inverse problem by maintaining locality, interpolation and smoothness, as well as adding a quadratic minimization function for penalizing the influence of far away landmark points. More details can be found in their paper, as it is not the subject of this thesis.

The Matlab implementation of the weighted averages was used, as described by Panozzo et al. (2013), and included it to replace the NRICP correspondence step. Given a sparse set of landmarks on a source mesh, the inverse problem was used to calculate the combination of landmark weights that defines the position of each source vertex. The forward problem was then used on the target mesh to find the location of each corresponding target vertex. Given the corresponding landmarks on the target mesh and the weights calculated for each source vertex, the corresponding target points can

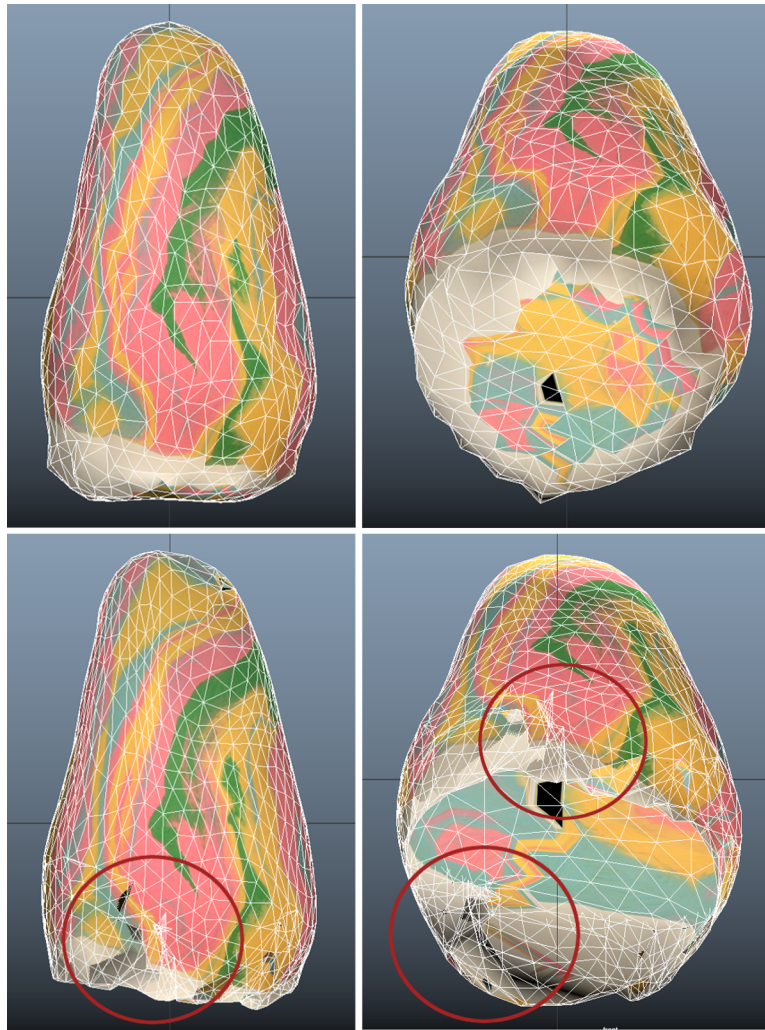


Figure 5-13: The top and bottom of a chunk of plasticine after a stretch motion. NRICP was used, with correspondences calculated using closest point (left) or weighted averages (right).

be established easily. Matlab's nearest neighbour procedure (`knnsearch`) was finally added to determine the closest vertex for each point calculated on the target mesh.

Figures 5-13 and 5-14 show the results for two simple model scans. The first one shows a compression, while the second shows a stretch. Notice that the result for weighted averages correspondences renders more uniform triangles than the original closest point method. The number of landmarks used was sparse, 24 for the compression example and 27 for the stretch example (figure 5-12).

The parts underneath the model were not covered by landmarks. This caused extrap-

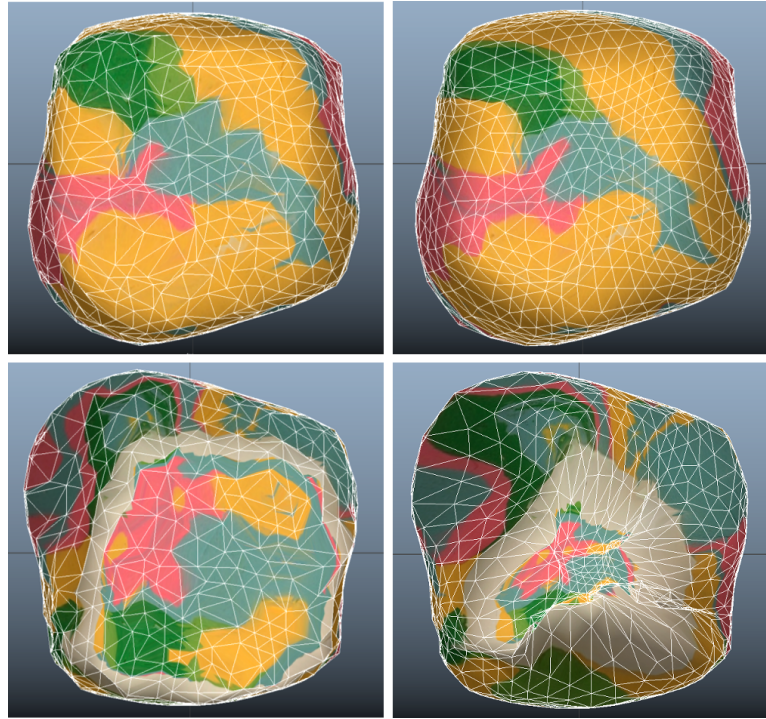


Figure 5-14: The top and bottom of a chunk of plasticine after using closest point correspondences with NRICP (left) and after using weighted averages correspondences with NRICP (right).

olation that distorts triangles to the point of triangle intersection. Due to landmarks being too far from particular vertices and the curvature of the surface between landmarks being too high, negative weights emerged. Panozzo et al. (2013) mention that they fix this issue by asking the user to place more landmarks on the surface. Since the artist's effort needed to be minimized, this method, although promising, was left for future experimentation.

Enhancement 4: Part Based Deformation

The enhancements presented so far are ideal for the *Animation by Shape* character class (figure 2-6). This is due to their reliance on material properties such as stiffness and volume preservation. Landmarks can be seen as application points for the artist's fingers, freely pushing and pulling the material. When limbs make an appearance, however, issues may occur in the NRICP due to self intersections (figure 3-16).

Animation by Skeleton characters need skeleton helpers to increase the changes of a good registration. Segmentation is also an intuitive extension to this idea, since it partitions the character into regions with high geometric or motion based similarities.

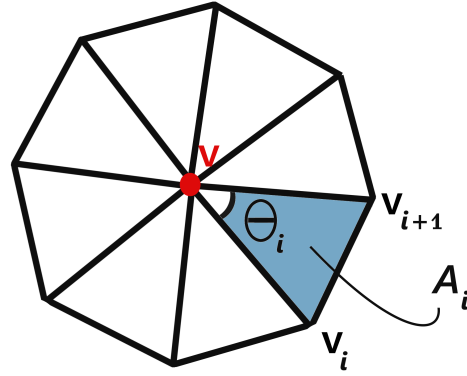


Figure 5-15: Triangle fan around current vertex v , surrounded by neighbours v_i . Notice the vertex angle Θ_i and corresponding triangle area A_i .

Segmentation

In the incipient stages of the project, the desire to speed up the registration process was attempted through rough geometric segmentation. This was to be a preliminary step before advancing to motion based segmentation. For the sake of timeline consistency, the two simple segmentation methods used are mentioned here.

A Gaussian curvature segmentation was first implemented, following the algorithm proposed by Zhang et al. (2002). This consists of three steps: Gaussian curvature estimation, boundary detection and region growing. Firstly the Gauss curvature for each vertex is approximated by the equation

$$\mathbb{K}(v) = \frac{3(2\pi - \sum_{i=1}^N \Theta_i)}{\sum_{i=1}^N A_i} \quad (5.8)$$

where N is the total number of neighbours, Θ_i is the $\angle v_i v v_{i+1}$ angle, v is the current vertex for which the curvature is calculated, v_i and v_{i+1} are its neighbour vertices and A_i is the area of the corresponding triangle (figure 5-15). Since Gauss curvature is an intrinsic property of a surface (do Carmo (1976)) it is, in theory, representation invariant and depends on the lengths and angles of the surface.

Secondly, boundaries of highly negative curvature are labeled as such if they are below a certain threshold. The remaining vertices are labeled as seeds. Lastly, the seed vertices form regions or segments by growing their influence iteratively, until a boundary is reached.

Another quick segmentation method tried was a planar split. The user picks three vertices to form a segmentation plane. Everything above that plane creates one segment, while everything below the plane creates another segment. Similarly to the Gaussian curvature method, this algorithm was implemented for simplicity and to experiment with registration time differences. Figure 5-19 shows two examples of these segmentation methods on a low resolution character mesh.

Skeletons

Segmentation of a model is linked to the piecewise rigid skeleton that approximates the shape. The only use of this procedure, however, was to deform one of the character source meshes to roughly align to the corresponding target mesh (figure 3-19). Since skeletons can provide large scale deformations, it would be useful to use them for better initialization of the registration procedure in the future.

5.2.2 Experiments and Results

A few promising experiments are next presented, following methods from the previous section. To reiterate the aim, using information about the material and the structure of the models can lead to better registrations. The material information comes in the form of local stiffness and volume preservation, while the structure of the model comes in the form of segmentation.

Experiments on Enhancement 1: Virtual Clay Deformation

One of the enhancements proposed for the original NRICP algorithm was to initialize it with a deformed version of the source mesh. Tests were done for both the global stiffness ($\lambda = 10$, $\mu = 10$, $\nu = 1000$) and the local IDW stiffness for the nonlinear deformation method described in chapter 4. The bold values in the tables below signify the smallest, and thus the preferred, results, with respect to the tables they are compared against.

The enhanced registration technique was compared to the original NRICP (Amberg et al. (2007)) for four Clay Jam characters scans. The average area and volume distortions for the original NRICP method (table 5.1) seemed to be similar to the proposed methods (tables 5.2 and 5.3). The distortions were calculated considering all the faces of a model. The average area and volume distortions were taken relative to their initial values. When analyzing the Hausdorff distances, however, the NRICP gives more accurate results when initialized by a deformed model.

Tables 5.4, 5.5 and 5.6 present the Hausdorff distances for the registered models after

using the original Amberg et al. (2007) method, after adding a localized IDW stiffness and a global stiffness deformation initialization step respectively. These results show that more accurate registrations can be attained when a deformed model initializes the registration.

Table 5.1: Average relative face area and volume distortions after registering the source mesh to the target mesh, with the original NRICP method. Negative values signify the mesh simplexes are shrinking, while positive values mean it is expanding.

Character	Faces	Lmrks	Face Area Distortion	Face Area σ	Face Volume Distortion	Volume σ
Blob Fish	3482	12	-0.0455	0.3449	0.0299	0.7566
Party Onion	3600	13	-0.0034	0.1811	0.0258	0.3578
Tick	2750	13	-0.0706	0.2560	-0.0707	0.5554
Hellidropter	1454	10	-0.0427	0.1019	-0.0732	0.1960

Table 5.2: Average relative face area and volume distortions after registering the source mesh to the target mesh, with a local stiffness deformation (inverse distance weights, propagated globally, exponent of 2, landmark factor = 1) + NRICP method. Negative values signify the mesh simplexes are shrinking, while positive values mean it is expanding.

Character	Faces	Lmrks	Face Area Distortion	Face Area σ	Face Volume Distortion	Volume σ
Blob Fish	3482	12	-0.0415	0.2697	-0.0086	0.5629
Party Onion	3600	13	-0.0052	0.2689	0.0619	0.6048
Tick	2750	13	-0.0725	0.3153	-0.0403	0.6234
Hellidropter	1454	10	-0.0345	0.1377	-0.0489	0.2799

Table 5.3: Average relative face area and volume distortions after registering the source mesh to the target mesh, with a global stiffness deformation ($\lambda = 10$, $\mu = 10$, $\nu = 1000$, landmark factor = 1) + NRICP method. Negative values signify the mesh simplexes are shrinking, while positive values mean it is expanding.

Character	Faces	Lmrks	Face Area Distortion	Face Area σ	Face Volume Distortion	Volume σ
Blob Fish	3482	12	-0.0431	0.2605	-0.0164	0.5405
Party Onion	3600	13	-0.0073	0.2357	0.0410	0.5068
Tick	2750	13	-0.0734	0.2872	-0.0589	0.5842
Hellidropter	1454	10	-0.0358	0.1376	-0.0514	0.2788

Figure 5-16 represents the in-between poses of the original NRICP applied to the Tick character. Figure 5-17 is the sequence of poses obtained after initializing the registration with a local IDW stiffness deformation. Notice that the latter adds more variety in the shapes obtained, which can then be used by the artist for extending the animation.

Table 5.4: Hausdorff distance after applying the original non-rigid registration (NRICP) method as proposed by Amberg et al. (2007), between the source and the target meshes. The time for each registration is also given in seconds.

Character	Faces	Landmarks	Hausdorff	Time(s)
Blob Fish	3482	12	0.0621	327.80
Party Onion	3600	13	0.0664	355.99
Tick	2750	13	0.1189	196.86
Hellidropter	1454	10	0.0353	43.93

Table 5.5: Hausdorff distance after registering the source mesh to the target mesh, with a local stiffness deformation (inverse distance weights, propagated globally, exponent of 2, landmark factor = 1) + NRICP method. The time for each registration is also given in seconds.

Character	Faces	Landmarks	Hausdorff	Time(s)
Blob Fish	3482	12	0.0236	379.44
Party Onion	3600	13	0.0336	370.85
Tick	2750	13	0.1038	245.80
Hellidropter	1454	10	0.0217	52.75

Table 5.6: Hausdorff distance after registering the source mesh to the target mesh, with a global stiffness deformation ($\lambda = 10$, $\mu = 10$, $\nu = 1000$, landmark factor = 1) + NRICP method. The time for each registration is also given in seconds.

Character	Faces	Landmarks	Hausdorff	Time(s)
Blob Fish	3482	12	0.0185	348.19
Party Onion	3600	13	0.0319	359.88
Tick	2750	13	0.1095	209.15
Hellidropter	1454	10	0.0217	50.86

Variety in this context signifies the localized control over the character deformation. The Tick can display frowning, dents and bulges of different areas, without affecting the rest of the mesh.

Experiments on Enhancement 4: Part Based Deformation

In the case of Gauss curvature based segmentation, some experiments were done with a low resolution version of the Rob character (figure 5-18). Good results were obtained in the regions with negative curvature (figure 5-19, top), but it failed in some cases, like the front of Rob’s face or the template’s torso, where the arms and torso are in the same segment. Planar segmentation was also used on Rob (figure 5-19, bottom) and on Hellidropter (figure 5-20).

A significant difference in the speed of registration was noticed, when segmentation was

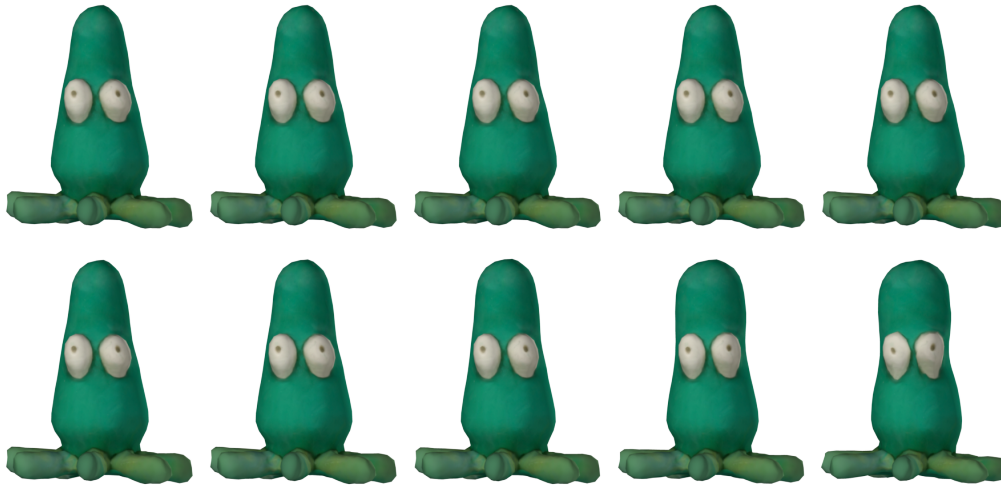


Figure 5-16: Sequence of in-between poses obtained on Tick character during the original NRICP method.

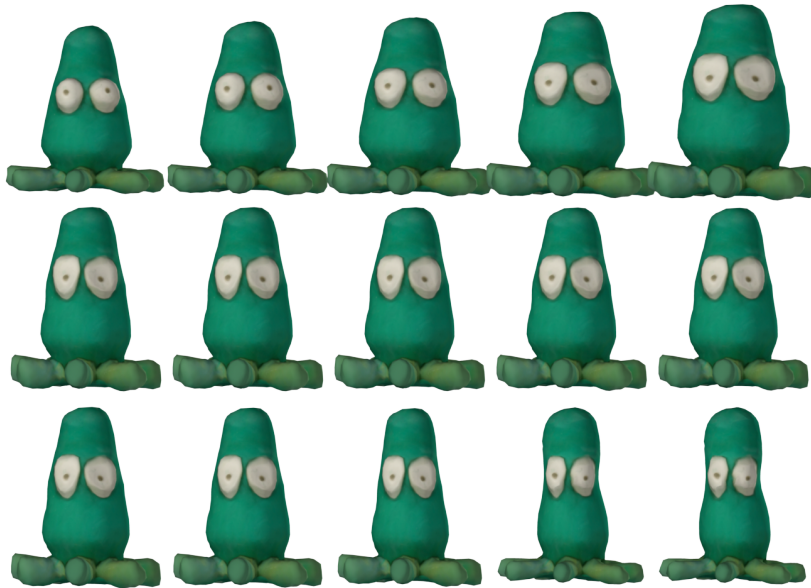


Figure 5-17: Sequence of in-between poses obtained on Tick character during the NRICP process initialized with a local IDW stiffness deformation.



Figure 5-18: Photograph of Rob character in first pose (left), photogrammetry result (middle) and photogrammetry result in a second pose (right).

employed. After a planar segmentation that took 0.000133 seconds for Hellidropter (figure 5-20), the results from table 5.7 were obtained. Notice that the sum of separate registrations for the propeller and head of the character is much lower and requires fewer landmarks than the registration for the whole body.

Table 5.7: Registration results for Hellidropter character for the whole body or for the separate segments (propeller and head in figure 5-20). This stiffness used was a global one, decreasing with a step of one unit per iteration.

Part of Hellidropter NRICP was used on	Stiffness (step)	Distance threshold (ϵ)	Landmarks	Iterations	Time (s)
Whole body	10 (-1)	1	60	8	26.4
Whole body	10 (-1)	3	60	8	9.61
Whole body	10 (-1)	5	60	8	9.74
Propeller	100 (-1)	3	10	6	1.97
Head	100 (-1)	3	10	3	0.74

5.2.3 Discussion

The *Nonrigid Registration* component of the E-StopMotion pipeline was presented, together with a few enhancements. These enhancements are linked to the material and structure of plasticine characters and aim to better guide the registration. Research question 4 was partially answered in the current work, as finding a registration method to suit all Clay Jam characters would require future investigation. Nevertheless, it was

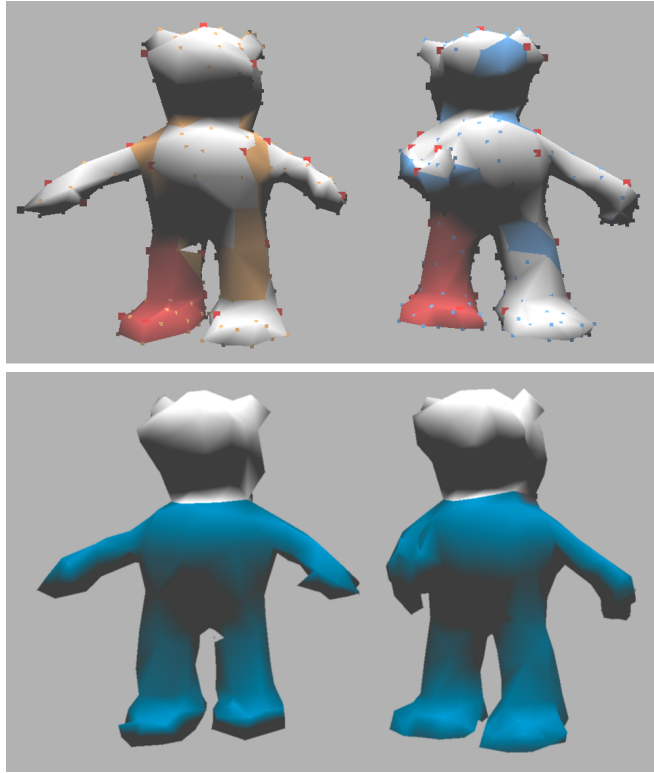


Figure 5-19: Gauss curvature segmentation (top) and planar segmentation of low resolution versions of Rob. In the top image, the source pose (left) has the right leg selected (red colour), similarly to the target pose (right), which means that a match will take place between the two segments. The segmentation was successful only on a few parts of the model (eg. right arm and leg). In the bottom image, a segmentation plane was selected at the base of the neck for the source (left) and target (right) poses for Rob.

shown how deformation and dense correspondences can aid the registration for such characters.

Limitations and Future Work

The non-rigid iterative closest point (NRICP) algorithm is extrinsic by design, as it views the mesh geometry as a whole and is not concerned with the intrinsics of any particular areas of the model. Although suitable for face deformations (Amberg et al. (2007)), experimentation rendered this technique unstable for matching character with more complex structures and deformations (eg. large rotations). This is mostly due to the aforementioned extrinsic design. Chang & Zwicker (2008) also confirm this empirical discovery, when they mention that the NRICP only approximates a non-rigid transformation and was not built to support large scale deformations.

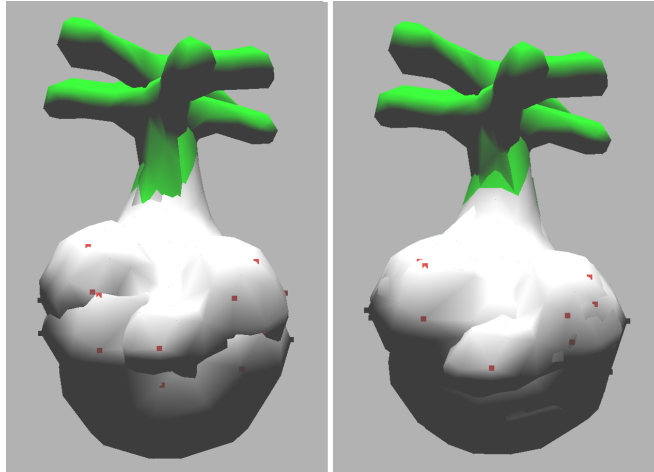


Figure 5-20: Planar segmentation of Helldropter. The white segment contains landmarks which are used to guide the non-rigid registration. The left image represents the overlapping source and target before registration. The right image represents the source and target after registration. Note that only the white segment is considered for the registration.

In order for the NRICP to deliver acceptable results, it would require a large number of helpers on both template and target, which are tedious to place by hand. In the future an automation for the helper selection could be done by character animation type, as classified in chapter 2. A segmentation by motion procedure (Chang & Zwicker (2008), Zheng et al. (2010)) can be adopted to identify whether a character displays animation by shape or by skeleton. Then the appropriate helpers (eg. landmarks, skeletons respectively) can be added to aid the non-rigid registration.

Landmark selection could also be replaced by salient point detection or sampling techniques in future work. The reason why such a method could be challenging is because plasticine can change shape significantly from frame to frame. This depends, of course, on the scale of deformation chosen. So for small and medium deformations, finding salient features that maintain the same properties from frame to frame could help aid the deformation.

Chapter 6

Conclusions

Having worked together with Fat Pebble for two years, the author managed to get an insight into the process of creating stop motion animation games. The work that goes into designing, modelling and photographing plasticine characters can be overwhelming for the artists. Also, their work is usually seen from a few angles, as photographed beforehand. An animation reconstruction pipeline was created to bring stop motion plasticine character animation into the 3D world.

The project's focus was mainly on the characters from the game Clay Jam. It started by analyzing the style and movement of these characters. A simple heuristic was proposed in chapter 2 to classify characters by the type of animation they display. It was observed that the majority of characters are made of parts that perform either an *Animation by Skeleton* or an *Animation by Shape*. This classification was then linked to the types of helpers (eg. landmarks, skeletons, segmentation) that can aid in subsequent deformation and registration steps.

Next, the definition and development of the pipeline were described in chapter 3. The animation reconstruction process started by performing acquisition and cleanup on a few *key poses* for each physical character. This resulted in a series of mesh scans with different connectivities. In order to obtain a sequence of meshes with the same connectivity, non-rigid registration was used (NRICP as in Amberg et al. (2007)). This was enhanced by adding a prior deformation step that makes use of the material properties of plasticine (chapter 4). These include volume preservation and local stiffness.

It was showed that a nonlinear, surface based deformation method can produce physically plausible deformations that imitate plasticine. This was done mainly by adding

local stiffness to the Fröhlich & Botsch (2011) thin shell deformation method. The deformation technique was used as an initialization step for the NRICP to enhance registration. A few physically plausible in-between poses were generated as a result, which could later be added to the stop animation.

After having the registered meshes, as-rigid-as-possible interpolation was used to generate more in-between poses. The algorithm is the one proposed in Ciucanu et al. (2018) and expanded in chapter 3. Plausible shapes can be generated using ARAP interpolation to enhance the original stop motion animation.

In the future, the focus should be on including the extended components from chapters 4 and 5 into the simplified pipeline presented in section 3.3. It would also be ideal to automate the process of classifying characters and adding the corresponding helpers. Helpers like skeletons will be experimented with in the future to improve deformations and registrations.

The author believes that the current work can be integrated into the pipeline of a stop motion animation studio. This could help artists bring their work to life in the CG world. By fusing handmade art with computer generated techniques, photorealism can be brought to the CG world and the stop motion art-form can be extended into the virtual world. Applications range from interacting with plasticine characters in games to editing stop motion digitally in film productions.

6.1 Acknowledgements

I would like to thank my supervisor Darren Cosker for his guidance, optimism and patience. I would also like to thank Yong-Liang Yang (Mac) for his help and advice during the development of the project. I thank Naval Bhandari and Shridhar Ravikumar for providing the original Matlab code for the ARAP interpolation and Non-rigid Registration (NRICP) algorithms respectively. Thanks also go to Xiaokun Wu (Kelvin) for his help while writing the MIG'18 paper.

I would like to thank my colleagues and friends, from the Computer Science Department, for their support, advice, encouragement, smiles and hugs. In no particular order: Joanna Tarko, Bingjie Yu, Maryam Sahar Naghizadeh, Jo Hyde, John Benardis, Christina Keating, Luca Benedetti, Daniela de Angeli, Andrea Aler Tubella, Andreas Theodorou, Cillian Dudley, Will Hua, Catherine Taylor, Zack Lyons, Thu Nguyen-Phuoc, Allesio Santamaria, David Sherratt, Andrew Lawrence, Horia Bogdan, Thomas James Williams, Sinéad Kearney, Daniel Finnegan.

Thanks also go to the Center for Digital Entertainment team and EPSRC for funding my research. Thank you to the former Fat Pebble team, Iain Gilfeather, the Technical Director, Chris Roe, the Art Director and Michael Movel, the Design Director. Thank you also to my former supervisors Brian Wyvill and Philip Willis.

Last, but not least, thank you to all the staff members from the Computer Science Department. I enjoyed working with you and soaking up your wisdom. In no particular order: Christian Richardt, Neill Campbell, Leon Watts, John Power, Fabio Nemetz, Joanna Bryson, Michael Wright, Peter Hall, Sarah Parry, Rebecca Knight, James Laird.

May we meet again on undiscovered plains of science!

Appendix A

Clay Jam Characters

Here is a list of the Clay Jam characters used in this thesis. The characters from figure A-1 were used in the experiments from chapters 3, 4 and 5, while the rest were only used in the classification process from chapter 2. Notice that the Tick character (figure A-1) had to be remodelled, since the original plasticine character had been lost.

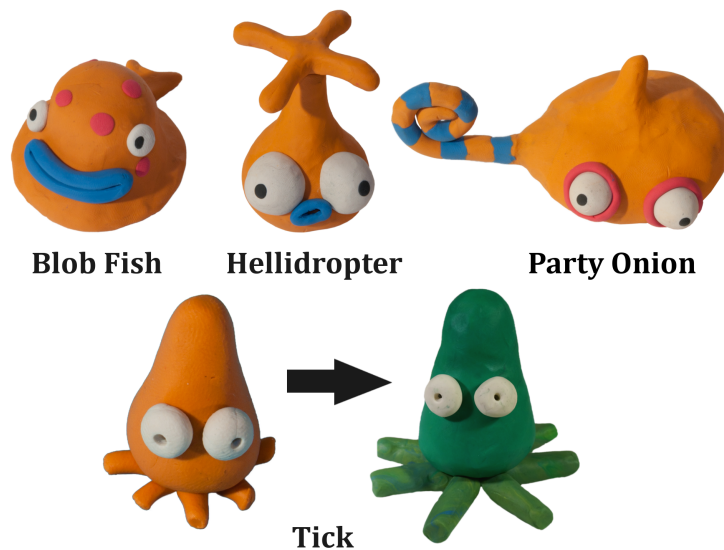


Figure A-1: Clay Jam characters used for experiments: Blob Fish (top left), Hellidropter (top middle), Party Onion (top right), Tick (bottom right), which was a copy of the original plasticine model (bottom left), which had been lost.



Ski Leech



Sausage Roll



Worm



Nobody Nose



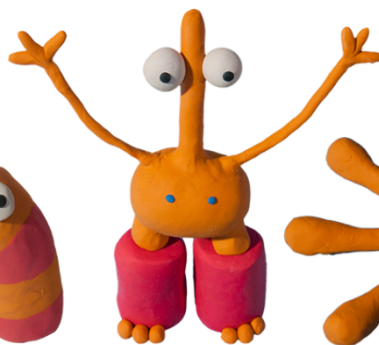
Bingo Winged Bat



Heavy Head



Slinky



**Bantam
Legs**



Rotoctopus

Figure A-2: Clay Jam characters used for classification (chapter 2): From top, left to right, Ski Leech, Sausage Roll, Worm, Nobody Nose, Bingo Winged Bat, Heavy Head, Slinky, Bantam Legs, Rotoctopus.

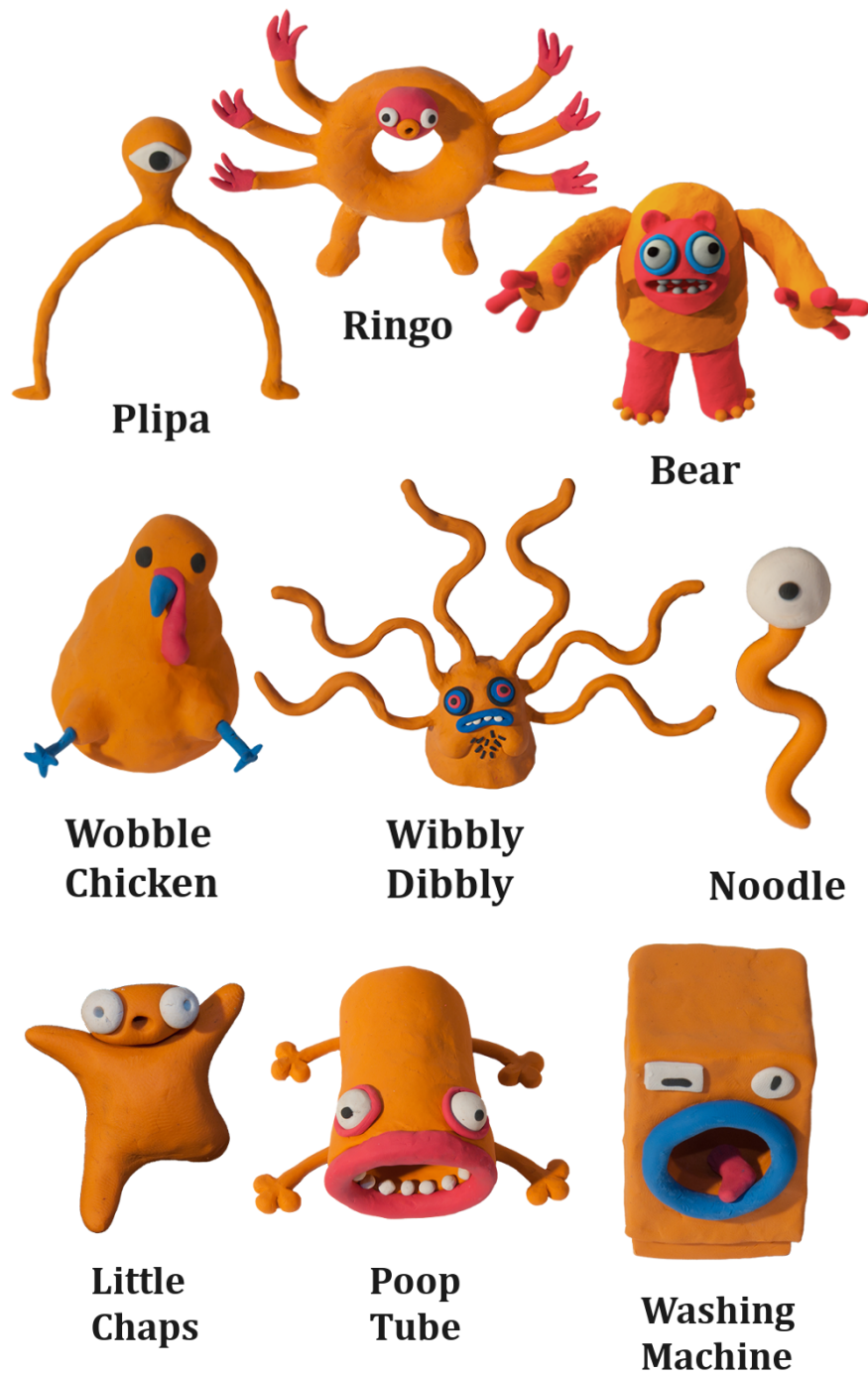


Figure A-3: Clay Jam characters used for classification (chapter 2): From top, left to right, Plipa, Ringo, Bear, Wobble Chicken, Wibbly Dibbly, Noodle, Little Chaps, Poop Tube, Washing Machine.



**Ground
Mouth**



**Monstre
Perillios**



Margo



Psych Sid



**Drake the Duck
Dragon**



**Bung it
Billy**



Big Jigger



**Wonder
Beard**

Figure A-4: Clay Jam characters used for classification (chapter 2): From top, left to right, Ground Mouth, Monstre Perillios, Margo, Psych Sid, Drake the Duck Dragon, Bung it Billy, Big Jigger, Wonder Beard.

Appendix B

Acronyms

- 2D = two dimensional
- 3D = three dimensional
- CG = computer graphics or computer generated
- ARAP = as-rigid-as-possible
- DSLR = digital single lens reflex
- FEM = finite element method
- ICP = iterative closest point
- IDW = inverse distance weights
- LOD = level-of-detail
- NRICP = non-rigid iterative closest point
- MLS = moving least squares

Bibliography

Agisoft (2018), ‘Photoscan’, <http://www.agisoft.com/>.

Alexa, M., Cohen-Or, D. & Levin, D. (2000), As-rigid-as-possible shape interpolation, *in* ‘Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques’, SIGGRAPH ’00, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, pp. 157–164.

Allen, B., Curless, B. & Popović, Z. (2003), ‘The space of human body shapes: Reconstruction and parameterization from range scans’, *ACM Trans. Graph.* **22**(3), 587–594.

Amberg, B., Romdhani, S. & Vetter, T. (2007), ‘Optimal step nonrigid icp algorithms for surface registration’, *2007 IEEE Conference on Computer Vision and Pattern Recognition* pp. 1–8.

Angelidis, A. & Wyvill, G. (2004), Animated sweepers: keyframed swept deformations, *in* ‘Proceedings Computer Graphics International, 2004.’, pp. 320–326.

Angles, B., Tarini, M., Wyvill, B., Barthe, L. & Tagliasacchi, A. (2017), ‘Sketch-based implicit blending’, *ACM Trans. Graph.* **36**(6), 181:1–181:13.

Anguelov, D., Srinivasan, P., Koller, D., Thrun, S., Rodgers, J. & Davis, J. (2005), ‘Scape: Shape completion and animation of people’, *ACM Trans. Graph.* **24**(3), 408–416.

Anguelov, D., Srinivasan, P., Pang, H.-C., Koller, D., Thrun, S. & Davis, J. (2004), The correlated correspondence algorithm for unsupervised registration of nonrigid surfaces, *in* ‘Proceedings of the 17th International Conference on Neural Information Processing Systems’, NIPS’04, MIT Press, Cambridge, MA, USA, pp. 33–40.

Arata, H., Takai, Y., Takai, N. K. & Yamamoto, T. (1999), Free-form shape modeling

- by 3d cellular automata, *in* ‘In International Conference on Shape Modeling and Applications’, Society Press, pp. 242–247.
- Au, O. K.-C., Tai, C.-L., Chu, H.-K., Cohen-Or, D. & Lee, T.-Y. (2008), ‘Skeleton extraction by mesh contraction’, *ACM Trans. Graph.* **27**(3), 44:1–44:10.
- Autodesk (2018a), ‘3ds max’, <https://www.autodesk.co.uk/products/3ds-max/overview>.
- Autodesk (2018b), ‘Maya’, <https://www.autodesk.co.uk/products/maya/overview>.
- Bærentzen, J. A. (1998), Octree-based volume sculpting.
- Bargteil, A. W., Wojtan, C., Hodgins, J. K. & Turk, G. (2007), ‘A finite element method for animating large viscoplastic flow’, *ACM Transactions on Graphics* **26**(3), 16.
- Baxter, W., Barla, P. & Anjyo, K.-i. (2008), Rigid shape interpolation using normal equations, *in* ‘Proceedings of the 6th International Symposium on Non-photorealistic Animation and Rendering’, NPAR ’08, ACM, New York, NY, USA, pp. 59–64.
- Berger, M. & Silva, C. T. (2012), ‘Nonrigid matching of undersampled shapes via medial diffusion’, *Computer Graphics Forum* **31**(5), 1587–1596.
- Besl, P. J. & McKay, N. D. (1992), ‘A method for registration of 3-d shapes’, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **14**(2), 239–256.
- Biasotti, S., Cerri, A., Bronstein, A. & Bronstein, M. (2016), ‘Recent trends, applications, and perspectives in 3d shape similarity assessment’, *Computer Graphics Forum* .
- Bill, J. R. & Lodha, S. K. (1995), Sculpting polygonal models using virtual tools, *in* ‘Proceedings of Graphics Interface ’95’, GI ’95, Canadian Human-Computer Communications Society, Toronto, Ontario, Canada, pp. 272–279.
- Blinn, J. F. (1982), ‘A generalization of algebraic surface drawing’, *SIGGRAPH Comput. Graph.* **16**(3).
- Bonarrigo, F., Signoroni, A. & Botsch, M. (2014), ‘Deformable registration using patch-wise shape matching’, *Graphical Models* **76**(5), 554–565.
- Botsch, M. & Kobbelt, L. (2004), ‘An intuitive framework for real-time freeform modeling’, *ACM Trans. Graph.* **23**(3), 630–634.

- Botsch, M. & Kobbelt, L. (2005), ‘Real-time shape editing using radial basis functions’, *Computer Graphics Forum* **24**(3), 611–621.
- Botsch, M., Kobbelt, L., Pauly, M., Alliez, P. & Levy, B. (2010), *Polygon Mesh Processing*, Ak Peters Series, Taylor & Francis.
- Botsch, M., Pauly, M., Gross, M. & Kobbelt, L. (2006), Primo: Coupled prisms for intuitive surface modeling, in ‘Proceedings of the Fourth Eurographics Symposium on Geometry Processing’, SGP ’06, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp. 11–20.
- Botsch, M. & Sorkine, O. (2008), ‘On linear variational surface deformation methods’, *IEEE Transactions on Visualization and Computer Graphics* **14**(1), 213–230.
- Bronstein, A., Bronstein, M. & Kimmel, R. (2008), *Numerical Geometry of Non-Rigid Shapes*, 1 edn, Springer Publishing Company, Incorporated.
- Bronstein, A. M., Bronstein, M. M., Guibas, L. J. & Ovsjanikov, M. (2011), ‘Shape google: Geometric words and expressions for invariant shape retrieval’, *ACM Trans. Graph.* **30**(1), 1:1–1:20.
- Bronstein, A. M., Bronstein, M. M. & Kimmel, R. (2006), ‘Generalized multidimensional scaling: A framework for isometry-invariant partial surface matching’, *Proceedings of the National Academy of Sciences* **103**(5), 1168–1172.
- Bronstein, A. M., Bronstein, M. M. & Kimmel, R. (2007), Rock, paper, and scissors: extrinsic vs. intrinsic similarity of non-rigid shapes, in ‘2007 IEEE 11th International Conference on Computer Vision’, pp. 1–6.
- Bronstein, A. M., Bronstein, M. M., Kimmel, R., Mahmoudi, M. & Sapiro, G. (2010), ‘A gromov-hausdorff framework with diffusion geometry for topologically-robust non-rigid shape matching’, *International Journal of Computer Vision* **89**(2), 266–286.
- Brown, B. & Rusinkiewicz, S. (2007), ‘Global non-rigid alignment of 3-D scans’, *ACM Transactions on Graphics (Proc. SIGGRAPH)* **26**(3).
- Budd, C., Huang, P., Klaudiny, M. & Hilton, A. (2013), ‘Global non-rigid alignment of surface sequences’, *International Journal of Computer Vision* **102**(1), 256–270.
- Cani-Gascuel, M. & Desbrun, M. (1997), ‘Animation of deformable models using implicit surfaces’, *IEEE Transactions on Visualization and Computer Graphics* **3**(1), 39–50.

- Cani, M.-P. & Angelidis, A. (2006), Towards virtual clay, *in* ‘ACM SIGGRAPH’, p. 67.
- Catmull, E. & Clark, J. (1978), ‘Recursively generated b-spline surfaces on arbitrary topological meshes’, *Computer-Aided Design* **10**(6), 350–355.
- Celniker, G. & Gossard, D. (1991), ‘Deformable curve and surface finite-elements for free-form shape design’, *SIGGRAPH Comput. Graph.* **25**(4), 257–266.
- Chakrabarty, J. (2009), *Applied Plasticity, Second Edition*, Mechanical Engineering Series, Springer US.
- Chang, W. & Zwicker, M. (2008), Automatic registration for articulated shapes, *in* ‘Proceedings of the Symposium on Geometry Processing’, SGP ’08, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp. 1459–1468.
- Chang, W. & Zwicker, M. (2009), ‘Range scan registration using reduced deformable models’, *Computer Graphics Forum* **28**(2), 447–456.
- Cheng, Z., Jiang, W., Dang, G., Martin, R. R., Li, J., Li, H., Chen, Y., Wang, Y., Li, B., Xu, K. & Jin, S. (2010), Non-rigid registration in 3d implicit vector space, *in* ‘2010 Shape Modeling International Conference’, pp. 37–46.
- Ciucanu, A., Bhandari, N., Wu, X., Ravikumar, S., Yang, Y.-L. & Cosker, D. (2018), E-stopmotion: Digitizing stop motion for enhanced animation and games, *in* ‘Proceedings of MIG 18: Motion, Interaction and Games’, MIG 18, ACM, New York, NY, USA.
- Crandall, S. H., Kurzweil, L. G. & Nigam, A. K. (1971), ‘On the measurement of poisson’s ratio for modeling clay’, *Experimental Mechanics* **11**(9), 402–413.
- Davis, H. & Snider, A. (1995), *Introduction to Vector Analysis*, Excellence in mathematics, Wm. C. Brown.
- DeRose, T., Kass, M. & Truong, T. (1998), Subdivision surfaces in character animation, *in* ‘Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques’, SIGGRAPH ’98, ACM, New York, NY, USA, pp. 85–94.
- Desbrun, M. & Gascuel, M.-P. (1994), Highly deformable material for animation and collision processing, 5th Eurographics workshop on animation and simulation.
- Desbrun, M. & Gascuel, M.-P. (1995), Animating soft substances with implicit surfaces, *in* ‘Proceedings of the 22Nd Annual Conference on Computer Graphics and Interactive Techniques’, SIGGRAPH ’95, ACM, New York, NY, USA, pp. 287–290.

- Dewaele, G. & Cani, M.-P. (2004), ‘Interactive global and local deformations for virtual clay’, *Graph. Models* **66**(6), 352–369.
- Dey, T. K., Fu, B., Wang, H. & Wang, L. (2015), ‘Automatic posing of a meshed human model using point clouds’, *Computers and Graphics* **46**, 14–24.
- do Carmo, M. P. (1976), *Differential Geometry of Curves and Surfaces*, Englewood Cliffs, NJ: Prentice Hall.
- Eisenberger, M., Lhner, Z. & Cremers, D. (2018), ‘Divergence-free shape interpolation and correspondence’.
- Elad, A. & Kimmel, R. (2003), ‘On bending invariant signatures for surfaces’, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **25**(10), 1285–1295.
- Ferley, E., Cani, M.-P. & Gascuel, J.-D. (2001), ‘Resolution adaptive volume sculpting’, *Graphical Models* **63**(6), 459–478.
- Fröhlich, S. & Botsch, M. (2011), ‘Example-driven deformations based on discrete shells’, *Computer Graphics Forum* **30**(8), 2246–2257.
- Funkhouser, T., Min, P., Kazhdan, M., Chen, J., Halderman, A., Dobkin, D. & Jacobs, D. (2003), ‘A search engine for 3d models’, *ACM Trans. Graph.* **22**(1), 83–105.
- Gal, R. & Cohen-Or, D. (2006), ‘Salient geometric features for partial shape matching and similarity’, *ACM Trans. Graph.* **25**(1), 130–150.
- Galyean, T. A. & Hughes, J. F. (1991), Sculpting: An interactive volumetric modeling technique, in ‘Computer Graphics (Proceedings of SIGGRAPH 91)’, Vol. 25, pp. 267–274.
- Ganapathi-Subramanian, V., Thibert, B., Ovsjanikov, M. & Guibas, L. (2016), ‘Stable region correspondences between non-isometric shapes’, *Computer Graphics Forum* **35**(5), 121–133.
- Gao, L., Chen, S.-Y., Lai, Y.-K. & Xia, S. (2016), ‘Data-driven shape interpolation and morphing editing’, *Computer Graphics Forum* **36**(8), 19–31.
- Gerszewski, D., Bhattacharya, H. & Bargteil, A. W. (2009), A point-based method for animating elastoplastic solids, in ‘Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation’, SCA ’09, ACM, New York, NY, USA, pp. 133–138.

- Goktekin, T. G., Bargteil, A. W. & O'Brien, J. F. (2004), 'A method for animating viscoelastic fluids', *ACM Trans. Graph.* **23**(3), 463–468.
- Greaves, G. N., Greer, A. L., Lakes, R. S. & Rouxel, T. (2011), 'Poisson's ratio and modern materials', *Nature Materials* **10**, 823. Review Article.
- Grinspun, E., Hirani, A. N., Desbrun, M. & Schröder, P. (2003), Discrete shells, in 'Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation', SCA '03, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp. 62–67.
- Gross, M. & Pfister, H. (2007), *Point-Based Graphics*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
- Hirshberg, D. A., Loper, M., Rachlin, E. & Black, M. J. (2012), Coregistration: Simultaneous alignment and modeling of articulated 3d shape, in A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato & C. Schmid, eds, 'Computer Vision – ECCV 2012', Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 242–255.
- Hontani, H., Matsuno, T. & Sawada, Y. (2012), Robust nonrigid icp using outlier-sparsity regularization, in '2012 IEEE Conference on Computer Vision and Pattern Recognition', pp. 174–181.
- Horn, B. K. P. & Schunck, B. G. (1981), 'Determining optical flow', *Artif. Intell.* **17**(1-3), 185–203.
- Huang, J., Shi, X., Liu, X., Zhou, K., Wei, L.-Y., Teng, S.-H., Bao, H., Guo, B. & Shum, H.-Y. (2006), 'Subspace gradient domain mesh deformation', *ACM Trans. Graph.* **25**(3), 1126–1134.
- Huang, Q.-X., Adams, B., Wicke, M. & Guibas, L. J. (2008), Non-rigid registration under isometric deformations, in 'Proceedings of the Symposium on Geometry Processing', SGP '08, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp. 1449–1457.
- Irving, G., Teran, J. & Fedkiw, R. (2004), Invertible finite elements for robust simulation of large deformation, in 'Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation', SCA '04, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp. 131–140.
- Jain, V., Zhang, H. & van Kaick, O. (2007), 'Non-rigid spectral correspondence of triangle meshes', *International Journal on Shape Modeling* **13**(1), 101–124.

- Jones, B., Thuerey, N., Shinar, T. & Bargteil, A. W. (2016), ‘Example-based plastic deformation of rigid bodies’, *ACM Trans. Graph.* **35**(4), 34:1–34:11.
- Joshi, P., Meyer, M., DeRose, T., Green, B. & Sanocki, T. (2007), ‘Harmonic coordinates for character articulation’, *ACM Trans. Graph.* **26**(3).
- Kanazawa, A., Kovalsky, S., Basri, R. & Jacobs, D. (2016), ‘Learning 3d deformation of animals from 2d images’, *Comput. Graph. Forum* **35**(2), 365–374.
- Keiser, R., Adams, B., Gasser, D., Bazzi, P., Dutré, P. & Gross, M. (2005), A unified lagrangian approach to solid-fluid animation, in ‘Proceedings of the Second Eurographics / IEEE VGTC Conference on Point-Based Graphics’, SPBG’05, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp. 125–133.
- Kim, V. G., Lipman, Y. & Funkhouser, T. (2011), ‘Blended intrinsic maps’, *ACM Trans. Graph.* **30**(4), 79:1–79:12.
- Kin-Chung Au, O., Tai, C.-L., Cohen-Or, D., Zheng, Y. & Fu, H. (2010), ‘Electors voting for fast automatic shape correspondence’, *Computer Graphics Forum* **29**(2), 645–654.
- Kraevoy, V. & Sheffer, A. (2004), ‘Cross-parameterization and compatible remeshing of 3d models’, *ACM Trans. Graph.* **23**(3), 861–869.
- Li, H. (2010), Animation Reconstruction of Deformable Surfaces, PhD thesis, ETH Zurich.
- Li, H., Adams, B., Guibas, L. J. & Pauly, M. (2009), ‘Robust single-view geometry and motion reconstruction’, *ACM Trans. Graph.* **28**(5), 175:1–175:10.
- Li, H., Sumner, R. W. & Pauly, M. (2008), Global correspondence optimization for non-rigid registration of depth scans, in ‘Proceedings of the Symposium on Geometry Processing’, SGP ’08, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp. 1421–1430.
- Li, X. & Iyengar, S. S. (2014), ‘On computing mapping of 3d objects: A survey’, *ACM Comput. Surv.* **47**(2), 34:1–34:45.
- Lipman, Y. & Funkhouser, T. (2009), ‘Möbius voting for surface correspondence’, *ACM Trans. Graph.* **28**(3), 72:1–72:12.
- Lipman, Y., Levin, D. & Cohen-Or, D. (2008), ‘Green coordinates’, *ACM Trans. Graph.* **27**(3), 78:1–78:10.

- Lipman, Y., Rustamov, R. M. & Funkhouser, T. A. (2010), ‘Biharmonic distance’, *ACM Trans. Graph.* **29**(3), 27:1–27:11.
- Liu, Y.-S., Yan, H.-B. & Martin, R. R. (2011), ‘As-rigid-as-possible surface morphing’, *Journal of Computer Science and Technology* **26**(3), 548–557.
- Lowe, D. G. (1999), Object recognition from local scale-invariant features, *in* ‘Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2’, ICCV ’99, IEEE Computer Society, Washington, DC, USA, pp. 1150–.
- Lubliner, J. (2008), *Plasticity Theory*, Dover books on engineering, Dover Publications.
- McDonnell, K. T. & Qin, H. (2002), ‘Dynamic sculpting and animation of free-form subdivision solids’, *Vis. Comput.* **18**(2), 81–96.
- McDonnell, K. T., Qin, H. & Wlodarczyk, R. A. (2001), Virtual clay: A real-time sculpting system with haptic toolkits, *in* ‘Proceedings of the 2001 Symposium on Interactive 3D Graphics’, I3D ’01, ACM, New York, NY, USA, pp. 179–190.
- Mitra, N. J., Flöry, S., Ovsjanikov, M., Gelfand, N., Guibas, L. & Pottmann, H. (2007), Dynamic geometry registration, *in* ‘Proceedings of the Fifth Eurographics Symposium on Geometry Processing’, SGP ’07, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp. 173–182.
- Moreton, H. P. & Séquin, C. H. (1992), ‘Functional optimization for fair surface design’, *SIGGRAPH Comput. Graph.* **26**(2), 167–176.
- Müller, M. & Gross, M. (2004), Interactive virtual materials, *in* ‘Proceedings of Graphics Interface 2004’, GI ’04, Canadian Human-Computer Communications Society, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, pp. 239–246.
- Niu, J., Lei, J. & He, J. (2017), ‘Radial basis function mesh deformation based on dynamic control points’, *Aerospace Science and Technology* **64**, 122 – 132.
- Noble, R., Zhang, K. & McDermott, R. (2004), An investigation of multiple tools actions for virtual sculpting using implicit surfaces, *in* ‘Proceedings Theory and Practice of Computer Graphics, 2004.’, pp. 24–31.
- O’Brien, J. F. (2002), Graphical modeling and animation of ductile fracture, *in* ‘Proceedings of the 29th International Conference on Computer Graphics and Interactive Techniques. Electronic Art and Animation Catalog.’, SIGGRAPH ’02, ACM, New York, NY, USA, pp. 161–161.

- OrbitVU (2018), ‘Alphashot360’, <https://orbitvu.com/alphashot-360>.
- Originals, A. (2016), ‘Credited as: Head of puppetry’, <https://www.youtube.com/watch?v=kl7aLqgDpE>.
- Ovsjanikov, M., Ben-Chen, M., Solomon, J., Butscher, A. & Guibas, L. (2012), ‘Functional maps: A flexible representation of maps between shapes’, *ACM Trans. Graph.* **31**(4), 30:1–30:11.
- Panozzo, D., Baran, I., Diamanti, O. & Sorkine-Hornung, O. (2013), ‘Weighted averages on surfaces’, *ACM Trans. Graph.* **32**(4), 60:1–60:12.
- Papazov, C. & Burschka, D. (2011), ‘Deformable 3d shape registration based on local similarity transforms’, *Computer Graphics Forum* **30**(5), 1493–1502.
- Pauly, M., Keiser, R., Adams, B., Dutré, P., Gross, M. & Guibas, L. J. (2005), ‘Meshless animation of fracturing solids’, *ACM Trans. Graph.* **24**(3), 957–964.
- Pebble, F. (2013), ‘Fat pebble games’, <http://www.fatpebble.com/>.
- Pebble, F. (2014), ‘Clay jam’, <https://play.google.com/store/apps/details?id=com.zynga.fatpebble.clayjam>.
- Pekelný, Y. & Gotsman, C. (2008), ‘Articulated object reconstruction and markerless motion capture from depth video’, *Computer Graphics Forum* **27**(2), 399–408.
- Perry, R. N. & Frisken, S. F. (2001), Kizamu: A system for sculpting digital characters, in ‘Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques’, SIGGRAPH ’01, ACM, New York, NY, USA, pp. 47–56.
- Piti, T. (2006), ‘Stop motion gains speed’.
URL: <http://www.adweek.com/brand-marketing/stop-motion-gains-speed-84628/>
- Popa, T., Julius, D. & Sheffer, A. (2006), ‘Material-aware mesh deformations’, *IEEE International Conference on Shape Modeling and Applications 2006* p. 22.
- Prager, W. & Hodge, P. (1951), *Theory of Perfectly Plastic Solids*, Applied Mathematics Series, John Wiley & Sons.
- Purves, B. J. (2016), *Stop-Motion Animation: Frame by Frame Film-making with Puppets and Models*, Vol. 2, Fairchild Books.
- Ramberg, W. & Osgood, W. R. (1943), ‘Description of stress-strain curves by three parameters’.

- Raviv, A. & Elber, G. (2000), ‘Three-dimensional freeform sculpting via zero sets of scalar trivariate functions’, *Computer-Aided Design* **32**(8), 513–526.
- Rustamov, R. M. (2010), ‘Barycentric coordinates on surfaces’, *Computer Graphics Forum* (29), 1507–1516.
- Schulz, C., von Tycowicz, C., Seidel, H.-P. & Hildebrandt, K. (2014), ‘Animating deformable objects using sparse spacetime constraints’, *ACM Trans. Graph.* **33**(4), 109:1–109:10.
- Selim, M. M. & Koomullil, R. P. (2016), ‘Mesh deformation approaches a survey’, *Journal of Physical Mathematics* **7**(2), 9.
- Sharf, A., Alcantara, D. A., Lewiner, T., Greif, C., Sheffer, A., Amenta, N. & Cohen-Or, D. (2008), ‘Space-time surface reconstruction using incompressible flow’, *ACM Trans. Graph.* **27**(5), 110:1–110:10.
- Shepard, D. (1968), A two-dimensional interpolation function for irregularly-spaced data, in ‘Proceedings of the 1968 23rd ACM National Conference’, ACM ’68, ACM, New York, NY, USA, pp. 517–524.
- Sorkine, O. (2005), Laplacian mesh processing, in Y. Chrysanthou & M. Magnor, eds, ‘Eurographics 2005 - State of the Art Reports’, The Eurographics Association.
- Sorkine, O. (2006), ‘Differential representations for mesh processing’, *Computer Graphics Forum* **25**(4), 789–807.
- Sorkine, O. & Alexa, M. (2007), As-rigid-as-possible surface modeling, in ‘Proceedings of the Fifth Eurographics Symposium on Geometry Processing’, SGP ’07, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp. 109–116.
- Sorkine, O., Cohen-Or, D., Lipman, Y., Alexa, M., Rössl, C. & Seidel, H.-P. (2004), Laplacian surface editing, in ‘Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing’, SGP ’04, ACM, New York, NY, USA, pp. 175–184.
- Stam, J. (1999), Stable fluids, in ‘Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques’, SIGGRAPH ’99, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, pp. 121–128.
- Stam, J. (2002), ‘A simple fluid solver based on the fft’, *J. Graph. Tools* **6**(2), 43–52.

- Stanculescu, L., Chaine, R. & Cani, M.-P. (2011), ‘Freestyle: Sculpting meshes with self-adaptive topology’, *Computers and Graphics* **35**(3), 614–622.
- Sumner, R. W. & Popović, J. (2004), ‘Deformation transfer for triangle meshes’, *ACM Trans. Graph.* **23**(3), 399–405.
- Sumner, R. W., Zwicker, M., Gotsman, C. & Popović, J. (2005), ‘Mesh-based inverse kinematics’, *ACM Trans. Graph.* **24**(3), 488–495.
- Süßmuth, J., Winter, M. & Greiner, G. (2008), ‘Reconstructing animated meshes from time varying point clouds’, *Computer Graphics Forum* **27**(5), 1469–1476.
- Tagliasacchi, A., Delame, T., Spagnuolo, M., Amenta, N. & Telea, A. (2016), ‘3d skeletons: A state-of-the-art report’, *Computer Graphics Forum* **35**(2), 573–597.
- Tam, G. K., Cheng, Z.-Q., Lai, Y.-K., Langbein, F., Liu, Y., Marshall, A. D., Martin, R., Sun, X. & Rosin, P. (2013), ‘Registration of 3d point clouds and meshes: A survey from rigid to nonrigid’, *IEEE Transactions on Visualization and Computer Graphics* **19**(7), 1199–1217.
- Tam, G. K. L., Martin, R. R., Rosin, P. L. & Lai, Y.-K. (2014), ‘An efficient approach to correspondences between multiple non-rigid parts’, *Computer Graphics Forum* .
- Team, T. G. (2018), ‘Gimp’, <https://www.gimp.org/>.
- Technologies, U. (2018), ‘Unity3d’, <https://unity3d.com/>.
- Terzopoulos, D. & Fleiseher, K. (1988), ‘Modeling inelastic deformation: Viscoelasticity, plasticity, fracture’, *Computer Graphics* **22**, 269–278.
- Terzopoulos, D., Platt, J., Barr, A. & Fleischer, K. (1987), ‘Elastically deformable models’, *SIGGRAPH Computer Graphics* **21**(4), 205–214.
- Terzopoulos, D., Platt, J. & Fleischer, K. (1989), Heating and melting deformable models (from goop to glob), in ‘Proceedings of Graphics Interface ’89’, GI ’89, Canadian Man-Computer Communications Society, Toronto, Ontario, Canada, pp. 219–226.
- Tevs, A., Berner, A., Wand, M., Ihrke, I., Bokeloh, M., Kerber, J. & Seidel, H.-P. (2012), ‘Animation cartography - intrinsic reconstruction of shape and motion’, *ACM Transactions on Graphic* **31**(2), 12:1–12:15.
- Thomas, F. & Johnston, O. (1981), *The Illusion of Life: Disney Animation*.

- Tonnesen, D. (1991), Modeling liquids and solids using thermal particles, in ‘Graphics Interface ’91’, GI ’91, pp. 255–262.
- van Kaick, O., Zhang, H., Hamarneh, G. & Cohen-Or, D. (2011), ‘A survey on shape correspondence’, *Computer Graphics Forum* **30**(6), 1681–1707.
- Vestner, M., Litman, R., Rodolà, E., Bronstein, A. M. & Cremers, D. (2017), ‘Product manifold filter: Non-rigid shape correspondence via kernel density estimation in the product space’, *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* pp. 6681–6690.
- Visual Computing Institute, R. A. U. (2018), ‘Open flipper’, <https://www.openflipper.org/>.
- von Funck, W., Theisel, H. & Seidel, H.-P. (2006), ‘Vector field based shape deformations’, *ACM Trans. Graph.* **25**(3), 1118–1125.
- Wampler, K. (2016), ‘Fast and reliable example-based mesh ik for stylized deformations’, *ACM Trans. Graph.* **35**(6), 235:1–235:12.
- Wand, M., Adams, B., Ovsjanikov, M., Berner, A., Bokeloh, M., Jenke, P., Guibas, L., Seidel, H.-P. & Schilling, A. (2009), ‘Efficient reconstruction of nonrigid shape and motion from real-time 3d scanner data’, *ACM Trans. Graph.* **28**(2), 15:1–15:15.
- Wicke, M., Ritchie, D., Klingner, B. M., Burke, S., Shewchuk, J. R. & O’Brien, J. F. (2010), ‘Dynamic local remeshing for elastoplastic simulation’, *ACM Transactions on Graphics*.
- Williams, R. (2009), *The Animator’s Survival Kit: A Manual of Methods, Principles and Formulas for Classical, Computer, Games, Stop Motion and Internet Animators*, Faber & Faber.
- Wojtan, C. & Turk, G. (2008), ‘Fast viscoelastic behavior with thin features’, *ACM Transactions on Graphics* **27**(3), 47:1–47:8.
- Wyvill, B., Guy, A. & Galin, E. (1999), ‘Extending the csg tree. warping, blending and boolean operations in an implicit surface modeling system’, *Computer graphics forum* **18**(2), 149–158.
- Xu, D., Zhang, H., Wang, Q. & Bao, H. (2005), Poisson shape interpolation, in ‘Proceedings of the 2005 ACM Symposium on Solid and Physical Modeling’, SPM ’05, ACM, New York, NY, USA, pp. 267–274.

- Yang, J., Li, K., Li, K. & Lai, Y.-K. (2015), ‘Sparse non-rigid registration of 3d shapes’, *Comput. Graph. Forum* **34**(5), 89–99.
- Yang, Y., Ma, W., Yoshiyasu, Y. & Ouhyoung, M. (2013), Deformation transfer based on stretchiness ratio, *in* ‘2013 28th International Conference on Image and Vision Computing New Zealand (IVCNZ 2013)’, pp. 160–165.
- Yoshiyasu, Y., Ma, W.-C., Yoshida, E. & Kanehiro, F. (2014), As-conformal-as-possible surface registration, *in* ‘Proceedings of the Symposium on Geometry Processing’, SGP ’14, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp. 257–267.
- Zell, E. & Botsch, M. (2013), ‘Elasti face: Matching and blending textured faces’, *NPAR* pp. 15–24.
- Zhang, H., Sheffer, A., Cohen-Or, D., Zhou, Q., Van Kaick, O. & Tagliasacchi, A. (2008), ‘Deformation-driven shape correspondence’, *Computer Graphics Forum* **27**(5), 1431–1439.
- Zhang, Y., Paik, J., Koschan, A., Abidi, M. A. & Gorsich, D. (2002), Simple and efficient algorithm for part decomposition of 3-d triangulated models based on curvature analysis, *in* ‘Proceedings. International Conference on Image Processing’, Vol. 3.
- Zhao, X., Su, Z., Gu, X. D., Kaufman, A., Sun, J., Gao, J. & Luo, F. (2013), ‘Area-preservation mapping using optimal mass transport’, *IEEE Transactions on Visualization and Computer Graphics* **19**(12), 2838–2847.
- Zheng, Q., Sharf, A., Tagliasacchi, A., Chen, B., Zhang, H., Sheffer, A. & Cohen-Or, D. (2010), ‘Consensus skeleton for non-rigid space-time registration’, *Computer Graphics Forum* **29**(2), 635–644.
- Zhou, Y., Lun, Z., Kalogerakis, E. & Wang, R. (2013), ‘Implicit integration for particle-based simulation elasto-plastic solids’, *Pacific Graphics 2013* **32**.