Efficient Facial Animation Integrating Euclidean and Geodesic Distance-Based Algorithms into Radial Basis Function Interpolation

MIGUEL RAMOS CARRETERO

A thesis submitted in partial fulfilment of the requirements of

Bournemouth University for the degree of

Master's by Research in Media and Communication

(3D Computer Animation)

Bournemouth
University

Supervisors:

Prof. Lihua You

Prof. Jian Jun Zhang

April 2019

Copyright Statement

Abstract

Miguel Ramos Carretero

Efficient Facial Animation Integrating Euclidean and Geodesic Distance-Based Algorithms into Radial Basis Function Interpolation

Facial animation has been an important research topic during the last decades. There has been considerable effort on the efficient creation of realistic and believable facial expressions. Among various approaches for creating believable movements in human facial features, one of the most common utilises motion capture. This thesis explores the current approaches on facial animation with this technology together with Radial Basis Function (RBF) interpolation, covering a review of Euclidean and geodesic distance-based algorithms, and proposing a hybrid approach that tries to take the advantages of the two previous methods aided by pre-processed distance data to fasten the computations. Using motion capture performance based on the Facial Action Coding System (FACS), the results are then evaluated with a wide range of facial expressions in both a realistic and a stylised facial model. The findings of this thesis show the advantage of the hybrid RBF approach proposed which, combined with pre-processed distance data, results in a more efficient and more accurate process for the generation of high-detail facial animation with motion capture.

List of Contents

List of Figures

List of Tables

List of Algorithms

Acknowledgements

I would like to express my gratitude to all the people that helped me during the development of this project:

Thanks to Prof. Jian Jung Zhang, Dr. Mike Board, Zoe Leonard and the rest of the CDE team and colleagues for their support and for giving me the opportunity to accomplish this project.

Thanks to Dr. Zhidong Xiao and Edward Sedgley for their support and help with the motion capture technologies and the arrangement of the studio.

Thanks to Kamila Bianchi and Josh Sobel for the 3D facial models.

Thanks to Michelle Wu, Viktorija Sipilova, and Kamila Bianchi for helping me with the writing of the thesis report.

Thanks to my supervisor Prof. Lihua You for his invaluable help and for believing in my research skills when difficulties appeared.

Thanks to the people at the NCCA and the MA3D program for their assistance and their help in the artistic and technical aspects of computer animation, which are the roots of this research.

Thanks to all my friends in Bournemouth and around the world for being there whenever I needed it.

Special thanks to my mother, my father, my sister and my brother for their constant support and love.

Thank you all!

Author Declaration

I hereby declare that this thesis is all my work and that I have acknowledged and referenced properly all the material and media that supports the basis of this research. The work of this thesis has been wholly undertaken at Bournemouth University.

Miguel Ramos Carretero
Bournemouth, December 2018

Definitions

Euclidean distance: Given two points A and B in a 3D space, the Euclidean distance is the length of the AB segment (Deza and Deza 2009).

Facial Action Coding System (FACS): A dictionary for the classification of human facial movements and synthesis of facial expressions, compiled by Ekman and Friesen (1978).

Facial expression: Defined movements in the human facial features that conveys or suggests a certain mood or emotion (Oxford Dictionary 2019).

Geodesic distance: Given two points A and B within a surface S, the geodesic distance is the length of the shortest path along S from A to B (Deza and Deza 2009).

Motion Capture: The process of sampling and capturing the motion of humans, animals and other objects in parametrical terms (Kitagawa and Windsor 2012).

Radial Basis Function (RBF): A mathematical function whose result is dependent on the distance between a given origin point and another target point. Different distance functions would determine different outcomes of the RBF (Mongillo 2011).

Realistic: Within an art context, the recreation of something, being a person or an object, as true and accurate as seen in real life (Oxford Dictionary 2019).

Stylised: Within an art context, the recreation of something, being a person or an object, taking a free licence to simplify its details, usually with an emphasis on a particular style (Cambridge Dictionary 2019).

Chapter 1: Introduction

The portrayal of high-quality animation is of great importance for the creation of appealing characters in both realist and stylised contexts in feature films and video-games. The massive use of computer graphic for the recreation of fantastic and imaginary creatures in filmmaking and visual effects makes very demanding the investigation of new methods and efficient techniques that create believable animation, specifically when working within realistic contexts (Ping et al. 2013). Believable animation is achieved through the correct portrayal of emotion and the illusion of thought process in the characters, giving adequate traits and reactions to them in terms of body and facial motion (Lasseter 1987).

A key feature for believability in animation is the facial expression and the acting of the characters. The sensitivity of the human eye towards expressions makes the face the most observed feature and the most prone to be subject of critique, given the high amount of subtleties that can be found in each expression (Orvalho et al. 2012). Also, the complexity of the structure and the muscles of the face makes it hard to simulate it properly (Cong et al. 2016). The theory of FACS, a thorough study for the classification of facial movements and the synthesis of facial expressions (Ekman and Friesen 1978), serves nowadays as an important reference for animators to study the level of complexity of human face movements. Facial animation, therefore, presents itself as a challenging area within animation both for the artist and the technical departments. To achieve good results in this matter, great effort is put in the articulation process of the face (commonly known as rigging), so animators can have enough control to articulate the desire expressions in the characters. Recent advances in this area have reached levels in which the portrayal of complex facial expressions has been achieved successfully, both in realistic and stylised context (see Figure 1.1). However, current rigging techniques for the face require a considerable amount of effort and resources when working with realism and subtle expressions and, in many cases, they result in cumbersome controllers difficult for animators to handle, getting in the way of the animation process when trying to achieve certain expressions or convey subtle changes such as micro-expressions (Orvalho et al. 2012). Finding better techniques to facilitate this workflow in a more fluid and organic manner is an important subject to study in computer animation.

Within the most common techniques for the animation of facial features, the use of motion capture technology has been gaining popularity in the last two decades (Bastos 2015), particularly in the entertainment industry within the context of films and video-games. The collection of motion data from performers varies widely in terms of methods, and there are several techniques to map the resulting captured data in a facial model. Some of the most common ones are those based on RBF, a robust technique that maintains a high level of detail in the animation (Fidaleo et al. 2000). However, these methods are usually costly in terms of computation and, while some variants use more precise techniques to achieve high level of detail in the animation, they can become very heavy to process.

This research explores some of the most common methods of RBF for facial animation and introduces some approaches to fasten the computations and to achieve efficient results without compromising the quality of the animation. First, the Euclidean and the geodesics distance-based algorithms are reviewed, followed by a new proposal of a hybrid method that tries to combine the advantages of both algorithms and improve the results in a quantitative and qualitative manner. These three methods are then evaluated using motion capture performance based on the theory of FACS.

Figure 1.1: Still images from *Coco* (Disney Pixar 2017), *Avengers: Infinity War* (Marvel Studios 2018) and *Hellblade: Senua's Sacrifice* (Ninja Theory 2017).

1.1. Research Aims

This project focuses on the study and review of the state-of-the-art of facial animation within the field of computer graphics, with an especial emphasis on the most recent methods for achieving more realistic and believable performance in character animation with motion capture technologies. This work also investigates the recent approaches based on the use of RBF techniques to create facial animation with a wide range of expressivity and with flexibility to be adapted in models of diverse nature, either realistic or stylised. Focusing on Euclidean and geodesics distance-based methods, this thesis proposes a new approach combining these two methods to achieve a more efficient solution with good quality results. In addition to this, this thesis aims to assess the results using the theory of FACS as the main tool for the evaluation of facial expressions.

1.1.1. Main Objectives

- The implementation of RBF techniques based on Euclidean and geodesics distance-based algorithms. The development of a new RBF technique based on a hybrid approach between those two algorithms.
- The creation of a full library of motion capture data based on the theory of FACS, used for the evaluation of the RBF algorithms implemented.
- The application of the motion capture data collected into both a realistic and a stylised facial model using the RBF algorithms implemented.
- The implementation of a Maya plugin containing all the functionalities developed in this project.
- An evaluation of the outcomes and a study of possible applications within facial rigging and animation.

1.1.2. Research Questions

This thesis investigates two main research questions:

1) How can RBF methods be optimised for obtaining efficient surface deformation based on motion capture?

2) How can facial animation be enhanced by motion capture techniques based on RBF methods?

1.2. Contributions

The creation of high-detailed facial animation is a topic of great interest within the fields of computer graphics and entertainment, as well as in other applications where there is a need of high-quality simulation of realistic facial animation. The following sections explore some of the possible applications of this research, some more concrete, some more speculative:

1.2.1. Industry

The world of animation, visual effects and entertainment has a high demand on the use of better techniques for the creation of high-detailed believable facial animation in both realistic and stylised contexts. Some recent examples such as *Coco* (Disney Pixar 2017) or *Avengers: Infinity War* (Marvel Studios 2018) require outstanding facial performance for many of its CGI characters, such as grandmother *Coco* or the super-villain *Thanos* (see Figure 1.1). To maintain the believability of these characters in their own contexts, facial performance requires a high level of accuracy and attention to detail, giving the animators the necessary tools to convey a wide range of facial expressions with their characters. As stated by Lasseter (1987), the illusion of thought process must be achievable by the animators.

1.2.2. Academia

New methods for facial animation are being investigated within academia, and many studies have been developed in relation to the biomechanics of the face (Ho et al. 2018; Lucas 2017), with applications in diverse areas such as medical sciences and robotics (Parke and Waters 2008). The need for more realistic facial animation seems of high importance for these fields. Within this domain, alternatives to the ones approached in this thesis are also being investigated, and other techniques such as tissue deformation and muscle systems are being used in recent research to try to achieve the next level of accuracy in facial animation (Wu et al. 2014).

1.2.3. Societal

The understanding of the facial expressions and the use of theoretical research such as the theory of FACS (Ekman and Friesen 1978) allows for the continuation on the investigation of human psychology and body language, particularly in relation to the study of the face movements and the cultural and social implications that certain traits infer in different people. Being the face the main element of communication among humans, it is of key importance the understanding of it and how it works (Orvalho et al. 2012), and this research puts in practice the theory and principles of the theory of FACS.

1.3. Limitations

Given the nature of the research and its cross-boundaries with certain aspects of psychology and art, it is important to clarify that the thesis has its focus mainly within the domain of computer graphics, although certain terms and concepts have been taken from other areas of research (see list of definitions at the beginning of this report).

In addition to this, given the scope of this project, which has been developed under one year of study, there are certain aspects that will not be covered. As explained in the following chapter, a very prone branch of research in facial animation is the use of muscle systems and physic-based approaches for the calculation of the deformation of the face. Though these methods are widening the possibilities that can be achieved in facial animation, this thesis focuses mainly on the fast processes related to the generation of animation via motion capture technology and RBF techniques. It is also important to point out that this project does not focus on the portrayal of complex textures or rendering techniques for achieving realism, but rather on the investigation of movement in the face and how to achieve it in a more believable manner. Therefore, areas such as realistic texturing or the use of techniques such as displacements are not considered in this work.

1.4. Publications

As part of the work of this thesis, the following research items were presented to the academic community:

Ramos, M., Li-Hua, Y., Zhi-Dong, X. *FACS-based Facial Animation for Realistic and Stylized Characters aided by Motion Capture (poster).* Accepted for the CDE Digital Catapult London 2018.

Ramos, M., Li-Hua, Y., Zhi-Dong, X., Jian-Jun, Z. *z (journal paper).* Accepted for the ICGST-CVIP Journal 2019.

1.5. Thesis Overview

The following chapters are arranged as follow: Chapter 2 covers a literature review on the topic of facial animation and surface deformation techniques based on RBF and motion capture. Chapter 3 exposes the methodology followed for the successful development of this research, including chosen implementation and research methods. Chapter 4 covers the technical development of the project and the implementation of a facial animation plugin for Maya. Chapter 5 focuses on the implementation of each of the RBF techniques within the plugin, evaluating the results of each method in quantitative and qualitative terms. Chapter 6 presents some final conclusions on the outcomes and an outline of possible paths for future work.

Chapter 2: Literature Review

This chapter contains an overview of the state-of-the-art of the main topics related to this project. This review is presented as a base for a better understanding of the following chapters of this research. Divided in sections, the review covers areas related to: computer facial animation, MPEG4 standard for facial codification, Facial Action Coding System, surface deformation algorithms based on RBF and motion capture technologies for facial animation, respectively.

2.1. Computer Facial Animation

Computer facial animation is understood as the generation of a digital model, either 2D or 3D, resembling the attributes of a human face (Parke and Waters 2008). The next sections cover an overview on computer facial animation followed by the most common techniques developed:

2.1.1. Overview

The beginning of computer facial animation research has its origins in 1972, in the work of Parke (1972) with the proposal of the first computer generated animated face, in a period when computer graphics was starting to develop and to gain more attention. This pioneer attempt would open a new branch of research, leading to the development of new techniques for facial animation such as the work of Gillenson (1974), who proposed the first interactive system to create 2D facial images, and Platt (1980) on the first facial model controlled by muscle simulation. These highlights would lead to the development of further research and the creation of the first computer animated shorts depicting facial animation, such as *Tony de Peltrie* (University of Montreal 1985) and *Tin Toy* (Disney Pixar 1988). Since then, many approaches have been developed in relation to the creation of more realistic facial animation, including the first milestones in the area of medicine to simulate facial tissue and biomechanics on the face (Pieper 1991; Sifakis et al. 2005).

In recent years, the techniques for the recreation of realistic facial animation have reached high levels of fidelity, with examples such as the work of Alexander et al. (2010), with the photorealistic recreation of the animated face of an actor, or Ichim et al. (2017) on the use of physics-based simulation for the realistic animation of facial muscles and flesh. The state-of-the-art of this field offers now a wide range of solutions adapted to each of the possible scenarios in which facial animation plays a key role, both in the entertainment industry and in more academic and scientific applications. Continuous research effort is being invested on achieving more accurate results and better performance and expressivity in digital facial models.

2.1.2. Techniques

The development within computer facial animation has brought up a wide range of techniques for the recreation of digital faces. The following sub-sections cover some of the most common methods used nowadays: bone-based with blendshapes, physics-based and motion capture based.

Bone-Based and Blendshapes

In general terms, facial animation can be considered a special case for body animation, and in many cases the methods and solutions employed for body rigging can be applied to facial rigging. As such, a common practice in facial rigging involves the use of a bone-based rig to set the deformation of its main features, such as the cheeks, the eyelids and the jaw. Along with this, blendshapes allow the artists to sculpt certain features and details of the face that are difficult to achieve with bones, and these two techniques in combination are usually enough to reproduce a believable set of expressions with high efficiency (McLaughlin et al. 2011). In a higher level of detail, a related approach for facial animation consists on the use of a dictionary of blendshapes, sculpting in the model all the necessary expressions for the performance, and then creating the animation by shape interpolation (Orvalho et al. 2012) based on techniques such as bilinear and spline interpolation (Arai et al. 1996). A pioneer work in film is the case of the creature *Gollum,* from *The Lord of the Rings* (New Line Cinema 2001), in which a total of 675 blendshapes were used to reproduce all the range of possible expressions for the animation of his face (Orvalho et al. 2012). Though this method can achieve high fidelity in the performance and expressivity of the character, it is restricted to the range of sculpted expressions, and requires high computational resources.

Physics-Based and Muscle Systems

Physics-based techniques rely on heavy calculations and precise models for the animation of the face, attempting to simulate the elastic properties of facial tissue and muscles. The use of muscle systems is one of the most accurate methods within this area of facial animation, consisting on the simulation of the whole underlying muscle anatomy of the face, and calculating how it deforms the skin tissue on top (Parke and Waters 2008). In recent years, the film industry has started using these techniques more frequently, with remarkable examples such as *Kong* (Warner Bros 2017), in which muscle systems were used to help in the animation of the expressions (Cong et al. 2016; Cong et al. 2017). Physics-based methods have the advantage of being able to reproduce almost any achievable expression on the face, but it requires a considerable effort for the correct sculpting of the facial muscles and the proper simulation of the deformation of the skin, being this the main reason why its use is not very widespread yet.

Muscle systems have been used beyond the entertainment industry, and in the case of medical science additional techniques have attempted to simulate the face in a realistic manner for its use in surgery training and medical research. In this domain, physic-based systems are widely used, and the simulation of the face in layers (bone, muscle, fat and skin) is bringing new levels of realism and accuracy (Murai et al. 2017).

Motion Capture Based

Motion capture has become another common approach for the creation of computer facial animation, based on the collection of expression and performance from real actors. The recreation of movement from motion capture usually relies on the use of surface deformation techniques applied to the facial model and driven by the data collected. Among those, RBF is one of the most commonly used for its robustness and efficiency (Man-dun et al. 2014). A remarkable example of this technology can be seen in the real-time motion capture performance for *Hellblade: Senua's Sacrifice* (Ninja Theory 2017). However, this technology still requires considerable amount of efforts to set it up and clean it to make it ready for use, and facial nuances and micro expressions can still be difficult to achieve.

Technique	Pros	Cons
Bone-based	• Fast and efficient. • Intuitive (artist friendly). • Good for stylized models.	• Limitation in details. • Difficult to achieve realism by itself.
Blendshapes	• Can bring a wide range of accurate expressions. • Works well together with bone-based systems. • Works well for both stylized and realistic models.	• Computationally expensive. • Needs large collection of blendshapes to work well. • High level of artist effort.
Physics-based	• High level of accuracy (anatomically correct). • Able to simulate anatomical properties such as fat and skin tissue. • Good for realistic models.	• Computationally expensive. • Difficult to achieve realistic results • High level of technical effort.
Muscle systems	• High level of accuracy (anatomically correct). • Works well together with additional physics-based simulations (i.e. fat jiggling). • Good for realistic models.	• Difficult to achieve realistic results. • Needs large collection of muscles to work well. • High level of technical and artist effort.
Motion capture	• Based on actor performance. • Able to capture subtle movements. • Works well for both stylized and realistic models.	• Difficult to transfer data without losing accuracy. • Might introduce noise during capture process. • High level of technical effort.

Table 2.1: Pros and cons of different facial animation techniques.

2.2. MPEG-4 Standard

Within the field of computer facial animation, the MPEG-4 is widely referenced to aid with the parametrisation of the face. This was developed by the ISO/IEC Moving Picture Experts Group (MPEG) for the standardisation of the codification of audio and visual digital data (Abrantes and Pereira 1999). One of the parts of this standard corresponds to facial features and the main points that define the face.

Figure 2.1 shows a diagram with the parametrisation of the face in a set of points. As it can be observed, this standard describes the main points that define the face, and all facial movements can be translated to a set or subset of these points. Many applications, such as facial recognition or the use of motion capture technologies make wide use of this standard. As described in Chapter 4, the methods used in this project make use of the MPEG-4 standard for the gathering of facial motion data.

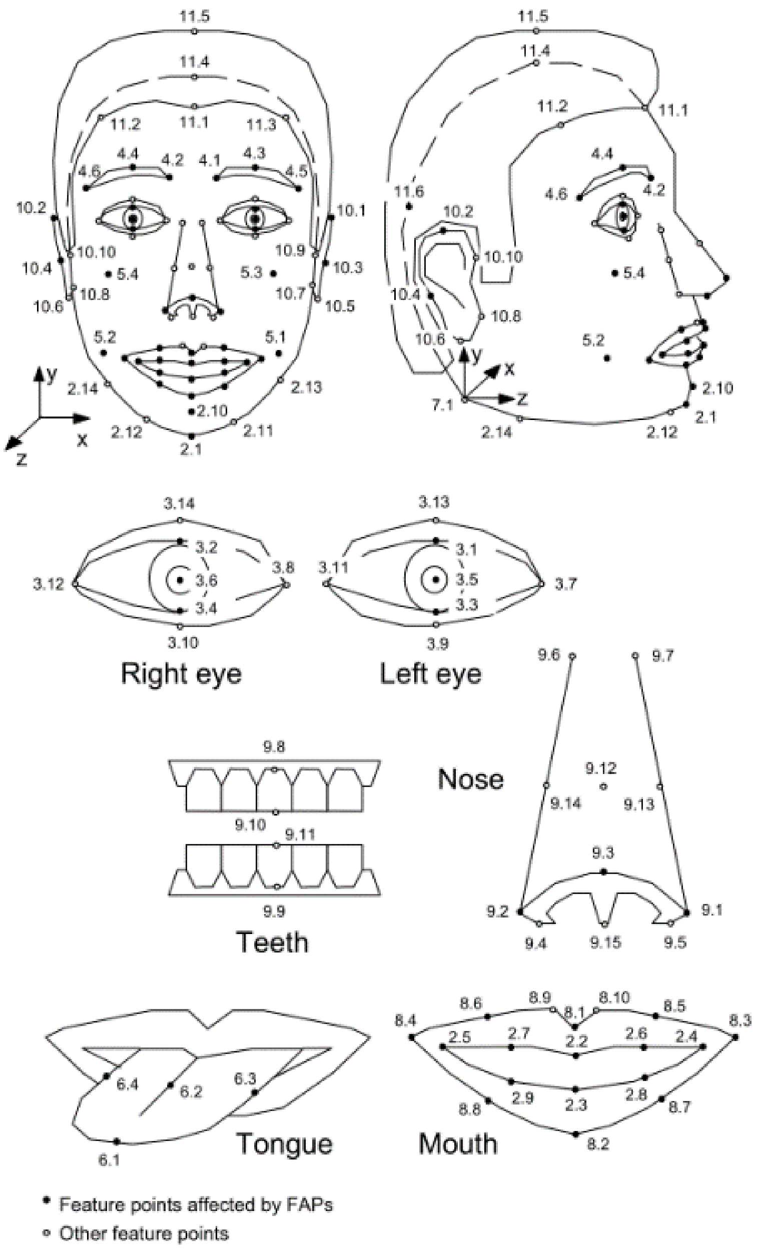


Figure 2.1: Feature-points grouping, as defined in the MPEG-4 standard. Black dots are facial animation parameters (FAP), and their description in the space define movements and expressions. White dots represent other feature points not directly related to facial movements but affected by the translational movement of the head (Abrantes and Pereira 1999).

2.3. Facial Action Coding System (FACS)

Within the field of psychology, the theory of FACS is the result of a research led by Ekman and Friesen (1978) to create a dictionary of facial movements and its categorisation in several action units (AU) according to the muscle areas and parts of the face used. This led the research for the exploration of the universality of all human expressions (Ekman and Rosenberg 1997), and it comprises in one single list how the different human facial features are used in combination to recreate facial movements. Table 2.1 lists a selection of some of the main FACS units, classified and divided by the author in six categories. As described in Chapter 4, the motion capture for facial animation of this project has a strong base on the theory of FACS.

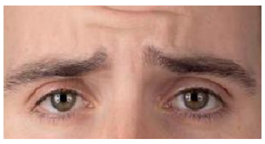
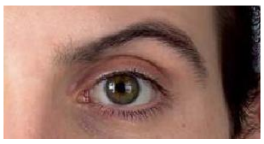
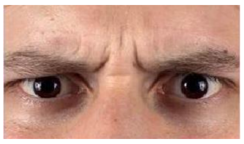
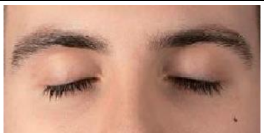
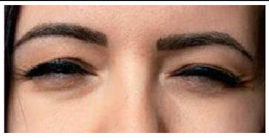
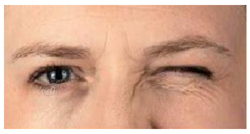
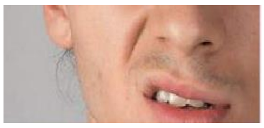
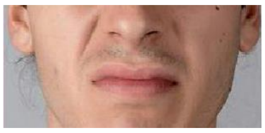
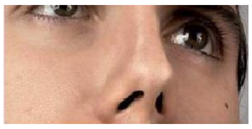
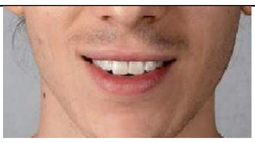
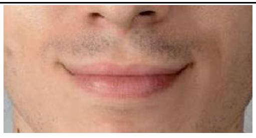
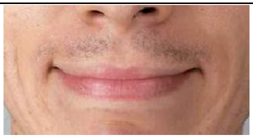
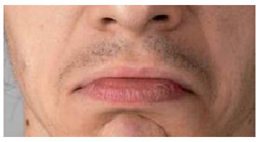
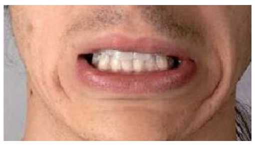
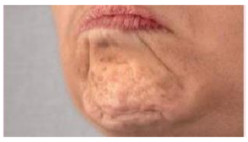
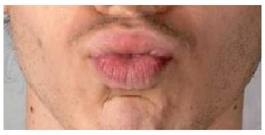
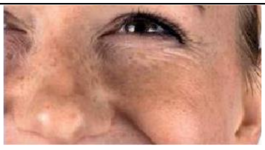
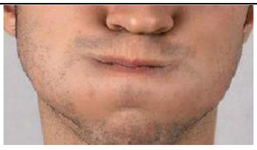
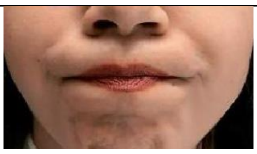
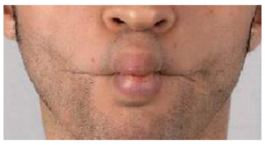
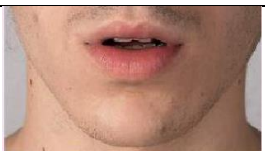
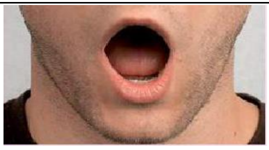
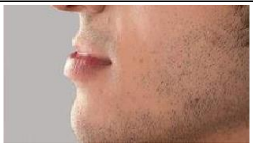
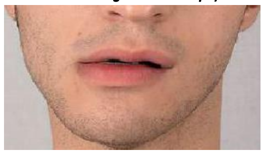
Eyebrows	 AU1 (*inner brow raiser*)	 AU2 (*outer brow raiser*)	 AU4 (*brow lowerer*)
Eyes	 AU43 (*lid drop*)	 AU44 (*squint*)	 AU46 (*wink*)
Nose	 AU10 (*upper lip raiser*)	 AU11 (*nasolabial deep*)	 AU39 (*nostril compress*)
Lips	 AU12 (*lip corner puller*)	 AU13 (*sharp lip puller*)	 AU14 (*dimpler*)
	 AU15 (*lip corner depress*)	 AU16 (*lower lip depressor*)	 AU17 (*chin raiser*)
	 AU18 (*lip pucker*)		
Cheeks	 AU6 (*cheek raiser*)	 AU33 (*check blow*)	 AU34 (*cheek puff*)
	 AU35 (*cheek suck*)		
Jaw	 AU26 (*jaw drop*)	 AU27 (*mouth stretch*)	 AU29 (*jaw thrust*)
	 AU30 (*jaw sideways*)		

Table 2.2: Selection of Facial Acting Coding System Units, extracted from Zarins (2017).

2.4. Radial basis function for surface deformation (RBF)

As a generic term, surface deformation algorithms deal with the definition of mathematical and physical processes to calculate the changes in shape of a given surface according to a certain input. These algorithms have wide applications in several fields, and it is at the core of 3D computer animation. Both for cloth or skin, these algorithms are the base layers from which animation software is built for character animation, and they rely strongly in the use of physic and geometrical calculations.

RBF methods are one of the most common algorithms to synthesise surface deformation. Proposed first by Hardy (1971), these techniques approximate the deformation of a surface considering a set of points that control how the deformation takes place. The algorithm for RBF depends on a basis function for the interpolation of data, using the distance between points as the main parameter. These functions can take different forms, being the most commons those based on Gaussian and Multiquadric functions (Mongillo 2011).

2.4.1. Algorithm overview

The RBF works in the following manner: given a geometrical mesh representing a surface and a set of control points over it, a function is defined to satisfy the displacement D_j of these control points:

$$D_j = f(m_j)\,,\ (0\ \leq j\ \leq M-1)\ \text{(2.1)}$$

$$f(m_j)\ =\ \sum W_j \cdot \varphi(\|m_j - m_k\|)\,,\ (0\ \leq\ j,\ k\ \leq M-1)\ \text{(2.2)}$$

where *M* is the total number of control points, m_j is the 3D coordinates of control point j and $\varphi(\|m_j - m_k\|)$ is the RBF function, being $\|m_j - m_k\|$ the norm of the distance between the two points. As stated previously, RBF functions can take different forms and the norm can also vary. One of the most common RBF approaches is the Gaussian technique (Eq. 2.3a), but others are also used such as the Multiquadric (Eq. 2.3b) or the Thin Plate Spline (Eq. 2.3c):

$$\varphi(\|m_j - m_k\|) =\ e^{-\|m_j - m_k\|/\gamma}\ \text{(2.3a)}$$

$$\varphi(\|m_j - m_k\|) =\ \sqrt{1 + (\|m_j - m_k\|)^2}\ \text{(2.3b)}$$

$$\varphi(\|m_j - m_k\|) =\ (\|m_j - m_k\|)^2\ \ln(\|m_j - m_k\|)\ \text{(2.3c)}$$

Considering *D* the vector of displacements and being *Φ* the resulting matrix of distances (Eq. 2.4), the vector of weights *W* is computed in Eq. 2.5:

$$D\ =\ \Phi \cdot W\quad \text{(2.4)}$$

$$W\ =\ \Phi^{-1} \cdot D\quad \text{(2.5)}$$

Finally, the weights W_j are used to compute the displacement of every vertex (Dv_i) of the surface in Eq. 2.6:

$$Dv_i = \sum W_j \cdot \varphi(\|v_i - m_j\|)\,,\ (0\ \leq\ j\ \leq M-1)\,,\ (0\ \leq\ i\ \leq N-1)\ \text{(2.6)}$$

where N is the total number of vertices.

The application of the vector displacement Dv_i in every vertex i results in the surface deformation of the given mesh (see Figure 2.2).

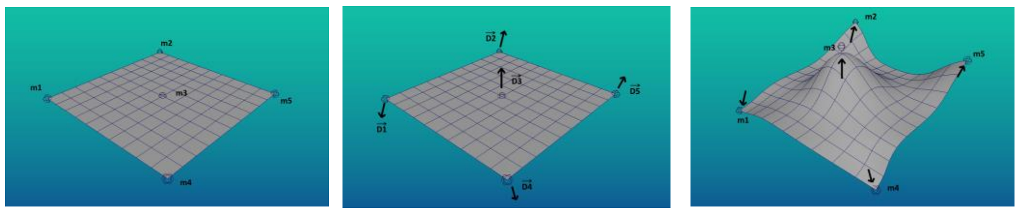


Figure 2.2: Illustration that exemplifies how the RBF algorithm works: once the markers are set on the mesh (left) and their displacements defined (middle), the RBF can calculate the displacement of each vertex of the mesh (right).

2.4.2. Euclidean and geodesic distance functions

In addition to the previous equations, there is a need of a specific norm to calculate distances. Traditional methods use the Euclidean norm: the geometrical distance in a 3D space between two points (straight-line distance, see Figure 2.3), which allows for simple and fast calculations, and gives enough accuracy for many applications with planar continuous surfaces.

Specific for computer facial animation, the geodesic norm has been proposed by several authors as a more accurate alternative to the Euclidean norm (Wan et al. 2012). This method obtains the distance of two points calculating the shortest path between them on a given surface. The calculation of geodesics has been object of research for many decades, and there have been many attempts to approximate a good solution for this problem (Jongmin Baek 2007). One of the approximations for the geodesic problem is the calculation of the mesh-based shortest path, based on the A-Star heuristic of Dijkstra’s search algorithm (Dijkstra 1959; Hart et al. 1968). Figure 2.3 shows how the geodesic distance algorithm calculates distances more accurately in non-planar non-continuous surfaces that present curvatures and holes.

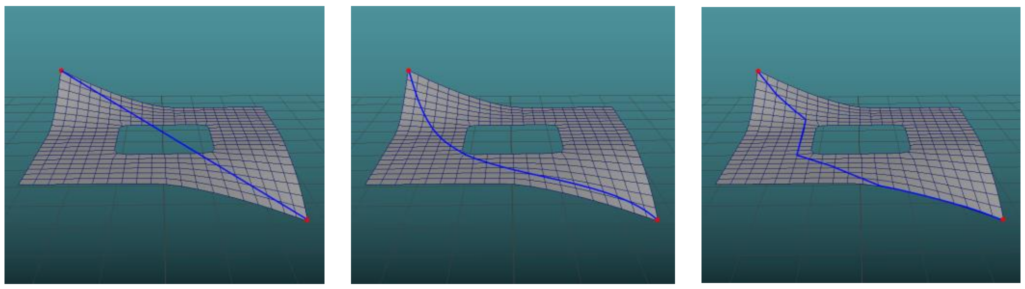


Figure 2.3: Application of distance norms in a non-planar non-continuous surface: Euclidean (left), continuous geodesics (centre) and mesh-based geodesics (right).

2.5. Motion Capture Technologies

Motion capture can be briefly defined as the process of sampling and capturing the motion of humans, animals and other objects in parametrical terms (Kitagawa and Windsor 2012). Next sections present an overview on the background of motion capture and its main techniques:

2.5.1. Overview

The first attempts of motion capture can be traced back prior the birth of computing: the invention of the zoopraxiscope in the late nineteenth century is marked as the beginning of the exploration in this area (Muybridge 1882). The invention of motion pictures at the beginning of the twentieth century would then lead to the implementation of new motion capture techniques, such as the invention of the rotoscope by Max Fleischer, a device that would allow artists to use real footage as a template to find the motion in the characters of their drawings (Menache 2000). The latter would be used by the first feature films of Walt Disney when attempting to recreate human features such as in *Snow White and the Seven Dwarfs* (Disney 1937). With the development of computer graphics in the 1970s, the research on digital motion capture gained more interest and resulted in its first uses in medicine and military applications (Kitagawa and Windsor 2012). Successful uses of this technology within the film industry would appear in the late 1990 and the beginning of the twentieth-first century with the spread of CGI, with remarkable examples such as *Gollum* from *The Lord of the Rings* (New Line Cinema 2001-2003) and the *Na'vi* creatures in *Avatar* (20th Century Fox 2009).

The development of motion capture has seen an increasing interest in recent years given to the wide variety of purposes in which is applied. Particularly for the case of computer facial animation, there has been recent research in the methods and techniques for the creation of facial animation from motion capture. Relevant research related to this project refers to the work of Ruhland et al. (2017), in which a new method is proposed for the synthesis of refined animation directly from motion capture data, and the work of Man-dun et al. (2014), in which an RBF technique is presented together with the capture of the face texture for the creation of realistic facial animation.

2.5.2. Techniques

The research and development of motion capture has spawned different type of technologies. Among those, optical and magnetic technologies are the ones that are more widely used (Kitagawa and Windsor 2012). The following sub-sections present an overview on them:

Marker-based optical motion capture

Optical systems capture the movement with a collection of cameras, usually arranged in circle, and a computer that synchronises and gathers the data captured. In these systems, the performers wear suits with reflective or light emitter markers that can be captured by the cameras (Guerra-Filho 2005). Light is an important component in these systems and needs to be treated carefully for the successful capture of the markers. Optical systems usually offer good accuracy in the data captured, which makes them particularly suitable for the capture of facial features and other subtle details of the performance (Herda et al. 2001). The main disadvantage of these system is the need of extensive post-processing to interpret the data correctly, demanding high technical expertise in order to get successful results (Kitagawa and Windsor 2012).

Magnetic motion capture

Magnetic systems capture the movement with a set of magnetic sensors. These sensors track the translations and the rotations during the performance, sending the information in real time to a central computer (O'Brien et al. 1999). This allows for faster post-processing, since the data is already identified from each sensor. One of the main disadvantages is their susceptibility to magnetic or electrical interferences, demanding the use of multi-channel techniques as proposed by Hashi et al. (2006). These limitations make them less suitable for facial capture, but they still perform well for the case of body animation (Menache 2000).

Markerless optical motion capture

These systems rely heavily on structured light pattern techniques, depth-cameras or RGB image processing, capturing the movement from an array of cameras and the silhouette of the performer, without requiring markers (Mündermann et al. 2006). As with marker-based optical techniques, light is an important factor for these systems. The major advantage is the versatility that they offer, being used in wider contexts and being more accessible for entertainment purposes, such as the case of Microsoft's *Kinect* (Zhang 2012). Their main disadvantage is the lack of accuracy in details like hands, fingers and facial features. Some authors have investigated on this issue, such as Shotton et al. (2011), who proposes post-processing the data with a database of poses and shapes to enhance the results, or Sundaresan and Chellappa (2005), who suggests the use of multiple calibrated depth sensors to increase accuracy.

Other systems

Other systems attempt to use mechanical technology for motion capture, such as exoskeletons or the use of ultrasonic systems, but their use is generally not suitable for the case of facial animation (Guerra-Filho 2005).

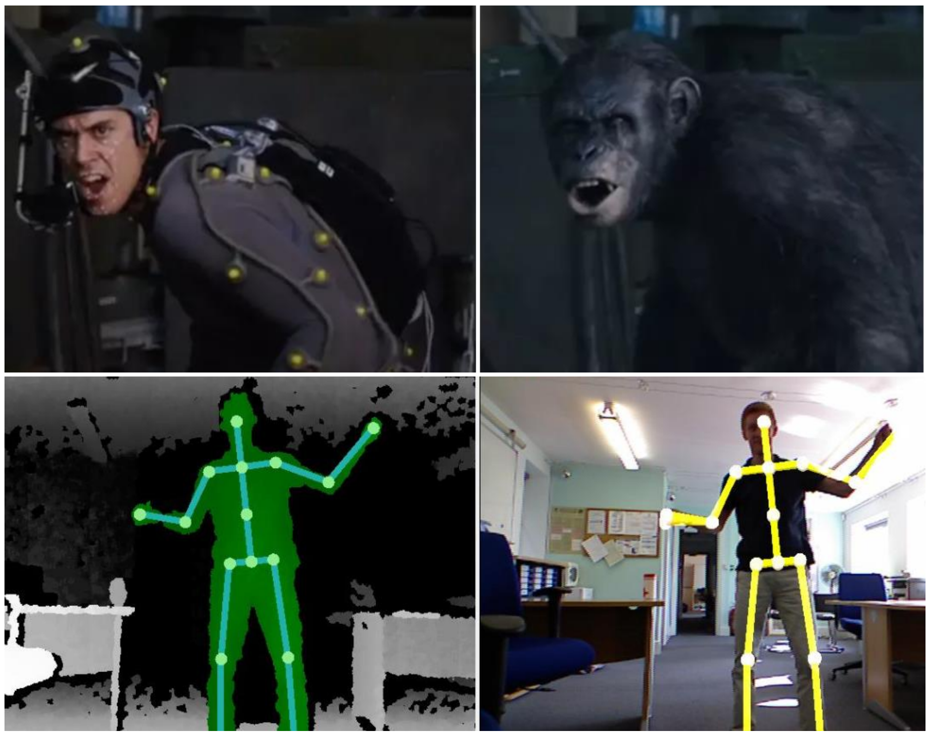


Figure 2.4: Above: Example of marker-based motion capture system for facial and body animation from *Dawn of the Planet of the Apes* (20th Century Fox, 2014). Below: Markerless motion capture system with *Kinect* technologies (Microsoft, 2010).

2.6. Summary

Chapter 2 has presented the main theoretical aspects that build the foundations of this project, covering an overview on computer facial animation, the current standards for the codification of facial features (MPEG-4 and FACS), an introduction to the RBF surface deformation and a quick overview on motion capture technologies.

The following chapter will cover the methodology followed for the development of this research.

Chapter 3: Research Methodology

This chapter focuses on the methodology followed for the development of this project, including the process used to keep track of the research progress, the approach for the software implementation and the research methods.

3.1. Specifications

Given the large amount of work required for this project, some guidelines and milestones were defined to allow for reflection, control and self-evaluation.

3.1.1. Milestones

On the early stages, the overall goals of the research were set, defining certain limitations and boundaries as well to prevent the project becoming unmanageable and out of scope. A specification document was set with this information, containing the main goals divided in three milestones:

Milestone I

- Document with a literature review covering the current state-of-the-art of facial animation.
- Analysis of the tools provided by Maya to approach the software development.
- Database of facial motion capture created using the theory of FACS as the base of the performance.
- Acquisition of high-poly realistic and high-poly stylised 3D models.
- First prototype of the plugin, covering the following functionalities:
    - Markers creation for the 3D model.
    - Markers calibration with motion capture data.

Milestone II

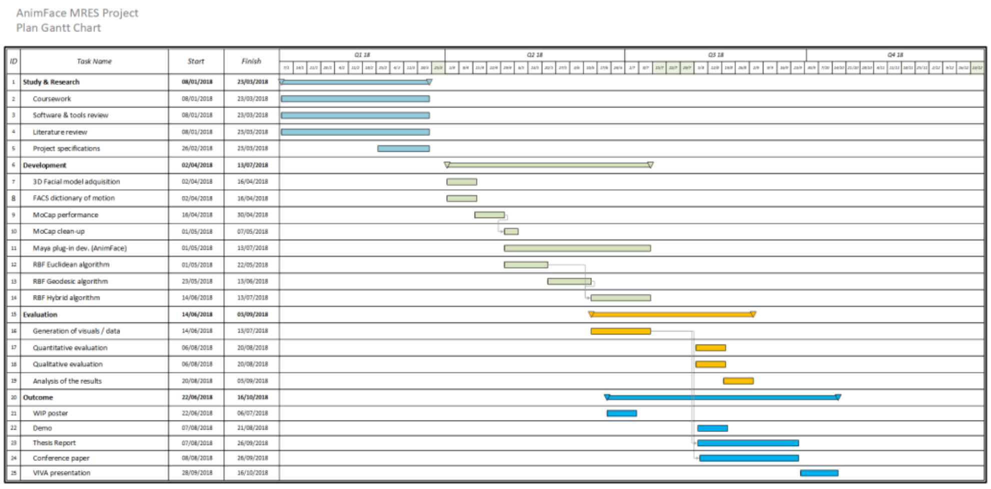
- Technical review of several methods for the mapping of motion capture data into a facial mesh.
- Second prototype of the plugin, covering the following functionalities:
    - Euclidean RBF algorithm for the transferral of motion capture on a facial mesh.
    - Geodesic RBF algorithm for the transferral of motion capture on a facial mesh.
- Poster showcasing the work-in-progress of the research at the CDE Digital Catapult Event.

Milestone III

- Third prototype of the plugin, covering the following functionalities:
    - Hybrid algorithm for the transferral of motion capture on a facial mesh.
    - Parametrization of variables in the algorithms (stiffness, frames, steps, etc.).
    - Artist-friendly Graphic User Interface for the use of the plugin in Maya.
- Demo reel showcasing an overview of the project.
- Conference paper to present the outcome to the research community.

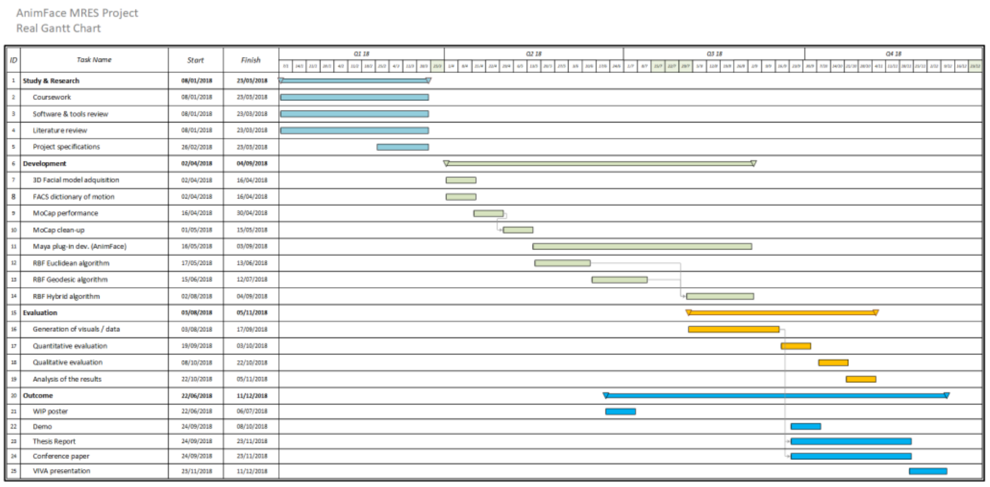
3.1.2. Gant chart diagrams

These diagrams show the planning and track of the project progression. The first Gantt chart (Figure 3.1) shows the planned schedule of the research. The second Gantt chart (Figure 3.2) shows the real timing. As it can be appreciated, the initial progress was followed according to plan, being the late stage of the project, (implementation) the one that took some weeks longer than originally planned:



ID	Task Name	Start	Finish
1	**Study & Research**	**08/01/2018**	**23/03/2018**
2	Coursework	08/01/2018	23/03/2018
3	Software & tools review	08/01/2018	23/03/2018
4	Literature review	08/01/2018	23/03/2018
5	Project specifications	26/02/2018	23/03/2018
6	**Development**	**02/04/2018**	**13/07/2018**
7	3D Facial model adquisition	02/04/2018	16/04/2018
8	FACS dictionary of motion	02/04/2018	16/04/2018
9	MoCap performance	16/04/2018	30/04/2018
10	MoCap clean-up	01/05/2018	07/05/2018
11	Maya plug-in dev. (AnimFace)	01/05/2018	13/07/2018
12	RBF Euclidean algorithm	01/05/2018	22/05/2018
13	RBF Geodesic algorithm	23/05/2018	13/06/2018
14	RBF Hybrid algorithm	14/06/2018	13/07/2018
15	**Evaluation**	**14/06/2018**	**05/09/2018**
16	Generation of visuals / data	14/06/2018	13/07/2018
17	Quantitative evaluation	06/08/2018	20/08/2018
18	Qualitative evaluation	06/08/2018	20/08/2018
19	Analysis of the results	20/08/2018	05/09/2018
20	**Outcome**	**22/06/2018**	**16/10/2018**
21	WIP poster	22/06/2018	06/07/2018
22	Demo	07/08/2018	21/08/2018
23	Thesis Report	07/08/2018	26/09/2018
24	Conference paper	08/08/2018	26/09/2018
25	VIVA presentation	28/09/2018	16/10/2018

Figure 3.1: Planned Gantt chart of the project.



ID	Task Name	Start	Finish
1	**Study & Research**	**08/01/2018**	**23/03/2018**
2	Coursework	08/01/2018	23/03/2018
3	Software & tools review	08/01/2018	23/03/2018
4	Literature review	08/01/2018	23/03/2018
5	Project specifications	26/02/2018	23/03/2018
6	**Development**	**02/04/2018**	**04/09/2018**
7	3D Facial model adquisition	02/04/2018	16/04/2018
8	FACS dictionary of motion	02/04/2018	16/04/2018
9	MoCap performance	16/04/2018	30/04/2018
10	MoCap clean-up	01/05/2018	15/05/2018
11	Maya plug-in dev. (AnimFace)	16/05/2018	03/09/2018
12	RBF Euclidean algorithm	17/05/2018	13/06/2018
13	RBF Geodesic algorithm	15/06/2018	12/07/2018
14	RBF Hybrid algorithm	02/08/2018	04/09/2018
15	**Evaluation**	**03/08/2018**	**05/11/2018**
16	Generation of visuals / data	03/08/2018	17/09/2018
17	Quantitative evaluation	19/09/2018	03/10/2018
18	Qualitative evaluation	08/10/2018	22/10/2018
19	Analysis of the results	22/10/2018	05/11/2018
20	**Outcome**	**22/06/2018**	**11/12/2018**
21	WIP poster	22/06/2018	06/07/2018
22	Demo	24/09/2018	08/10/2018
23	Thesis Report	24/09/2018	23/11/2018
24	Conference paper	24/09/2018	23/11/2018
25	VIVA presentation	23/11/2018	11/12/2018

Figure 3.2: Real Gantt chart after completing the project

3.2. Research Methods

Prior starting the project, a literature review was covered based on the areas of interest related to computer facial animation and rigging. This review took over a couple of months to complete, being an essential step to understand the overall state-of-the-art and the possibilities for new paths of development. A document was created, which evolved into the Chapter 2 of this thesis.

3.2.1. Research for Computational Modelling

After the literature review was completed, three articles were selected as the main pillars of the research:

- *Fast Individual Animation Based on Motion Capture Data* (Man-dun et al. 2014): A semi-automatic facial animation technique based on the RBF techniques with

Euclidean norm, using the bases of the MPEG4 standard to set the position of the markers in the face. The captured texture of the performer is mapped it in the facial model, attempting to achieve a realistic digital face.

- *Geodesics Distance-Based Realistic Facial Animation Using RBF Interpolation* (Wan et al. 2012): A variation on the approach of RBF algorithms, this research covers the implementation of the calculation of geodesic distances in a mesh, being used for the calculations of the RBF. The results show that the geodesic works better than standard Euclidean distances, especially in areas such as the mouth and the eyes.
- *A FACS Validated 3D Human Facial Model* (Cosker et al. 2010): A method for the creation and evaluation of a scanned 3D facial model using the theory of FACS to validate the captured data.

3.2.2. Experimentation

For the experimentation, the evaluation of the results was approached using both qualitative and quantitative methods. A comparative evaluation was carried out by the author, comparing the real footage of the FACS performance with the final animation outcomes. For the quantitative evaluation, the following parameters were considered when dealing with the analysis and comparison of results:

- Number of vertices in the model
- Number of frames simulated
- RBF processing time
- Vertex displacement time (ignoring key-framing or other processes from Maya)

3.3. Implementation Methods

In parallel to the theoretical research carried out during this project, a considerable amount of work was invested in the development of the algorithms to generate the facial animation in the 3D models from motion capture data. For this reason, certain software engineering methods were followed along with the specification of the programming tools needed for the successful development of the code.

3.3.1. Computational Modelling

The algorithms for facial animation were developed under six separate Python scripts (see Chapter 4), each building a functionality whose outcome would be the input for the next one. An incremental-iterative approach was used in the development, using Agile Methodologies that allowed for more flexible programming and quick feedback.

3.3.2. Off-The-Shelf Components and Software

The computational models for facial animation were fully developed under Maya 2017 API with the programming language Python, using some added functionalities from MEL (Maya Embedded Language). The SDK tools of this software were used to embed the program into the core system of Maya, allowing for better performance and faster results.

For the facial meshes, two models were acquired, one resembling a realistic face, and another one representing a more stylised human face. The geometry of both models was reduced to make calculations faster without compromising the level of detail.

The motion capture data was acquired by the own author’s performance using motion capture equipment and the software programs Cortex and Motion Builder, which allowed for the manipulation the motion capture data from Maya. Chapter 4 will cover more details about the motion capture process.

3.4. Summary

This chapter has covered the work done for the correct management and track of the project, and the chosen methodologies in terms of software implementation and research.

Next chapter presents the project development, including the process for motion capture acquisition and the implementation of the Maya plugin.

Chapter 4: Project Development

This chapter focuses on the overall pipeline for the project development, from the preliminary phase, where the performance for motion capture based on FACS took place, to the development of the software pipeline to transfer the motion capture data to the realistic and stylised 3D facial models.

4.1. Preliminary steps

Prior to implementing the algorithms for facial animation, two preliminary steps were required: first, to acquire a sample of suitable facial 3D models where the algorithms could be applied and, second, to gather motion capture data following the theory of FACS (see Chapter 2).

4.1.1. Gathering 3D Models

Two humanoid 3D models were used for the mapping of the motion capture, one being realistic and another one stylised (see Figure 4.1). The topology of both models was cut around the limits of the facial features, hiding other parts such as the ears, the neck and the back of the head. The models were modified and reduced to be under 4000 vertices, attempting to keep a high level of detail without compromising the performance for the mapping algorithms. The realistic face was set up to 2746 vertices and the stylised one to 3774.

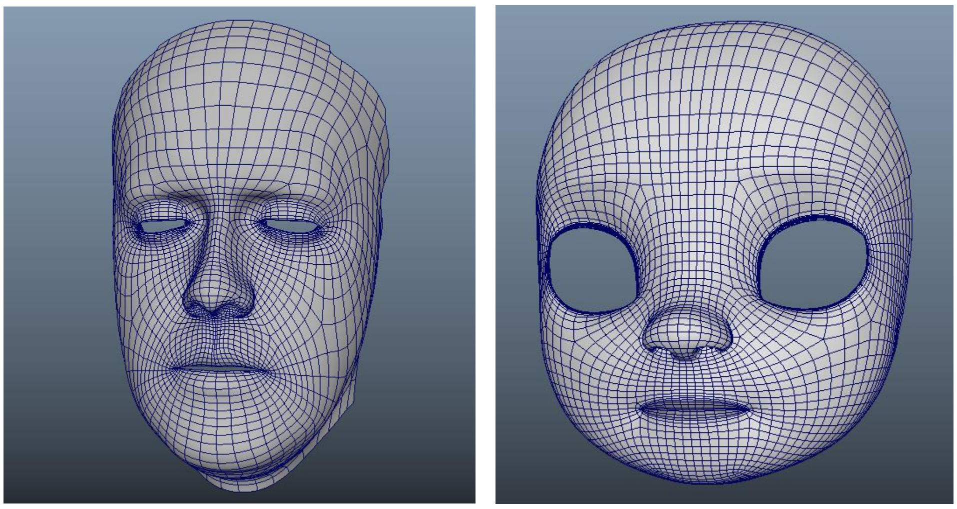


Figure 3.1: Images of the realistic face (left, 2746 vertices) and the stylised face (right, 3774 vertices), after extracting them from the main models and reducing their topology.

4.1.2. Capturing the motion of the face

The process of collecting the motion capture data was developed following the theory of FACS (Ekman and Friesen 1978), a descriptive collection of facial movements that comprises all the possible motions of the human facial muscles (see Chapter 2). In order to simplify the performance, the acting of the movements was divided into 6 separate groups: eyebrows, eyes, nose, lips, cheeks and jaw (see Chapter 2, Table 2.2 to find the FACS units corresponding to each group). The author acted as the subject for the performance of each of the movements. Every instance was repeated twice, and the best among the two was chosen for post-processing.

The setting of the motion capture equipment is shown in Figure 4.2. The chosen motion capture technology was based on an optical system (see Chapter 2) due to its flexibility and the high-level of detail that is able to capture, which was essential to obtain the details of the facial expressions. The set-up of the system consisted of 12 infrared cameras set around in a circle of 5 meters of diameter, with the subject in the centre. In addition to this, an extra regular camera was positioned in front of the subject to record the experiment for reference purposes and for the qualitative evaluation (see Chapter 5). Following the guidelines of the MPEG4 standard (see Chapter 2, Figure 2.1), 41 reflective markers were placed onto the subject's face, with the aim of capturing the motion in the most relevant areas, including the eyelids. Figure 4.3 shows the arrangement of the motion capture markers on the face. Table 4.1 shows the id numbers of the markers and their given tag.

After the session was completed, the motion capture data was cleaned up with the software *Cortex Motion Analysis* to fix the broken trajectories and to establish the relationship between the data point and the facial markers. The resulting data was exported into a 3D data file for its use in Autodesk Maya 2017.

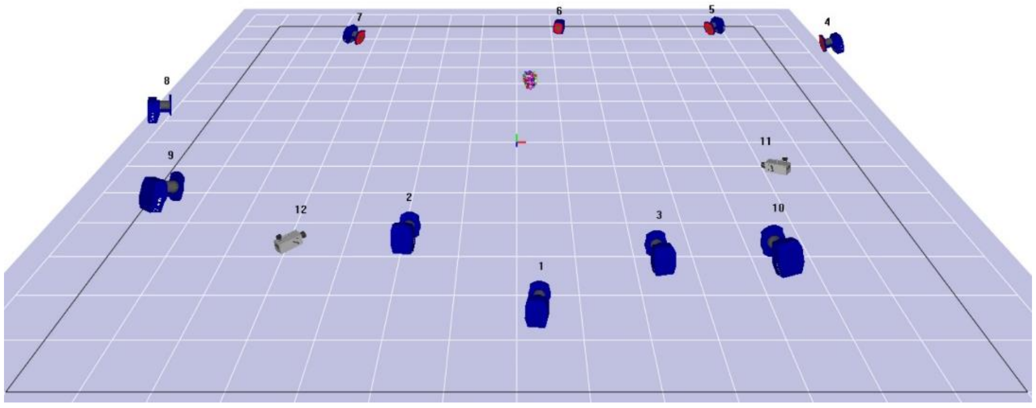
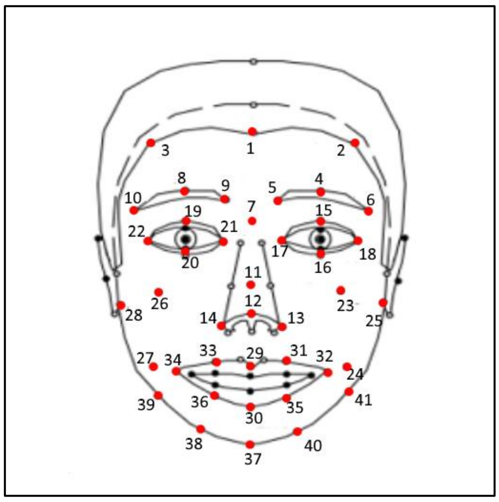


Figure 4.2: Diagram of the motion capture system set-up



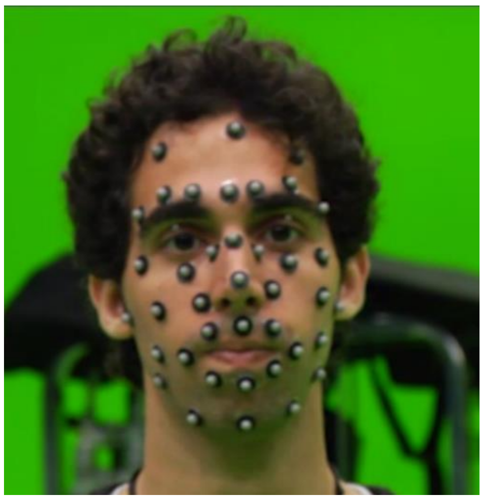


Figure 4.3: Images showing the layout of the markers on the MPEG-4 diagram with id numbers (left) and the arrangement of the markers in the performer (right).

1	*ForeHead_M*	**12**	*NoseHead*	**23**	*Lcheek_Up*	**34**	*Mouth_Rend*
2	*ForeHead_L*	**13**	*NoseHead_L*	**24**	*Lcheek_Down*	**35**	*Mouth_DownL*
3	*ForeHead_R*	**14**	*NoseHead_R*	**25**	*Lcheek_Outside*	**36**	*Mouth_DownR*
4	*Leyebrow_M*	**15**	*Leye_Up*	**26**	*Rcheek_Up*	**37**	*Chin*
5	*Leyebrow_Inside*	**16**	*Leye_Down*	**27**	*Rcheek_Down*	**38**	*Chin_R1*
6	*Leyebrow_Outside*	**17**	*Leye_Inside*	**28**	*Rcheek_Outside*	**39**	*Chin_R2*
7	*Eyebrow_Center*	**18**	*Leye_Outside*	**29**	*Mouth_Up*	**40**	*Chin_L1*
8	*Reyebrow_M*	**19**	*Reye_Up*	**30**	*Mouth_Down*	**41**	*Chin_L2*
9	*Reyebrow_Inside*	**20**	*Reye_Down*	**31**	*Mouth_UpL1*		
10	*Reyebrow_Outside*	**21**	*Reye_Inside*	**32**	*Mouth_Lend*		
11	*NoseTop*	**22**	*Reye_Outside*	**33**	*Mouth_UpR1*		

Table 4.1: Id numbers and tags for each facial marker

4.2. Software architecture

Once the motion data was captured and processed to be used in Maya, the next step required the proper transfer of the data for animating the 3D facial models, trying to keep as much information as possible. With this goal in mind, several steps were implemented in order to transfer the data of the motion capture markers to others positioned in the 3D models, and then applying the deformation algorithms to transfer the animation to them. To facilitate this process, a plug-in tool was designed for Maya 2017 named *AnimFace*.

A list of requirements was written to outline all the necessary functionalities for the algorithms to run properly:

- *Component A:* Functionality to create and group a set of markers that can be attached to a 3D facial model.
- *Component B:* Functionality to be able to transfer motion capture data onto the face markers.
- *Component C:* Functionality to be able to apply the animation of the markers onto a 3D facial model, using RBF surface deformation algorithms to build the facial features accordingly.
- *Component D:* Functionality to be able to calculate distances with different norms and be able to store them in distance matrices.
- *Component E:* A user graphic interface able to orchestrate all parameters and all functionalities from a single UI window.

Based on these main requirements, the software implementation was divided in five separate components. Each component was implemented by one or more Python scripts (see Figure 4.4). To make the scripts easily usable in Maya as a standalone application, all the components were organised together under a single user interface. The script *animFace_UI_plugin.py* orchestrates all the functionalities. Figure 4.5 shows a screenshot of this interface. As it can be noticed, the

UI includes several parameter options as well as possibilities of customising the frame range to run the calculations. As described in the following section, an optimisation for distance calculations was proposed with the use of distance matrices, which is included in one of the components.

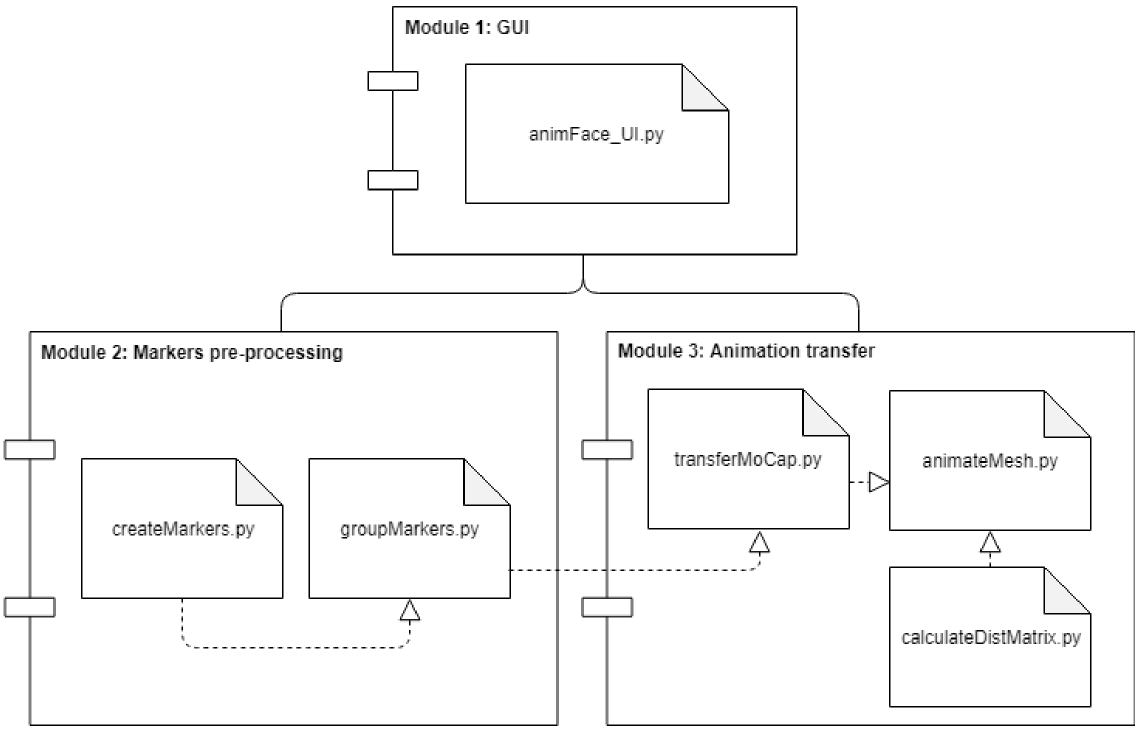
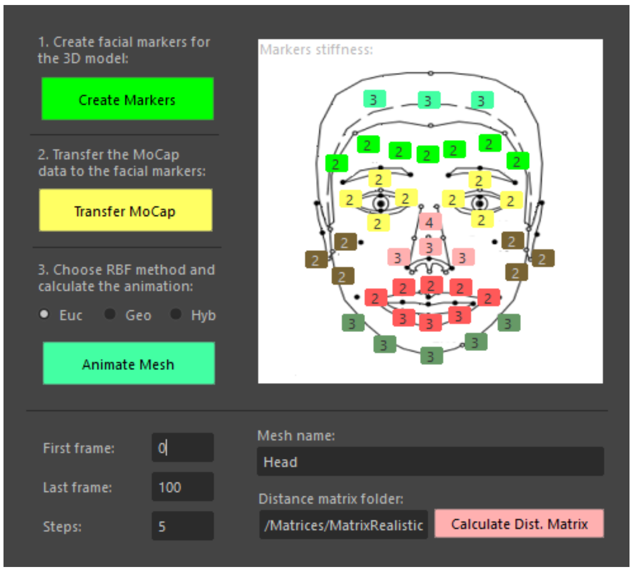


Figure 4.4: UML Component Diagram of the *AnimFace* plugin. Module 1 contains the Graphic User Interface functionality (component E). Module 2 contains the functionality for marker creation (component A). Module 3 contains the python scripts for the motion capture data transfer and the RBF surface deformation for animation with the optional use of distance matrices (components B, C and D, respectively). Discontinued arrows shows how the data flows between scripts.



1. Create facial markers for the 3D model:
Create Markers
2. Transfer the MoCap data to the facial markers:
Transfer MoCap
3. Choose RBF method and calculate the animation:
Animate Mesh
Markers stiffness:
Calculate Dist. Matrix

● Euc ● Geo ● Hyb

First frame: 0

Last frame: 100

Steps: 5

Mesh name: Head

Distance matrix folder: /Matrices/MatrixRealistic

Figure 4.5: A screenshot of the main UI window designed by the author for the *AnimFace* plugin.

4.3. Components development

In the following sections each step for the creation of the *AnimFace* plugin components is covered. An overview of the pseudo-code algorithms for the interface is included, as well as the reasons behind each implementation decision. For clarity purposes of this report, a separate chapter will be dedicated to the details behind the RBF algorithms (see Chapter 5). This part will focus on the development of the components that set the 3D environment and the interface ready for those algorithms to be executed.

4.3.1. Component A: Markers creation

In order to transfer the motion captured to the 3D model, a set of markers had to be created for the digital mesh. These were then set on the facial model by the user prior to applying the motion capture transformation. Though there is extensive research for the automatic calculation of facial feature point detection for a wide range of facial models (Wang et al. 2018), given that this project was limited to just two 3D models, manual marker placement was chosen as the most convenient option. An initial template was provided to reduce the amount of manual work by the user. The script *createMarkers.py* creates a default set of 41 joints, properly named and set in a default position, ready to be placed by the user onto the facial model. A secondary script, *groupMarkers.py*, arranges the markers in groups to ease the motion capture transfer in later steps. These functionalities are executed under the *Create Markers* button of the *AnimFace* UI.

To increase the accuracy during the placement of the markers, the snap tool provided by Maya was used to place the markers in correspondence of the vertex positions on the 3D models. Figure 4.6 shows the results for the realistic and the stylised model with the markers placed after this step.

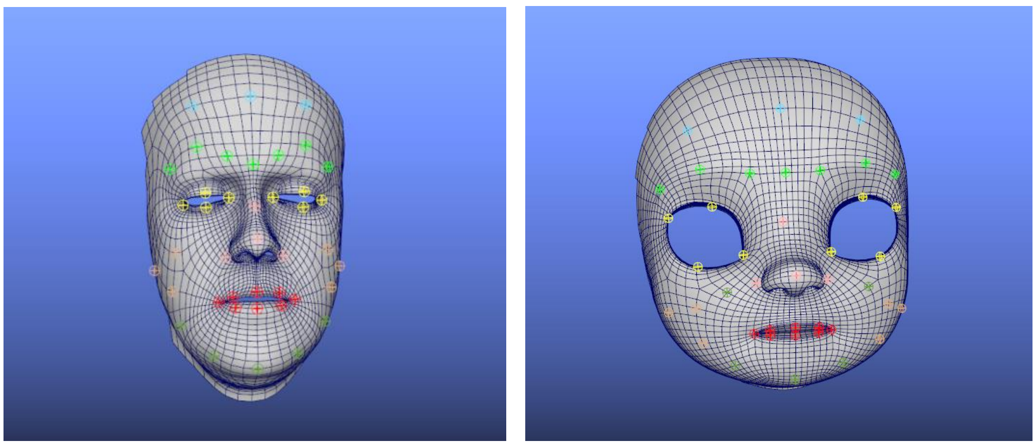


Figure 4.6: Image of the realistic and the stylised facial model with the markers.

4.3.2. Component B: Motion capture transfer

Once the markers were properly set on the 3D facial models, the motion capture data was then imported into the scene. This data was presented in Maya in the form of 41 spheres representing the control points of the motion captured, each of them associated with movement information (keyframes) in every frame. Each sphere was properly tagged according to the given name of each marker (see Table 4.1).

The script *transferMoCap.py* was created to transfer the motion capture data onto the facial markers placed on the 3D facial models. This script calibrates the position of the digital markers compared with the motion capture data. More complex calibrations have been investigated for cases when the morphology of the geometry is highly distorted (Abdul-Massih et al. 2017). For the scope of this project, given that both 3D models used are closely humanoid, a 1:1 scale calibration was used. Algorithm 4.1 shows the pseudo-code of the transfer procedure implemented in the script. This functionality is executed under the *Transfer MoCap* button of *AnimFace* UI.

```
offsets = new list [number of markers]

#Calculate the offset of the markers:
for x in number of markers:
        get marker position x
        get mocap position x
        offsets[x] = marker group position + marker position x –
                (mocap group position – mocap position x)

#Transfer mocap vectors to markers:
for x in number of frames:
        go to frame[x]
        for y in number of markers:
                get mocap position y
                new position = mocap position y + offsets[y]
                move marker y to new position
```

Algorithm 4.1: Pseudo-code for *transferMocap_plugin.py*

4.3.3. Component C: Animation algorithms

After the models were set with the animated markers and the motion capture was transferred to them, the next step calculated the deformation of the facial mesh according to the displacement information of the markers. These calculations were all based on the RBF method (see Chapter 2). A total of three algorithms were proposed to calculate the surface deformation. Chapter 5 will explain in detail the implementation of the different RBF surface deformation techniques.

The script *animateMesh.py* contains the implementation of the main architecture of the deformation algorithm, which orchestrate the surface deformation for the 3D facial models according to the position of the markers in every frame. Algorithm 4.2 shows the details of the pseudo-code of this script. This functionality can be executed under the *Animate Mesh* button of the *AnimFace* UI.

4.3.4. Component D: Matrix distance calculation

To avoid redundant calculations for each of the iterations of the deformation algorithms, an additional feature was added to the tool that allowed the use of distance matrix data stored in pre-processed files. These pre-processed files were calculated as an additional step and stored in a distance matrix file containing all the distances from one vertex to another, first by Euclidean and then by geodesic operations. The use of this tuple allowed for more flexibility in the deformation algorithms, easing the need for calculating the distances during the process to animate the mesh, and significantly improving computation time (see Chapter 5).

The script *calculateDistMatrix.py* contains the implementation. The generation of distance matrix files is executed under the *Calculate Dist. Matrix* button of *AnimFace* UI, given that there is a valid path on the *Distance Matrix Folder* textbox.

```
for (x=0, x < number of frames, x=x+step):

    go to frame[x]

    #Calculate the markers displacement:
    markers displacement = new list []
    for y in number of markers:
        markers displacement [y] = marker position y –
                                   marker y initial position

    #Calculate the RBF matrix of the markers and the weights:
    RBF matrix = new matrix [number of markers] [number of markers]
    for y in number of markers:
        for z in number of markers:
            RBF matrix [y][z] = calculate RBF (marker y, marker z)

    weights = new list [number of markers] filled with zeroes
    weights = invert matrix (RBF matrix) * markers displacement

    #Calculate the displacement of every vertex in the mesh:
    for y in number of vertices:
        displacement = 0
        for z in number of markers:
            RBF value = calculate RBF (vertex y, marker z)
            displacement += weights[z] * RBF value
        new position = position vertex y + displacement
        move vertex y to new position
```

Algorithm 4.2: Pseudo-code for *animateMesh_plugin.py*

4.3.5. Component E: User Interface

The UI was developed on the script *animFace_UI.py* using the SDK for Maya. In addition to the main buttons to execute the component functionalities, additional input fields were added to allow the user to set the frame range in which the algorithm works, the step value of the RBF algorithm, the name of the 3D facial mesh to deform, and the stiffness for each marker, used by the RBF algorithms. The range 1-9 determines how flexible the surface is around that point, being 1 the most flexible, and 9 the stiffest.

4.4. Summary

This chapter detailed the process followed to acquire the motion capture data and the tool created in Maya to transfer it to the 3D facial models. An overview of the software architecture for the *AnimFace* tool was covered, including a detailed explanation of the implementation of each of the components of this plugin.

The next chapter presents the details of the three RBF surface deformation algorithms developed, including the results and evaluations for each of these algorithms in quantitative and qualitative terms.

Chapter 5: RBF Surface Deformation Algorithms

This chapter explains in detail the development of the RBF surface deformation algorithms, including the implementation, the results, and a final evaluation comparing them in a qualitative and quantitative manner. Three different variants of the RBF technique were implemented: one based on the Euclidean distance, the second one using geodesics and the third one as a hybrid of the two previous methods. This chapter describes the insights of each one of them. All the three deformation algorithms aim to satisfy the RBF Gaussian distance function:

$$\varphi(\|m_j - m_k\|) = e^{-\|m_j - m_k\|/\gamma} \quad (5.1)$$

Figure 5.1 shows how the pipeline of the program works and how the data flows, following the functionalities of the components explained in Chapter 4. When the data reaches the RBF algorithm, this is then executed with one of the three RBF techniques implemented.

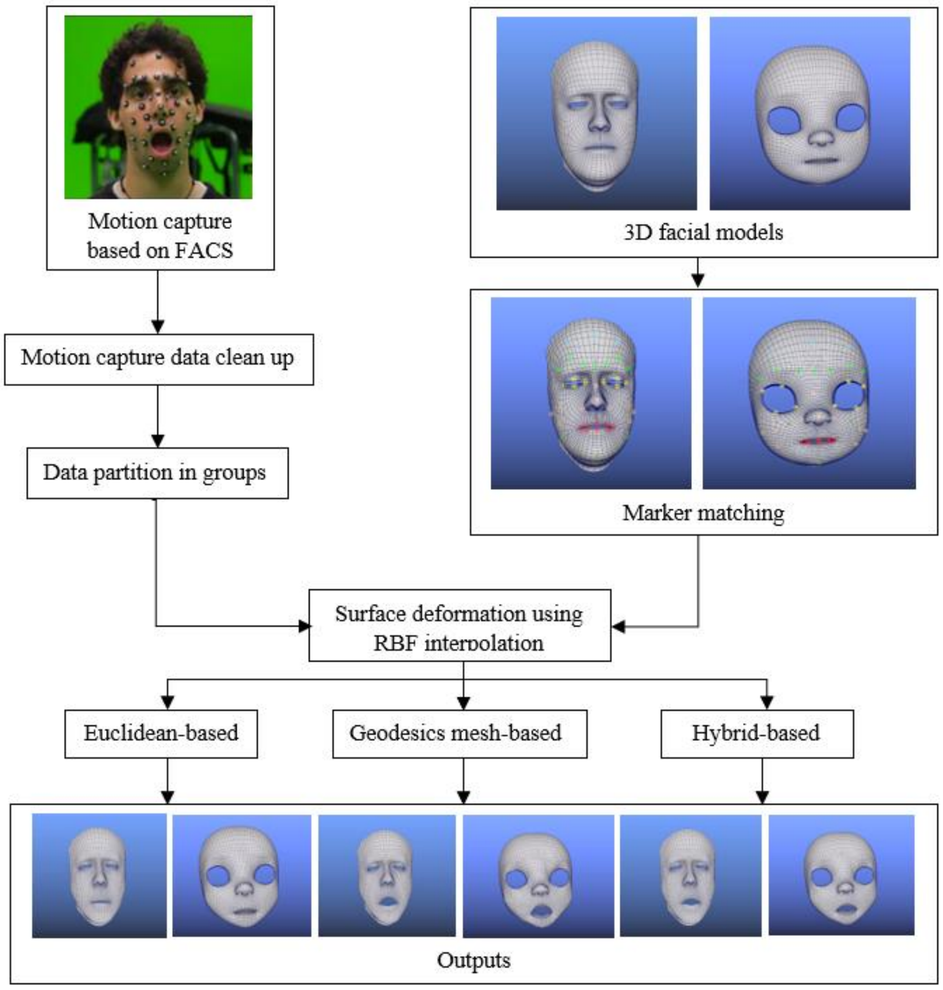


Figure 5.1: Pipeline diagram detailing the framework of the *AnimFace* plugin. Once the motion capture data is acquired, it goes through a clean-up process to make it ready for the algorithm. The 3D models are then prepared with the markers on top, and the RBF interpolation algorithm is executed with one of the three possible variants: Euclidean, Geodesic of Hybrid.

5.1. Surface deformation using RBF with Euclidean distance

Following a similar approach as the one presented by Man-dun et al. (2014), the first surface deformation algorithm implemented was based on an approximation of the RBF method using Euclidean distance.

5.1.1. Implementation

This method satisfies the RBF Gaussian distance equation (Eq. 5.1) using the ordinary metric of measuring the length of the segment that joints two points of interest in a 3D space:

$$\|p - q\| = \sqrt{(p_x - q_x)^2 + \left(p_y - q_y\right)^2 + (p_z - q_z)^2} \quad (5.2)$$

This method is the simplest way to solve the RBF distance function as it does not require heavy calculations. See Figure 2.3 in Chapter 2 for an example about how the RBF Euclidean distance works.

5.1.2. Results

Table 5.1 shows the average times of the Euclidean calculations after computing three instances of the algorithm with and without the help of a distance matrix. The results of this algorithm show that it is a fast-computing method that takes little resources and that is able to generate an output with good performance. The tests performed with and without distance matrix show no significant difference in the results, and the time for the RBF calculations is very similar in both cases.

In terms of qualitative results, this algorithm is able to map the general facial gestures onto the facial model, resulting in smooth results for each frame. However, this method lacks the capability of mapping the motion properly on the cavity areas, such as the mouth and the eyes. As mentioned by Wan et al. (2012), the Euclidean distance approach with RBF is not well suited for surface deformation of non-continuous meshes with holes, such as the case of the face. Artifacts appear when trying to transfer movements such as opening the mouth or closing the eyes, which result in stretched lips and eyelids. This is due to the nature of the distance calculations, which always take a straight line between two points. When these two points belong to areas seemingly close (e.g. the upper and the lower lip or the upper and the lower eyelid), the calculation of the deformations will stretch these out together. Figure 5.2 shows an example of the appearance of these artifacts when trying to replicate a gesture with closed eyelids and an open mouth.

Euclidean RBF Calculations	
With distance matrix	**Without distance matrix**
4.66 s	9.38 s

Table 5.1: Average time of the Euclidean RBF calculations for 2746 vertices.

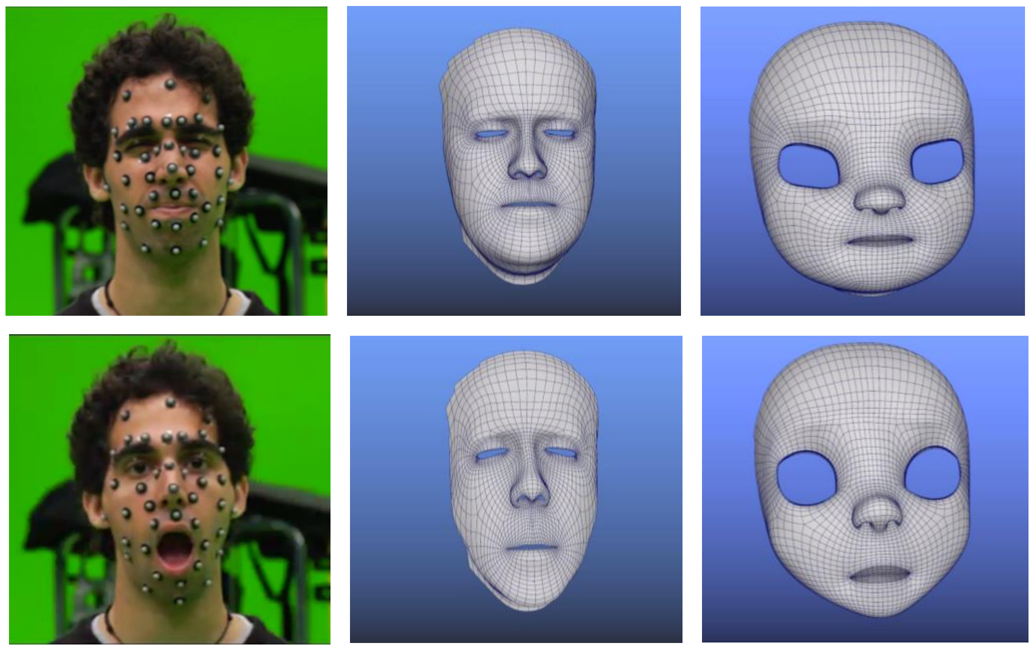


Figure 5.2: Example of the results of the Euclidean distance algorithm, compared with the real footage (FACS AU44 and AU27). Notice the mesh stretching appearing on the mouth.

5.2. Surface deformation using RBF with geodesic distance

This is the second approach to the RBF technique, based on the findings of Wan et al. (2012). The main goal of this algorithm is to find a more accurate distance calculation for non-continuous surfaces with holes, such as the mouth and the eyes in the face. The idea behind this is to avoid the stretching artefacts that appears with methods such as the Euclidean distance, as it was mentioned in the previous section. Given the fact that the geodesic problem is a complex one, two simplified heuristics of the algorithm were developed within the scope of this project.

5.2.1. Implementation

In graph theory, geodesic distance is the shortest path between two points along the edges of the graph, known also as mesh-based distance. This is a very common problem in the area of discrete maths, and extensive research has been done in order to find efficient algorithms to solve it (Jongmin Baek 2007). The most common one is the Dijkstra algorithm and its heuristic approximation A* (see Chapter 2 for more details).

Algorithm 5.1 presents a heuristic of the geodesic mesh-based distance with the application of recursion. This method was tested and then discarded given the high amount of time required to do the calculations and the lack of results obtained in areas where the mesh was not continuous. This heuristic also presented high amount of inaccuracies, distorting the facial features and the small gestures.

```
Geodesics_Recursive (vertex A, vertex B, number of iterations):

    selection A = [vertex A]
    selection B = [vertex B]

    for y in number of iterations:

        #Grow selections
        selection A = select vertices around selection A
        selection B = select vertices around selection B

        #Check if there are intersections
        intersection = intersection (selection A, selection B)

        if (intersection > 0):

            #Trivial case: vertices are adjacent to each other
            if (vertex A exists in intersection) and (vertex B exists in
            intersection):
                return Euclidean distance (vertex A, vertex B)

            #Calculate middle vertex and apply recursivity
            else
                temporal distance = max. float value

                middle vertex = null

                for v in intersection:

                    aux distance = Euclidean distance (vertex A, v)
                    +
                            Euclidean distance (v, vertex
                        B)

                    if (aux distance < temporal distance):
                        temporal distance = aux distance
                        middle vertex = v

                return Geodesics_Recursive (vertex A, middle vertex,
                number of iterations) + Geodesics_Recursive (middle
                vertex, vertex B, number of iterations)
```

Algorithm 5.1: Heuristic of geodesic mesh-based distance using recursion, discarded after unsuccessful tests and lack of results (presented here for continuity purposes).

An alternative approach was developed using the third-party Maya tool *Shortest Edge Path* (Autodesk 2018), a search heuristic to find the shortest path on a mesh between two given vertices (see Chapter 2), which aided the calculations and presented better results than the previous algorithm based on recursion. Algorithm 5.2 presents the pseudo-code of this heuristic using the auxiliary Maya tool.

```
Geodesics_Shortest_Edge_Path (vertex A, vertex B):

    #Trivial case: same vertex
    if vertex A == vertex B:
        return 0

    #Get the list of vertices of the shortest path from A to B
    edge path = Maya Shortest Edge Path (vertex A, vertex B)

    dist = 0

    #Get the total distance
    for y in (length (edge path) – 1):
        dist += Euclidean distance (edge path[y], edge path[y+1])

    return dist
```

Algorithm 5.2: Heuristic of geodesic mesh-based distance using the aid of the Maya Shortest Edge Path tool (Based on a path-finding algorithm).

5.2.2. Results

Table 5.2 shows the average times of the geodesic calculations after computing three instances of Algorithm 5.2 with and without a distance matrix. Contrary to the Euclidean method, this algorithm requires more dense operations to perform the calculations needed to give a final output. The main reason for the increase of these calculations is the complexity of the heuristic, which needs to calculate the distance between each vertex and each marker in order to solve the RBF interpolation. Given this, the time for RBF calculations was, on average, 462.51 s (8 min approx.). For this case, the use of a precomputed distance matrix was of high significance and running again the algorithm with the aid of this matrix reduced the time of RBF calculations to 5.05 s.

In relation to qualitative results, the approximation of the geodesic algorithm was able to calculate more accurately the deformation of the facial model based on the motion capture data. In this case, the mouth and the eye movements were more accurately matched, and movements such as mouth opening, and eye closing were able to be transferred properly. Figure 5.3 shows some of the results of this algorithm with some of the most noticeable expressions in which this approximation performs better.

Geodesic RBF Calculations	
With distance matrix	**Without distance matrix**
5.05 s	462.51 s

Table 3.2: Average time results of the geodesic RBF calculations for 2746 vertices.

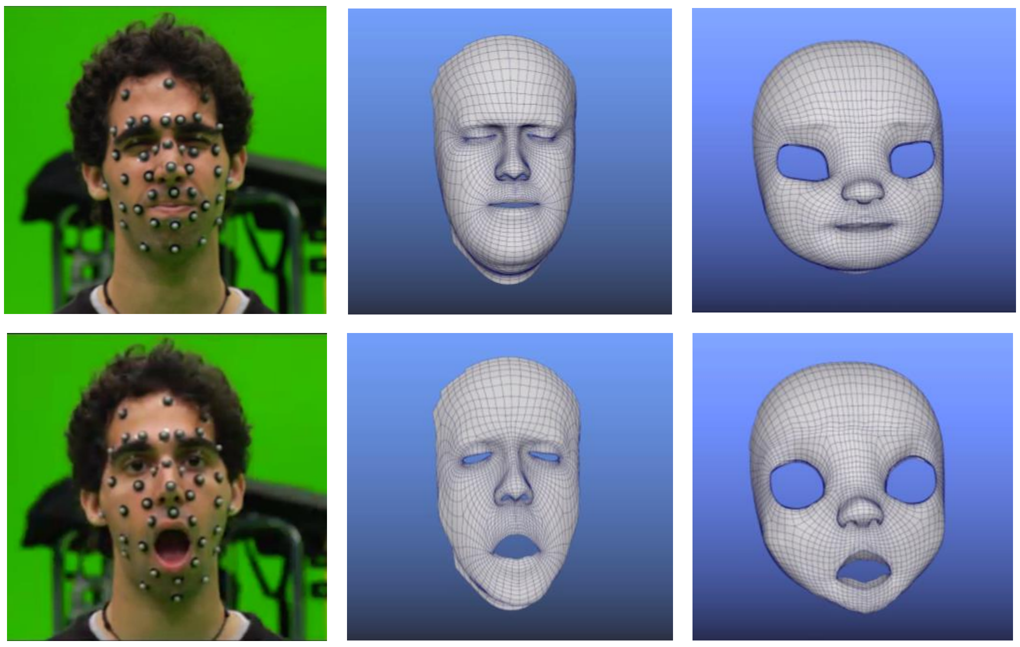


Figure 5.3: Example of the results of the geodesic algorithm, compared with the real footage (FACS AU44 and AU27).

5.3. Surface deformation using RBF with hybrid distance function

To try to take advantage of each of the algorithms described in the previous sections, a third approach was proposed in which a hybrid of the Euclidean and geodesic algorithm was implemented.

5.3.1. Implementation

Given that the calculation of the geodesic mesh-based distance is costlier, these calculations were limited to critical areas of the model that required this algorithm, leaving the other areas to normal Euclidean calculations. To do this, the vertices around the holes of the face (mouth and eyes) were grouped in Maya sets (being about 20% of the total number of vertices). Figure 5.4 shows an image depicting these vertices in yellow. In this algorithm, vertices that fall in one of these areas follow the geodesic algorithm, while vertices that start falling off from these areas are gradually interpolated with the Euclidean algorithm.

To calculate the interpolation between the two distances, a weight list was created to store the weight of the vertices, assigned in the range [0.0, 1.0] depending on how close the vertex is to a geodesic region. The decay function used for the weight was based on a Gaussian curve with gamma value equal to 2.

Once the weights were calculated, the calculation of the distance was decided based on these weights. When the weight is 0, the geodesic calculations were ignored, using only the Euclidean distance. For the rest, the following interpolation function was used:

*Distance = Geodesic_Distance(v,w)*weight + Euclidean_Distance(v,w)*(1-weight)*

Algorithm 5.3 and Algorithm 5.4 show the pseudocode for the calculation of the weights and the distance interpolation, respectively:

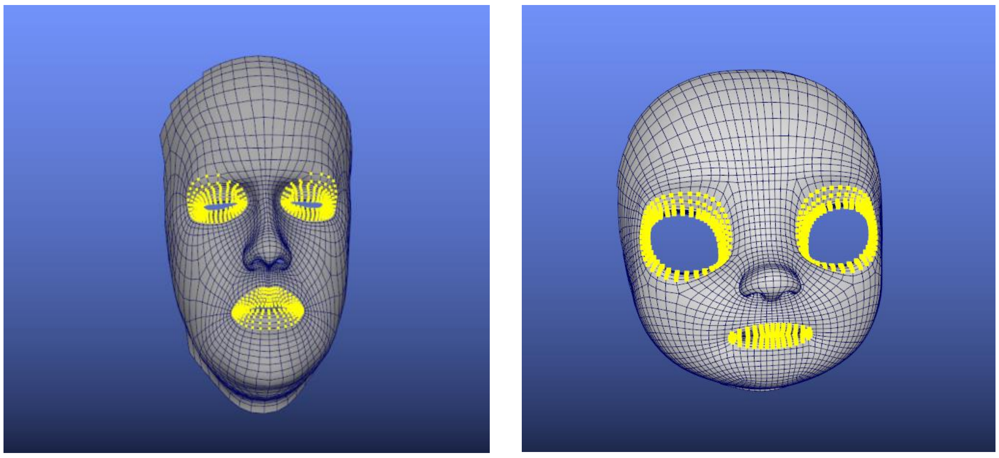


Figure 5.4: Geodesic vertices (yellow) for the realistic and the stylised model.

```
weights = new list [number of vertices]

for v in number of vertices:

        #If inside the geodesic area, the weight is 1
        if (v in geodesic vertices):
                weights[v] = 1

        #Else, the weight depends on the distance to the closest geodesic vertex
        else:
                distance = max. float value
                for g in geodesic vertices:
                        new distance = Euclidean distance (g, v)
                        if (aux distance < distance):
                                distance = new distance
                weights[v] = Calculate Gaussian(distance, gamma=2)
```

Algorithm 3.3: Weight calculation to interpolate the geodesic influence of a vertex.

```
Hybrid_Distance (weight, vertex A, vertex B):

        distance = 0

        if (weight == 0):
                distance = Euclidean distance (vertex A, vertex B)
        else:
                distance = Geodesic distance (vertex A, vertex B) * weight +
                           Euclidean distance (vertex A, vertex B) * (1 – weight)

        return distance
```

Algorithm 5.4: Pseudo-code of the calculation of the hybrid distance.

5.3.2. Results

Table 5.3 shows the average times of the hybrid calculations after computing three instances of the algorithm with and without matrix. As a hybrid between the two previous methods, the results of this algorithm gave a performance ranked in a middle point between the Euclidean and geodesic methods, with an average of 288.82 s (5 min approx.) for the RBF calculations. This algorithm was also developed to be compatible with the use of distance matrices. By employing them, the time for RBF calculations was significantly reduced to 4.82 s.

For the qualitative results, the hybrid algorithm successfully transferred the movements of the motion capture data and applied it correctly to all areas of the face, including the mouth and the eyelids. Mouth opening and eye closing were animated properly. Figure 5.5 show some examples of the results of this algorithm. The main issue with this algorithm was the appearance of small artifacts in the interpolation areas between Euclidean and geodesics, which required the adjustment of the Gaussian gamma parameter to an appropriate value (set to 2) to generate a smooth decay function in the interpolation.

Hybrid RBF Calculations	
With distance matrix	**Without distance matrix**
4.82 s	288.82 s

Table 5.3: Average time results of the hybrid RBF calculations for 2746 vertices.

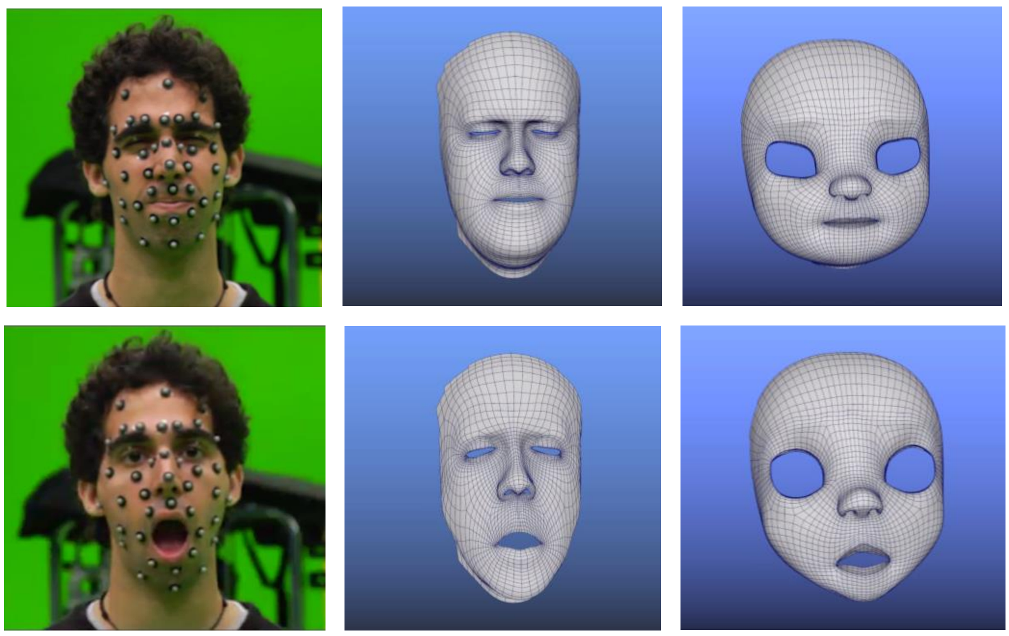


Figure 5.5: Results of the hybrid algorithm compared with real footage (FACS AU44 and AU27).

5.4. Comparative Evaluation

A demo reel was created as part of the outcomes of this project to present the results of the algorithms, including the animations and the comparison with real footage. Appendix A presents a complete compilation of the results of the RBF algorithms for the relevant frames in each of the FACS-based motion capture groups. It is especially interesting to focus on the differences encountered in relation to the mouth and eye movements for each of the methods, and the appearance of artifacts.

5.4.1. Quantitative Assessment (Performance analysis)

The RBF algorithms were tested on a 2.80 GHz Intel Core i7-7700HQ CPU with 16 GB of RAM and a Nvidia GeForce GTX 1050Ti graphic card. All the quantitative calculations were performed under an idle operating Windows 10 system within Autodesk Maya 2017. For accuracy purposes in the comparison, the keyframing time was ignored.

Table 5.4, Figure 5.6 and Figure 5.7 show the compilation of the quantitative results. For each, the calculations were performed with and without matrix distance:

	With dist. matrix	**Without dist. matrix**
Euclidean RBF calculations	4.66 s	9.38 s
Geodesic RBF calculations	5.05 s	462.51 s
Hybrid RBF calculations	4.82 s	288.82 s

Table 5.4: Compilation of quantitative results of the RBF algorithms for 2746 vertices.

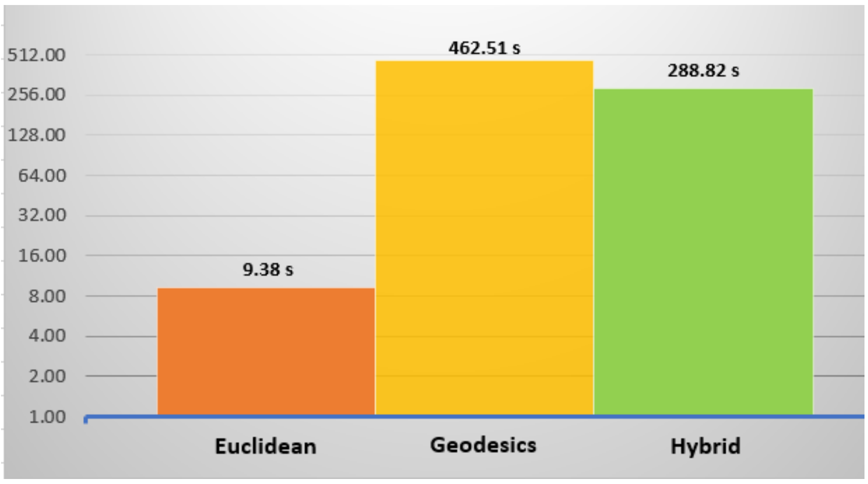


Figure 5.6: Time comparison between the different RBF calculations for 2746 vertices without distance matrix (logarithmic scale).

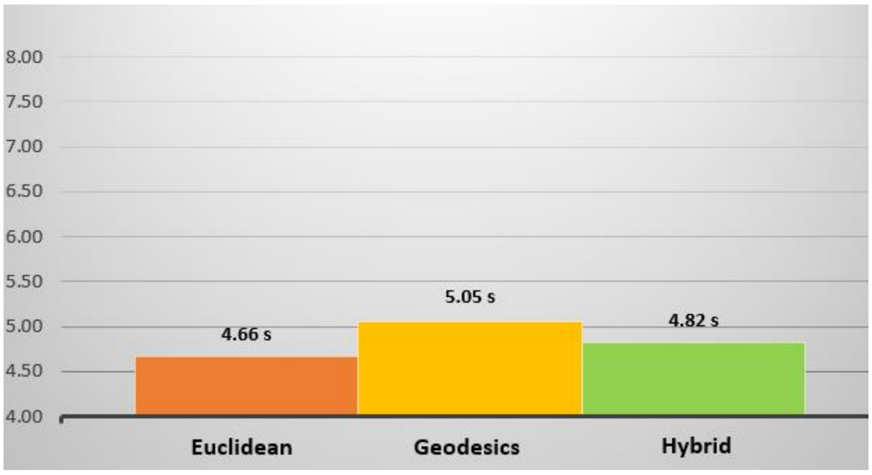


Figure 5.7: Time comparison between the different RBF calculations for 2746 vertices with distance matrix.

As it can be seen, when distance matrices are not used the Euclidean algorithm runs much faster than the other two, given the lack of complexity in the calculation of the distances. Following this, the hybrid algorithm is the second best in terms of performance, while the geodesic appears to be the slowest one in these cases. As mentioned in previous sections, this is due to the number of instances required on the path-finding heuristic for the calculation of the shortest edge path.

It is interesting to notice that, when using distance matrices, the calculations speed up very significantly, and there is little difference in the time required to complete computations between the three methods. This reveals the efficiency of pre-processing data for these algorithms, avoiding redundant calculations and making them more flexible and useful in separate running instances. Table 5.5 shows the times for calculating each of these matrices.

	Euclidean	**Geodesics**
Total time	9.277 s	547.124 s
Time per vertex	3.8 ms	199.24 ms

Table 5.5: Calculation time for the Euclidean and the geodesic distance matrices.

5.4.2. Qualitative Assessment (Comparison with real footage)

As mentioned in Chapter 4, the motion capture data was recorded by performing a selection of facial expressions based on the theory of FACS. The RBF algorithms were then used to transfer the motion capture onto two types of models, one realistic and one stylised. This section shows a highlight of the results of these transfers for each of the FACS groups, included a qualitative comparison carried out by the author against the real footage. Appendix A contains the compilation of all the results for each of the FACS units. The following subsections refer to each of those tables. In addition to this, a demo presents some of these comparisons between the real footage and the animated 3D models.

Eyebrows group

Table A.1 in Appendix A shows the results of this group. As it can be seen from the images, these facial features do not show much difference in the 3D models between the three algorithms, given the fact that the eyebrow deformations happen almost completely in areas where the surface is continuous (without holes). The deformations are more accentuated in the geodesic and the hybrid cases. In general, the three algorithms are all suitable for the generation of good results, with the Euclidean approach being the one that would give better performance.

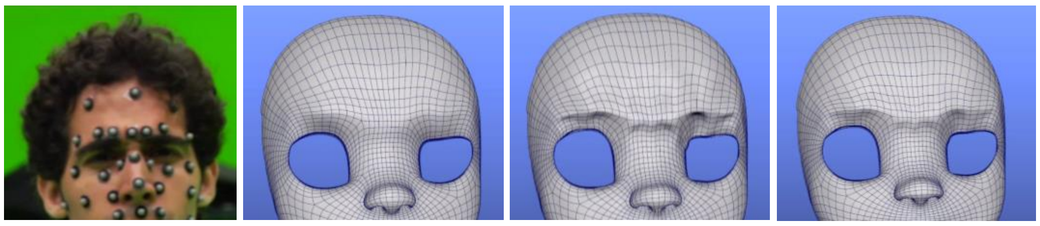


Figure 5.8: Detail of the AU4 results (Euclidean, geodesics and hybrid) for the stylised model.

Eyes group

Table A.2 in Appendix A shows the results of this group. In these cases, there is a major difference in the results when analysing the eyelid areas. The Euclidean algorithm is not able to deal properly with surface deformation in non-continuous areas, and therefore the results of this algorithm lack enough detail. In contrast, geodesics and hybrid cases present results closer to the real footage. When comparing these two, a better grade of smoothness is noticeable in the hybrid results (see Figure 5.9). It is also noticeable the limitation of the algorithms in the stylised model, given the fact that the eyes do not completely close. This is possibly related to the density of the mesh around the eyelids.

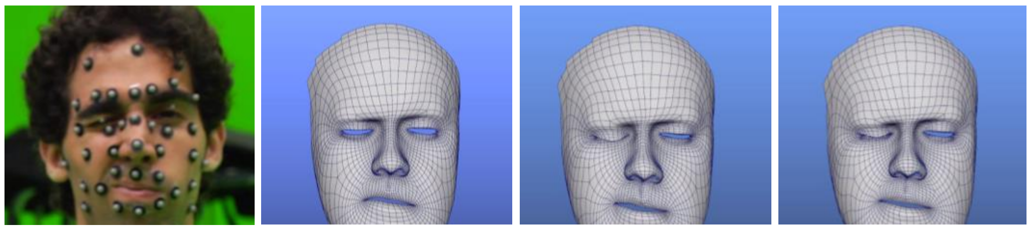


Figure 5.9: Detail of the AU46 results (Euclidean, geodesics and hybrid) for the realistic model. Notice the smoothness of the hybrid case (right), presenting a more polished result.

Nose group

Table A.3 in Appendix A shows the results of this group. The results of the algorithms present many similarities, with more accentuated features in the geodesics and the hybrid cases. It is interesting to remark the unit AU39 (nostril compressor), which presents many artifacts in all the cases, possibly due to the fact that it is a very extreme feature of the face (see Figure 5.10). For this last case, the deformations that approach better the reference are the geodesic and hybrid instance.

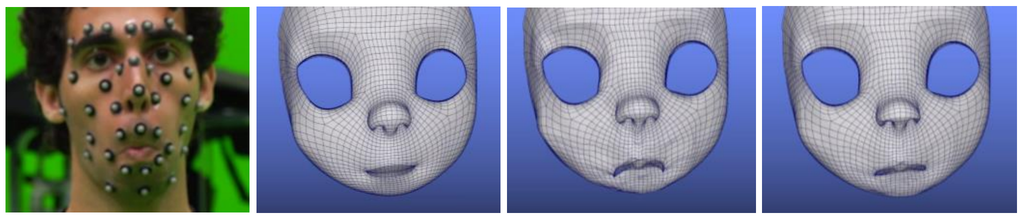


Figure 5.10: Detail of the AU39 results (Euclidean, geodesics and hybrid) for the stylised model, with the hybrid case (right) being the one presenting smoother results.

Lips group

Table A.4, Table A.5 and Table A.6 in Appendix A show the results of this group. For these cases, the benefits of the geodesics distance calculations can be observed in areas of the mouth. The geodesic algorithm is the one that reaches the extreme poses better. The hybrid approach reaches similar levels of accuracy in the poses, giving also better results in terms of smoothness of the features. FACS unit AU14 (dimpler) exemplifies this: the nasolabial fold can be seen both in the geodesic and the hybrid instances (see Figure 5.11).

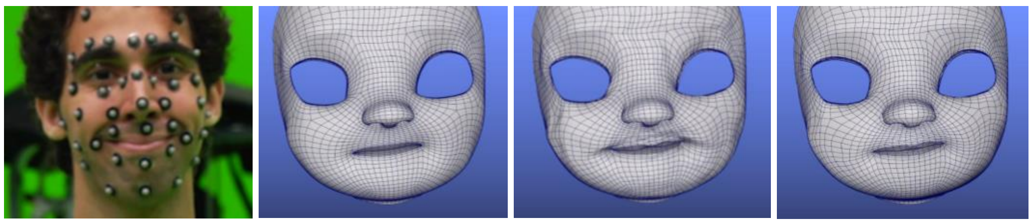


Figure 5.11: Detail of the AU14 results (Euclidean, geodesics and hybrid) for the stylised model. Notice the nasolabial fold, accentuated in the geodesic (middle) and hybrid (right) cases.

Cheeks group

Table A.6 and Table A.7 of Appendix A show the results of this group. Similarly to the previous group, this group presents some extreme poses in the features that are achievable with more smoothness by the hybrid algorithm. The Euclidean instances also suffice for this group, since the mouth remains closed in all the cases. Despite reaching slightly more extreme poses with geodesics, the hybrid case is the best options to generate smoother results of this group (see Figure 5.12).

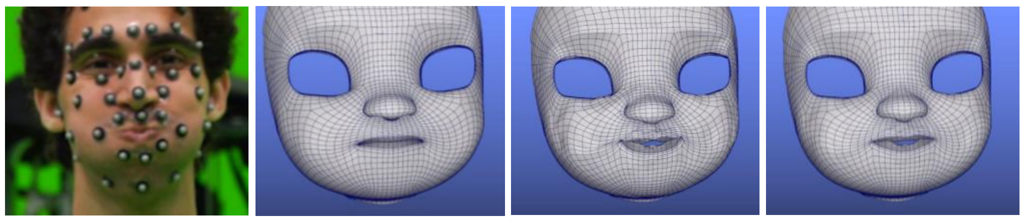


Figure 5.12: Detail of the AU33 results (Euclidean, geodesics and hybrid) for the stylised model. Notice the more extreme poses in geodesics (middle) and hybrid (right) cases.

Jaw group

Table A.7, Table A.8 and Table A.9 in Appendix A show the results of this group. In these cases, except for the FACS units AU26 (jaw drop) and AU27 (jaw stretch) which contains open mouths, the facial features can be accomplished with very similar results by any of the three algorithms. Same as in the previous groups, the expressions get slightly more accentuated in the geodesics instance (see Figure 5.13).

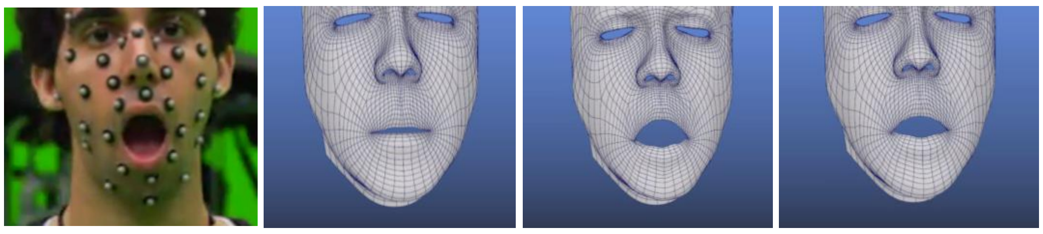


Figure 5.13: Detail of the AU27 results (Euclidean, geodesics and hybrid) for the stylised model. Notice the more accurate expression of the open jaw in geodesics (middle) and hybrid (right) cases.

5.5. Summary

This chapter presented the implementation and results of the RBF algorithms, including a quantitative and qualitative evaluation for each of them. The results showed that, overall, the hybrid approach is the one that finds the best compromise between performance, results and accuracy, both in quantitative and qualitative terms.

The next chapter will conclude this thesis with some final thoughts of the author and an overview of possible paths to extend this research in future work.

Chapter 6: Conclusions and Future Work

This chapter summarizes the findings of this project, including a list of the outcomes generated, and reflects on some conclusions based on them. It also proposes new possible paths to continue the research with future work.

6.1. General Findings

The main scope of this project was to explore the methods related to the transfer of motion capture data into facial models using RBF techniques, and to test different algorithms based on this technique to get efficient and good quality results. Three main steps were followed to accomplish this: first, to capture the motion capture data efficiently and with a wide range of variation in the expressions; second, to create a plugin for Maya able to read the motion capture data and transfer it to a facial model using RBF algorithms; and third, to make a quantitative and qualitative evaluation to test the final animation outcomes.

The following lines present a reflection on the two research questions of this thesis:

1) How can RBF methods be optimized for obtaining efficient surface deformation based on motion capture?

The results of the evaluation reiterated the advantages and disadvantages of the Euclidean methods, as stated before by Man-dun et al. (2014) and Wan et al. (2012), as well as the exploration of a simple heuristic for the geodesic method. A third approach was proposed using a hybrid algorithm that attempted to use the advantages of the two previous methods, which probed to give positive results in terms of performance and animation quality. The main disadvantage of the geodesic method was found to be the time for the calculations, which was solved by proposing the use of pre-processed distance matrices, significantly reducing the algorithm time in all the three methods and avoiding redundant calculations.

2) How can facial animation be enhanced by motion capture techniques based on RBF methods?

Though this question remained open during the whole research process, the use of RBF techniques for the generation of detailed deformation in facial models opens some possibilities for the creation of high-quality animation rigs. A known technique for the creation of animation rigs is the use of blendshapes generated from the outcomes of the motion capture (McLaughlin et al. 2011). Some famous examples in relation to this technique can be seen in the work of *The Lord of the Rings* with the creature *Gollum* (Orvalho et al. 2012) or, more recently, in the work of *Kong* (Cong et al. 2017).

The main advantage that motion capture presents in this area is the possibility of generating a large amount of expressions without the need of manual sculpting or the creation of a complex muscle system, which results in more efficient times and in higher realism, especially when dealing with photorealistic characters. Being able to generate believable expressions is key in an industry in which the demand for realism keeps growing. Figure 6.1 shows some examples of the potential of this technique using blendshapes generated from the models of this project.

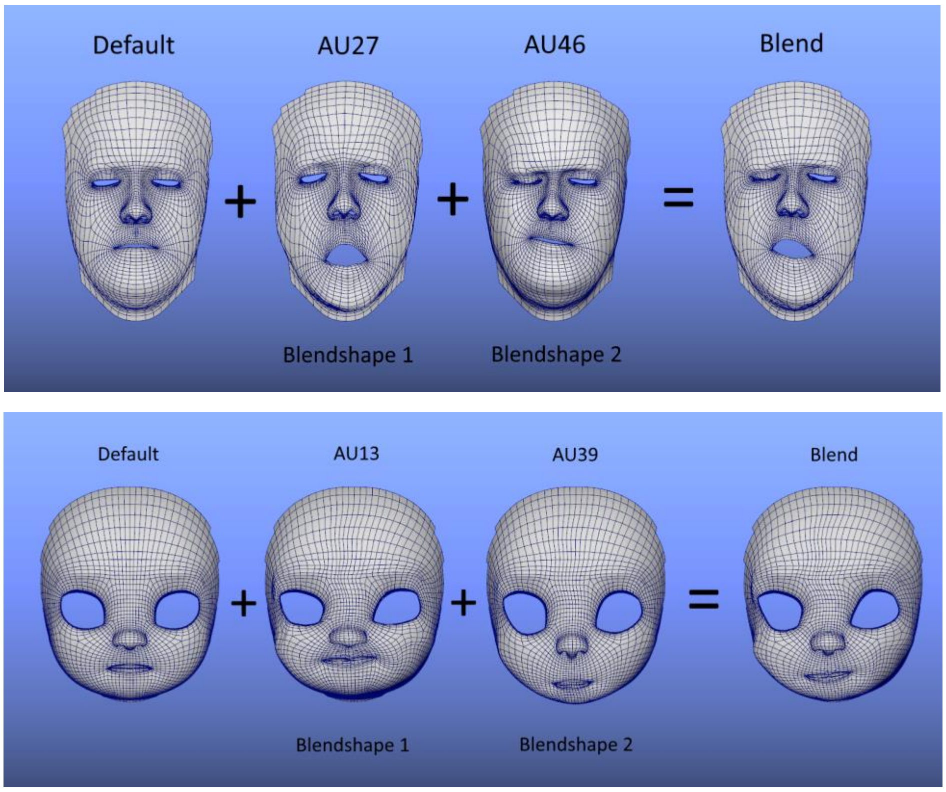


Figure 6.1: Examples of how to create blendshapes. On the top, the realistic facial model was set with the blendshapes of AU27 (weight 0.8) and AU46 (weight 0.9). On the bottom, the stylised facial model was set with the blendshapes of AU13 (weight 0.8) and AU39 (weight 0.9).

6.2. Outcomes

The following lines present the outcomes that were generated from this project:

- *AnimFace:* A plugin created for Maya for the transfer of motion capture data onto 3D facial models. This plugin includes tools for the generation of markers for the digital faces and the transfer of motion capture using the RBF algorithms with the three variants explored in this research, all integrated in a completely functional UI (Accessible at: *github.com/bigoMay/AnimFace/tree/master/PythonScripts*).
- *Facial motion capture data:* A collection of motion capture data based on the theory of FACS, divided in six categories by the different regions of the face (eyebrows, eyes, nose, lips, cheeks and jaw), along with a record of the real footage of the performance (Accessible at: *github.com/bigoMay/AnimFace/tree/master/MoCapData*).
- *Animated facial models:* two 3D facial models, one realistic and another stylised, animated by the transfer of facial motion capture data using the algorithms of the *AnimFace* tool.
- *Demo:* An explanatory video rendered with some of the examples of the animated facial models, compared with the real footage, which illustrates the use of the *AnimFace* tool (Accessible at: *vimeo.com/289380770*).
- *Poster:* A research poster submitted for the CDE Digital Catapult Event 2018 in London containing a brief explanation of the RBF algorithms developed for this project.
- *Paper:* A journal paper submitted to ICGST-GVIP 2019 to present the results of this project to the research community.

6.3. Future Work

This work is the result of a master's project developed within the area of computer animation. In general terms, the goals of this project were to explore the area of motion capture for facial animation and the use of efficient algorithms based on RBF for the generation of surface deformations in both realistic and stylised models.

The results of this research show some positive outcomes when dealing with the development and use of transferred motion capture in 3D facial models. The method for motion capture data with the equipment and the performance script described was successful in relation to the capture of expressive motion to create our facial animation. Though the efforts of cleaning the data slowed the process and forced to fix certain trajectories manually, the data kept consistent and within a standard high-quality level throughout the entire process. Better facial rigs for motion capture could be used in the future to improve acquisition accuracy. Also, when making the evaluation, there are some limitations in the qualitative comparison, given that is was carried out solely by the author. A more robust perceptual experiment could bring more significant results to this.

In relation to the algorithms, both 3D models responded generally well to the animation transfer, and the RBF methods developed approximated the surface deformation with expected results. Among them, the hybrid algorithm that combined the Euclidean and the geodesic methods showed an overall smoother result in terms of animation quality. Also, the use of distance matrices proved to be a very significant achievement in the project towards optimizing the calculation times. In all the cases, however, some small imperfections could be noticed during the animation playbacks, most likely caused by the fact the algorithms ignored the small translations of the head during the motion capture performance. Further ways of improving these algorithms would be to consider this information when generating the animation. In addition, the use of ground truth facial data for the qualitative comparison could help in evaluating the quality and accuracy of the algorithms.

In relation to future work, several branches of development are open from here. On one line of development, there are possibilities to optimize further the RBF algorithm to bring more accurate facial deformation and to be more computational efficient. The use of the Python Maya plugin could be further profiled for better performance and integrated using code compiled directly for the 3D software. On another line of work, other techniques alternative to RBF have been proposed, one of them particularly taking more attention: the use of Partial-Differential Equations for the calculation of how a mesh is deformed given a certain force vector. PDE algorithms have been suggested for future work in order to be able to evaluate the results of both approaches in terms of animation accuracy and computational efficiency. Finally, another proposal for future work would be to create a facial rig based on blendshapes directly extracted from the resulting facial animation, as mention in one of the previous sections, attempting artist-friendly controls and driven keys to enable more expressive facial animation.

The approaches of this project presented an outline of a much larger task in the attempt to explore new possibilities for the creation of expressive facial animations in both realist and stylised characters. This thesis covered the efforts on the use of motion capture data with RBF to sculpt a wide range of facial movements based on FACS, but new techniques have been proposed and suggested for future work. Among them, the use of muscle systems keeps growing interest in the industry, and new ways to match motion capture with these new methods could open new interesting paths of research. The final goal would be to reach more accurate and efficient techniques for better facial rigging and animation, which is of high demand nowadays in a wide range of areas within film-making, feature animation, videogames and simulations.

References

Abdul-Massih, M., Yoo, I. and Benes, B., 2017. Motion style retargeting to characters with different morphologies. *Computer Graphics Forum*, 36(6), 86–99.

Abrantes, G.A. and Pereira, F., 1999. MPEG-4 facial animation technology: survey, implementation, and results. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(2), 290–305.

Alexander, O., Rogers, M., Lambeth, W., Chiang, J.Y., Ma, W.C., Wang, C.C. and Debevec, P., 2010. The digital Emily project: Achieving a photorealistic digital actor. *IEEE Computer Graphics and Applications*, 30(4), 20-31.

Arai, K., Kurihara, T. and Anjyo, K., 1996. Bilinear interpolation for facial expression and metamorphosis in real-time animation. *The Visual Computer*, 12(3), 105–116.

Avatar, 2009 [film]. Directed by James Cameron. USA-UK: 20th Century Fox.

Avengers: Infinity War, 2018 [film]. Directed by Anthony Russo and Joe Russo. USA: Marvel Studios.

Bastos, P.T.B., 2015. *Easy Facial Rigging and Animation Approaches*. Thesis (PhD). University of Porto.

Cambridge Dictionary, 2019. *English Cambridge Dictionary* [online]. Cambridge: Cambridge University Press. Available from: dictionary.cambridge.org [Accessed 1 April 2019]

Coco, 2017. [film]. Directed by Lee Unkrich and Adrian Molina. USA: Disney Pixar.

Cong, M., Bhat, K.S. and Fedkiw, R., 2016. Art-directed muscle simulation for high-end facial animation. *ACM SIGGRAPH Symposium on Computer Animation*, 119–127.

Cong, M., Lan, L. and Fedkiw, R., 2017. Muscle simulation for facial animation in Kong: Skull Island. *ACM SIGGRAPH 2017 Talks*, 21.

Cosker, D., Krumhuber, E. and Hilton, A., 2010. A FACS validated 3D human facial model. *Proceedings of the SSPNET 2nd International Symposium on Facial Analysis and Animation*, 12.

Dawn of the Planet of the Apes, 2014. [film]. Directed by Matt Reeves. USA: 20th Century Fox.

Deza, M.M. and Deza, E., 2009. *Encyclopedia of distances*. Berlin, Heidelberg: Springer.

Dijkstra, E.W., 1959. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1), 269–271.

Ekman, P. and Friesen, W.V., 1978. *Manual for the facial action coding system*. Consulting Psychologists Press.

Ekman, P. and Rosenberg, E.L., 1997. *What the face reveals: Basic and applied studies of spontaneous expression using the Facial Action Coding System (FACS)*, USA: Oxford University Press.

Fidaleo, D., Noh, J.Y., Kim, T., Enciso, R. and Neumann, U., 2000. Classification and volume morphing for performance-driven facial animation. *International Workshop on Digital and Computational Video,* 10.

Gillenson, M.L., 1974. *The interactive generation of facial images on a CRT using a heuristic strategy.* Thesis (PhD). The Ohio State University.

Guerra-Filho, G., 2005. *Optical Motion Capture: Theory and Implementation*. RITA, 12(2), 61–90.

Hardy, R.L., 1971. Multiquadric equations of topography and other irregular surfaces. *Journal of geophysical research*, 76(8), 1905–1915.

Hart, P.E., Nilsson, N.J. and Raphael, B., 1968. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2), 100–107.

Hashi, S., Toyoda, M., Yabukami, S., Ishiyama, K., Okazaki, Y. and Arai, K.I., 2006. Wireless magnetic motion capture system for multi-marker detection. *IEEE transactions on magnetics*, 42(10), 3279–3281.

Hellblade: Senua's Sacrifice, 2017 [game]. Directed by Tameem Antoniades. UK: Ninja Theory.

Herda, L., Fua, P., Plänkers, R., Boulic, R. and Thalmann, D., 2001. Using skeleton-based tracking to increase the reliability of optical motion capture. *Human movement science*, 20(3), 313–341.

Ho, L.C., Klaassen, M.F. and Mithraratne, K., 2018. Biomechanics of Three-Dimensional Face. *The Congruent Facelift*, 33–65.

Ichim, A.E., Kadleček, P., Kavan, L. and Pauly, M., 2017. Phace: Physics-based face modeling and animation. *ACM Transactions on Graphics (TOG)*, 36(4), 153.

Jongmin Baek, A.D.K.R., 2007. *Finding Geodesics on Surfaces*. USA: Stanford University. Technical Report.

Kitagawa, M. and Windsor, B., 2012. *MoCap for artists: workflow and techniques for motion capture*. 1st edition. Focal Press.

Kong: Skull Island, 2017 [film]. Directed by Jordan Vogt-Roberts. USA: Warner Bros.

Lasseter, J., 1987. Principles of traditional animation applied to 3D computer animation. *ACM Siggraph Computer Graphics*, 21 (4), 35–44.

Lucas, J.B., 2017. The Physiology and Biomechanics of Skin Flaps. *Facial plastic surgery clinics of North America*, 25(3), 303–311.

Man-Dun, Z., Xin-Jie, G., Shuang, L., Zhi-Dong, X., Li-Hua, Y. and Jian-Jun, Z., 2014. Fast Individual Facial Animation Framework Based on Motion Capture Data. *Journal of Donghua University (English Edition)*, 1(3), 8.

McLaughlin, T., Cutler, L. and Coleman, D., 2011. Character rigging, deformations, and simulations in film and game production. *ACM SIGGRAPH 2011 Courses*, 5.

Menache, A., 2000. *Understanding motion capture for computer animation and video games*. San Francisco: Morgan Kaufmann.

Mongillo, M., 2011. Choosing basis functions and shape parameters for radial basis function methods. *SIAM Undergraduate Research Online*, 4(190–209), 2-6.

Mündermann, L., Corazza, S. and Andriacchi, T.P., 2006. The evolution of methods for the capture of human movement leading to markerless motion capture for biomechanical applications. *Journal of neuroengineering and rehabilitation*, 3(1), 6.

Murai, A. et al., 2017. Dynamic skin deformation simulation using musculoskeletal model and soft tissue dynamics. *Computational Visual Media*, 3(1), 49–60.

Muybridge, E., 1882. *The Attitudes of Animals in Motion Illustrated with the Zoopraxiscope.* Alexandria: Library of Alexandria.

O'Brien, J.F., Bodenheimer Jr, R.E., Brostow, G.J. and Hodgins, J.K., 1999. *Automatic joint parameter estimation from magnetic motion capture data*. Georgia Institute of Technology.

Orvalho, V., Bastos, P., Parke, F.I., Oliveira, B. and Alvarez, X., 2012. A Facial Rigging Survey. *Eurographics (STARs)*, 183–204.

Oxford Dictionary, 2019. English Oxford Living Dictionary [online]. Oxford: Oxford University Press. Available from: en.oxforddictionaries.com [Accessed 1 April 2019]

Parke, F.I., 1972. Computer generated animation of faces. *Proceedings of the ACM annual conference,* 1, 451–457.

Parke, F.I. and Waters, K., 2008. *Computer facial animation*. New York: AK Peters/CRC Press.

Pieper, S.D., 1991. *CAPS-Computer-aided plastic surgery*. Thesis (PhD). Massachusetts Institute of Technology.

Ping, H.Y., Abdullah, L.N., Sulaiman, P.S. and Halin, A.A., 2013. Computer Facial Animation: A Review. *International Journal of Computer Theory and Engineering*, 5(4), 658.

Platt, S.M., 1980. *A system for computer simulation of the human face*. University of Pennsylvania.

Ruhland, K., Prasad, M. and McDonnell, R., 2017. Data-Driven Approach to Synthesizing Facial Animation Using Motion Capture. *IEEE computer graphics and applications*, 37(4), 30–41.

Shotton, J., Fitzgibbon, A.W., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A. and Blake, A., 2011. *Real-time human pose recognition in parts from single depth images*, 2(3).

Sifakis, E., Neverov, I. and Fedkiw, R., 2005. Automatic determination of facial muscle activations from sparse motion capture marker data. *ACM transactions on graphics (TOG)*, 24(3), 417–425.

Snow White and the Seven Dwarfs, 1937 [film]. Directed by David Hand et. al. USA: Disney.

Sundaresan, A. and Chellappa, R., 2005. Markerless motion capture using multiple cameras. *Computer Vision for Interactive and Intelligent Environment (CVIIE'05)*, 15–26.

The Lord of the Rings, 2001 [film]. Directed by Peter Jackson. New Zealand-USA: New Line Cinema.

Tin Toy, 1988 [film]. Directed by John Lasseter. USA: Disney Pixar.

Tony de Peltrie, 1985 [film]. Directed by Pierre Lachapelle et. al. Canada: University of Montreal.

Wan, X., Liu, S., Chen, J.X. and Jin, X., 2012. Geodesic distance-based realistic facial animation using RBF interpolation. *Computing in Science and Engineering*, 14(5), 49–55.

Wang, N., Gao, X., Tao, D., Yang, H. and Li, X., 2018. Facial feature point detection: A comprehensive survey. *Neurocomputing*, 275, 50–65.

Wu, T., Hung, A. and Mithraratne, K., 2014. Generating facial expressions using an anatomically accurate biomechanical model. *IEEE Transactions on Visualization and Computer Graphics*, (1).

Zarins, U., 2017. *Anatomy of Facial Expressions*. EU: Anatomy Next.

Zhang, Z., 2012. Microsoft kinect sensor and its effect. IEEE multimedia, 19(2), 4–10.

Appendix A: Compilation of Results

The following tables show the results of the algorithms for each of the FACS units captured:

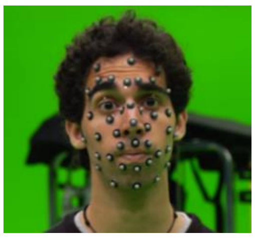
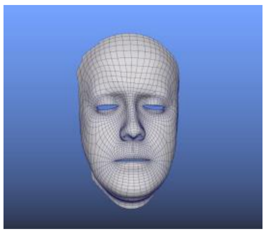
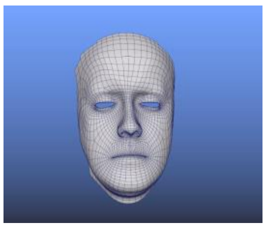
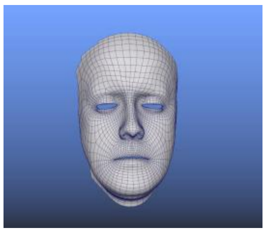
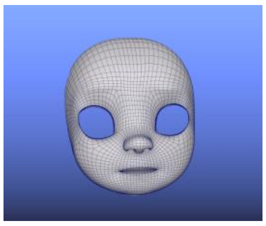
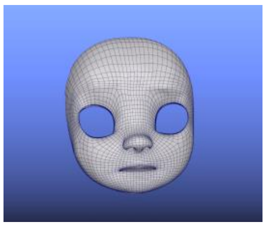
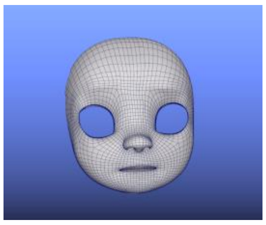
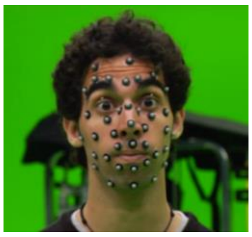
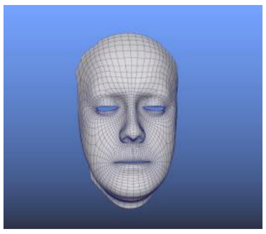
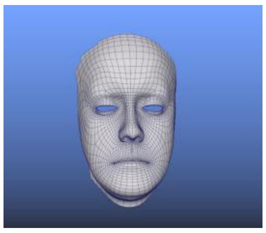
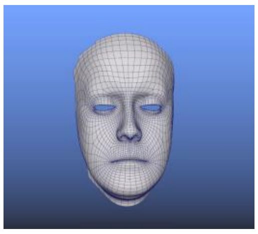
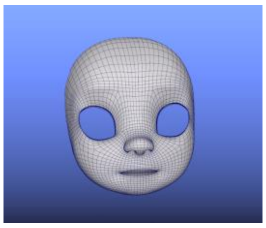
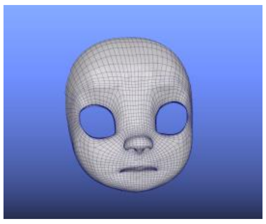
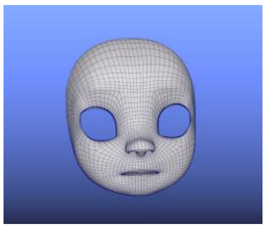
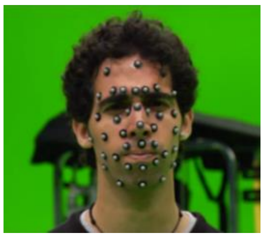
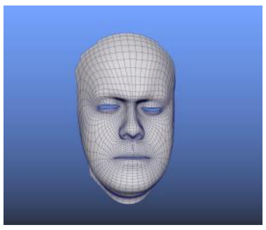
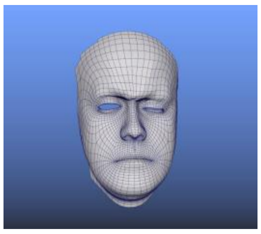
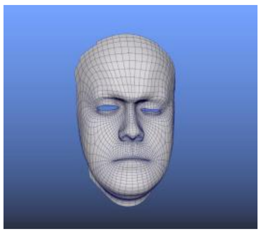
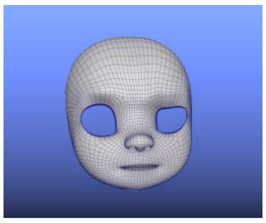
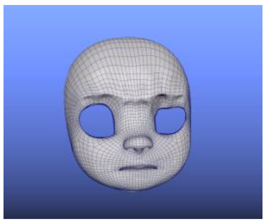
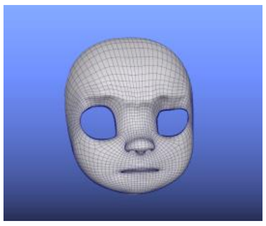
Footage / FACS Unit	Euclidean	Geodesics	Hybrid
 **AU1** **Inner brow raiser**			
			
 **AU2** **Outer brow raiser**			
			
 **AU4** **Brow lowerer**			
			

Table A.1: FACS units AU1, AU2 and AU4 (Eyebrows group).

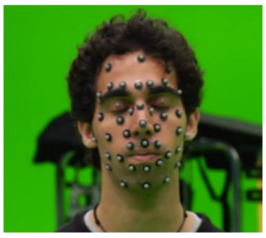
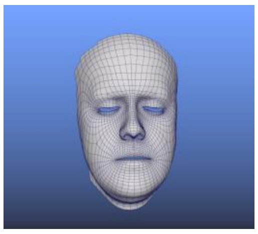
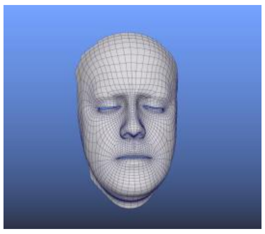
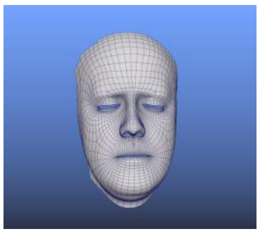
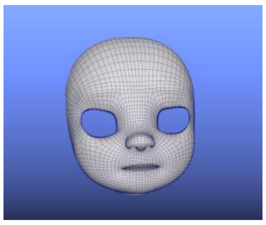
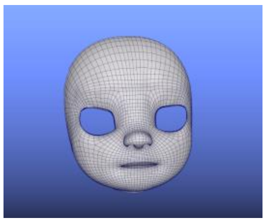
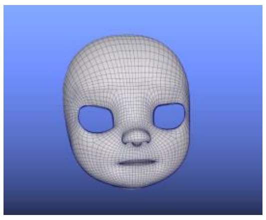
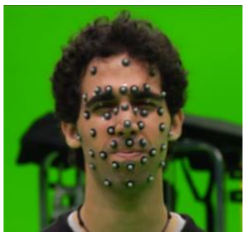
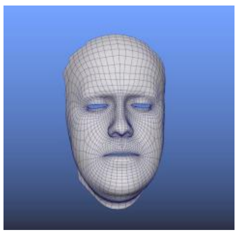
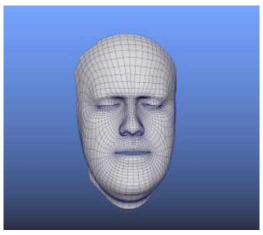
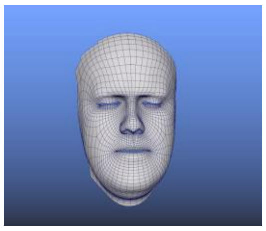
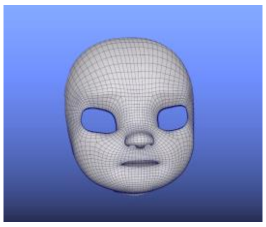
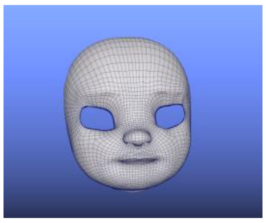
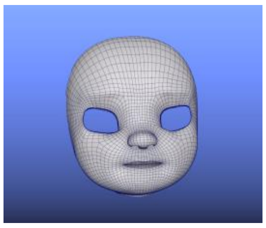
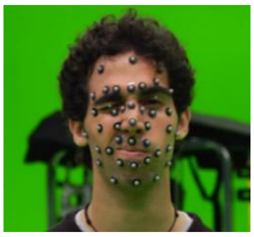
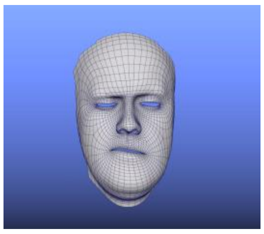
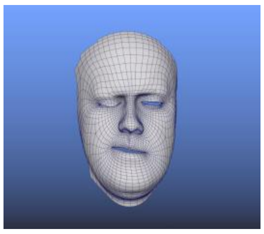
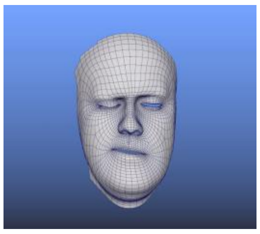
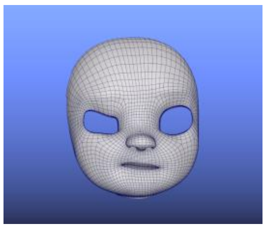
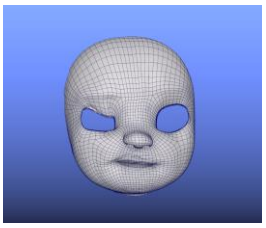
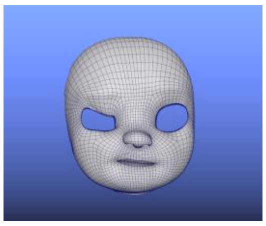
Footage / FACS Unit	Euclidean	Geodesics	Hybrid
 AU43 Eyes closed			
			
 AU44 Squint			
			
 AU46 Wink			
			

Table A.2: FACS units AU43, AU44 and AU46 (Eyes group).

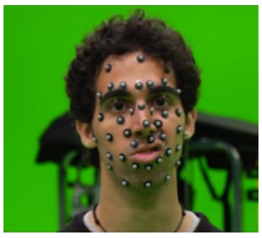
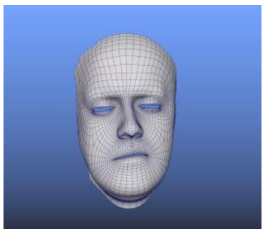
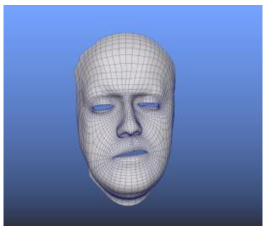
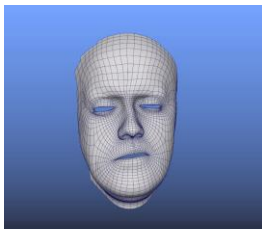
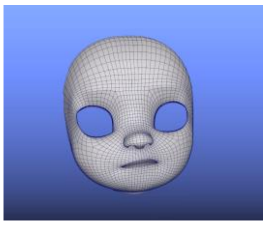
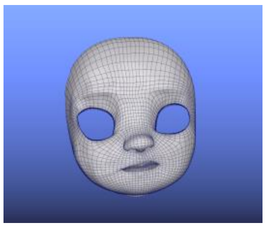
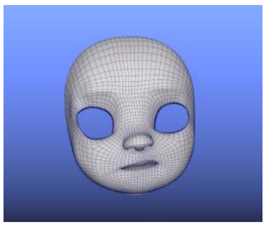
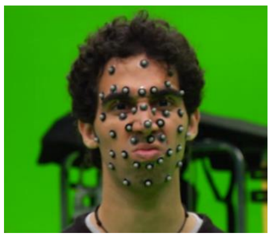
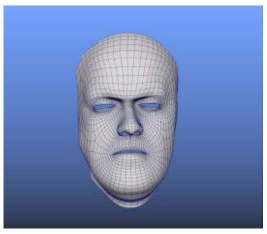
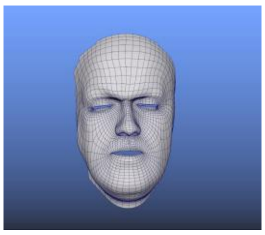
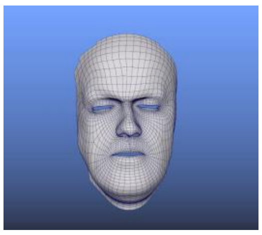
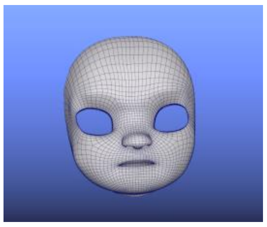
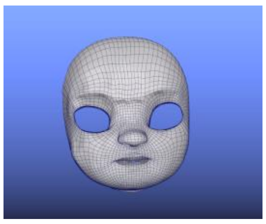
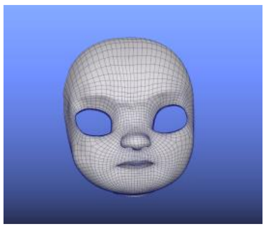
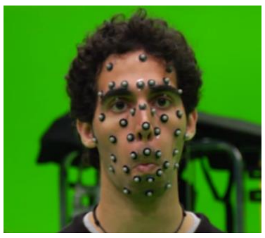
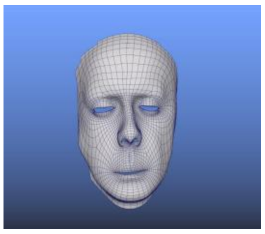
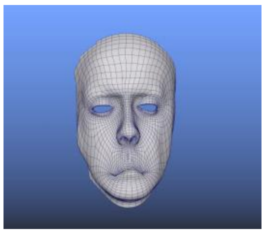
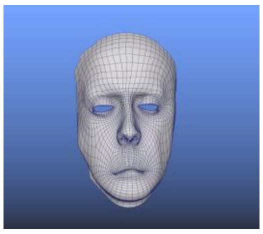
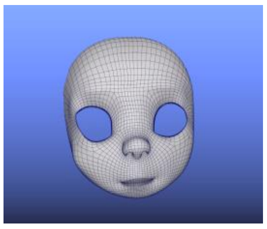
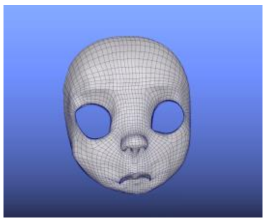
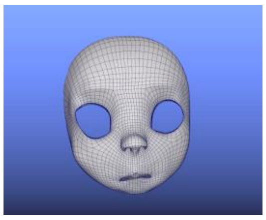
Footage / FACS Unit	Euclidean	Geodesics	Hybrid
 AU10 Upper lip raisere			
			
 AU11 Nasolabial deep			
			
 AU39 Nostril compressor			
			

Table A.3: FACS units AU10, AU11 and AU39 (Nose group).

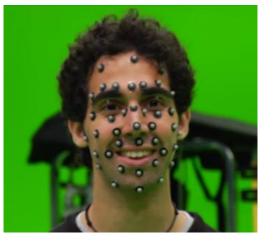
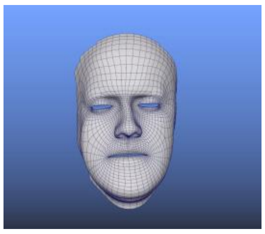
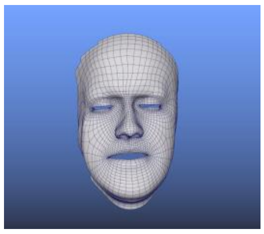
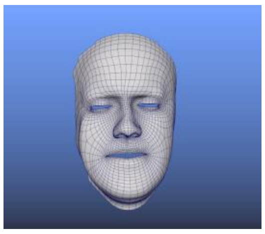
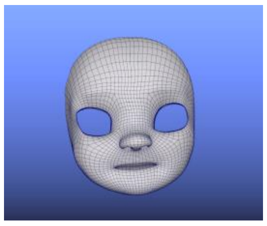
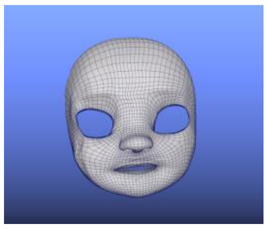
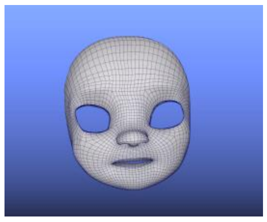
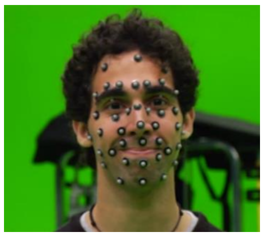
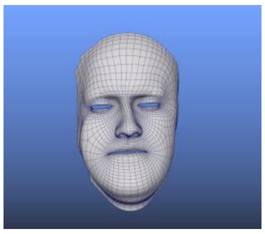
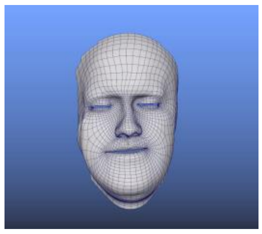
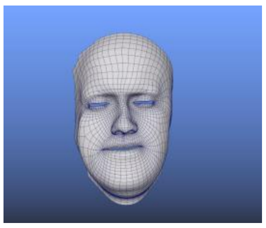
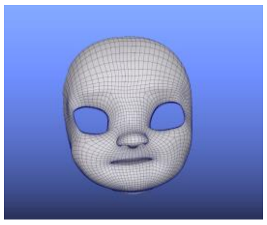
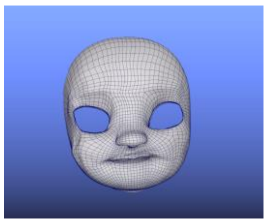
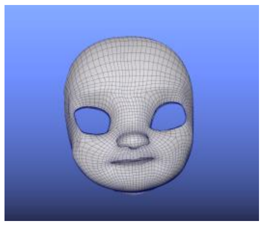
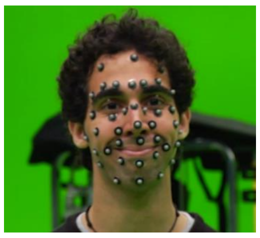
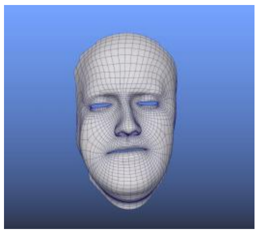
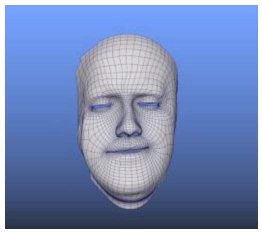
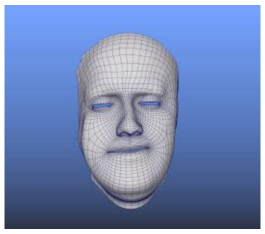
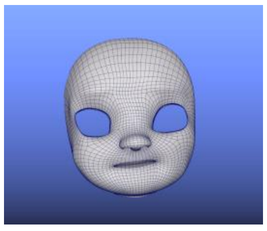
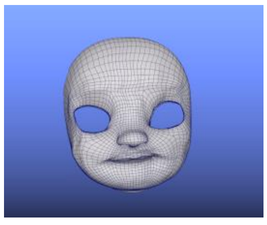
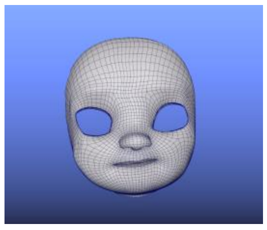
Footage / FACS Unit	Euclidean	Geodesics	Hybrid
 AU12 Lip corner puller			
			
 AU13 Sharp lip puller			
			
 AU14 Dimpler			
			

Table A.4: FACS units AU12, AU13 and AU14 (Lips group).

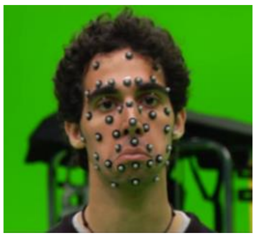
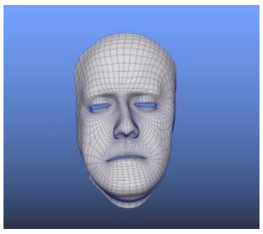
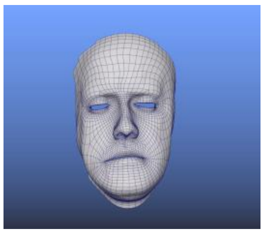
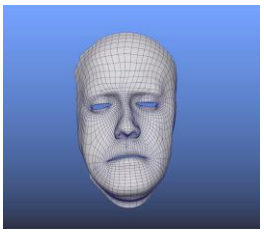
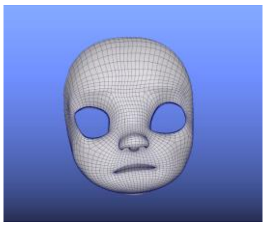
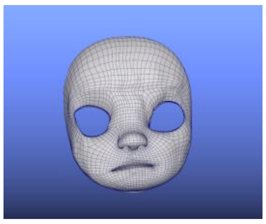
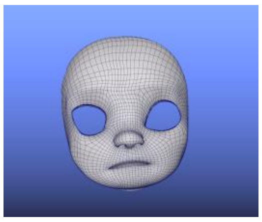
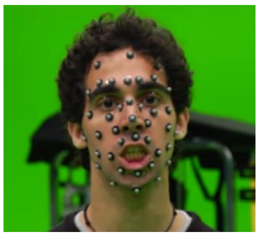
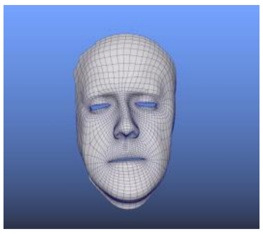
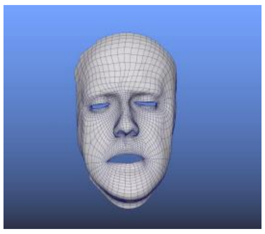
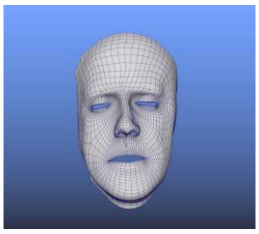
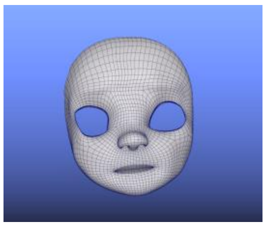
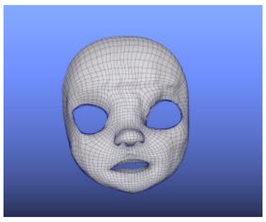
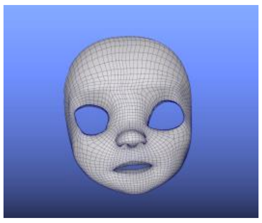
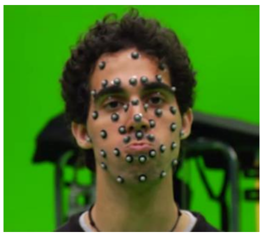
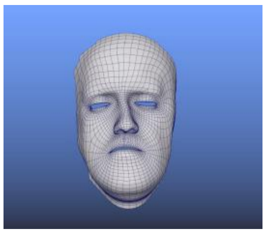
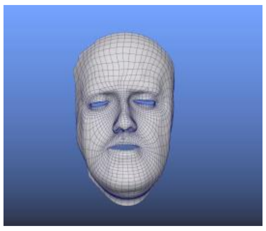
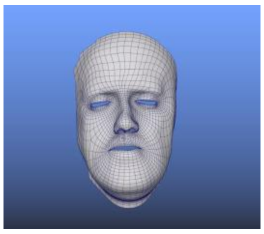
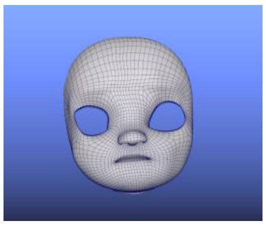
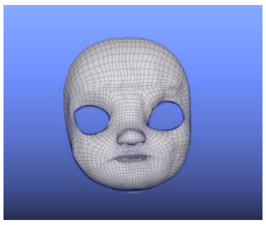
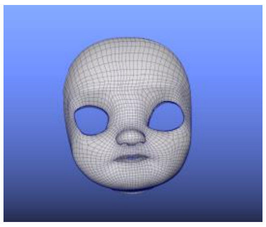
Footage / FACS Unit	Euclidean	Geodesics	Hybrid
 **AU15** **Lip corner depressor**			
			
 **AU16** **Lower lip depressor**			
			
 **AU17** **Chin raiser**			
			

Table A.5: FACS units AU15, AU16 and AU17 (Lips group).

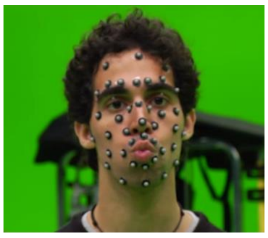
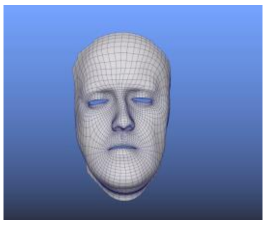
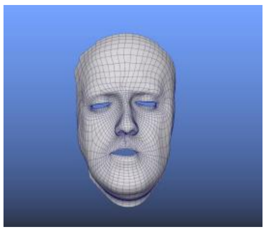
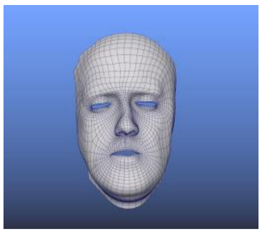
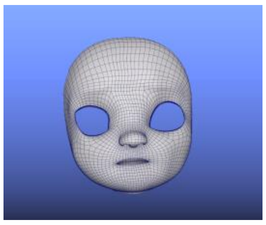
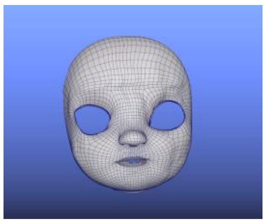
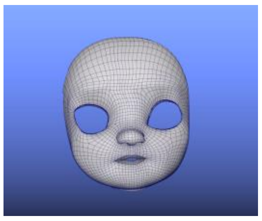
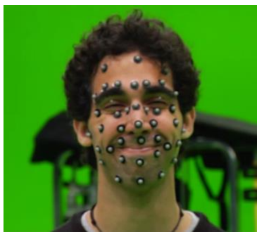
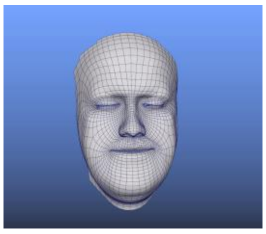
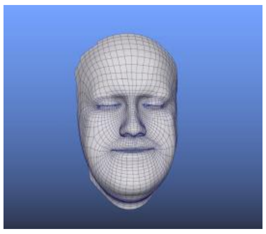
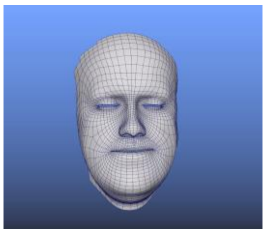
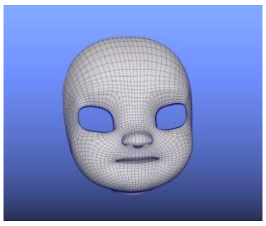
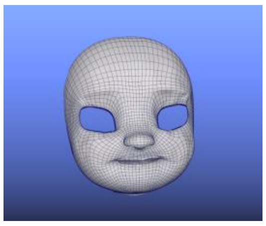
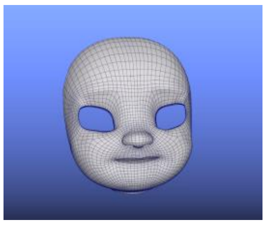
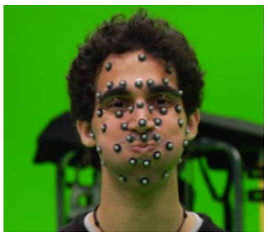
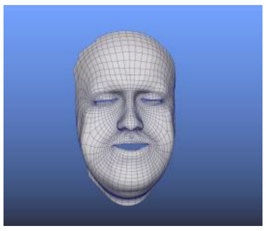
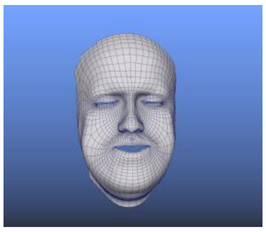
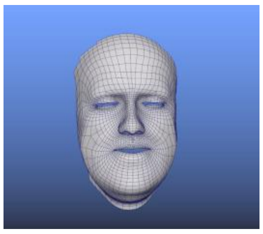
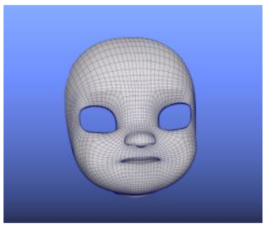
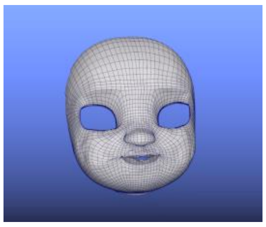
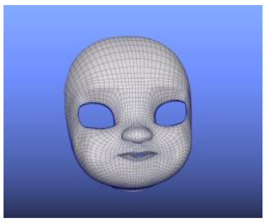
Footage / FACS Unit	Euclidean	Geodesics	Hybrid
 AU18 Lip pucker			
			
 AU6 Cheek raiser			
			
 AU33 Cheek blow			
			

Table A.6: FACS units AU18 (Lips group). FACS unit AU6 and AU33 (Cheeks group).

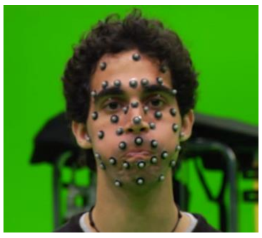
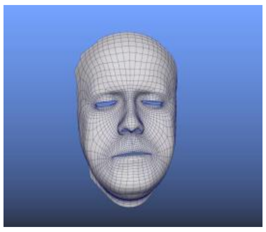
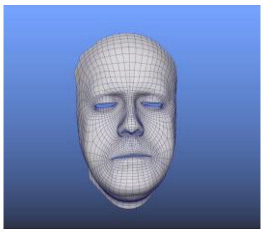
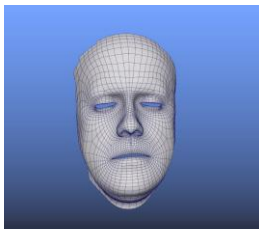
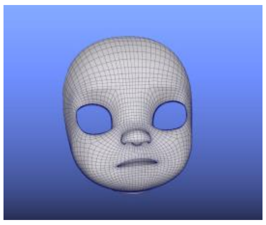
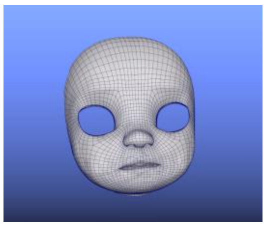
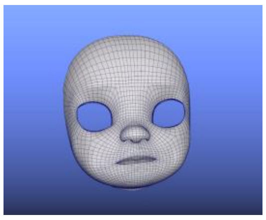
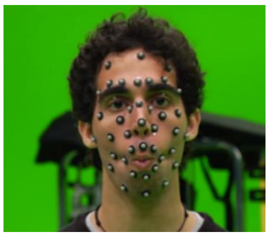
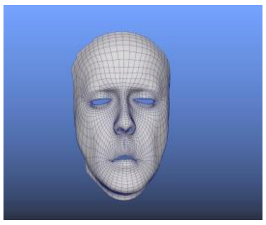
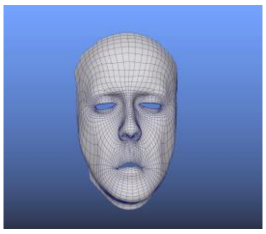
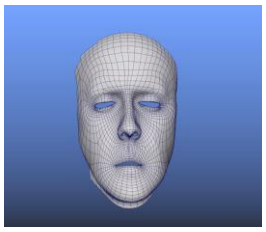
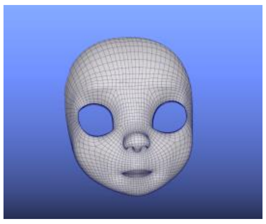
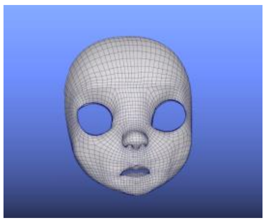
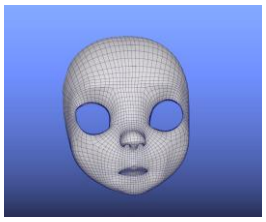
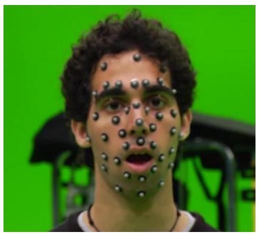
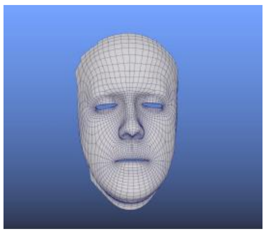
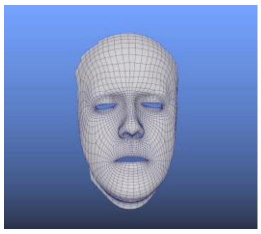
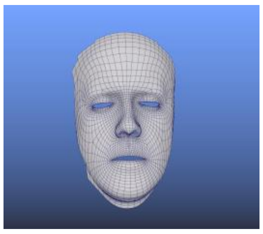
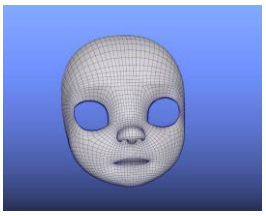
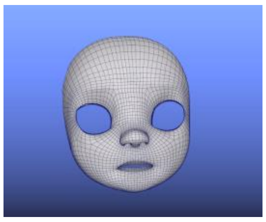
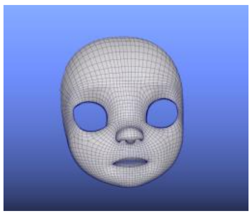
Footage / FACS Unit	Euclidean	Geodesics	Hybrid
 **AU34** **Cheek puff**			
			
 **AU35** **Cheek suck**			
			
 **AU26** **Jaw drop**			
			

Table A.7: FACS units AU34 and AU35 (Cheeks group). FACS unit AU26 (Jaw group).

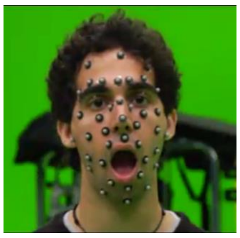
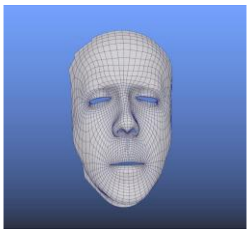
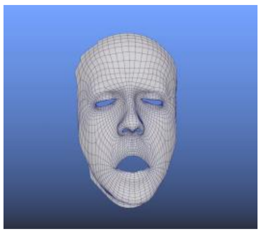
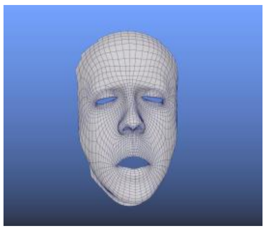
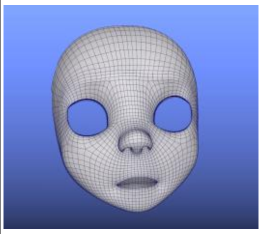
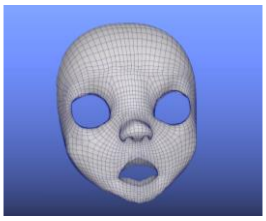
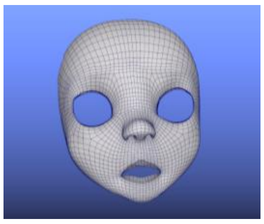
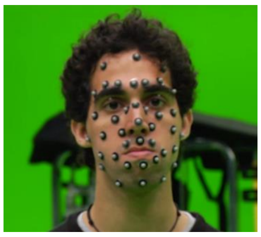
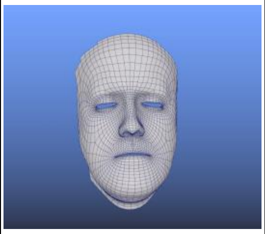
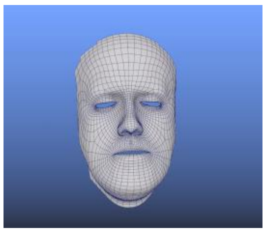
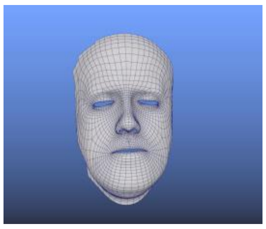
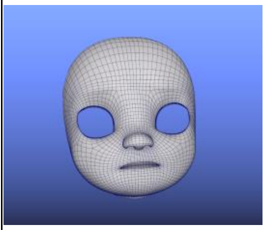
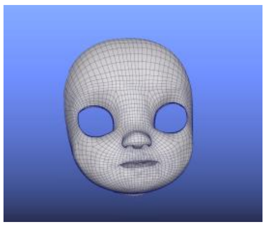
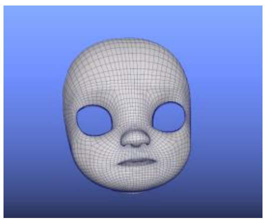
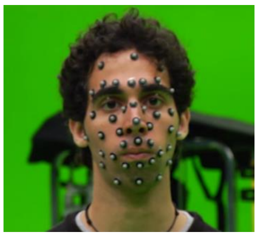
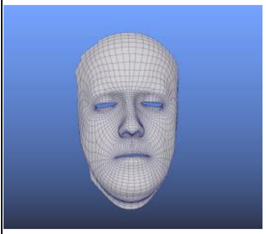
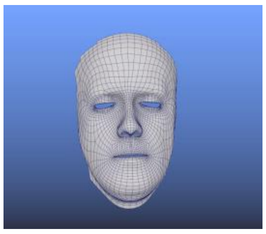
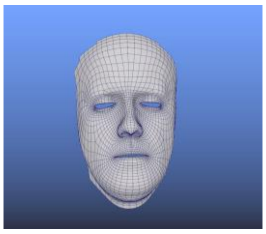
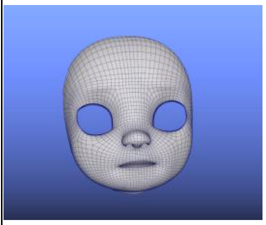
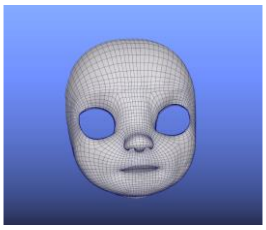
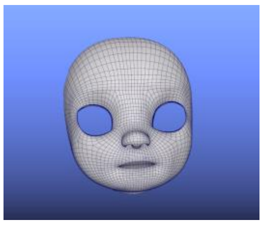
Footage / FACS Unit	Euclidean	Geodesics	Hybrid
 **AU27** **Jaw stretch**			
			
 **AU29.1** **Jaw thrust front**			
			
 **AU29.2** **Jaw thrust back**			
			

Table A.8: FACS units AU27, AU29.1 and AU29.2 (Jaw group).

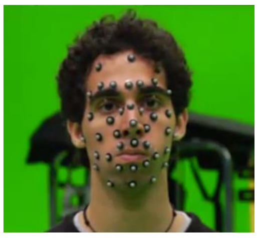
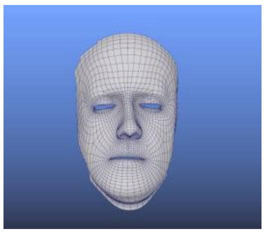
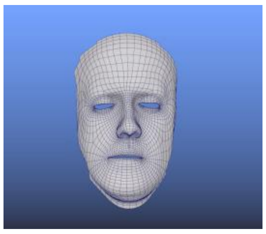
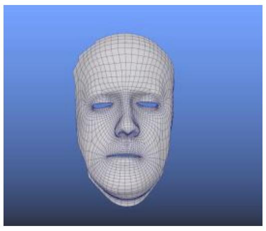
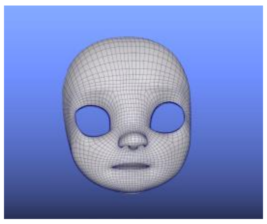
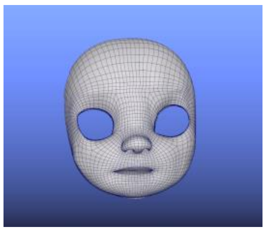
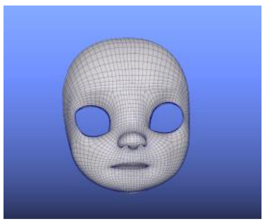
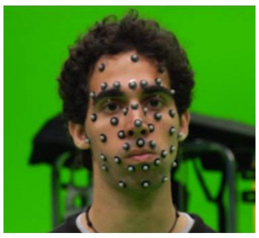
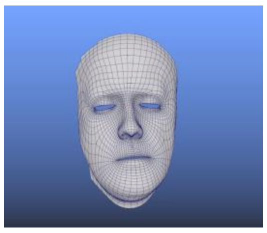
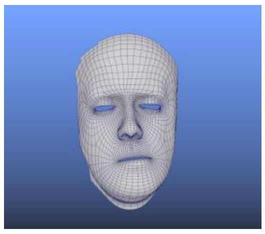
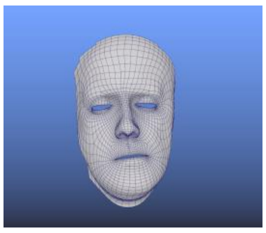
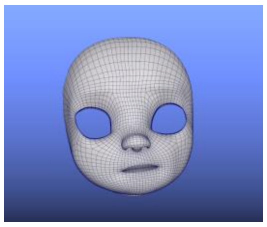
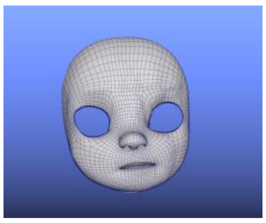
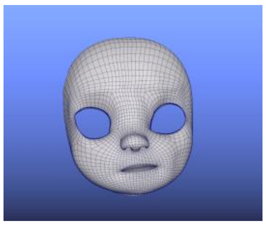
Footage / FACS Unit	Euclidean	Geodesics	Hybrid
 **AU30.1** **Jaw to the right**			
			
 **AU30.2** **Jaw to the left**			
			

Table A.9: FACS units AU30.1 and AU30.2 (Jaw group).