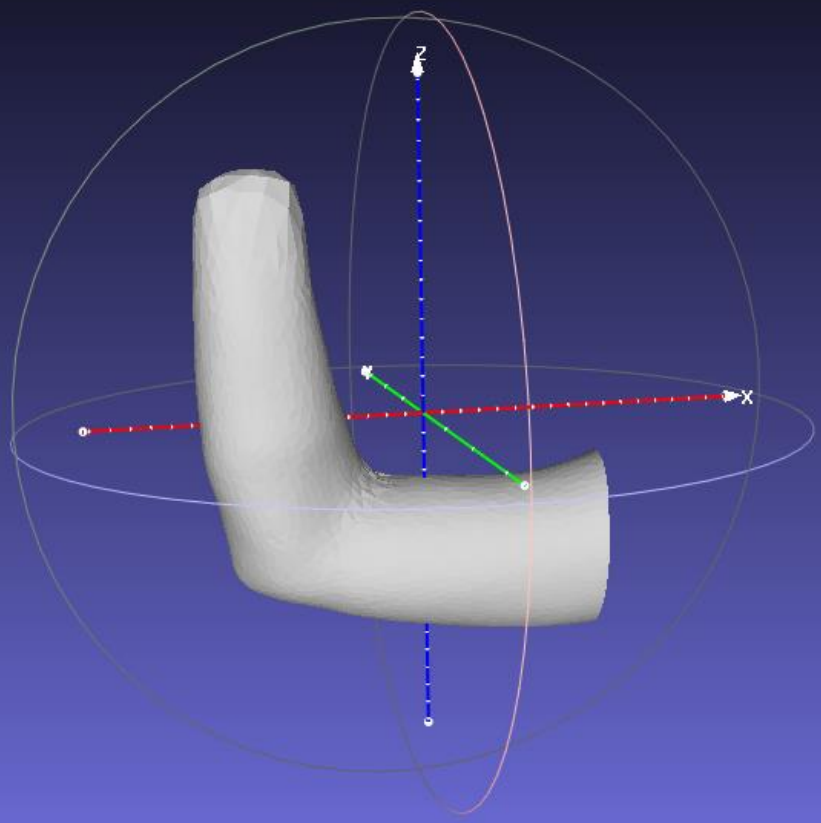
Technische Universiteit Delft

Integrating statistical shape modelling into a silhouette-based 3D reconstruction process to model a transradial defect

S.W.S. Goes



TU Delft

Integrating statistical shape modelling into a silhouette-based 3D reconstruction process to model a transradial defect

by

S.W.S. Goes

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on December 14, 2018 at 02:00 PM.

Student number:	4083741	
Project duration:	May 1, 2018 – December 14, 2018	
Thesis committee:	Dr. ir. D. H. Plettenburg,	TU Delft, daily supervisor
	Dr. ir. G. Smit,	TU Delft, support supervisor
	Dr. ir. T. Huysmans,	TU Delft, support supervisor
	Ir. J. S. Cuellar Lopez,	PhD candidate, support supervisor

This thesis is confidential and cannot be made public until December 31, 2020.

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft Delft University of Technology

Abstract

Background: In low- and middle-income countries there is a high demand for prosthetic devices. An automatic system, in which hand prostheses are manufactured with 3D printers can potentially offer a solution for patients having a transradial defect in these areas. As part of such a system, a detailed 3D model of the residual limbs is needed. In order to make the process of creating this 3D model more accessible to low- and middle-income countries, a new silhouette-based 3D reconstruction process is observed which can be implemented in a smartphone application.
Objective: Measure the accuracy of the observed method.
Methods: A database of artificial residual limbs and an experimental algorithm is created. This algorithm consists of two parts. The first part simulates the process of capturing pictures of a residual limb with a smartphone camera. The second part performs an automatic silhouette-based 3D reconstruction.
Results: When the reconstruction method is performed on a known residual limb shape with three silhouette images, the highest measured 3D reconstruction accuracy is 1.12 ± 0.57 mm. When the method is performed on an unknown residual limb shape with three silhouette images, the highest accuracy is 6.48 ± 2.15 mm.
Conclusion: This work presents a technique for reconstructing a residual limb by means of silhouette images. The observed method can be considered as a promising 3D reconstruction approach for prosthetic designing. The method could be improved by having access to a larger database of residual limb shapes and by analysing and finding the optimal input arguments for the optimiser.

Clinical Relevance: The observed method provides a low-cost and accessible approach to model a residual limb for the design of a fitting prosthetic socket that can be manufactured by a 3D printer.

Keywords: *3D printing, 3D scanning, Statistical Shape Modelling, SSM, residual limb, transradial amputee*

Preface

The second year of my master's programme was a tremendous learning experience for me. Prior to the thesis project, I went to Nepal for a three month internship at a disaster recovery and development organisation called Disaster Hack. This nongovernmental organisation established a partnership with a leprosy hospital to investigate 3D scanning and printing technology for the development of prosthetic devices. Together with a local prosthetist, I performed some 3D scanning experiments with low-cost scanners to model residual limbs. Also, I used an available 3D printer to produce an orthotic device that was created based on a patients 3D model. Unfortunately, the device was not a perfect fit, which was caused by a bad model. This experience encouraged me to do further research in the field of 3D scanning. At the Delft University of Technology, a research project is set up to create a higher production rate of prosthetic devices in low-income countries by using 3D scanning and printing technology. With the experience from Nepal, I was highly motivated to contribute to this project. As a result, this thesis project was conducted in the final stage of my studies in Biomedical Engineering.

I would like to thank everyone involved in the thesis project for their contribution in any shape or form. This project would not have been possible without your contributions. Some of you deserve a special mention. First of all, I would like to thank Toon Huysmans for his involved and pro-active attitude. We spent many hours of discussing specific subjects and you gave clear explanations on certain problems. Your feedback was always positive and helped me tremendously. I would like to thank Dick Plettenburg, who supported my idea of going to Nepal and who assisted me as daily supervisor during the whole project. Many thanks to Juan Cuellar Lopez, who was was very helpful, always positive and who gave me feedback on certain parts of the report before the final version. I would like to thank Gerwin Smit, for his tips and trick he gave about structuring the project and the report. During the summer I worked together with Gale Dewo. It was very nice to discuss certain problems with him as we were working on the same subject. And last of all, I would love to thank my wonderful friends and family who have supported me every step of the way.

Steven Goes
Den Haag, December 2018

Contents

List of Abbreviations

3DP	3D Printing
FoV	Field of View
GPA	Generalised Procrustes Analysis
LMICs	Low- and Middle Income Countries
PC	Principal Component
PCA	Principal Component Analysis
SD	standard deviation
SfS	Shape-from-Silhouette
SSIC	Statistical Shape Image Comparison
SSM	Statistical Shape Model
SVD	Singular Value Decomposition
VH	Visual Hull
VW	Viewing Width

1 Introduction

1.1. Background

About 30 million people in low- and middle-income countries (LMICs) are in need of orthotic and prosthetic devices [1]. However, only 5-15% of these people have access to such devices due to their high cost, a shortage of trained personnel and a lack of good infrastructure [2–4]. Three-dimensional printing (3DP) enables a rapid and cost-efficient way of producing prostheses and has promising influence on the inadequate supply of prosthetic devices in LMICs [5, 6]. Three-dimensional scanning of the residual limb can replace the casting process during the traditional socket fabrication since a digital 3D model is needed to design a prosthesis for 3DP. Three-dimensional scanning can be divided in passive and active methods [7]. In active scanning, the target is illuminated or scanned with some form of electromagnetic radiation, such as a laser or a white light. In passive scanning, a 3D model is created from a number of pictures. The latter is otherwise known as photogrammetry and uses trigonometry to calculate 3D points. Active scanners are mostly expensive and less accessible than ordinary cameras used for photogrammetry. A reasonable camera can be found in any smartphone [8]. Furthermore, smartphones are becoming the leading handset type in almost all countries globally by 2025 [9]. Because of this accessibility in combination with its camera, motion sensors and internet functionalities, the smartphone can be seen as an ideal tool for 3D scanning in LMICs. Using a smartphone for 3D modelling has already shown its potential in the medical field [10–12]. Several apps for 3D scanning exists but most of them need extra hardware or calibration tools [13]. Compared to other photogrammetry methods, Shape-from-Silhouette (SfS) is computationally simple [14] and the scanning procedure can be performed marker-less [15]. However, the digital model from SfS is often an inaccurate representation of the real shape. These SfS model outputs are also known as visual hull (VH). But the availability of large-scale anthropometric surveys using 3D body scanning technologies has enabled researchers to predict 3D body shapes from partial inputs, such as a small set of silhouette images [16, 17]. A statistical shape model (SSM) is designed with the survey models and can be used to morph and fit within the object outline in the silhouette image. This silhouette-based shape estimation will be called the statistical shape image comparison (SSIC) method. It compares projections of the SSM with silhouette images to validate and morph the 3D SSM into the desired shape. This technique has brought potential benefits to the e-commerce of wearable products but might just as well be used in the medical field for wearable devices such as prostheses.

1.2. Problem definition

People in LMICs are in need of prostheses. Manufacturing prostheses by means of 3D printing as part of an automated process can be a solution to meet the demand. In order to produce a custom-made prosthesis for 3D printing, a detailed 3D model of the patients residual limb is needed. Images of the limb taken by a smartphone can be used in combination with prior knowledge of stump shapes to construct a 3D model. The SSIC method is based on this idea and can be performed automatically. However, it is unclear how accurate this method is and if it is useful for prosthetic designing.

1.3. Goal

The goal is to measure the accuracy of the SSIC method using an artificial database of 3D residual upper limb models.

1.4. Thesis project

This thesis is part of a TU Delft Global Initiative project named "Access to prosthetics thanks to 3D printing and a smartphone app". The goal of the project is to make prosthetic devices easy accessible by automating the design process. A prosthetic hand model with no need for assembly, a parametric socket design, cheap materials for 3DP and a smartphone as a residual limb measuring tool will enable a fast and low-cost production. The process will be an end-to-end solution where the patient will scan his residual limb as input and a fitting prosthesis will be delivered as output. In between is a black box, where a fitting hand prosthesis is designed by the computer and automatically 3D printed. The project is meant particularly for Colombia.

1.5. Outline of this report

- **Chapter 2** describes the designing process of the SSM. The image acquisition setup will be discussed and the underlying principle of projecting 3D points onto a 2D plane will be explained as this is a fundamental part of the SSIC method. The image comparison error is defined and the SSIC output model validation methods are discussed.
- **Chapter 3** contains the results of the SSIC method. Furthermore, the VH construction results are shown and the influence of different input arguments for the optimiser are presented.
- **Chapter 4** interprets the results from chapter 3. The SSIC method accuracy is compared with the literature and some recommendations are given for a real life setup.
- **Chapter 5** concludes this research project.

2 Methods

2.1. Overview

To establish the goal defined in section 1.3, an experimental algorithm is created. One part of the algorithm is an image acquisition setup in a 3D rendering software. This will simulate the process of capturing pictures of a residual limb with a smartphone. The other part is the automatic silhouette-based 3D reconstruction process: the SSIC method. This chapter will describe the design of this experimental algorithm. First, a database of artificially constructed 3D transradial defects is composed. This database is made for two reasons: 1) the samples from the database are used as objects for the digital photo shoot and 2) several samples from the database are used to create an adjustable 3D model, also known as statistical shape model (SSM). The samples must meet certain requirements for the latter reason, therefore the description of composing the database will be done in combination with the explanation of statistical shape analysis in section 2.2. Next, an image acquisition setup is designed in section 2.3. The setup will enable the user to define the cameras position, pose and intrinsic parameters, such as the focal length. The projection, a linear transformation, of a 3D point onto a 2D plane is elaborated in more detail as this is a fundamental part of the SSIC method. Section 2.4 will describe how the SSIC method combines the SSM, the rendered images and the projection information to automatically reconstruct a 3D model. Finally, the validation of the SSIC method will be performed in two ways. Firstly, the performance of the SSIC method will be compared with the current practice. Secondly, the accuracy of the reconstructed models will be measured. How this is done will be described in section 2.5.

2.2. Designing an adjustable 3D model

2.2.1. Defining a 3D shape

A three-dimensional shape describes the external boundary form of the object. It is invariant to Euclidean similarity transformations, such as shifting and scaling [18–20]. A commonly used representation for a shape in computer graphics is a triangle mesh. It is a simplified version of the real world object and consists of 3D points connected by triangular surfaces. The points are called vertices, defined by their Cartesian coordinates. A mathematical representation of an n-point shape, S_i, in a 3D space is then:

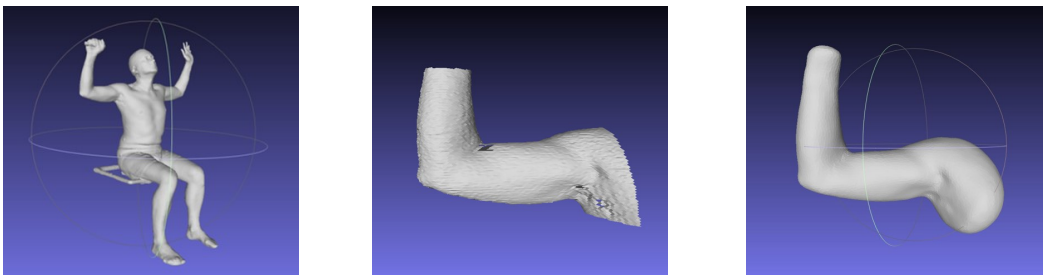
$$S_i = \left[x_1, y_1, z_1, x_2, y_2, z_2, \cdots, x_n, y_n, z_n\right]^T \tag{2.1}$$

The shape variability in a given set of shapes can be analysed with statistical methods. This will allow to create a shape similar to the shapes in the set. It is basically done by calculating a mean shape from a normal distribution of shapes and deform it in certain directions within the variance of the data. Such a morphable model is called a statistical shape model (SSM). This can be translated to the problem discussed in this thesis. A digital shape of a residual limb can be created with the variability values extracted from a dataset of residual limb shapes. For this, a training set of shape samples (formulated as in equation 2.1) is needed to obtain the statistical values. Important are the mean shape and the variance of the data. The next subsection will describe how the database of shapes is constructed. Furthermore, a detailed description of data handling is given. In the end, the formula is provided that constructs a digital residual limb from a training set of residual limb meshes.

2.2.2. Creating a shape database

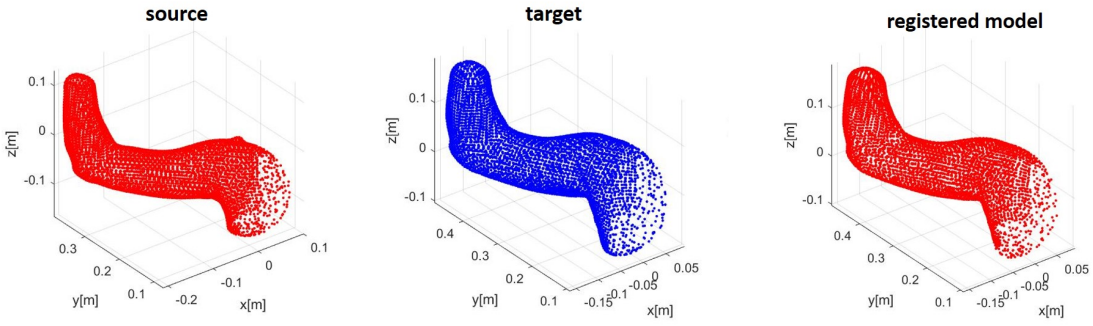
The database of 3D stumps is not only needed for the SSM. The models are also used as objects for the photo shoot in a 3D computer graphics software. The photos taken are then used for 3D reconstruction as mentioned in chapter 1. Making pictures and 3D scans of patient participants to acquire images and 3D models is time-consuming to conduct and to realise because of the requirements of a research ethics committee. Therefore an artificial database of residual limbs is constructed. The 3D stump models are established by modifying 41 accurate scans of healthy humans (figure 2.1a) aged between 18-65 years from the CAESAR database [21, 22]. First, the right arm and shoulder are separated from the body. All parts except the right arm and shoulder are removed. Secondly, the hand is cut off somewhere between the wrist and the cubital fossa, i.e. the elbow pit. The cutoff plane is perpendicular to the line between the cubital fossa and the wrist (see figure 2.1b). To create a diverse database, the cutoff point varies on the line between the cubital fossa and the wrist. Thirdly, a Screened Poisson Surface Reconstruction [23] is performed to fill the gaps and give it a natural look at the transradial cut (see figure 2.1c). This process will result in a artificial database of 41 transradial amputated limbs, where the cut is placed at different points in the forearm. This set of residual limbs will be called the raw database.

The next step is to make the dataset homologous, meaning that all the models in the database will have the same number of vertices with point-to-point correspondence between models (vertices 1 in model 1 is related to vertices 1 in model 2, 3 etc.). The raw database will then become a registered database. Creating a registered database is an iterative process where constantly the same mesh (source mesh) is rigidly aligned with other meshes (target meshes) and then non-rigidly deformed into their shapes. This is called a non-rigid Iterative Closest Point (non-rigid ICP) algorithm. The registered database is basically a database with source meshes in different shapes, therefore having the same number of vertices. The rigid alignment would have been problematic if the shoulders were not attached to the arms. The models would then have a boomerang shape, therefore making it difficult for the algorithm to determine which part is superior or inferior to the elbow in the alignment process. An example of the registration process is shown in figure 2.2.



(a) (b) (c)

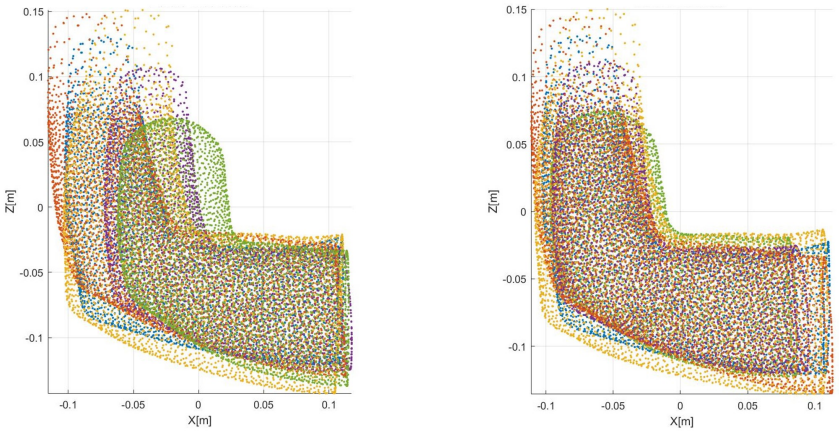
Figure 2.1: CAESAR data for residual limb model construction. (a) A subject from the CAESAR database (b) Residual limb after a Screened Poisson Surface Reconstruction and (c) One sample of the residual limb database.



(a) (b) (c)

Figure 2.2: non-rigid ICP, model samples are presented as a point cloud where each point is a vertex. (a) A source model (b) A target model (c) the target model after registration. This registered model is coherent to the source model and other registered models.

After the registration process, the shoulder part can be removed due to its unnatural shape. In the model, a cutoff plane is used to remove the shoulder part. This cutoff plane is manually placed in the part superior to the elbow, closer to the shoulder than the elbow, so that there is enough room for prosthetic designing. The vertices that are removed in one model, are also removed in the other models due to coherence of vertices between models. Every model will have the same percentage of its shape removed. This removal method is shape independent thus retaining the shape variability in the registered database. All the models in this registered database are translated and rotated to the centre of the world coordinate frame. From this registered database it is decided to take 35 samples for the training set, the remaining 6 samples will be used as a test set. Both the training set and the test set are used for the photo shoot in the 3D rendering software. An orthopaedic advisor validated and approved the shapes to be used. For the SSM, only the training set of 3D shapes is aligned in a common co-ordinate frame [19]. Generalised Procrustes Analysis (GPA), named after the Greek bandit who fitted passengers to his iron bed by stretching them or cutting off their legs, is the most popular method [20]. It basically calculates the mean shape from the set of shapes and minimises the mean squared distance between every shape and the calculated mean shape (see figure 2.3). The GPA brings the training set into a 'shape space' [18] and can thereafter be used for statistical analysis to generate a statistical shape model. In the next subsection the parameters to describe the variance in the training set will be calculated. It consists of applying a statistical procedure to find the most important variances. With that knowledge, a 3D shape can be created from a number of pictures.



(a) (b)

Figure 2.3: Generalised Procrustes Analysis (GPA), model samples are presented as point clouds. (a) A side view of the first five model samples before GPA. (b) A side view of the first five model samples after GPA.

2.2.3. Modelling of shape variability

The distribution parameters of the data can be determined when it is homologous and aligned with GPA. The data consist of shapes described by points in a 3D Cartesian coordinate system. The variance of the point distribution in different directions can be described by the eigenvectors and corresponding eigenvalues. Eigenvectors are a set of orthogonal axes, each pointing towards the main directions of variation. The eigenvalue is a number describing how spread out the data is in the direction of the eigenvector. The eigenvector with the highest eigenvalue describes the most important variance of the data. Principal component analysis (PCA) is a statistical procedure that observes the data, extracts the eigenvectors and eigenvalues and stores the eigenvalues in a decreasing magnitude with the corresponding eigenvectors accordingly. The eigenvector with the highest eigenvalue, thus the direction with the highest variance in the data, is the first principal component (PC). In case of shape change, PCA allows to find most important directions to transform a model. Least important directions have a small variance and can therefore be neglected, reducing the dimensionality. Each shape is represented by a vector S_i of coordinates (equation 2.1). The set of shapes will be represented as a matrix by concatenating all the shapes. This is called the matrix of shapes, where n_s is the total number of samples in the training dataset:

$$M = \begin{bmatrix} S_1 & S_2 & S_3 & \cdots & S_{n_s} \end{bmatrix} \tag{2.2}$$

Principal component analysis is usually explained via an eigen-decomposition of the covariance matrix. However, it can also be performed via singular value decomposition (SVD) of the data matrix: the matrix of shapes M. This method is computationally less demanding by using smaller matrices [24]. The matrix of shapes needs to have a zero mean to perform PCA via SVD. Therefore the mean shape is calculated:

$$\overline{S} = \frac{1}{n_s} \sum_{i=1}^{n_s} S_i \tag{2.3}$$

The mean shape also describes the best guess to determine a shape of a residual limb. This shape is a good starting point to use for the shape creation. The mean is then subtracted from each shape to solely focus on the variance. This results in a row-centred matrix of shapes:

$$X_c = \left[S_1 - \overline{S}, \quad S_2 - \overline{S}, \quad S_3 - \overline{S}, \quad \cdots \quad , S_{n_s} - \overline{S} \right] \tag{2.4}$$

The eigenvectors and eigenvalues can be extracted from the covariance matrix of X_c. The covariance matrix is a symmetric matrix so it can be diagonalized:

$$C = PLP^T \tag{2.5}$$

where C is the covariance matrix of the data X_c, P is an orthogonal matrix consisting of only eigenvectors, and L is a diagonal matrix with eigenvalues λ in the decreasing order on the diagonal. Through algebra, it is shown that SVD on the centred data, X_c, results in the same matrices but more efficiently [19, 24, 25]. The eigenvectors, or shape modes, can be adjusted to nonrigidly change the shape of a model (more on this in subsection 2.2.4). The most important modes can be determined knowing the eigenvalues over the whole dataset. A 'scree plot' (figure 2.4), can show the influence of the modes on the data. If the eigenvalue of a principal component is high, the influence of that mode is high on the shape change.

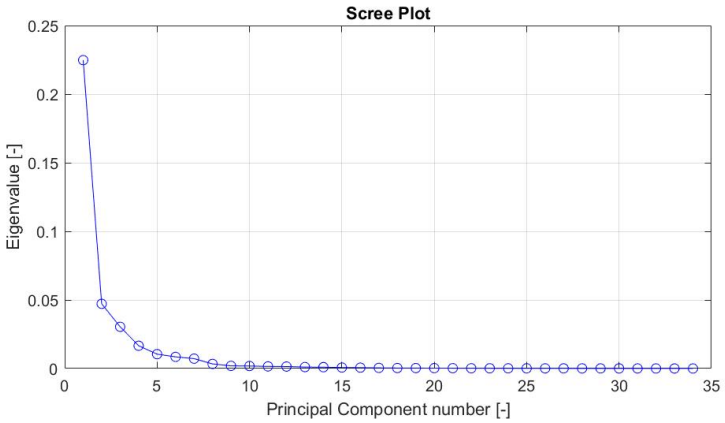


Figure 2.4: Scree plot of the training dataset

From this graph, it is clear that the principal components are ordered in a decreasing manner. A dimensionality reduction can be performed. The influence is calculated and shown in table I.

Table I: The influence of the principal components on the data

PC [-]	The fraction of total variance in the data [%]
1	62
1, 2, 3	83
1, 2, 3, 4, 5, 6, 7	95

The 7 first principal components are accounting for 95% of the total variance in the data. This indicates that with only the first 7 PCs , a relatively reliable model can be made. When a digital model of a residual limb is constructed, the 7 first principle components will make a residual limb that is good for 95% of the population.

2.2.4. Adjusting a 3D shape

All the statistical parameters are now available to create a 3D adjustable model. Using the obtained shape modes **P** and the corresponding mode variances λ, a new shape X can be created with the following linear equation:

$$X = \overline{S} + \mathbf{P}b \tag{2.6}$$

where $\overline{S}$ is the mean shape from the training set presented as a $3n$ x 1 vector, with n the number of vertices, **P** is a matrix with eigenvectors (i.e. the principal components) presented as a $3n$ x $(m-1)$ matrix, with n the number of vertices and m the number of samples from the training set and b is a vector with weight values [19], presented as $b = [b_1, b_2, b_3, \ldots, b_{m-1}]^T$. The dimensions of the principal components are at most $m-1$ [26]. Vector b can adjust the eigenvectors in the **P** matrix, therefore having influence on the shape change. However, the values in b are constraint to keep the shape within the normal distribution of shapes from the training set:

$$-3\sqrt{\lambda_i} \leq b_i \leq 3\sqrt{\lambda_i} \tag{2.7}$$

The 3D model X is a result of tweaking the mean shape by adjusting the weight values in b as described in formula 2.6. This adjustable model is called a statistical shape model (SSM) as it uses statistical information to modify the shape. The weight vector b has a considerable impact on the shape outcome. Parameter adjustments of the weight vector can be done by using silhouette images. It has already been shown that the shape of a foot can be constructed with 3 silhouette pictures as weight inputs [27]. The principle behind this method will be discussed in section 2.4. For the design of a transradial prosthesis, the same method can be applied. With 3 pictures of the patients stump and a dataset of stump shapes, a 3D model of the patients stump can be constructed. Using stump models from the available database of stump shapes described in subsection 2.2.2, a photo shoot can be carried out in the computer with 3D rendering software. The next section will describe how a rendering scene is set up for image acquisition.

2.3. Generating images of a 3D model

2.3.1. Setting up a rendering scene

A graphics application software is used to setup a digital photo shoot. The software should be able to easily change different parameters like the type of camera, the location of each camera, the angle of each camera and the number of cameras. Scripting the setup is therefore useful. Blender, an open-source 3D computer graphics software, is well suited for this job. If the script is run, the computer will setup the digital scene, take photos of the residual limb mesh and extract the camera parameters of each camera (see figure 2.5). The camera parameters will further be discussed in subsection 2.3.2.

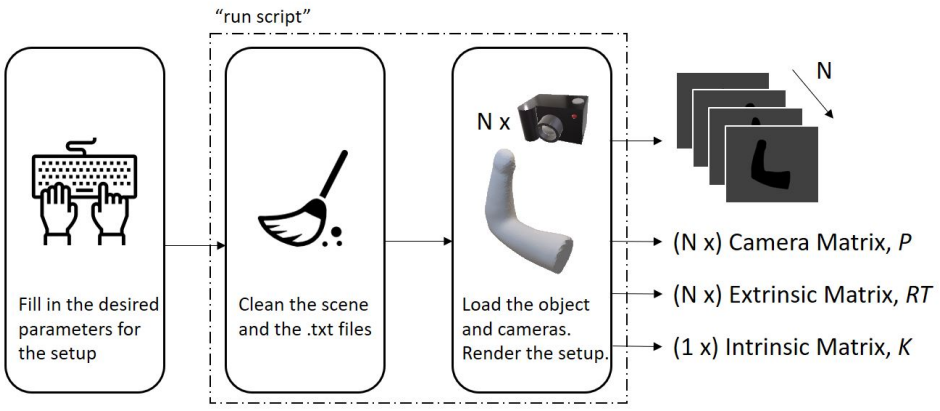


Figure 2.5: A visual flowcharts from the Blender source code

The camera's position in the test setup is described with spherical coordinates (r, ϕ, θ), where r is the radial distance, θ the azimuthal angle and ϕ the polar angle (see figure 2.6a). The radial distance can be set to any positive value, but 0.5 m is mostly used in this thesis as the residual limb is clearly visible on the pictures within the field-of-view (FoV) of each camera. It is not possible to put a full sphere of cameras around a residual limb in real life because the body of the patient will stand between the cameras and the limb, so the camera position will be constraint to be located on half of the sphere around the object. Furthermore, the azimuthal angle should start from $\frac{1}{2}\pi$ because the residual limbs in the constructed database are aligned with the x-axis. The Cartesian coordinates of each camera's centre can be calculated with the following formulas:

$$C_X = r \cdot sin(\phi) \cdot cos(\theta) \tag{2.8}$$

$$C_Y = r \cdot sin(\phi) \cdot sin(\theta) \tag{2.9}$$

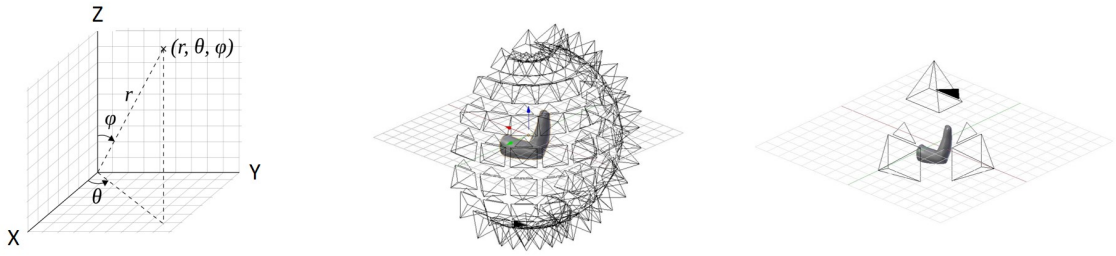
$$C_Z = r \cdot cos(\phi) \tag{2.10}$$

With the following constraints on the angles ϕ and θ:

$$0 \leq \phi \leq \pi \tag{2.11}$$

$$\frac{1}{2}\pi \leq \theta \leq 1\frac{1}{2}\pi \tag{2.12}$$

The azimuthal and polar angle will determine the number of cameras. Within the loop to position the cameras, the angle parameters will have an equal value on the step size to generate cameras equally on the spherical area around the object. The angle steps are: $\frac{1}{2}\pi$, $\frac{1}{3}\pi$, $\frac{1}{4}\pi$, $\frac{1}{5}\pi$, $\frac{1}{6}\pi$, $\frac{1}{7}\pi$, $\frac{1}{8}\pi$, $\frac{1}{9}\pi$, $\frac{1}{10}\pi$. This results in 9 camera setups with the following number of cameras: 5, 10, 17, 26, 37, 50, 65, 82 and 101 (see figure 2.6b). Also a camera setup with 3 cameras on the orthogonal axes is taken into account (figure 2.6c). This sets the number of camera setups to 10. When the cameras are positioned, the "look-at" direction of all cameras is set to the centre of the world coordinate frame where the residual limb mesh is located. The number of cameras and the camera locations can be determined by filling in the radius value and the angle step size. The type of camera, with the number of pixels can also be chosen and will be discussed in subsection 2.3.2.



(a) (b) (c)

Figure 2.6: (a) A representation of spherical coordinates. (b) A camera setup with 101 cameras on a half spherical area. (c) A camera setup with 3 cameras on the 3 orthogonal axes.

The SSIC method is investigated in this thesis report as it can be done by a smartphone which is easy accessible. Another practical argument to use this method is that it can make models with a small amount of silhouette images as mentioned in chapter 1. It is interesting to investigate the quality of 3D models created by the SSIC method when only 1, 2 or 3 cameras are used. When such a small amount of photos need to be taken, the method becomes low effort and easier to perform by the user. Many experiments are therefore conducted with only 3 cameras as shown in figure 2.6c, because that occupies all the orthogonal axes and can hypothetically produce enough information for reasonable models [28]. To reconstruct a 3D model from pictures, it is important to know the calibration parameters of the camera. Camera calibration is the process of determining accurate specifications for the camera's intrinsic and extrinsic parameters. These parameters consist of focal length, number of pixels in the pixel grid, object-camera distance, object-camera relative rotation and more. It is important to know how a 3D point is projected onto a 2D image to generate the calibration parameters in Blender. Therefore the next section will explain how these parameters are obtained and how they are related to the projection process.

2.3.2. Modelling a camera

Pinhole camera model

This section will show a mathematical model to describe how 3D world points get projected into 2D pixel coordinates. The camera is considered to be an ideal pinhole camera: the aperture is described as a point and no lenses are used to focus light. The pinhole is the optical centre and also the centre of the camera coordinate system, in figure 2.7 this is C. In a physical pinhole model, the light rays will cross the pinhole and afterwards land on the image plane, causing an upside-down image behind the pinhole (from a perspective of a 3D point in space). In computer vision, the image plane is flipped and placed between the centre of the camera coordinate system, C, and the object in the 3D world (see figure 2.7). In that case, the image is not upside-down which makes the results of projections easier to understand. The sequence of transformations can be described by a matrix equation. The matrix equation consists of two matrices: the extrinsic matrix and the intrinsic matrix. Multiplying these matrices with a 3D point will result in its 2D location on the pixel grid. Both matrices will now be explained, starting with the extrinsic matrix.

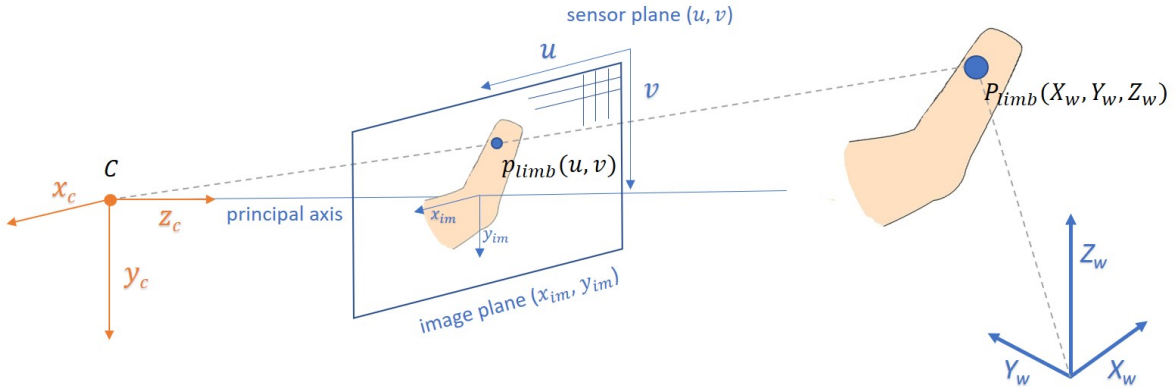


Figure 2.7: The pinhole camera model with a residual arm as an object.

Extrinsic matrix

The extrinsic matrix is a rigid transformation matrix. This means that the point in space will be unaffected but the point of view is changed. When looking at figure 2.7, the transition can be formulated as follows:

$$P_{limb}(X_w, Y_w, Z_w) \Rightarrow P_{limb}(x_c, y_c, z_c) \tag{2.13}$$

Blender provides the location and orientation of the camera in the world coordinate frame. The extrinsic matrix can be determined using this information. Converting the Euler angles into a rotation matrix and using the location as a translation vector will describe the camera location in the world coordinate frame. Inverting this transformation matrix will result in describing the world coordinates in the camera frame. That result is the extrinsic matrix. So the extrinsic matrix is created by making a rigid transformation matrix that describes the camera's pose and then take it's inverse.

The following will describe the transformation matrix that describes the camera pose. Given a set Euler angles $(\theta_x, \theta_y, \theta_z)$, the rotation matrix can be described as follows

$$R_c = Rot(Z,\theta_z) \cdot Rot(Y,\theta_y) \cdot Rot(X,\theta_x) \tag{2.14}$$

$$R_c = \begin{bmatrix} cos(\theta_z) & -sin(\theta_z) & 0 \\ sin(\theta_z) & cos(\theta_z) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} cos(\theta_y) & 0 & sin(\theta_y) \\ 0 & 1 & 0 \\ -sin(\theta_y) & 0 & cos(\theta_y) \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\theta_x) & -sin(\theta_x) \\ 0 & sin(\theta_x) & cos(\theta_x) \end{bmatrix} \tag{2.15}$$

$$R_c = \begin{bmatrix} r_{c1,1} & r_{c1,2} & r_{c1,3} \\ r_{c2,1} & r_{c2,2} & r_{c2,3} \\ r_{c3,1} & r_{c3,2} & r_{c3,3} \end{bmatrix} \tag{2.16}$$

The world coordinates of the cameras centre (C_X, C_Y, C_Z) are used to make the translation vector:

$$C = [C_X, C_Y, C_Z]^T \tag{2.17}$$

The following equation will describe the transformation matrix:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = R_c \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} + C \tag{2.18}$$

By adding a one at the bottom of the world coordinates we can simplify equation 2.18:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} r_{c1,1} & r_{c1,2} & r_{c1,3} & C_X \\ r_{c2,1} & r_{c2,2} & r_{c2,3} & C_Y \\ r_{c3,1} & r_{c3,2} & r_{c3,3} & C_Z \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{2.19}$$

The transformation matrix that describes the camera's pose is then $[R_c|C]$. The matrix is made square by adding an extra row of [0,0,0,1] (see equation 2.20). This will allow to further decompose this matrix into a rotation followed by translation. Then the extrinsic matrix $[R|t]$ is obtained by inverting the camera's pose matrix:

$$\left[\begin{array}{c|c} R & t \\ \hline \mathbf{0} & 1 \end{array}\right] = \left[\begin{array}{c|c} R_c & C \\ \hline \mathbf{0} & 1 \end{array}\right]^{-1} \tag{2.20}$$

$$= \left(\left[\begin{array}{c|c} I & C \\ \hline \mathbf{0} & 1 \end{array}\right] \left[\begin{array}{c|c} R_c & 0 \\ \hline \mathbf{0} & 1 \end{array}\right] \right)^{-1} \tag{2.21}$$

$$= \left[\begin{array}{c|c} R_c & 0 \\ \hline \mathbf{0} & 1 \end{array}\right]^{-1} \left[\begin{array}{c|c} I & C \\ \hline \mathbf{0} & 1 \end{array}\right]^{-1} \tag{2.22}$$

$$= \left[\begin{array}{c|c} R_c^T & 0 \\ \hline \mathbf{0} & 1 \end{array}\right] \left[\begin{array}{c|c} I & -C \\ \hline \mathbf{0} & 1 \end{array}\right] \tag{2.23}$$

$$= \left[\begin{array}{c|c} R_c^T & -R_c^T C \\ \hline \mathbf{0} & 1 \end{array}\right] \tag{2.24}$$

As seen in equation 2.23, the inverse of an orthonormal matrix is its transpose and the translation vector C is made negative. We can now define the relationship between the extrinsic matrix $[R|t]$ and the cameras pose $[R_c|C]$:

$$[R|t] = [R_c^T | -R_c^T C] \tag{2.25}$$

$$[R|t] = \begin{bmatrix} r_{1,1} & r_{1,2} & r_{1,3} & t_x \\ r_{2,1} & r_{2,2} & r_{2,3} & t_y \\ r_{3,1} & r_{3,2} & r_{3,3} & t_z \end{bmatrix} \tag{2.26}$$

The extrinsic matrix can transform a 3D point in the world frame to the camera frame:

$$\begin{bmatrix} x_C \\ y_C \\ z_C \end{bmatrix} = [R|t] \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{2.27}$$

$$\begin{bmatrix} x_C \\ y_C \\ z_C \end{bmatrix} = \overbrace{\begin{bmatrix} r_{1,1} & r_{1,2} & r_{1,3} & t_x \\ r_{2,1} & r_{2,2} & r_{2,3} & t_y \\ r_{3,1} & r_{3,2} & r_{3,3} & t_z \end{bmatrix}}^{\text{extrinsic matrix } [R|t]} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{2.28}$$

Intrinsic matrix

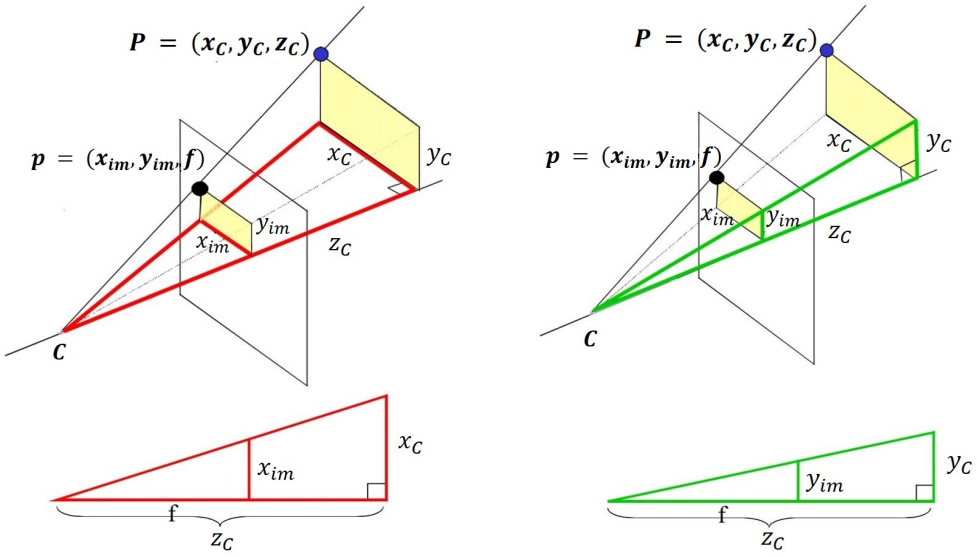
The intrinsic matrix, commonly referred as the K matrix, is composed of quantities intrinsic to the camera: focal length, f, principle point (u_0, v_0) which is the point where the principal axis intersects with the image plane (see figure 2.7) and pixel skew s. In this thesis it is assumed that pixels have no skew (s = 0), and are square. It is also safe to assume that the principal point is at the centre of the image if an image has not been cropped. The scanning tool is a smartphone, therefore intrinsic parameters of a smartphone are used. In this thesis, the parameters of an iPhone 6 are used [1].

The extrinsic matrix enables to look at a point in space from the camera's local frame perspective. From this frame it is easy to project a 3D point on the image plane by using the rule of similar triangles as it is shown in figure 2.8. The 2D location on the image plane is then described as follows:

$$x_{im} = x_C \frac{f}{z_C} \tag{2.29}$$

$$y_{im} = y_C \frac{f}{z_C} \tag{2.30}$$

All the 2D coordinates on the image plane are at a constant distance of $z_{im} = f$ from the camera's centre. This constant value can be neglected. The result is that the coordinates on the image plane are defined with respect to the centre of the plane: the principal point.



(a) Similar triangles X

(b) Similar triangles Y

Figure 2.8: Similar triangles X & Y [29]

The final step in the transformation process is to describe points on the image plane in terms of pixels. For this, the image plane is treated as a grid of discrete elements, ordered from top to bottom, and left to right. The origin of the pixel grid therefore lies on the top-left corner of the image plane (see figure 2.9). This is a translation and can be described by adding the principal point coordinates to the image locations in the image coordinate frame:

$$u = x_C \frac{f}{z_C} + u_0 \tag{2.31}$$

$$v = y_C \frac{f}{z_C} + v_0 \tag{2.32}$$

[1] The parameters are found at `https://www.devicespecifications.com`.

This can be combined in a matrix notation. An extra row with a one is added to eventually simplify the notation for 3D coordinates. The following will show why:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} x_C \frac{f}{z_C} + u_0 \\ y_C \frac{f}{z_C} + v_0 \\ 1 \end{bmatrix} \tag{2.33}$$

Both sides are multiplied with z_c:

$$\begin{bmatrix} u\, z_C \\ v\, z_C \\ z_C \end{bmatrix} = \begin{bmatrix} x_C\, f + u_0\, z_C \\ y_C\, f + v_0\, z_C \\ z_C \end{bmatrix} \tag{2.34}$$

This is simplyfied as follows:

$$z_C \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \overbrace{\begin{bmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix}}^{\text{intrinsic matrix K}} \begin{bmatrix} x_C \\ y_C \\ z_C \end{bmatrix} \tag{2.35}$$

The matrix shown in equation 2.35 is the intrinsic matrix assuming that there is no skew and pixels are square. In formulas 2.19 and 2.33 an extra dimension to the coordinates is added with the value 1 to simplify linear calculations. When this extra dimension is added, the coordinates are also known as homogeneous coordinates. Now it is possible to combine the sequence of transformations shown in equations 2.28 and 2.35 in one matrix, called the camera matrix or (camera) projection matrix P.

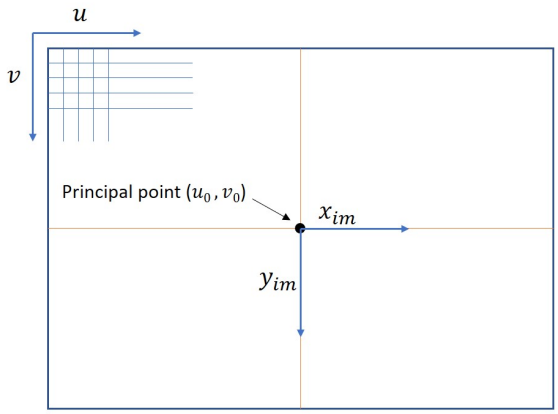


Figure 2.9: The image plane and pixel plane. Both coordinate frames are on the same plane but have a different origin.

Camera matrix

The extrinsic matrix and the intrinsic matrix can be combined in one matrix. The equation of the sequence of transformations has the following form:

$$w \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{2.36}$$

$$\begin{bmatrix} uw \\ vw \\ w \end{bmatrix} = \overbrace{\begin{bmatrix} f \cdot r_{11} + u_0 \cdot r_{31} & f \cdot r_{12} + u_0 \cdot r_{32} & f \cdot r_{13} + u_0 \cdot r_{33} & f \cdot t_x + u_0 \cdot t_z \\ f \cdot r_{21} + v_0 \cdot r_{31} & f \cdot r_{22} + v_0 \cdot r_{32} & f \cdot r_{23} + v_0 \cdot r_{33} & f \cdot t_y + v_0 \cdot t_z \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix}}^{\text{camera matrix P}} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{2.37}$$

It is known that $w = z_C$ as shown in equation 2.35. Factor w is also known as a scale factor. As we can see by going from equation 2.34 back to 2.33 the pixel coordinates are acquired by dividing the first two coordinates by the last.

The object is a polygon mesh, consisting of vertices and faces. Each vertex has X_w, Y_w, Z_w coordinates. So for object i with n vertices the following equation converts the 3D object to a picture:

$$\begin{bmatrix} u_{i,1}w_{i,1} & u_{i,2}w_{i,2} & \cdots & u_{i,n}w_{i,n} \\ v_{i,1}w_{i,1} & v_{i,2}w_{i,2} & \cdots & v_{i,n}w_{i,n} \\ w_{i,1} & w_{i,2} & \cdots & w_{i,n} \end{bmatrix} = \begin{bmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} X_{i,1} & X_{i,2} & \cdots & X_{i,n} \\ Y_{i,1} & Y_{i,2} & \cdots & Y_{i,n} \\ Z_{i,1} & Z_{i,2} & \cdots & Z_{i,n} \\ 1 & 1 & \cdots & 1 \end{bmatrix} \tag{2.38}$$

Then dividing the the first two rows by the last will result in the pixel coordinates of the vertices in the pixel frame:

$$\frac{1}{\begin{bmatrix} w_{i,1} & w_{i,2} & \cdots & w_{i,n} \end{bmatrix}} \begin{bmatrix} u_{i,1}w_{i,1} & u_{i,2}w_{i,2} & \cdots & u_{i,n}w_{i,n} \\ v_{i,1}w_{i,1} & v_{i,2}w_{i,2} & \cdots & v_{i,n}w_{i,n} \\ w_{i,1} & w_{i,2} & \cdots & w_{i,n} \end{bmatrix} = \begin{bmatrix} u_{i,1} & u_{i,2} & \cdots & u_{i,n} \\ v_{i,1} & v_{i,2} & \cdots & v_{i,n} \\ 1 & 1 & \cdots & 1 \end{bmatrix} \tag{2.39}$$

The SSIC method compares images taken in Blender with the projections of the SSM. Both images must have the same point of view and intrinsic parameters in order to make a reliable comparison. Therefore the projection matrix (equation 2.36) extracted from Blender is used in the 3D reconstruction process in MATLAB. The next section will explain how this is done and how the SSIC algorithm is organised.

2.4. Constructing the SSIC method

In the previous sections the SSM and the projection matrix have been discussed. This section will explain how both can be used to create a model of a residual limb from a set of silhouette images. A silhouette image, I_{ssm}, can be created from the 3D statistical shape model by using a projection matrix and a threshold on the pixel intensity values. This image (I_{ssm}) can be compared to the silhouette image taken of the patient, I_p, where the latter image is taken as ground truth. During calculations the pixel grid of a silhouette image is represented as a matrix with binary values, where the object value is "1" and the background has a value of "0". Overlapping the images will generate a certain error value that is used to measure the correctness of the statistical shape change. The error can be expressed as the number of nonzero values after subtracting I_{ssm} from I_p. If the images are exactly the same, the result of subtracting I_{ssm} from I_p will generate a matrix with only zeros thus having no nonzero values meaning no error. The error can be described more specifically as the number of incorrect pixels per pixel grid. This can be mathematically explained. First, I_{ssm} is subtracted from I_p to get a matrix with nonzero values:

$$I = | I_p - I_{ssm} | \tag{2.40}$$

Then, all the nonzero values are summed up and divided by the total amount of elements in I to get the local error from one camera viewpoint, i.e. one image comparison:

$$e_{cam} = \frac{1}{(m \cdot n)} \sum_{i=1}^{m} \sum_{j=1}^{n} I_{ij} \tag{2.41}$$

Finally the global error, expressed in percentage, over all viewpoints is calculated with the following formula:

$$E = \frac{1}{N_{cam}} \Big(\sum_{cam=1}^{N_{cam}} e_{cam} \Big) \cdot 100\% \tag{2.42}$$

The comparison of pictures while changing the SSM can be seen as an optimisation problem. The goal is to minimise the global error E. Three different sets of parameters have influence on finding the right shape: the translation vector T, the rotation matrix R and the weight values b. The initial position of the projected SSM is at the centre of the image. If, for example, the object in I_p is more to the left of the image, the SSM must have the freedom to translate to the left with respect of the camera's centre before projection. This way, the

projected SSM appears more to the left in I_{ssm} thus having a better fit with I_p in this example. To be able to translate and rotate the SSM, the following formula is used:

$$\mathbf{X}_{ssm} = \mathbf{R} \cdot \mathbf{X}^T + \mathbf{T} \tag{2.43}$$

where $\mathbf{R}$ is the same 3 x 3 matrix as in formula 2.15, $\mathbf{X}^T$ is X from formula 2.6 transformed to a 3 x n matrix, with n representing the number of vertices and $\mathbf{T}$ is a matrix of 3 x n where every column is the same translation vector $[T_X, T_Y, T_Z]^T$. So the parameters that the optimisation process will optimise are: $\theta_x, \theta_y, \theta_z, T_x, T_y, T_z$ and a number of weight values, b, depending on the number of principal vectors in $\mathbf{P}$ (see formula 2.6). The process of SSIC is visually explained in figure 2.11. The angle of view (AOV), 2α, and the known distance, D, between the camera's centre and the object centre can be used to calculate the viewing width VW. The viewing width is used to validate the outcome of the projected X_{ssm} after translation. As shown in figure 2.10, the VW is calculated by using similar triangles. This is previously done in figure 2.8 but the physical representation of the pinhole model makes it much clearer. By substituting $\alpha = tan^{-1}(\frac{u_0}{f})$ into $VW = 2\ D\ tan(\alpha)$ the viewing width is calculated.

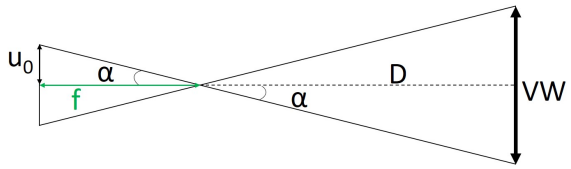


Figure 2.10: Top-down view of the pinhole model (physical representation). The geometrical parameters are used to calculate the viewing width (VW)

As mentioned in subsection 2.3.2, the camera's intrinsic parameters are adjusted so that they match the parameters of an iPhone 6. This is done because this device is readily available and can be used if real life tests are planned. But the camera sensor has a high resolution pixel grid: 2448 x 3264 pixels (8MP rear camera). The algorithm constructed in MATLAB to simulate the SSIC, downgrades both I_p and I_{ssm} to a size of 630 x 840 pixels. The 630 x 840 is the default window setting of MATLAB. It makes the algorithm more efficient but lower resolution images will result in a less accurate 3D model. Using the SSIC method in real life with a smartphone will require instructions on the smartphone for posing the camera with respect to the residual limb. Cameras must be calibrated to get accurate 3D reconstructions. With a digital test setup, it can be investigated how uncalibrated cameras have effect on the 3D reconstruction. This can be done by slightly changing the camera pose while keeping the projection matrices constant. A link to the source code of the SSIC method can be found in Appendix A.

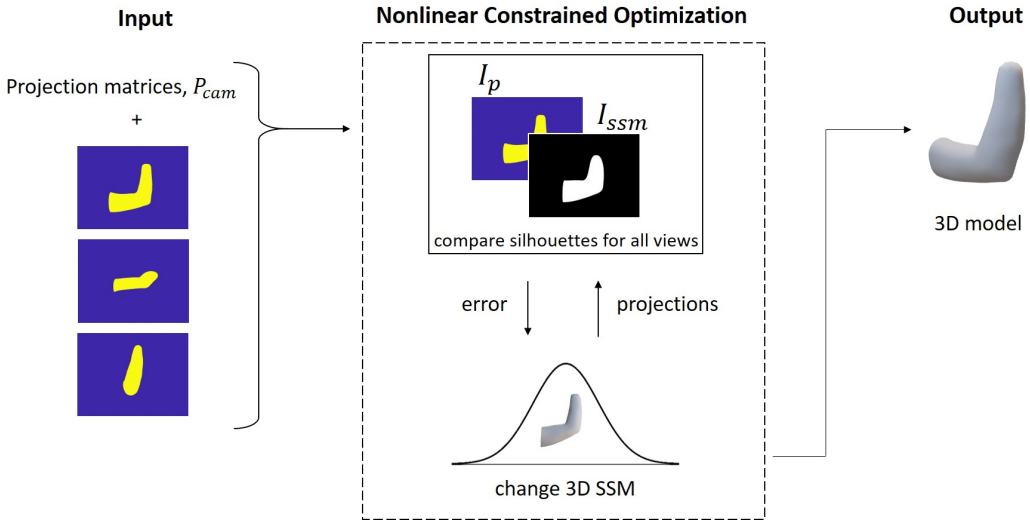


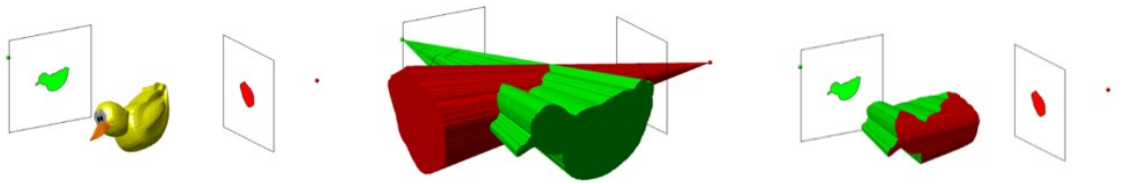
Figure 2.11: Statistical Shape Image Comparison (SSIC) method visual flowchart

2.5. Validating the SSIC method

2.5.1. Comparing the SSIC method with the VH construction method

This section consist of two parts. First, the SSIC method will be compared with the common practice in SfS algorithms. As mentioned in chapter 1, this common practice is the visual hull (VH) construction. The concept of the VH construction will be described to have a better understanding on its working principle. This will give more insight in the difference between the performance of the SSIC method and the VH construction method. The VH construction is mainly observed in this thesis and used as a comparative method to validate accuracy outcome and speed performance of the SSIC method. The VH will be compared with the original 3D model from which the photos are taken by means of the Hausdorff distance. The Hausdorff distance will be explained in subsection 2.5.2. Second, it is described how the accuracy of the SSIC method is measured in subsection 2.5.2. This is done by measuring the distance between the original 3D model from the database and the reconstructed 3D model.

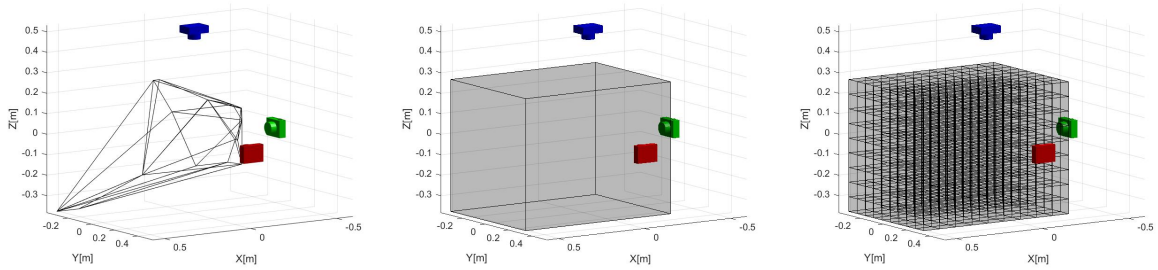
A visual hull is a volume of intersecting cones coming from different camera viewpoints. It is defined by Laurentini [30, 31], as the largest object that consists with all possible silhouettes. The object projection in the silhouette will determine the cross sectional shape of the cone as the cone intersects with the image. Therefore, the silhouette can be seen as a mask where the cone rays only pass through the area that labels the object in the picture. A visual hull of a duck is presented in figure 2.12 . The cones only pass the mask within the contour of the object (see figure 2.12b). In figure 2.12c a visual hull is constructed out of the intersecting cones. The visual hull is always an overestimation considering that a finite set of silhouettes is used [32]. In order to calculate the 3D coordinates of the object point in space, the cameras must be calibrated, meaning that the camera intrinsic and extrinsic parameters must be known.



(a) (b) (c)

Figure 2.12: Visual Hull of a duck [33]. (a) Two silhouette images are presented with a duck in between. (b) Visual cones are intersecting. (c) A resulting VH

There are several methods to construct a mesh out of intersecting cones. The voxel based method is popular as it does not suffer from numerical instabilities that generate incomplete or corrupt surface models. A common field of view can be generated by back projecting rays from all the calibrated cameras (figure 2.13a). A bounding box is constructed that encloses the field of view (figure 2.13b). Then the bounding box will be segmented into smaller blocks: voxels (figure 2.13c).



(a) (b) (c)

Figure 2.13: Constructing the bounding box for the VH. (a) A common field of view. (b) A bounding box. (c) Voxels within the bounding box

A criteria function $f(x,y,z)$ is used to label each voxel as being inside, outside or on the boundary of the visual hull. The outcome of the function will be -1 if the projection of the voxel lies in the background of all the silhouettes (the black area in the picture) and will be +1 if the voxel projection is in the foreground (the white area in the picture) of all the silhouettes. When the function results in $f(x,y,z) = 0$, part of the voxel lies inside the foreground and a part of the voxel lies inside the background (figure 2.14a). Such a voxel is called transverse, meaning that any of its edges intersects with the boundary of the objects contour. Every transverse voxel will be decomposed in smaller triangular pyramids (tetrahedron, figure 2.14b) where every corner is assessed to lie in- or outside of the object. With 4 corners per tetrahedron, each having the option to lie in- or outside (1 or 0) of the object makes $2^4 = 16$ different options of corner combinations per tetrahedron (figure 2.14c). Each of these 16 corner combinations is related to a certain polygonization solution that can be done within the tetrahedron (figure 2.14d). These solutions are available in a list that works as a digital lookup table. This method is called an Implicit Surface Polygonizer [34]. The model consists of a low number of vertices (around 300). A patch remesher algorithm is used to give the model a comparable amount of vertices with the SSM while keeping the same shape. The procedure subdivides the edges with more vertices.

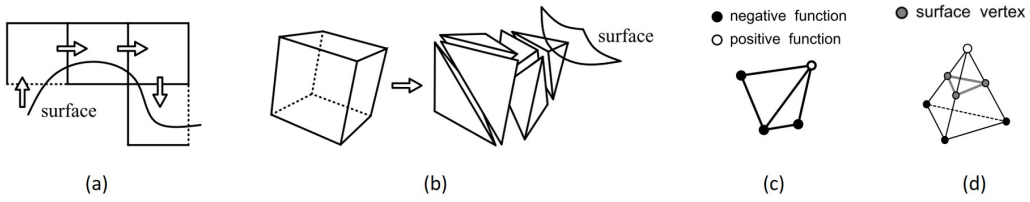


Figure 2.14: Bloomenthal's implicit surface polygonizer [34]. (a) Path of voxel evaluation. (b) Polygonizing cell decomposition. (c)Tetrahedron classification. (d) Polygonization

2.5.2. Measuring the accuracy of the SSIC method

The reconstructed 3D models produced by the SSIC method are compared with the original 3D model from which the photos are taken. The reconstructed models would ideally have the same shape as the original model. A 3D model constructed by the SSIC method is compared to its original model by means of Euclidean distance. The training set, the test and the SSM have the same amount of vertices due to the point-to-point correspondence described in subsection 2.2.2. The Euclidean distance can be used between the modified SSM and the original model from the training set or test set. The VH on the other hand does not have the same amount of vertices. The Hausdorff distance is used instead. This is a common metric for comparing meshes and is later during this project also used in comparing the SSIC method models with the original models (see section 4.2). The Hausdorff distance is the maximum distance of a set to the nearest point in the other set [35]. It is defined as follows:

$$H(A,B) = \max\Big\{h(A,B),\ h(B,A)\Big\} \tag{2.44}$$

Where A is a point cloud of the first 3D model and B is the point cloud of the second 3D model. h(A,B) and h(B,A) are the one sided Hausdorff distances, h(A,B) is described as follows:

$$h(A,B) = \max_{a\in A}\Big\{\min_{b\in B}\{d(a,b)\}\Big\} \tag{2.45}$$

MeshLab uses a sampling approach to compute the above formula taking a bunch of points over a model A and searching for each a the closest point b on a model B. MeshLab can create more vertices on the surface of a model. This will closely approximate a surface to surface measurement as more points are used.

This finalises the chapter. In this chapter, it is presented how the experimental algorithm is organised. Also, the validation methods are discussed for finding out how accurate the SSIC method is and how it performs compared to the current practice. The next chapter will present the results.

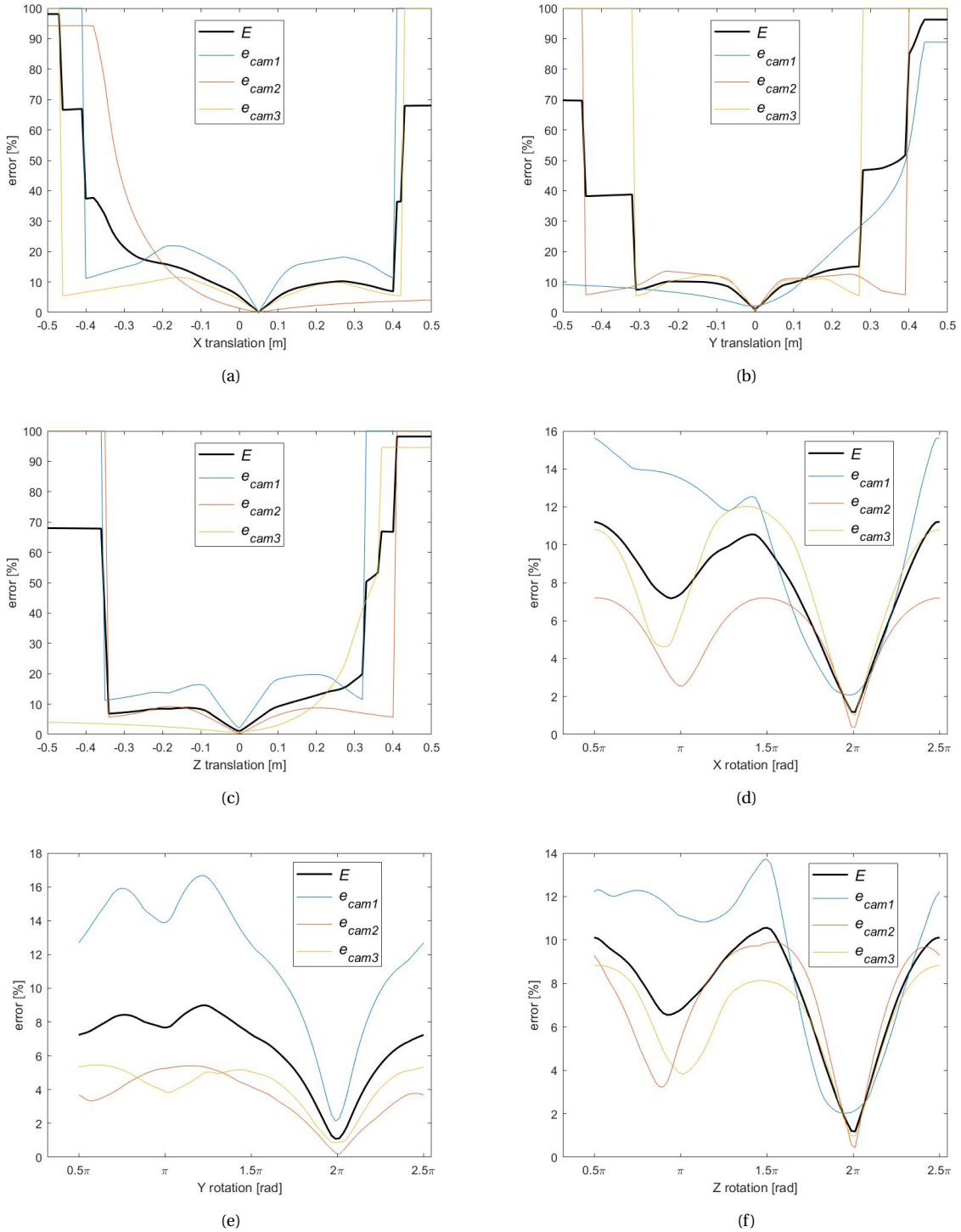
3
Results

3.1. Overview
This chapter will demonstrate the results of three separate items. First, section 3.2 illustrates how reliable the experimental algorithm is by showing some verification results. Second, section 3.3 will discuss the results of the SSIC method. The influence of the optimiser step size will first be shown. Then, the accuracy of 3D reconstructions made with the SSIC method will be presented. Reconstructions of models taken from both the training set and test set will be discussed. Furthermore, the results of a 3D reconstruction made with uncalibrated cameras is demonstrated and the influence of the number of cameras used is presented. Third, section 3.4 will discuss the results of the current practice method: the VH creation.

3.2. Verification of the experimental algorithm
The first experiments are intended to validate if the formula for the SSM (equation 2.6) is working as it should be. For this, the first sample in the training set, L_1, is used as a reference model. The b vector from L_1 is extracted by reordering formula 2.6:

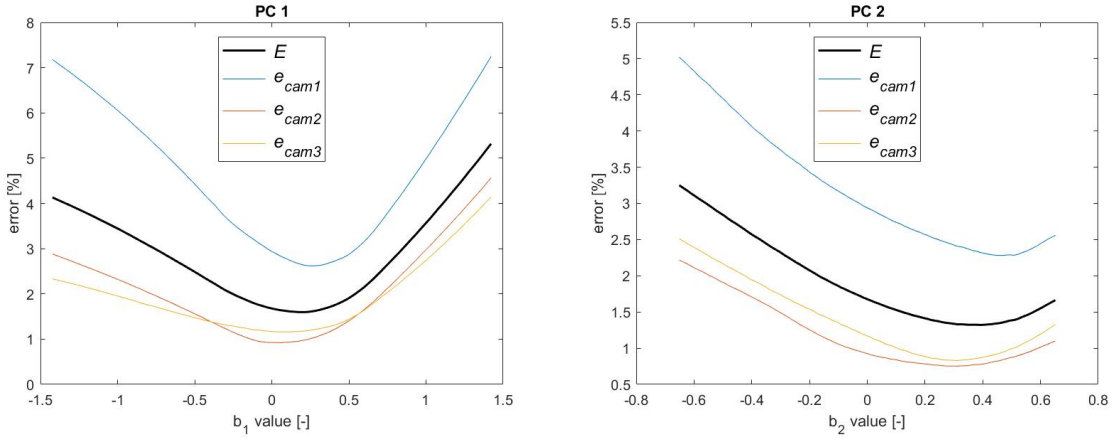
$$b_1 = \mathbf{P}^T \cdot (L_1 - \overline{S}) \tag{3.1}$$

When using $b = b_{L1}$ in formula 2.6, it is expected that $X_1 = L_1$. This is indeed the case so creating a 3D residual limb with the mean shape, a matrix of eigenvectors and weight values is performing as expected. Sample L_1 is now used to validate if the projection of this 3D model, the silhouette image, is correct after translating and rotating the sample in the 3D digital space. First the translation is observed by changing **T** in formula 2.43. The model is positioned in Blender at X_w = 0.06 m, Y_w = 0 m and Z_w = 0 m. The setup is rendered and 3 images (3 x I_p) with corresponding projection matrices are provided. Then, X_1 is translated from X_W = -0.5 m to X_w = 0.5 m in steps of 0.01 m, while keeping Y_w and Z_w zero. Because $X_1 = L_1$, the global error (equation 2.42) should be 0 at X_w = 0.06 as I_p and I_{ssm} are identical from all viewpoints. The translation results are shown in figures 3.1a, 3.1b and 3.1c. The rotation matrix **R** from formula 2.43 is validated as well. Now the location of X_1 is kept constant at [0.06, 0, 0] but the model is rotated around the X- Y- and Z-axis respectively. The results are shown in figures 3.1d, 3.1e and 3.1f. It is clearly visible that finding the right location and rotation of the object is an optimisation problem as the black line of E is at a minimum at the optimal **R** and **T** values. As shown in figure 3.2, the same applies for finding the right b value for different principle components. The cameras in Blender are validate by checking the VW. Looking through camera 1 ($C = [0, 0.5, 0]^T$), the VW at distance D = 0.5 m is calculated to go from X_w=-0.29 m to X_w=0.29 m. As shown in figure 3.4, this is indeed the case so the intrinsic camera parameters can be trusted. Only when the whole object is out of the VW, the local camera error e will become 100 %. Therefore e_{cam1} becomes 100% when $-0.4m > X_w > 0.4m$ (see e_{cam1} in figure 3.1a).



(a) (b) (c) (d) (e) (f)

Figure 3.1: The error is calculated by comparing I_p with I_{ssm} when translated and rotated. (a) X translation of X_1. (b) Y translation of X_1. (c) Z translation of X_1. (d) Rotation of X_1 around the X axis. (e) Rotation of X_1 around the Y axis. (f) Rotation of X_1 around the Z axis.



(a) (b)

Figure 3.2: Eigenvalues of the principle component (PC). (a) Error by changing the variance of PC1. (b) Error by changing the variance of PC2.

In MATLAB a function called `fmincon` is used to perform the optimisation that will find the the optimal parameters (**T**, **R** and b) related to the lowest global error. The performance of the optimiser is dependent on the input arguments. Important input arguments are the tolerances and stopping criteria, the upper- and lower bound of the search field and the initial values of the optimising parameters **T**, **R** and b. The first input argument that is observed is the step size of the optimiser. Zooming in at the translation graph will show that the line is not smooth but has an impulse character (figure 3.3). The global error E is determined by subtracting the pixel matrix of I_{ssm} from I_p. If X_{ssm} is then slightly translated with a step smaller than half of the pixel size (pixel location is rounded), the projection I_{ssm} will be unchanged. Therefore the optimiser will think it found its optimal location.

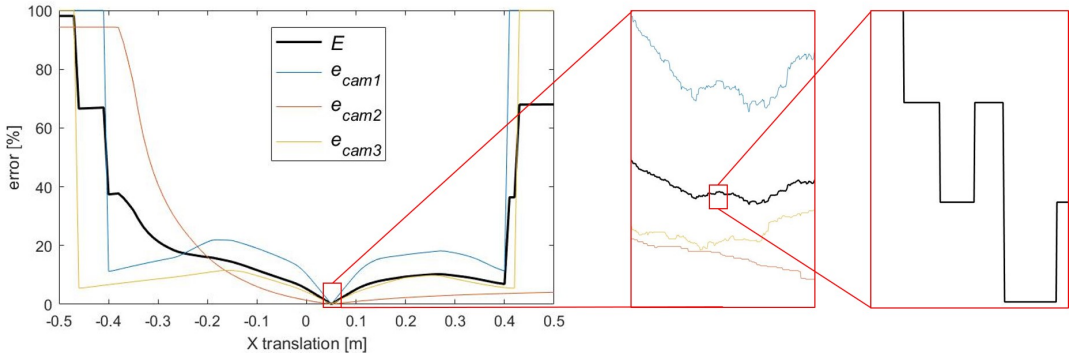
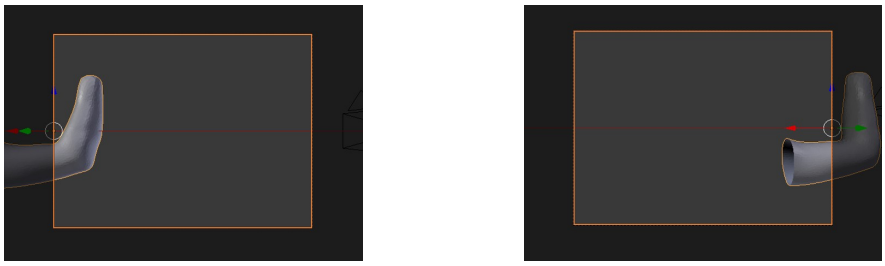


Figure 3.3: Zoomed in on the landscape of the translation.



(a) (b)

Figure 3.4: Looking through camera 1 in Blender (the orange rectangle). (a) When the centre of the object is at X_w=0.29. (b) When the centre of the object is at X_w=-0.29.

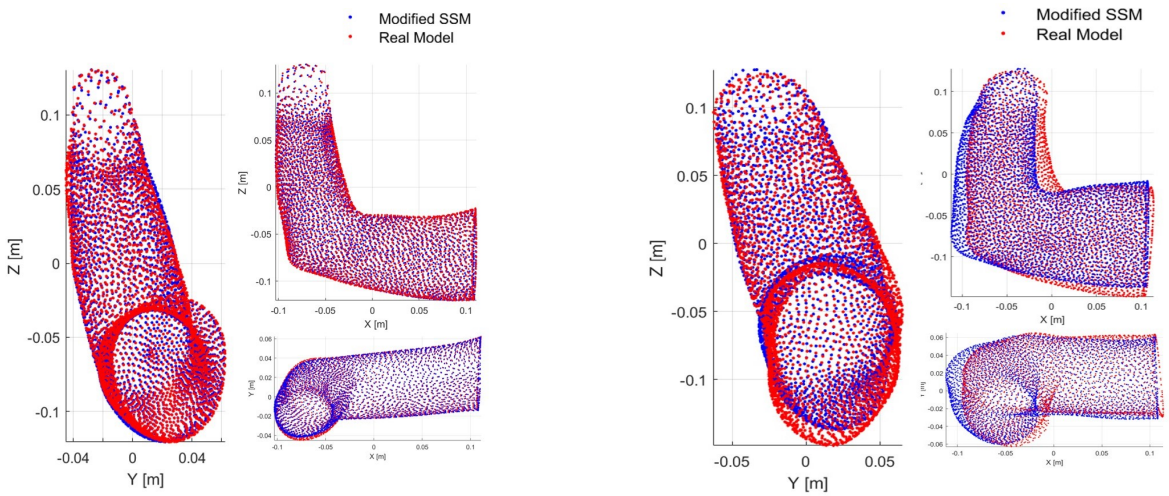
3.3. Results of the SSIC method

The performance of the optimiser is dependent on the input arguments. There are several local minima where the optimiser will terminate but the global minimum is obviously preferred. The influence of the optimiser step size is observed to find out how important such a input argument is for the resulting 3D model. Important to note is that the steps are done per parameter in the initial vector. The initial vector only contains b values, $[b_1 \ \dots \ b_{34}]^T$ in this case (there are 34 principle components), because the sample is positioned in Blender with zero translation and zero rotation thus neglecting the related parameters $\theta_x, \theta_y, \theta_z, T_x, T_y, T_z$ in the initial vector. The result of the accuracy are found in table II.

Table II: SSIC results with a 3 camera setup. Influence of different optimiser stepsizes.

Optimiser step size	Time (min)	Mean dist (mm)	Max dist (mm)	SD (mm)	Global error (%)
$b/10$	5	3.123	20.854	1.854	0.422
$b/50$	10	2.065	6.210	0.852	0.292
$b/100$	13	1.516	4.814	0.613	0.230
$b/500$	35	1.118	3.914	0.565	0.145
$b/1000$	13	2.507	10.973	1.254	0.347

The optimising process for the statistical shape image comparison method has been carried out and several parameters have been investigated. Experiments with SSIC method are carried out to see how the method performs on a sample out of the training set and test set. A step size of 0.001 over all parameters is used for the optimiser and the sample is positioned in Blender with zero translation and zero rotation. The results are shown in figure 3.5 and table III.



(a) (b)

Figure 3.5: SSIC 3D model reconstruction results. (a) SSIC with the original model form a training set, sample 01. (b) SSIC with the original model from the test set, sample 41.

Table III: SSIC difference between a sample from the test set and from the training set

	Time (min)	Mean dist (mm)	Max dist (mm)	SD (mm)	Global error (%)
sample training set (sample 01)	34	1.804	5.064	0.873	0.116
sample test set (sample 41)	35	12.506	28.145	3.196	1.362

The SSIC method is performed on more samples from the test set. The results are shown in table IV. The SSM is trained on 35 samples from the training set. The shape variability of the SSM is therefore limited to this amount of shapes. Using math, the closest shape approximation of a test set sample can be calculated. This

is done by calculating the b vector of a sample from the test set as shown in equation 3.1 and than use this test sample b vector in equation 2.6 to modify the SSM. By calculating the closest approximation possible, it can be validated if the SSIC method is performing well with the limited shape variability. Figure 3.7 illustrates the models created by the SSIC method in the left column and the optimal shapes possible with the SSM in the right column. The left column of figure 3.7 is related to table IV.

Table IV: Accuracy values for the SSIC method performed on several samples from the test set

	Time (min)	**Mean dist (mm)**	**Max dist (mm)**	**SD (mm)**
sample 40	≈30	9.348	28.811	4.139
sample 39	≈30	16.333	37.237	3.298
sample 38	≈30	6.483	25.867	2.145
sample 37	≈30	17.980	37.267	3.953
sample 36	≈30	21.832	33.042	6.410

Then a test was carried out to see how well a 3D model is constructed by using a uncalibrated camera with the SSIC method. For this a setup with 3 cameras is used as shown in figure 2.6c. A standard format of projection matrices is used where the distance from the cameras to the objects centre (z_C, the look-at direction from the cameras pose) is constantly 0.5 m. The cameras are then positioned at different z_C distances from the object. The distance for cameras 1, 2 and 3 are respectively set to 0.5m, 0.4m and 0.6m. Then the SSIC method is performed with the standard format of projection matrices, meaning that the I_p from the second and the third viewpoint have a false related projection matrix. The result is shown in figure 3.6 and table V.

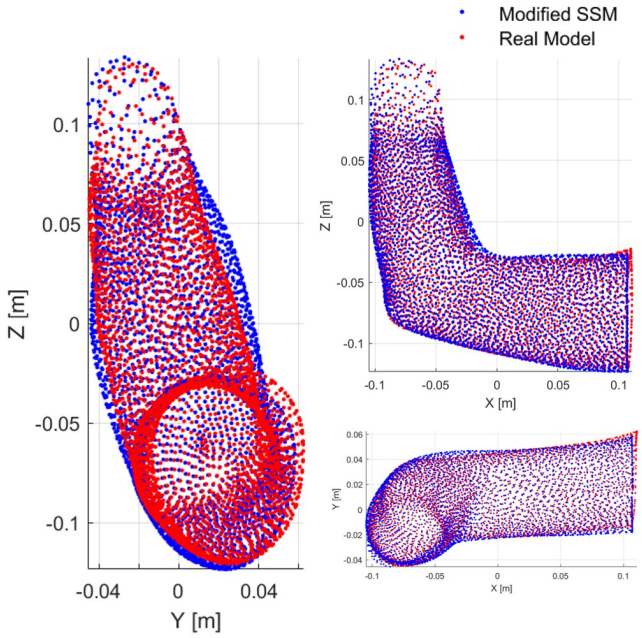


Figure 3.6: SSIC results with a 3 camera setup. In this case, the cameras are modelled as uncalibrated cameras. The sample is taken from the training set.

Table V: SSIC results when the SSIC method is performed with an uncalibrated camera

	Time (min)	**Mean dist (mm)**	**Max dist (mm)**	**SD (mm)**	**Global error (%)**
sample training set with uncalibrated camera	15	4.068	14.178	1.793	2.300

The influence of the number of pictures, i.e. the number of cameras, used for the SSIC method is explored. The optimiser step size of $b/10$ is chosen as this input argument runs relatively fast.

Table VI: SSIC results of optimising with with a different number of cameras. Step Tolerance is 1e-8. Optimiser step size is $b/10$. These results are of a 3D reconstruction of a model from the training set.

Number of cameras used	Time (min)	Mean dist (mm)	Max dist (mm)	SD (mm)	Global error (%)
1	2	2.666	16.628	1.714	0.593
2	5	3.344	16.900	1.657	0.670
3	5	3.123	20.854	1.854	0.422
5	14	2.727	7.795	1.054	0.447
10	25	2.712	21.083	1.660	0.550
17	40	3.162	14.699	1.543	0.570
26	150	2.953	16.117	1.462	0.513
37	73	3.367	14.321	1.688	0.538
50	138	3.199	16.324	1.631	0.550
65	261	3.320	15.012	1.681	0.534
82	309	3.291	15.903	1.599	0.540
101	413	3.245	8.311	1.543	0.544

The next two tables show the difference between the reconstruction of a training dataset sample and a test set sample using 1 or 2 cameras. The results are shown in table VII and VIII. These results of table VII differ from the results shown in table VI as another step size is used.

Table VII: SSIC results of a sample from the training set with only one or two images. Side view is taken with 1 camera, side and frontal view are taken with 2 cameras. A step size of 0.001 over all parameters in the b vector is used.

Number of cameras used	Time (min)	Mean dist (mm)	Max dist (mm)	SD (mm)	Global error (%)
1	5	2.158	8.554	1.066	0.240
2	11	2.736	8.924	1.363	0.346

Table VIII: SSIC results of a sample from the test set with only one or two images. Side view is taken with 1 camera, side and frontal view are taken with 2 cameras. A step size of 0.001 over all parameters in the b vector is used.

Number of cameras used	Time (min)	Mean dist (mm)	Max dist (mm)	SD (mm)	Global error (%)
1	9	12.325	27.812	5.135	1.677
2	18	11.668	24.554	5.018	1.306

Finally, the Hausdorff distance is measured between samples created by the SSIC method and their original model. Two samples are used: a sample from the training set and a sample from the test set. These are the same samples as shown in figure 3.5. As mentioned in section 2.5.2, a software package called MeshLab is used to calculate the Hausdorff distance between a created model and its original. The cutoff plane of the samples is also changed before measuring the Hausdorff distance again. This will indicate if the cutoff area at the proximal part of the limb has a considerable impact on the distance error. The results are shown in appendix B. The Euclidean distances in table III can be compared with the distances in figure B.1 and B.3, because these distances are measured for the same models.

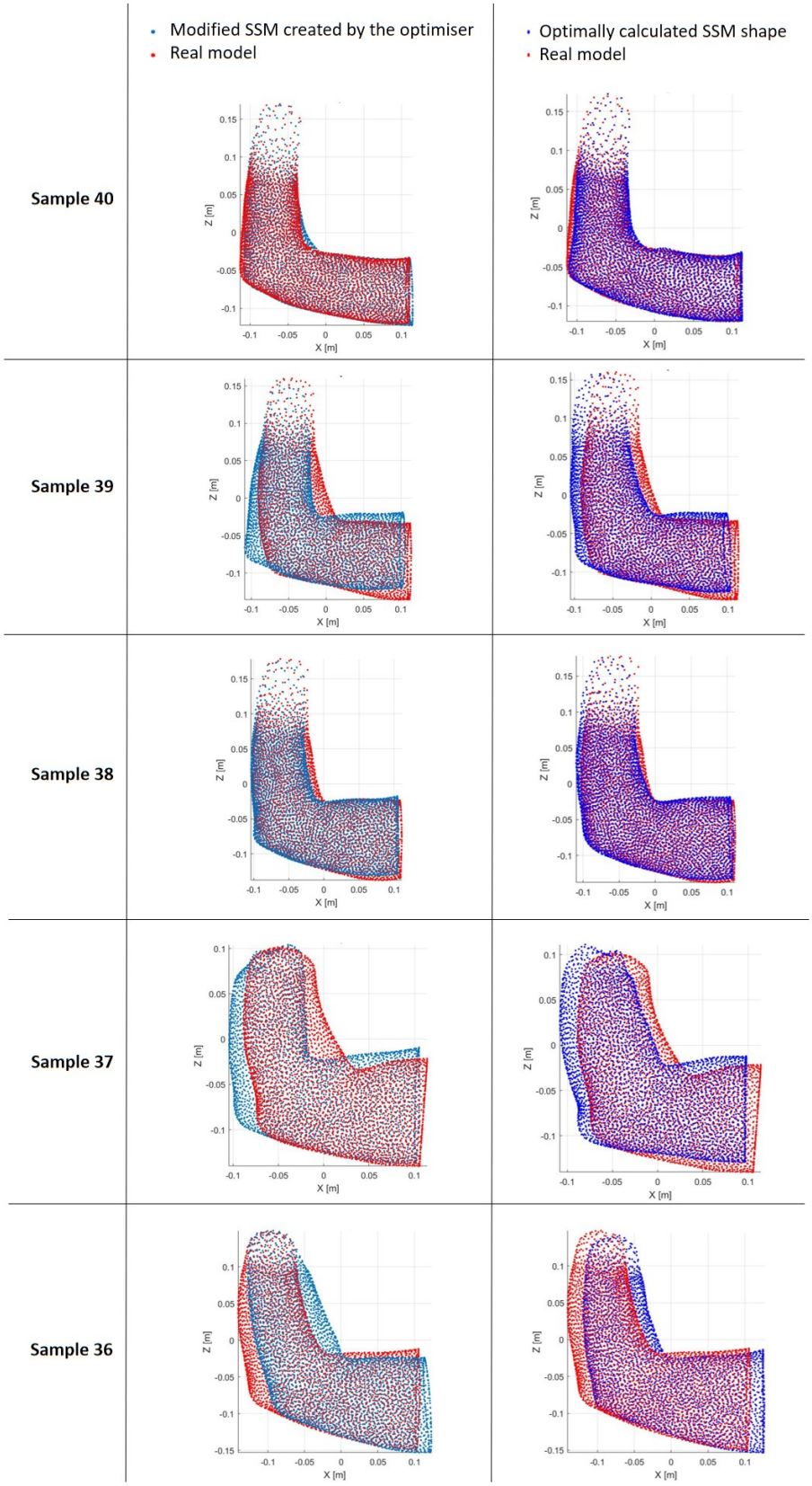
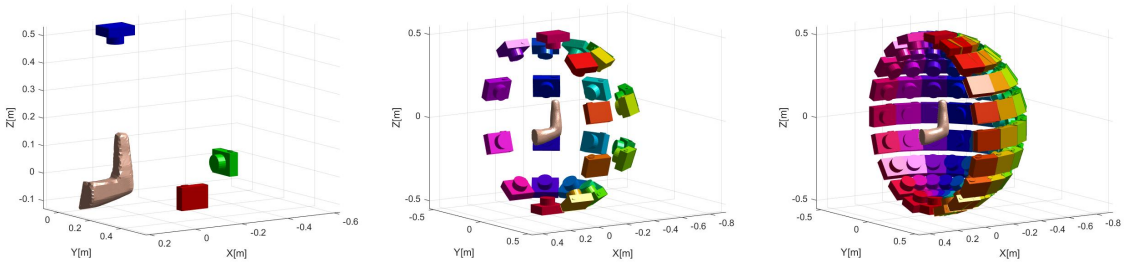


Figure 3.7: Side view plots of 5 test samples. The blue models in the left column are the created 3D models by the SSIC method. The blue models in the right column are the optimally calculated shapes possible with a SSM based on 35 shapes. The red model is in both columns the original model.

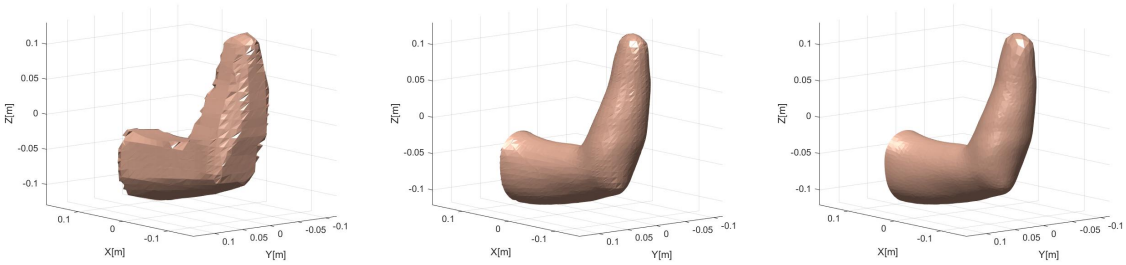
3.4. Results of the VH construction method

The results of the SSIC methods were shown in the previous section. The other conventional method of constructing a 3D model from silhouette images is the visual hull. A VH can be constructed by importing the data from Blender into MATLAB. In figure 3.8 the setup of 3, 26 and 101 cameras in MATLAB is shown with the resulting VH. Figure 3.9 will show the quality of the reconstructed 3D model between the 3 and the 101 camera setup with respect to the original model. The Hausdorff distance between the VH and the original per camera setup is shown in figure 4.1a. Figure 4.1b demonstrate the time it takes to make the VH.



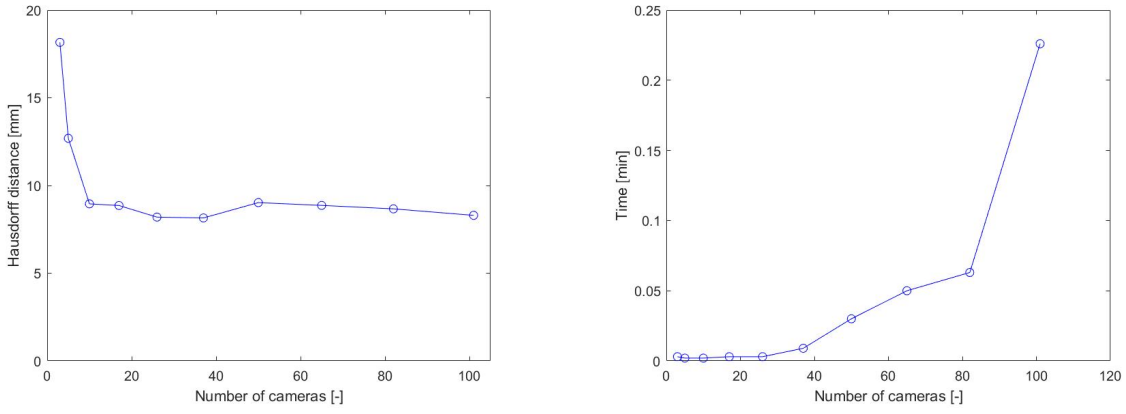
(a) (b) (c)

Figure 3.8: VH 3D model reconstruction results. (a) The VH from the 3 cam setup. (b) The VH from the 26 cam setup. (c) The VH from the 101 cam setup.



(a) (b) (c)

Figure 3.9: VH 3D model reconstruction results. (a) The VH from the 3 cam setup. (b) The VH from the 101 cam setup. (c) The original model from which the photos are taken.



(a) (b)

Figure 3.10: VH 3D model reconstruction results. (a) Hausdorff distance of the VH per set of cameras. (b) The VH reconstruction time of the algorithm.

4

Discussion

4.1. Commentary on the SSIC algorithm

The graphs in figure 3.1 clearly shows that an optimiser can be used to get the X_{ssm} model at the right position with the right pose in each projection I_{ssm}. Figure 3.2 shows that finding the right b values is also an optimisation problem. Thus the optimiser is able to find the global minimum error E which corresponds to the optimal parameter vector: $[\theta_{x_{opt}}, \theta_{y_{opt}}, \theta_{z_{opt}}, T_{x_{opt}}, T_{y_{opt}}, T_{z_{opt}}, b_{1_{opt}} \dots b_{34_{opt}}]^T$. The optimal parameter vector is the input for the shape creation to translate, rotate and morph the SSM to resemble the residual limb from which the photos are taken. The MATLAB function `fmincon` is a nonlinear constrained solver used to find the optimal parameters. Opposed to other solver functions available in MATLAB, `fmincon` has the option to put constraints on the parameter vector. This is a useful feature as the the FOV is a constraint on the translation, a rotation is constrained to be within the range of 0 - 2π rad and the weight values of the b vector are constraint as defined by equation 2.7. There are many input arguments that have influence on the performance of the solver. Only the finite step size of the solver is investigated, but parameters such as the stopping criteria, the function tolerance and the stepping tolerance can be investigated in future work.

There are also some remarks on the processing of the images in MATLAB and the influence of the screen resolution. Let's take sample L_1 from the training set. The SSM can be transformed in the exact same model by substituting b_1 from equation 3.1 into the equation 2.6 to get X_1. Both models, X_1 and L_1, are located at the same location and no rotation is applied, thus having the exact same pose. Now projecting both on the image plane of the same camera, camera 1 for example (or viewpoint 1), will result in the exact same picture ($I_p = I_{ssm}$). The local error, e_{cam1}, should be zero when I_{ssm} is subtracted from I_p (equation 2.40 and 2.41). However, this is not the case: the local error e_{cam1} is 0.0624%. This is also shown in figure 3.3, where error line do not touch the zero x-axis. That an error of zero cannot be achieved is probably caused by the fact that MATLAB reduces the resolution of the images taken in Blender to a default resolution of 630 x 840 pixels. Some of the information of I_p is lost during reduction, but it does not influence the performance of the optimiser. Finally, the projected SSM and the Blender images are presented in a MATLAB figure. Then, snapshots are taken of these figures with a function called `getframe` to further process them for error calculations. The resolution of the snapshot might not match the number of elements in the MATLAB figure because the function is dependent on the screen resolution. To see if that had some influence on the laptop used, a MATLAB figure was saved as a JPEG picture. The properties of the image gave the pixel resolution of the picture. This resolution was compared with the binary matrix size of the snapshot in MATLAB. They were both 630 x 840, thus the resolution of the screen had no influence.

4.2. SSIC accuracy evaluation

Active 3D scanners are tested on their accuracy in a research done by Dickinson et al. [36]. In their research they 3D printed a residual limb and scanned it with three different active scanners. The scans were compared with the 3D model that was used for 3D printing. The best scanner, Creaform 3D VIUScan, had a Euclidean distance of up to 0.20 mm across 95 percent of the surface and a maximum of 2.5 mm. The scanner with the highest Euclidean distance was the "Sense 3D" from 3D systems, having a mean distance of 1.40 mm across 95 percent of the surface and peaked at 4.0 mm. Looking at table II, the SSIC with 3 cameras can achieve a

mean Euclidean distance of 1.118 mm with ± 0.565 mm SD. This is comparable with the "Sense 3D", a device that costs $419. Three comments on this result is that: 1)it is done with a model from the training set, 2) the centre of the SSM is already aligned with the centre of the sample model, therefore neglecting the optimisation process for the translation and rotation and 3) the photos are taken with ideal conditions. These conditions resulted in easy image segmentation, the perfect projection matrices and a motionless object.

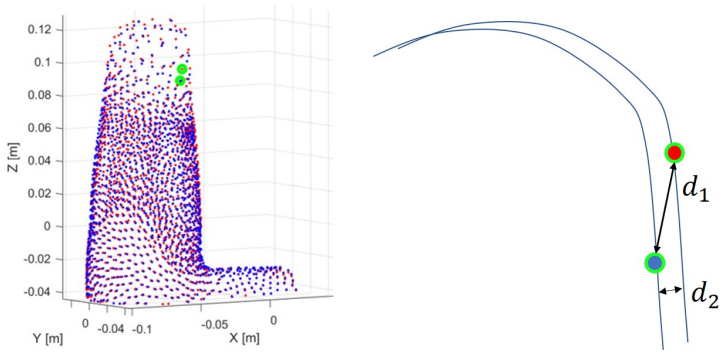
It is shown in figure 3.5 and table III that the SSIC method does not perform so well on a sample from the test set as opposed to the performance on a sample from the training set. The 3D reconstruction of the sample from the test set could be better performed if the training set of residual limb models was bigger. The training set created for the SSM of a residual limb is done with 35 models in this thesis. In a paper written by Ballester et al. [16] a database of 761 scans was used to make a SSM. Another research constructed a SSM with 700 models [17] and other researchers were able to use even more models [21, 37–39]. In that case, the SSM is better trained on a bigger variety of shapes and could therefore perform better.

One of the main restrictions of the SSIC method is the creation of the projection matrix. The camera intrinsics and extrinsics should be known to start the optimisation process and find the right shape. In the image acquisition setup build in this thesis project, the camera projection matrix can easily be extracted. The SSIC method is therefore observed under the most ideal conditions. But using uncalibrated cameras is simulated by using projection matrices that are unrelated to the cameras pose. As seen in figure 3.6 and table V, the performance is less accurate compared to the model reconstructed in figure 3.5a. This illustrates the importance of the right pose to make an accurate 3D reconstruction.

Also the influence of the number of camera viewpoints is investigated. It has little effect on the accuracy of the SSIC method. The VH construction on the other hand became more accurate with more camera view points as shown in figure 3.9. After using 10 cameras, the VH has an accuracy of a consistent value, whereas the number of view points has almost no effect on the accuracy of the SSIC method (see table VI). It is also shown in table VI that the amount of time it takes to run an algorithm, the time complexity, is linearly growing with the number of cameras. The VH is much faster (figure 4.1b). This concludes that the SSIC is performing better with a low amount of viewpoints compared to the conventional SfS 3D reconstruction method.

4.3. Commentary on the accuracy measurement

The goal of this thesis is to measure the accuracy of the SSIC method. This is done by using the Euclidean distance. This measuring method was chosen as it is an easy way to calculate because the models have point-to-point correspondence. However, there are some concerns about this measuring strategy. The points that define the maximum Euclidean distance between between the original model and the reconstructed model are observed more closely. Figure 4.1a illustrates two overlapping models where the points encircled in green define the maximum Euclidean distance between these two models.



(a) (b)

Figure 4.1: (a) Point clouds of the original model and the modified SSM. The corresponding point with the greatest Euclidean distance is encircled in green. (b) The maximum Euclidean distance between corresponding points of two comparable models, d_1 and the distance between the surfaces of the models, d_2

For a fitting prosthesis, it is more important to measure the surface to surface distance (d_2 in figure 4.1b). It

is therefore unknown if the Euclidean distance is a useful metric for measuring the accuracy. Another commonly used metric to compare mesh models is the Hausdorff distance [40–43]. This metric is already used for comparing the VH with the original model as the two models have an unequal amount of vertices that are also unrelated. Because the VH was closed at the proximal part of the arm while the original model has a open gap at the proximal part. The vertices that were closing the mesh were removed so that the Hausdorff distance was not calculated for these points as those points were not important for the shape. Points with a value of $X_w > 0.1$m were cut away for both the original model and the VH. A suggestion for the measuring method is to use a patch remesher for the target mesh to give it more vertices. Then by using the one-sided Hausdorff distance, the closest points are searched and used for calculating the distance between the 2 models. Because of the high density of points, it is more likely to find a point at distance d_2 (figure 4.1). Figures B.1 and B.3 shows the Hausdorff measurement on a sample from the test set and a sample from the training set. It differs from the Euclidean measurements in table III. The maximal distance is lower when looking at the figures B.1 and B.3. This indicates that the surface-to-surface distance has a higher accuracy as the maximum distance is lower.

4.4. Recommendations for real life implementation

The use of a smartphone as a scanning tool is investigated in this thesis under ideal circumstances. It is useful to know in advance what the SSIC method is capable of. To perform this method in real life, some things have to be taken into account. No lighting is involved during the rendering process in Blender. But during scene projection, Blender clearly separates the foreground from the background by assigning a lower pixel intensity value to the background than to the foreground. This is done by default and make the conversion to a silhouette easy, as the foreground and background are already separated. The algorithm used in this thesis sets a threshold on the pixel intensity values to assign a white colour to the foreground and a black colour to the background. This process is also known as image segmentation. In real life, people need to pay attention to the lighting of the scene and the residual limb. Another concern on the image segmentation process is shown in figure 4.2b. In real life not only the limb is on in the picture, but other human body parts as well (figure 4.2a an 4.2b). This thesis focused on a loose residual limb as object. However, this can be solved by setting up a clever photo shoot that does not require too much materials. Punching the residual limb through a newspaper can already make a difference. The training set of residual limbs should be constructed with real patients. There is already a study conducted that scanned 81 rectified casts of transtibial residual limbs [44]. For this research a comparable procedure should be done with patients having a transradial defect. With highly accurate expensive scanner in the high income countries, a database can be constructed for the LMICs. A calibrated camera is needed for calculating the projection matrix. The calibration process often requires extra tools, but as described in chapter 1 the method for 3D reconstruction should be made accessible and extra hardware makes the method less accessible. To overcome this problem, several solutions are thought of. In a research performed by Parrilla et al. [17] an A4 size paper is used as an orientation field for the smartphone to scan feet. In another research, some information about the person to be scanned (height, size etc.) [16] is filled in by the user. Based on that information, a full body shape outline is predicted and projected onto the smartphone screen. When the person to be scanned is within the boundaries of the outline, the smartphone knows the pose and distance to the person, thus it can calculate the projection matrix. As shown in figure 4.2c, something comparable can be done for the SSIC method. A scanning area of interest is projected onto the screen of a smartphone. The user should then fit the residual limb to be scanned into this area. The optimiser is able to translate and rotate the SSM to find the right location where the projections are aligned with the pictures taken.

(a) (b) (c)

Figure 4.2: (a) Camera top-down view in Blender of a patient. (b) Camera side view in Blender of a patient. (c) SSIC smartphone application guidance during patient scanning.

5

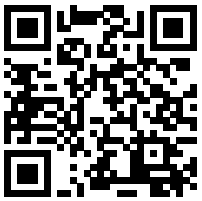
Conclusion

The accessibility of prosthetic devices must be improved in low- and middle-income countries to meet the demand. The fabrication process of prosthetic devices can be automated with 3D scanning and 3D printing technologies which enables a rapid and cost-efficient way of producing prostheses. As part of such a system, a silhouette-based 3D reconstruction method to model a residual upper limb for the design of a fitting hand prosthesis is investigated in this thesis. The purpose of the reconstruction method is to generate an accurate model, a duplicate of the residual limb, by using a smartphone camera. Therefore the goal of this thesis is to measure the accuracy of the observed 3D reconstruction method. An experimental algorithm is created that serves two purposes. Firstly, the process of shooting a picture is simulated to generate information for the 3D reconstruction. Secondly, it performs an automatic 3D reconstruction based on information from the first part and a priori knowledge. The 3D reconstruction is data-driven since it uses a parameterised training dataset of artificially created upper extremity stump shapes. Afterwards, the reconstructed model is compared with the original model from which the photos are rendered to determine the accuracy of the silhouette-based 3D reconstruction method. Considering a camera setup with three orthogonal viewpoints, the highest accuracy measured is 1.12 ± 0.57 mm when a model from the training dataset is photographed and reconstructed. If models from the test dataset, therefore an unknown shapes, are photographed and reconstructed with the same camera setup, the highest accuracy is then 6.48 ± 2.15 mm. These results are close to the accuracy of current 3D scanners used for prosthetic designing. It shows that the accuracy can be measured more accurately by giving models a denser vertex distribution and find the closest point, thus approximating a surface-to-surface distance. The performance of the 3D reconstruction can be increased in two ways. In the first place, the number of models in the training data set should be increased. In the second place, the input arguments of the optimiser should be investigated further as that can gain more optimal solutions for the 3D reconstruction. The observed 3D reconstruction method is compared with the current practice in the field of 3D reconstructions based on silhouette images. For less than 10 silhouette images, the observed method is performing more accurately. However, the time complexity is much higher than the current practice. It is also shown that the observed method can achieve an accuracy of 2.16 ± 1.07 mm with only 1 silhouette image. A few things must be considered for real-life implementation. The residual limb must visually be clearly separated from the background for image segmentation. Calibrating the camera is an essential part for reliable results. These requirements are met with clear instructions on a smartphone application. Even though some requirements are needed, this study shows that a data-driven silhouette-based 3D reconstruction process can make reasonable 3D models of residual limbs automatically. The reconstruction process can be optimised and real-life tests can be done in future research. It is a promising method that provides a cheap gateway to obtain a hand prosthesis in low resource settings.

A

Source Code

The following link will guide you to the GitHub page where the source code of this thesis project can be found. This will include the Python script for Blender to render a scene for image acquisition, the MATLAB files needed for the SSIC method and the artificial database of residual limbs.



`https://github.com/stevengoes/SSIC`

B

Hausdorff distance figures

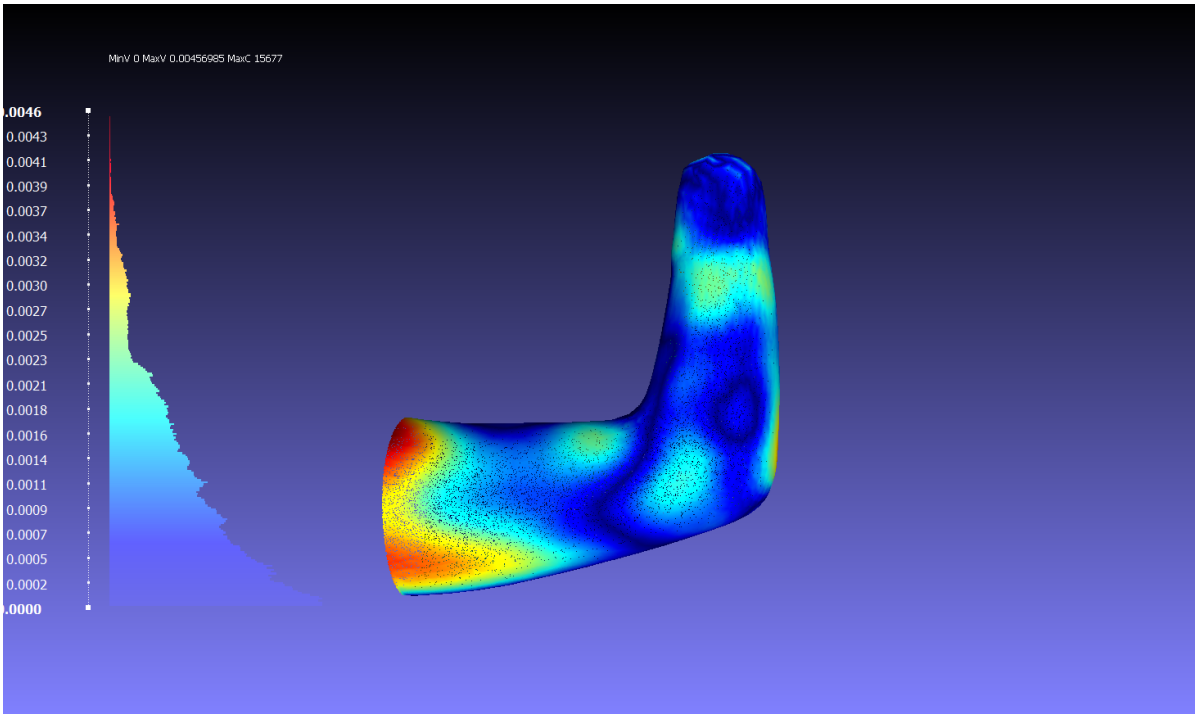


Figure B.1: Hausdorff distance measured in MeshLab between two models. The first model is a model created with the SSIC method where the silhouette images of sample 01 are used. A step size of 0.001 is used for the optimiser and the sample is positioned in Blender with zero translation and zero rotation. The second model is sample 01. In other words the reconstruction is compared with the original. On the left side a histogram is shown with the distances in meters.

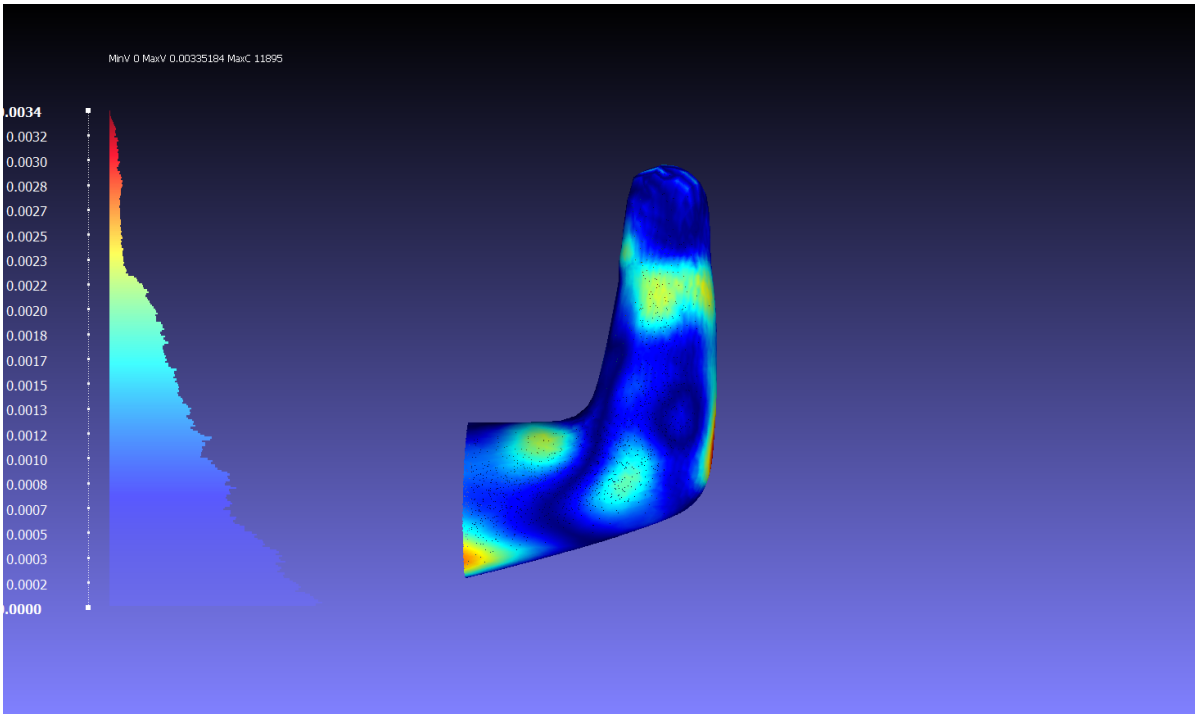


Figure B.2: Hausdorff distance measured in MeshLab. Sample 01 from the training set is taken as object for the SSIC method. A step size of 0.001 is used for the optimiser and the sample is positioned in Blender with zero translation and zero rotation. The Hausdorff distance is measured between the model created by the SSIC method and the original sample (i.e. sample 01). All 3D point with a value of X>0.05m are removed. On the left side a histogram is shown with the distances in meters.

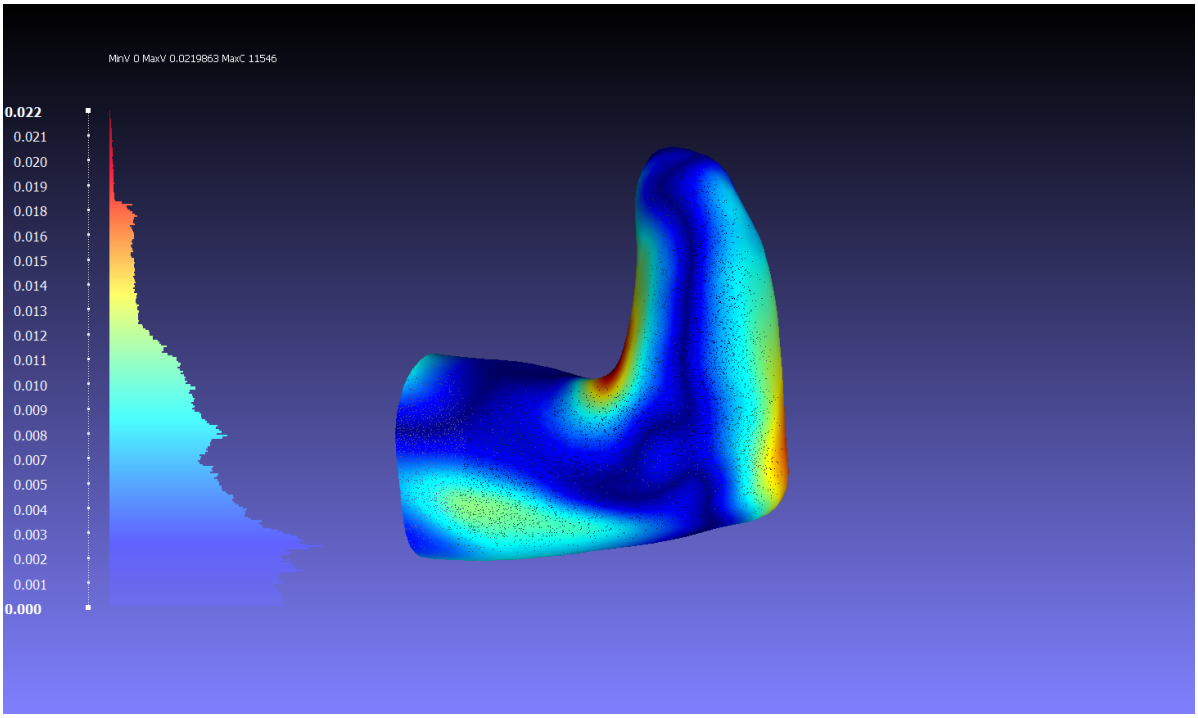


Figure B.3: Hausdorff distance measured in MeshLab between two models. The first model is a model created with the SSIC method where the silhouette images of sample 45 are used. A step size of 0.001 is used for the optimiser and the sample is positioned in Blender with zero translation and zero rotation. The second model is sample 41. In other words the reconstruction is compared with the original. On the left side a histogram is shown with the distances in meters.

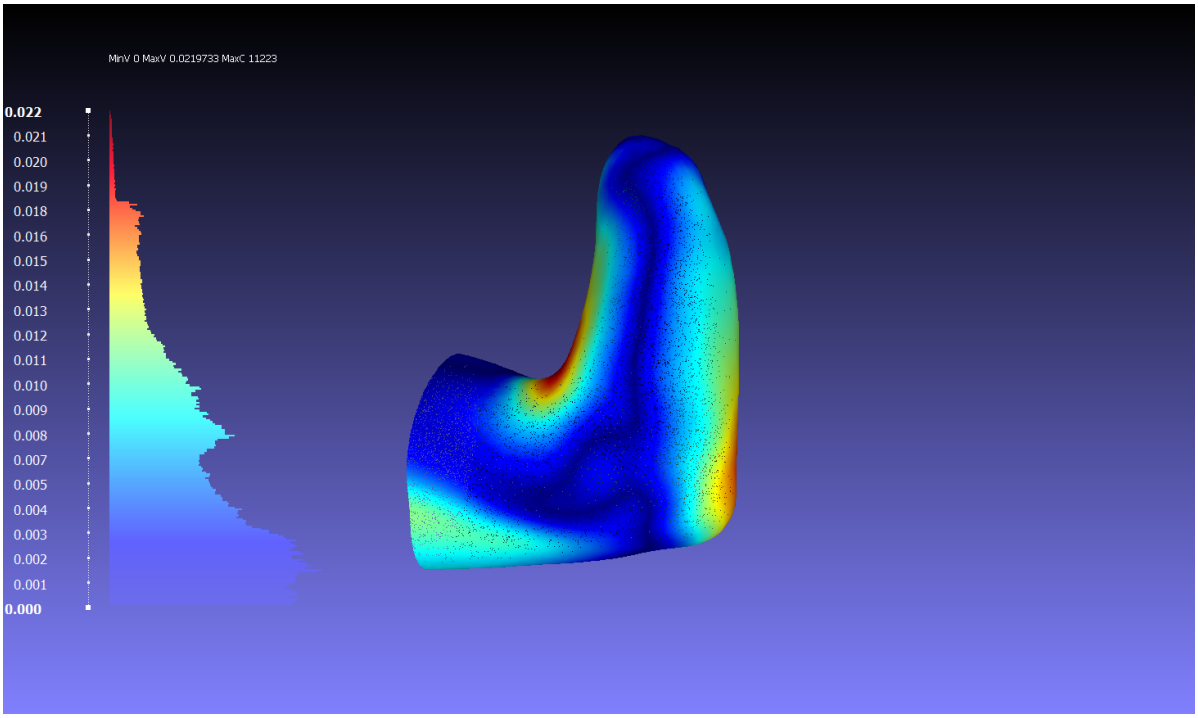


Figure B.4: Hausdorff distance measured in MeshLab between two models. The first model is a model created with the SSIC method where the silhouette images of sample 45 are used. A step size of 0.001 is used for the optimiser and the sample is positioned in Blender with zero translation and zero rotation. The second model is sample 41. In other words the reconstruction is compared with the original. All 3D point with a value of X>0.05m are removed. On the left side a histogram is shown with the distances in meters.

Bibliography

[1] World Health Organisation. World Report on Disability - Summary. Technical Report WHO/NMH/VIP/11.01, 2011. URL http://www.who.int/disabilities/world_report/2011/report.pdf.

[2] Chapal Khasnabis. Standards for Prosthetics and Orthotics Service Provision 2015-2017 work plan. Technical report, 2015. URL http://www.who.int/phi/implementation/assistive_technology/workplan_p-o_standards.pdf.

[3] International Society for Prosthetics and Orthotics and World Health Organization. Guidelines for training presonnel in developing countries for prosthetics and orthotics services. Technical report, 2005. URL https://afro.who.int/sites/default/files/2017-06/who_guidelines_training_personnel_en.pdf.

[4] Ryan Schmidt, Ginger Coons, Vincent Chen, Timotheius Gmeiner, and Matt Ratto. 3D-printed prosthetics for the developing world. *SIGGRAPH 2015: Studio on - SIGGRAPH '15*, (c):1–1, 2015. doi: 10.1145/2785585.2792535.

[5] Emelie Strömshed. *The Perfect Fit - Development process for the use of 3D technology in the manufacturing of custom-made prosthetic arm sockets*. PhD thesis, Lund University, 2016.

[6] Laura E Diment, Mark S Thompson, and Jeroen Hm Bergmann. Three-dimensional printed upper-limb prostheses lack randomised controlled trials: A systematic review. *Prosthetics and orthotics international*, 42(1):7–13, 2 2018. ISSN 1746-1553. doi: 10.1177/0309364617704803.

[7] Ming C Leu, Hoda A Elmaraghy, Andrew Y.C. Nee, Soh Khim Ong, Michele Lanzetta, Matthias Putz, Wenjuan Zhu, and Alain Bernard. CAD model based virtual assembly simulation, planning and training. *CIRP Annals - Manufacturing Technology*, 62(2):799–822, 2013. ISSN 00078506. doi: 10.1016/j.cirp.2013.05.005.

[8] Petri Tanskanen, Kalin Kolev, Lorenz Meier, Federico Camposeco, Olivier Saurer, and Marc Pollefeys. Live Metric 3D Reconstruction on Mobile Phones. In *2013 IEEE International Conference on Computer Vision*, pages 65–72. IEEE, 2013. ISBN 978-1-4799-2840-8. doi: 10.1109/ICCV.2013.15.

[9] GSM Association. The Mobile Economy 2018. Technical report, 2018. URL https://www.gsmaintelligence.com/research/?file=061ad2d2417d6ed1ab002da0dbc9ce22&download.

[10] O. Ciobanu, G. Ciobanu, and M. Rotariu. Photogrammetric Scanning Technique and Rapid Prototyping Used for Prostheses and Ortheses Fabrication. *Applied Mechanics and Materials*, 371:230–234, 2013. ISSN 1662-7482. doi: 10.4028/www.scientific.net/AMM.371.230.

[11] Octavian Ciobanu and Mariana Rotariu. Photogrammetric Scanning and Applications in Medicine. *Applied Mechanics and Materials*, 657:579–583, 2014. ISSN 16627482. doi: 10.4028/www.scientific.net/AMM.657.579.

[12] Amaia Hernandez and Edward Lemaire. A smartphone photogrammetry method for digitizing prosthetic socket interiors. *Prosthetics and Orthotics International*, 41(2):210–214, 2017. ISSN 0309-3646. doi: 10.1177/0309364616664150. URL https://doi.org/10.1177/0309364616664150.

[13] 3D scanning apps for smartphones (updated October 2018). URL https://www.aniwaa.com/best-3d-scanning-apps-smartphones/.

[14] David Schneider. Visual Hull. In *Computer Vision: A Reference Guide*, pages 866–869. Springer US, Boston, MA, 2014. ISBN 978-0-387-31439-6. doi: 10.1007/978-0-387-31439-6{_}211.

[15] Fendy Santoso, Matthew A. Garratt, Mark R. Pickering, and Md Asikuzzaman. 3D Mapping for Visualization of Rigid Structures: A Review and Comparative Study. *IEEE Sensors Journal*, 16(6):1484–1507, 2016. ISSN 1530437X. doi: 10.1109/JSEN.2015.2498940.

[16] Alfredo Ballester, Eduardo Parrilla, Ana Piérola, Jordi Uriel, Paola Piqueras, Beatriz Nácher, and Julio A Vivas. Data-driven three-dimensional reconstruction of human bodies using a mobile phone app. *International Journal of the Digital Human*, 1(4):361–388, 2016. doi: https://doi.org/10.1504/IJDH.2016.084581.

[17] Eduardo Parrilla, Alfredo Ballester, Clara Solves-Camallonga, Beatriz Nácher, Sergio Antonio Puigcerver, Jordi Uriel, Ana Piérola, Juan Carlos González, and Sandra Alemany. Low-cost 3D foot scanner using a mobile app. *Footwear Science*, 7(sup1):S26–S28, 2015. ISSN 1942-4280. doi: 10.1080/19424280.2015.1038308.

[18] Mikkel B. Stegmann and David Delgado Gomez. A brief introduction to statistical shape analysis. *Informatics and Mathematical Modelling*, (March):1–15, 2002.

[19] Tim Cootes, Er Baldock, and J Graham. An introduction to active shape models. *Image Processing and Analysis*, pages 223–248, 2000. ISSN 1098-6596. doi: 10.1017/CBO9781107415324.004.

[20] Tobias Heimann and Hans Peter Meinzer. Statistical shape models for 3D medical image segmentation: A review. *Medical Image Analysis*, 13(4):543–563, 2009. ISSN 13618415. doi: 10.1016/j.media.2009.05.004.

[21] SAE International. CAESAR: Civilian American and European Surface Anthropometry Resource Project. URL `http://store.sae.org/caesar/`.

[22] K.M. Robinette, H. Daanen, and E. Paquet. The CAESAR project: a 3-D surface anthropometry survey. In *Second International Conference on 3-D Digital Imaging and Modeling (Cat. No.PR00062)*, pages 380–386. IEEE Comput. Soc, 1999. ISBN 0-7695-0062-5. doi: 10.1109/IM.1999.805368.

[23] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics*, 32(3):1–13, 2013. ISSN 07300301. doi: 10.1145/2487228.2487237. URL `http://dx.doi.org/10.1145/2487228.2487237`.

[24] Toon Huysmans, Jan Sijbers, and Verdonk Brigitte. Automatic construction of correspondences for tubular surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(4):636–651, 2010. ISSN 01628828. doi: 10.1109/TPAMI.2009.93.

[25] Jonathon Shlens. *A Tutorial on Principal Component Analysis*. PhD thesis, Carnegie Mellon University, 2005. URL `http://arxiv.org/abs/1404.1100`.

[26] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3D faces. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, pages 187–194, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co. ISBN 0201485605. doi: 10.1145/311535.311556.

[27] 3Dabout.me. URL `https://www.3dabout.me/nl/`.

[28] Lars Mundermann, Stefano Corazza, Ajit M Chaudhari, Eugene J Alexander, and Thomas P Andriacchi. Most favorable camera configuration for a shape-from-silhouette markerless motion capture system for biomechanical analysis. In *Proceedings of SPIE - The International Society for Optical Engineering*, volume 5665, pages 278–287, 2005. doi: 10.1117/12.587970.

[29] Robert Collins. *Lecture 12 : Camera Projection*. PhD thesis, Penn State University, Pennsylvania, 2007. URL `http://www.cse.psu.edu/~rtc12/CSE486/lecture12.pdf`.

[30] Aldo Laurentini. How far 3D shapes can be uderstood from 2D silhouettes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):188–195, 1995.

[31] Aldo Laurentini. The Visual Hull Concept for Silhouette-Based Image Understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(2):150–162, 1994. ISSN 01628828. doi: 10.1109/34.273735.

[32] Stefano Corazza, Lars Mündermann, Emiliano Gambaretto, Giancarlo Ferrigno, and Thomas P Andriacchi. Markerless motion capture through visual hull, articulated ICP and subject specific model generation. *International Journal of Computer Vision*, 87(1-2):156–169, 2010. ISSN 09205691. doi: 10.1007/s11263-009-0284-3.

[33] Keith Forbes. *Calibration, Recognition, and Shape from Silhouettes of Stones*. PhD thesis, University of Cape Town, 2007. URL `https://www.dip.ee.uct.ac.za/publications/theses/PhDKeith.pdf`.

[34] Jules Bloomenthal. An implicit surface polygonizer. *Graphics gems IV*, 1:324–349, 1994. ISSN 2072-4292. doi: 10.3390/rs2030833.

[35] Günter Rote. Computing the minimum Hausdorff distance between two point sets on a line under translation. *Information Processing Letters*, 38(3):123–127, 5 1991. ISSN 00200190. doi: 10.1016/0020-0190(91)90233-8.

[36] Alexander S Dickinson, Joshua W Steer, Christopher J Woods, and Peter R Worsley. Registering methodology for imaging and analysis of residual-limb shape after transtibial amputation. *Journal of Rehabilitation Research and Development*, 53(2):207–218, 2016. ISSN 0748-7711. doi: 10.1682/JRRD.2014.10.0272.

[37] Ales Jurca, Tomaz Kolsek, and Tina Vidic. DOROTHY Mass Foot Measurement Campaign. In *Proceedings of 1st International Conference on 3D Body Scanning Technologies*, pages 338–344, Lugano, Switzerland, 2010. doi: http://dx.doi.org/10.15221/10.338.

[38] Sandra Alemany, Alfredo Ballester, Eduardo Parrilla, Jordi Uriel, Jorge González, Beatriz Nácher, Juan Carlos González, and Álvaro Page. Exploitation of 3D Body Databases to Improve Size Selection on the Apparel Industry. In *Proceedings of 4th International Conference on 3D Body Scanning Technologies*, pages 456–466, 2013. doi: http://dx.doi.org/10.15221/13.456.

[39] Zongyi Xu, Qianni Zhang, and Shiyang Cheng. Multilevel active registration for kinect human body scans: from low quality to high quality. *Multimedia Systems*, 0(0):1–14, 2017. ISSN 09424962. doi: 10.1007/s00530-017-0541-1.

[40] P. Cignoni, C. Rocchini, and R. Scopigno. Metro: Measuring Error on Simplified Surfaces. *Computer Graphics Forum*, 17(2):167–174, 1998. ISSN 01677055. doi: 10.1111/1467-8659.00236.

[41] Paolo Cignoni. Measuring the difference between two meshes, 2010. URL `http://meshlabstuff.blogspot.com/2010/01/measuring-difference-between-two-meshes.html`.

[42] Michael Guthe, Pavel Borodin, and Reinhard Klein. Fast and accurate Hausdorff distance calculation between meshes. *Journal of WSCG*, 13(2):41–48, 2005. ISSN 1213-6972.

[43] Nicolas Aspert, Diego Santa-Cruz, and Touradj Ebrahimi. MESH: Measuring errors between surfaces using the Hausdorff distance. *Proceedings - 2002 IEEE International Conference on Multimedia and Expo, ICME 2002*, 1:705–708, 2002. ISSN 0166-4328. doi: 10.1109/ICME.2002.1035879.

[44] Peter Worsley, Joshua Steer, Chris Smith, and Alex Dickinson. Classifying residual limb shape in transtibial amputees. In *15th International Society of Prosthetics and Orthotics World Congress*, number June, page 1, Lyon, 2015. ISPO.