CHALMERS
UNIVERSITY OF TECHNOLOGY

Image Analysis for Modelling Infant Limb Movement

A Pilot Project to Predict Neurological Dysfunction

Bachelor thesis in Signals and Systems
Course SSYX02-15-03

Dennis Sångberg
Eric Karlsson
Knut Nordenskjöld
Lukas Jönsson
Marie Liljenroth
Pia Damsten

Department of Signals and Systems

Chalmers University of Technology
Gothenburg, Sweden 2015

Bachelor's thesis 2015:03

Image Analysis for Modelling Infant Limb Movement

A Pilot Project to Predict Neurological Dysfunction

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Signals and Systems
Division of Signal processing and Biomedical engineering
Chalmers University of Technology
Gothenburg, Sweden 2015

Image Analysis for Modelling Infant Limb Movement
A Pilot Project to Predict Neurological Dysfunction

Dennis Sångberg (Engineering Physics)
Eric Karlsson (Engineering Mathematics)
Knut Nordenskjöld (Mechanical Engineering)
Lukas Jönsson (Electrical Engineering)
Marie Liljenroth (Engineering Mathematics)
Pia Damsten (Biotechnology)

Supervisor: PhD student Yixiao Yun, Signals and Systems
Examiner: Professor Irene Yu-Hua Gu, Signals and Systems

Bachelor Thesis 2015:03
Department of Signals and Systems
Division of Signal processing and Biomedical engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2015

Abstract

Currently health care is in need for an efficient and affordable method to predict neurological diseases in infants. A sign of neurological disease is abnormal limb movement. Previous research indicates both the human eye and computer-based video analysis can distinguish between an infant's abnormal and normal limb movement. This project aims to aid in the development of a new, suitable video analysis method, which meets the needs of the health care.

In this project, a series of video-recordings of infants were recorded at Östra Hospital in Gothenburg. A computer program was developed to create 3D models of the infants' movements from the videos captured. The program, created in MATLAB©, consists of two major parts. The first part of the program tracks visible markers placed on the infants' bodies. The second part of the program projects these tracks from 2D to 3D.

The program can model the infants' movements seen in the videos, as tracks in 3D. The program needs to be further developed in order to generate models that can be used for predicting neurological diseases. Improved models and more data could arguably make this method a useful tool in diagnosis of infants.

Sammanfattning

Sjukvårdssystemet har för tillfället behov av en ny, effektiv och billig teknik för att förutse neurologiska sjukdomar hos spädbarn. Exempel av symptom på neurologiska sjukdommar är onormala rörelsemönster hos spädbarn. Tidigare forskning visar att onormala rörelsemönster kan detekteras både av människa och med hjälp av datorbaserad videoanalys. Syftet med det här projektet är att ta fram en ny, användbar metod genom bildanalys för att förutspå neurologiska sjukdomar hos spädbarn.

I projektet utfördes en rad videoinspelningar av spädbarn vid Östra Sjukhuset i Göteborg. Ett program, med syfte att göra 3D modeller av spädbarnen, skapades i MATLAB©. Programmet består av två huvudelar. Den första delen skapar rörelsebanor i 2D genom att följa markörer på spädbarnets kropp. Den andra projicerar dessa röselsebanor från 2D till 3D.

Det visade sig möjligt att modellera rörelsebanor i 3D med hjälp av de erhållna videorna och det skapade programmet. Programmet bör dock vidareutvecklas för att kunna generera modeller som gör det möjligt att förutspå neurologiska sjukdomar. Mycket tyder på att vidareutvecklade modeller samt ytterligare data kan göra denna metod till ett användbart redskap för diagnostisering av spädbarn.

Acknowledgements

We would like to thank the families who allowed us to video record the movements of their infant childrens' limbs. These recordings were the basis for this project. We would also like to thank Doctor Magnus Thordstein and Doctor Anders Flisberg from Sahlgrenska University Hospital who helped provide background knowledge and assisted us with the measurements. Finally, we would like to give a special thanks to our examiner Professor Irene Yu-Hua Gu and our supervisor PhD student Yixiao Yun, who discussed project issues with us and provided valuable criticism throughout the course of the project. Without their guidance and support this project would not have been possible.

Gothenburg, May 2015

Glossary

Neonatal period The first four weeks in an infant's life.

Central nervous system (CNS) Contains the brain and spinal cord which coordinates and influences the whole body's movement.

(Brain) Plasticity The ability of the brain to evolve throughout a person's life. By learning the brain plasticity changes physically, functionally and chemically.

Neonate - A newborn child.

Preterm children Children born after a pregnancy significantly shorter than normal.

General movements Uncontrolled movements such as flailing legs and arms.

Cerebral palsy A classification permanent movement disorders which appears in early childhood.

SIFT Short for Scale-Invariant-Feature-Transform. An algorithm for detecting and describing local features in an image.

DoG Short for Difference of Gaussian, see 2.1 for further explanation.

Keypoint An point of interest which the SIFT algorithm has produced, describing a feature in an image.

Intensity Refers to a pixel value in an grayscale image, it ranges from 0 to 255 where 0 is black and 255 is white.

Candidate-set The set of keypoints the SIFT-algorithm produces.

HSV (Hue-Saturation-Value) A common cylindrical-coordinate representation of points in an RGB (Red-Green-Blue) colour model. Where H corresponds to the angular-coordinate, V to the axial-coordinate and saturation to the radial-coordinate. The values range from 0 to 1.

Contents

1 Introduction

In the neonatal period it is difficult to evaluate the functioning level of an infant's central nervous system (CNS), particularly regarding higher functionality which can not be assessed at such an early age. Nevertheless, this assessment is important since treatment or supportive measures should be started as early as possible to prevent complications of the CNS. If preventative measures can be started while the plasticity of the nervous system is high, meaning the brain is in major development, the CNS is more likely to develop without complications.

1.1 Background

The clinical method used today to evaluate the CNS can easily distinguish the neonates who have had a serious brain injury from the healthy babies who have been in a risk of brain injury but survived without obtaining sequela. Infants who have been exposed to complications during birth such as lack of oxygen, premature birth, or being small in relation to the expected size have an increased risk of developing neurological disabilities. However, identifying infants with a minor brain injury can be a major difficulty. This is a problem because it is desirable to identify neonates at risk as early as possible, to determine if they have any latent neurological diseases. The current method can not evaluate the functioning level of the CNS until 1-2 years after birth, at which point the CNS has deteriorated significantly[13].

A diagnostic tool for the functional assessment of the young nervous system has been devised by H.F.R Prechthl.[1] Prechthl discovered the correlation between an infant's general movements and the infant's CNS. General movements are described as seemingly uncontrolled movements, such as flailing arms and legs[3]. In Prechtl's method, a child's general movements are examined in a five-minute film, to determine whether they are normal or abnormal. The movements are categorized in sub-categories which are described in detail in "Prechtl's assessment of general movements: a general diagnostic tool for the functional assessment of the young nervous system"[1]. The method is used to examine 9 - 20 week post-term infants. The method identifies a certain type of general movement that indicate Celebral Palsy, which is a general term for different types of neurological dysfunctions that affect coordination and movements.

By implementing Prechtl's method qualitative assessment of general movements has

been successfully made using computer-based video analysis[5]. The method has been implemented by modifying existing MATLAB© code. The implementation shows great potential as it has managed to successfully diagnose infants without human input.

1.2 Purpose

This is a pilot project to investigate in the possibility to detect neurological diseases of neonates by image analysis, instead of clinical examination. The method is tested for one week post-term infants. The main purpose is to track the motion of the limbs in 3D, as a step for evaluating the possibility to detect neurological dysfunction by tracking the limbs' movement.

This project is relevant for doctors and patients in the development of methods for simpler detection of neurological dysfunctions at hospitals. This method will ideally help decrease the long-term complications of infants and their families.

This project is a collaboration between Chalmers University of Technology and Sahlgrenska University Hospital in Gothenburg. The measurements will be done at Östra Hospital neonatal unit in Gothenburg and the main work will be done at Chalmers.

The project is a basis for further development and finally an easy-to-use software.

This project attempts to solve the two main issues:

- Tracking markers in 2D image planes.
- Modelling baby arm movement with 3D trajectories by 2D-3D mapping.

1.3 Limitations

The measurements used in this project are limited to the number of available infants and the measurements done at Östra Hospital.

The tracking is restricted to finding the markers of each infant's limbs and not the movement of the whole limb.

The project is limited to providing a program adapted for the project's sets of video measurements.

The project is limited in time to 13 weeks.

This project will not provide a program with the purpose of deciding whether the filmed infant has a neurological dysfunction or not. The project is restricted to provide a 3D-model of the filmed infants' limb motion.

The software written in the project will complete the task of modelling movement in 3D-space but will not be optimized for time efficiency or usability.

The project is limited to the method of image analysis to model infants' limb motion. The use of accelerometers or other methods to model limb motion will not be considered in this project.

2 Theoretical Framework

The theoretical framework in this project considers two major topics: the Scale Invariant Feature Transform (SIFT) and 2D-3D mapping. SIFT is an algorithm which is used in the project for detection of markers and tracking. The geometry and associated equations of 2D-3D mapping are explained in the 2D-3D section.

2.1 Scale-Invariant Feature Transform (SIFT)

This algorithm provides a method for extracting distinctive invariant features from an image. It was first published in 1999 by David Lowe at the University of British Columbia [6]. There are four main steps of the algorithm:

1. Scale-space extrema detection: The first part of the algorithm searches all scales and locations to find extrema points in the image. This gives a set of candidate keypoints.

2. Keypoint localization: Computes a more accurate location of the keypoint. Keypoints with low measured stability are discarded.

3. Orientation assignment: One or more orientations (directions) are assigned to each keypoint location based on local image gradient directions. This provides invariance to rotation.

4. Keypoint descriptor: The local image gradients are measured at the selected scale in the region around each keypoint. These are transformed into a representation which provides distinctiveness and some level of invariance to distortion and illumination.

2.1.1 Scale Space Extrema Detection

The scale space is defined as a function $L(x,y,\sigma)$, produced by convolution of the variable-scale Gaussian $G(x,y,\sigma)$ and the input image $I(x,y)$, where x,y are room variables and σ is the standard deviation for the Gaussian distribution.

$$L(x,y,\sigma) = G(x,y,\sigma) * I(x,y)$$

Where * is the convolution operator in x and y.

The operation smoothes the Image I and for larger values of σ more detail is removed. This effect can be seen in Figure 2.1. This works for a continuum of values of the parameter σ but in this project only a discrete set will be considered.

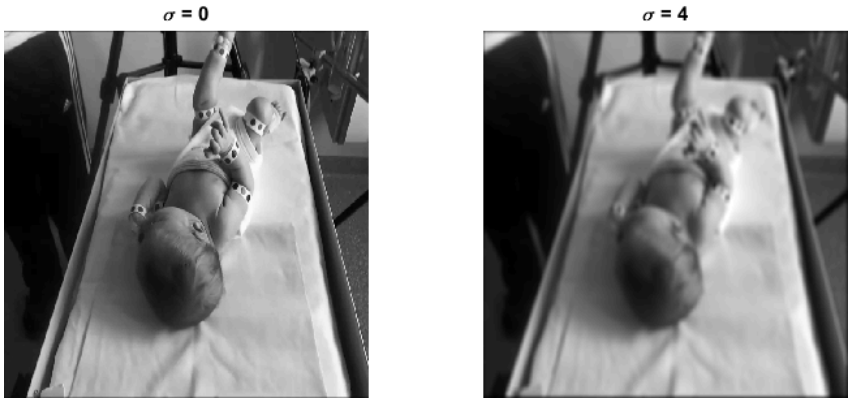


Figure 2.1: *Illustration of the effect of increasing the value of* σ

.

For detection of stable keypoints in scale-space, David Lowe proposes scale-space extrema detection in the difference of Gaussian (DoG) function [7]. $D(x,y,\sigma)$ is defined as the difference between two Gaussian functions convolved with the image. This can be computed as

$$D(x,y,\sigma) = (G(x,y,k\sigma) - G(x,y,\sigma)) * I(x,y) = L(x,y,k\sigma) - L(x,y,\sigma)$$

Where k is a scale constant.

The DoG also provides a good approximation of the scale-normalized Laplacian of Gaussian, $\sigma^2\nabla^2 G$. It has been shown that this image function produces the most stable features compared to other image functions such as the Hessian or Harris corner function [8]. The domain of DoG also defines the scale space.

The extrema is defined as a maxima or minima of the 26 neighbours of a pixel in 3×3 regions at the current and adjacent scales in the DoG images. This is illustrated in Figure 2.3. An extrema point is then considered a keypoint candidate for further evaluation.

In the paper by David G. Lowe from 2004 [7] the optimal values of k, σ, and sampling frequency in scale and spatial domain is discussed. The full understanding of that discussion is not significant to this project where focus is using SIFT as a tool. The

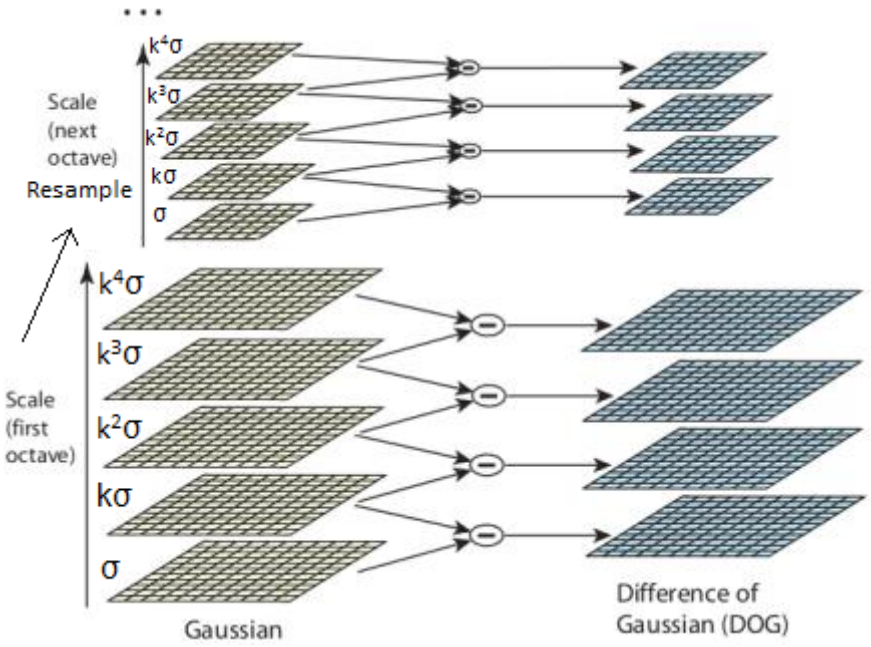


Figure 2.2: *Illustration of the DoG scale-space computation adapted from: [7]. The image is resampled for each octave which corresponds to a doubling of sigma. In each octave there are a preset number of levels to be computed. In this figure the DoG scale-space has 4 levels.*

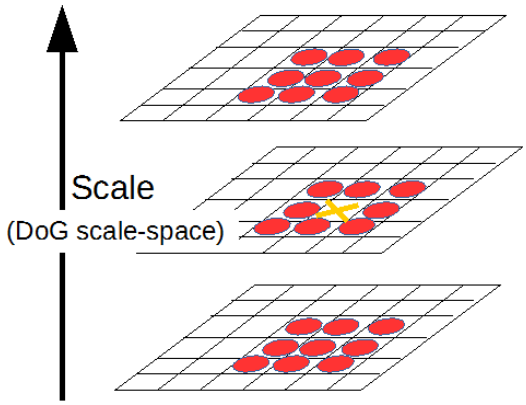


Figure 2.3: *The pixel indicated by a cross is compared to the pixel's neighbours in scale-space for extrema detection. The figure was adapted from: [7].*

optimal values found in the paper are already implemented in the VLFeat toolbox and not changed in this project.

2.1.2 Keypoint Localization

The scale-space extrema detection produces redundant keypoint candidates, some of which are unstable. Therefore, it is desirable to eliminate some points in low contrast areas or keypoints on edges in the image, such as silhouettes. This can be

done by fitting the keypoints to the nearby data for accurate location, scale, and ratio of principal curvatures (a measure of how a surface bends by different amounts in different directions).

To obtain a more accurate position, a quadratic Taylor expansion of $D(x,y,\sigma)$ with the candidate keypoint as origin is used

$$D(\mathbf{x}) = D + \frac{\partial D^T}{\partial \mathbf{x}}\mathbf{x} + \frac{1}{2}\mathbf{x}^T\frac{\partial^2 D}{\partial \mathbf{x}^2}\mathbf{x}$$

where D and its derivatives are evaluated at the candidate point and $\mathbf{x} = (x,y,\sigma)$ is the offset from this point. The accurate position of the extrema is found by taking the derivative of the expansion and letting it be zero. A candidate keypoint offset larger than 0.5 indicates the extrema lies closer to another candidate.

To discard the low contrast points the value of the Taylor expansion is computed at the offset. If that value is less than 0.03 the candidate keypoint is discarded.

Elimination of poorly localized keypoints along edges (sensitive to noise) is done by comparing the principal curvature along the edge with the principal curvature across the edge. If the principal curvature across the edge is greater than along the edge, the $D(x,y,\sigma)$ peaks are poorly defined and these keypoints are discarded. These points can be found by evaluating the eigenvalues of the second-order Hessian matrix

$$\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$$

If the ratio between the largest and the smallest eigenvalue is greater than 10 the candidate is considered poorly localized and is discarded.

2.1.3 Orientation Assignment

In order to achieve independence of image rotation, each keypoint is assigned one or more orientations based on local image properties. The keypoint descriptors can be represented relative to the rotation, thus becoming independent of rotation. The scale of the keypoint is used to select the Gaussian smoothed image L with the closest scale. For the image sample $L(x,y)$ at this scale the gradient magnitude $m(x,y)$, and orientation $\theta(x,y)$ is precomputed using pixel differences:

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}$$

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

A histogram is created from the gradient orientations computed for a neighboring region around the keypoint. The histogram has 36 bins covering 360 degrees. Each gradient orientation is weighed by its gradient magnitude and a Gaussian circular window at $\sigma = 1.5$ times the scale of the keypoint. The highest peak in the histogram determines the orientation of the keypoint. In addition, all peaks within 80% of the highest peak also creates a new keypoint at the same location but with a different orientation.

2.1.4 Keypoint Descriptor

The keypoint descriptor provides a highly distinctive representation of the keypoint which can be used to match features between different images even with the presence of change in illumination-conditions or change in 3D-viewpoint between images. This representation is obtained by computing gradients in a 16×16 region around the keypoint. These gradients are used to create a set of 16 histograms consisting of the orientation of the gradients, with 8 bins in each histogram, such that each bin contains samples from a 4×4 sub region from the original 16×16 region. The gradient magnitudes are weighed by a Gaussian circular window with $\sigma = 1.5$ times the width of the descriptor which is illustrated by the overlaying circle in the Figure 2.4. The descriptor is represented as a 128-dimensional vector with all values of the histograms. All computations are made on an image sample L relative to the keypoint's orientation and with the same scale as the keypoint to maintain independence to scale and rotation of the image feature.

2.1.5 Matching

Matching features or keypoints between images are done by measuring the Euclidean distance between a descriptor vector in one image and a descriptor vector in another image. If the distance is small enough there is a high probability the same feature is found in both images.

2.2 Multiple View Geometry

Mapping information from 2D images to 3D space is a multistep process. The theory in this section describes the mathematical functions and algorithms involved in each step. The main segments of the 2D-3D mapping are:

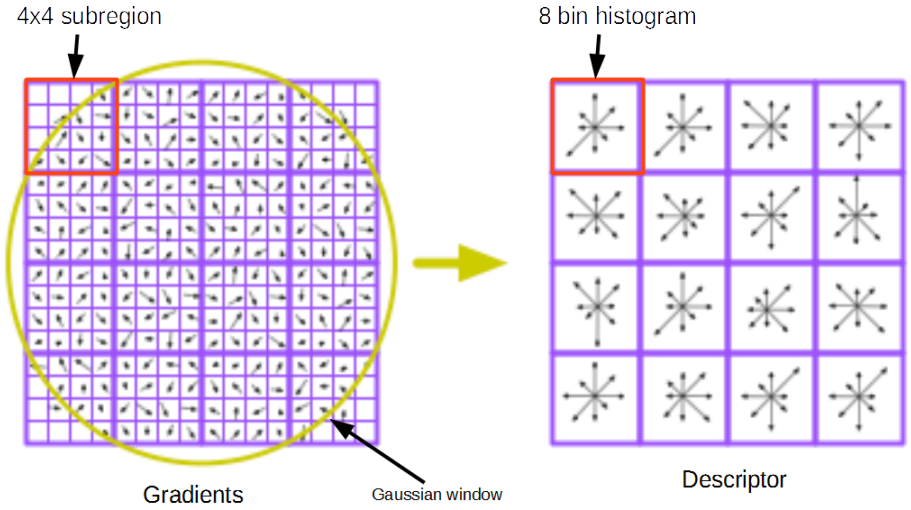


Figure 2.4: *Illustration of how the gradients are weighed by the Gaussian window (indicated by the yellow circle) and combined into a descriptor.*

- Calibration of the cameras: Calibrating parameters for each camera and computing camera matrices.
- Projection between image and world: Compute the relation between real world coordinates and the coordinates in the images.
- Epipolar geometry: Combine information between cameras to create a position in 3D.

2.2.1 Camera Calibration, Computation of Extrinsic and Intrinsic Parameters

Camera calibration is the calibration of inner (intrinsic) parameters and outer (extrinsic) parameters of the camera. Currently, free algorithms for calibration are available online. These algorithms typically require the use of a chessboard-pattern captured at different angles by the camera for the parameters to be calculated.

The intrinsic parameters include focal length, principal point, lens distortion, and skewness[11]. These are shown graphically in Figure 2.5.

Extrinsic parameters describe the orientation of the camera in the space and are composed of the translation and rotation vector[11]. These are represented in Figure

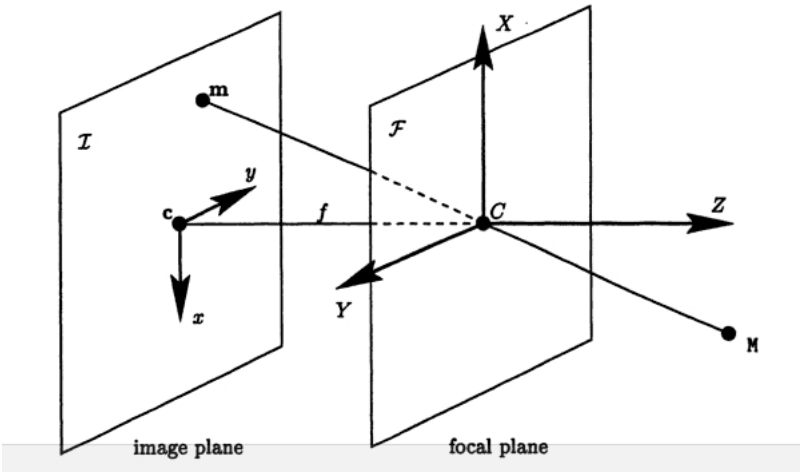


Figure 2.5: *The figure shows intrinsic parameters for the ideal pinhole camera model. In the image plane, c is the principal point and f is the focal length. A point M in space will be projected to position m in the image plane. C is the origin of the camera coordinate system and the optical center[11].*

2.6.

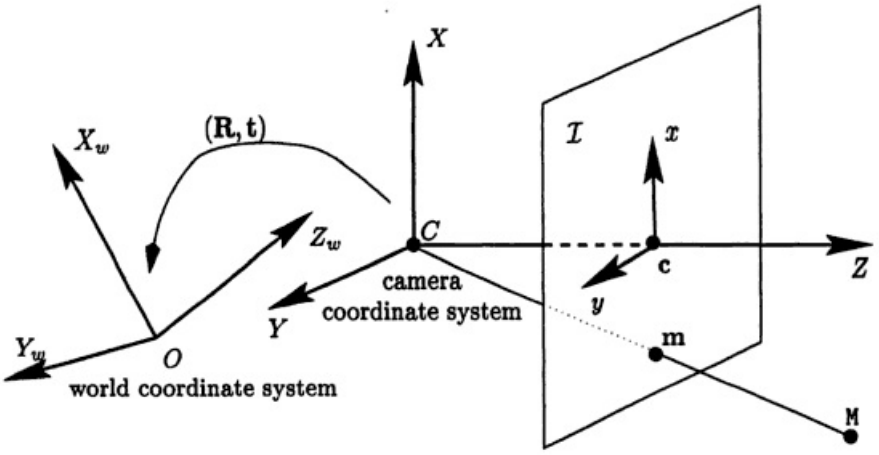


Figure 2.6: *Representation of the relation between world coordinate system and camera coordinate system. C is the optical center and the origin of the camera coordinate system, and M is a point in space. Shifting between these two coordinate systems is achieved by using rotation matrix R and translation vector T [11].*

2.2.2 Projection Between Image, Camera, and World Coordinates

The camera matrices are calculated from the intrinsic and extrinsic parameters. The matrices enable transformation between image, camera, and world coordinates.

The translation and rotation vector allow computation of the extrinsic matrix equation:

$$\mathbf{X_c} = \mathbf{R} \cdot \mathbf{X_w} + \mathbf{T} \tag{2.1}$$

Where $\mathbf{X_w} = [X_w, Y_w, Z_w]$ is the position in real world coordinates, R is the 3×3 rotation matrix computed by applying Rodrigue's formula on the rotation vector obtained by camera calibration[14], T is the 3×1 translation vector which shows the position of the world origin in relation to camera origin, and $\mathbf{X_c} = [X_c, Y_c, Z_c]$ is the position in camera coordinates.

The extrinsic matrix, Equation(2.1), allows transformation of a point in world coordinates, where the origin is a point in the 3D room, into camera coordinates. A representation of the relation between world coordinate system and camera coordinate system is shown in Figure 2.6. In camera coordinates the origin will be the nodal point of the camera[10].

The point in a 3D room can be further projected onto an image plane by using the ideal pinhole camera model shown in Equation (2.2). This model assumes the camera has strictly linear properties[10]. The pixel coordinate of the point $\mathbf{X_p} = [X_p, Y_p, 1]$ is given by

$$\mathbf{X_p} = \mathbf{KK} \cdot \mathbf{X'_c} \tag{2.2}$$

Where $\mathbf{X_c} = \left[\frac{X_c}{Z_c}, \frac{Y_c}{Z_c}, 1\right]$ is the normalized position vector in camera space, and $\mathbf{KK}$ is the intrinsic matrix

$$\mathbf{KK} = \begin{bmatrix} f_1 & af_1 & x_0 \\ 0 & f_2 & y_0 \\ 0 & 0 & 1 \end{bmatrix}$$

Where f_1 and f_2 is the focal length, x_0 and y_0 is the principal point in pixel coordinates, and a is the skew coefficient. The skew coefficient corrects distortion of angles in the image. In order to determine the skew coefficient the chessboard needs to be viewed from different angles.

The process of converting between different coordinate spaces can also be reversed. In other words, a pixel point in the image plane can be projected to camera or world coordinates. The projection from the 2D image plane to the 3D room will however, be restricted since the depth of the point can not be determined. The projection from a pixel point in the image plane will therefore be a direction vector from which we can create a line in the 3D world space, extending between the nodal point of

the camera and the pixel point's x- and y-coordinates, both in world space. Every coordinate on that line can possibly be the original position of the point.

Finding the original coordinate in 3D space will require information about the point from cameras in different angles. This will be explained in the next section about epipolar geometry.

2.2.3 3D Mapping Using Epipolar Geometry

The relation between how an object in space is projected onto different image planes is called epipolar geometry. Epipolar geometry describes how one scene is projected differently onto image planes depending on the camera's viewpoint. The epipolar geometry also helps derive information about a scene based on the different projections.

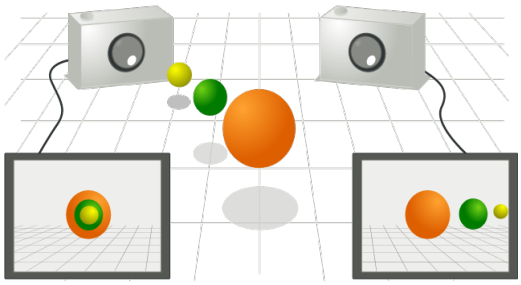


Figure 2.7: *Illustration of how projections differ for cameras at different angles. CC BY-SA 3.0[16]*

The projection of a pixel point to world coordinates will result in a line with an infinite number of possible positions for the point. This line projected onto another camera in the same room will represent the epipolar line of that particular point.

Placing two cameras in one room and viewing the projection of the principal point of one camera in the image plane of the second camera will result in an epipolar point or epipole. The plane, built up from the epipolar lines and epipoles, is referred to as the epipolar plane[12].

In an ideal case the projection lines will intersect at the position where the object exists. The intersection point determines the actual position of a point in space based on the projection onto two different images. In practice, the epipolar lines will not intersect at the object's original position in space, because of failure in determining the camera parameters exactly. A method for finding the best approximation has to be used.

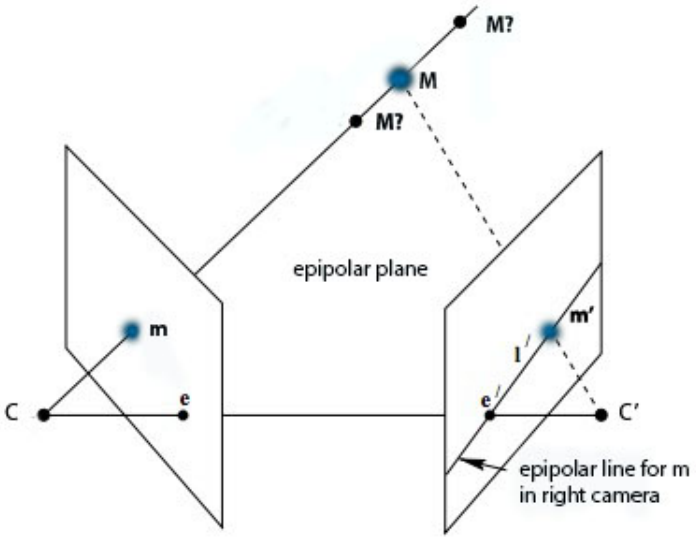


Figure 2.8: *Visualization of epipolar geometry. The figure shows camera centers C and C', and the epipoles e and e'. The projections of M to the left and right image is marked as m and m' respectively. These projection lines allow for computation of the epipolar plane.*

2.2.4 Finding the Best Approximation for 3D Coordinates

The sought 3D point will be found where the distance between the two projection lines is the smallest. The smallest distance between these lines means the vector combining them are perpendicular to both projection lines. This distance is represented with the vector **PQ**, combining the points P and Q in Figure 2.9. This is known as the method of least squares.

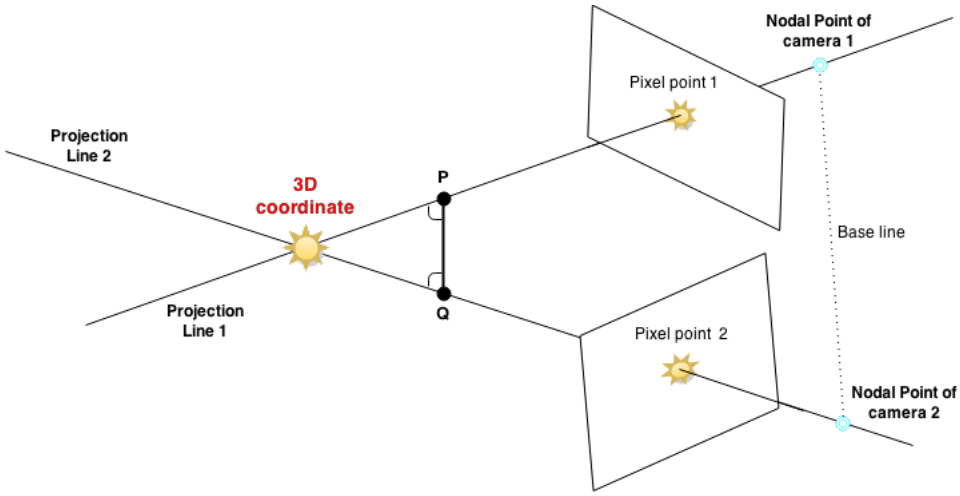


Figure 2.9: *The projection lines from both cameras are supposed to cross each other at the 3D point in world coordinates. An example of where the vector* **PQ** *could be situated is also shown.*

To construct a line mathematically, an intersection point and a vector determining the direction are needed. These can be derived from the camera calibration and one image capturing the object. The known nodal point, $\mathbf{X_c^n} = [0, 0, 0]^T$ can easily be converted to world coordinates using Equation (2.1).

With one known pixel coordinate the projection line can be estimated using the nodal point. The pixel coordinate is transformed to the world system by the combined expression

$$\mathbf{X_w^p} = \mathbf{R^{-1}}((\mathbf{KK^{-1}X_p^p})' - \mathbf{T})$$

These are two point vectors in world coordinates, which intersect in the desired projection line. Subtracting these from each other, the line's direction $\mathbf{d}$ is obtained:

$$\mathbf{d} = \mathbf{X_w^n} - \mathbf{X_w^p}$$

The equation describing the projection line $\mathbf{p}$ can then be defined by:

$$\mathbf{p} = \mathbf{d} \cdot t_i + \mathbf{X_w^n}$$

with some parameter t_i. This can be evaluated for multiple cameras and thus three lines can be constructed.

$$\mathbf{PQ} = \mathbf{p} - \mathbf{q} = \mathbf{d}_p \cdot t_p + \mathbf{X_{w,p}^n} - \mathbf{d}_q \cdot t_q + \mathbf{X_{w,q}^n}$$

The points P and Q will be somewhere on the projection lines for some parameters t_p and t_q. These parameters can be found using $\mathbf{PQ}$ and must be perpendicular to both projection lines. These parameters will give a final estimation of the 3D coordinate. These t_i can be found using the least square method. The parameters are estimated to be at the shortest distance from the base line when combining the two nodal points. The 3D coordinate will be considered to be in the middle of the vector $\mathbf{PQ}$.

3 Video Recordings

Five infants were recorded at Östra Hospital during three sessions. All recordings were produced by us, especially for this collaboration between Sahlgrenska and Chalmers.

Three different cameras were used, placed in three different viewpoints around the bed and set to see the whole baby. When three cameras are used, instead of two, it is easier to see the relevant markers in at least two cameras at once, and this is essential to find a good model at the end of the project. The camera setup is shown in Figure 3.1.

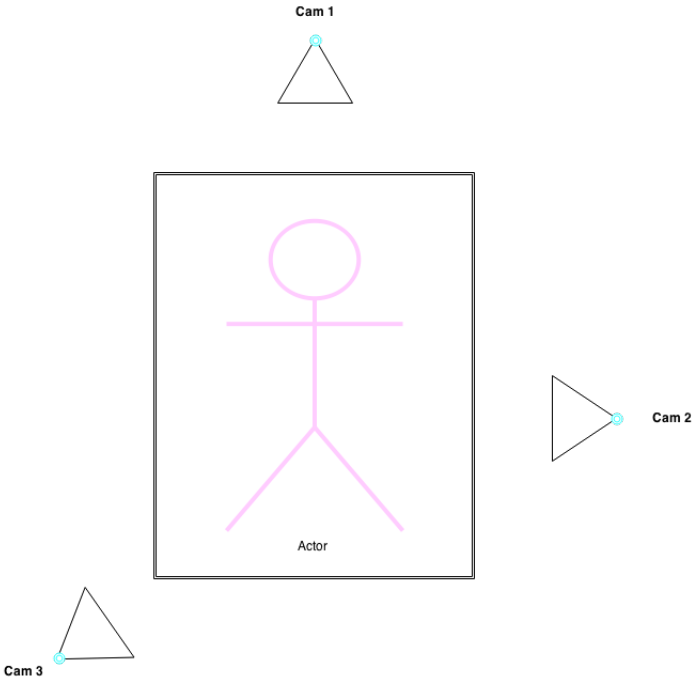


Figure 3.1: *Illustration of the camera setup used during video recordings at Östra Hospital*

This arrangement was used in all recordings, except for the first one. The three camera views are shown in Figure 3.3 and the setup of the first recording is captured in Figure 3.2.

The required equipment includes 3 web-cameras, a laptop, an external hard drive, recording software, 3 camera tripods, a chessboard, paper straps with dots used as markers, sticky-tape to fasten the straps on the infant's limbs, a small bed with

Figure 3.2: *The camera setup arrangement for the first video recording.*

(a) Camera one (b) Camera two (c) Camera three

Figure 3.3: *Images showing the view of the three cameras during calibration.*

bright linens for the infant to lie on, a heating mattress to keep the neonate warm, and something to raise the edge of the mattress to prevent the infant from rolling off the bed.

The videos are recorded in the avi format. The recording software used is Veedub64. In this project 30 frames per second (fps) is the chosen frame rate and the video encoding format used is H.264. The video is captured in 480 x 640 resolution and is recorded without audio to reduce the size of the files.

Figure 3.4: *Illustration of the markers attached to the infants' limbs.*

Camera calibration is required to make a 3D-model. In order to calibrate the cameras, the cameras capture the chessboard in different angles as shown in Figure 3.3. Further explanation of camera calibration is given in section 4.5.1.

Since the program can not manage to record three videos simultaneously, three different instances of the program need to run in parallel. A picture of this is shown in Figure 3.5. All three recordings are started manually so the videos do not start at the same point in time. In order to create a 3D model all three videos need to be synchronized.

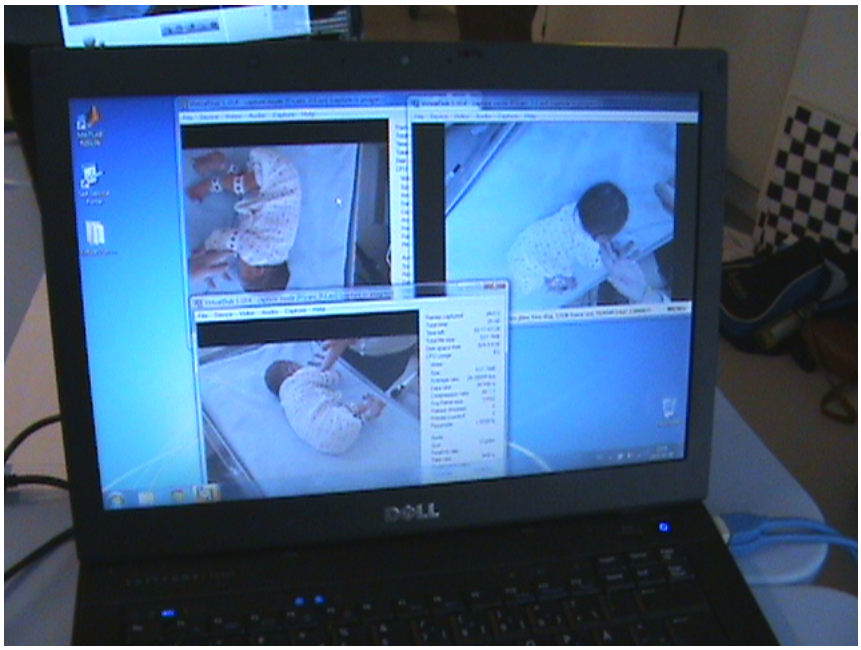


Figure 3.5: *Three videos of the recording running in parallel.*

The infants are prepared with markers on the infants' legs and arms, see Figure 3.6. The markers consist of 1 cm in diameter dots in different colours, which have been printed on white paper. To record the rotation of the infants' limbs, two different colours are used on every second dot. A band with dots is fixed around the infants' limbs using tape. There are eight sets of markers: two on each arm and two on each leg. The sets of markers are positioned on the wrist, over the elbow, on the ankle, and on the knee.

To be able to calibrate the extrinsic parameters the chessboard will initially lay on the bed before it is pulled away. The cameras are recalibrated with the chessboard between each video to ensure correct calibration. Then, without pausing the recording, the infant is placed on the bed. The infants are filmed for 10 - 15 minutes.

Ensuring the cameras are fixed is essential. If the cameras are moved slightly the camera calibration is not valid and the conversion to a 3D coordinate system will be skewed. Calibration is the only way to convert the camera coordinates to a real world coordinate system.

Figure 3.6: *The infant prepared with markers during first session.*

4 Method

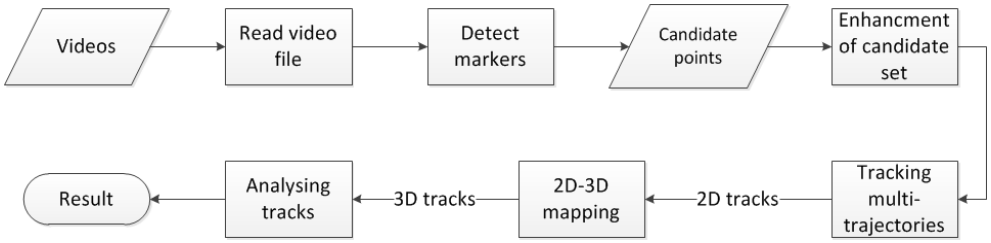


Figure 4.1: *Illustration of how the project work was performed. Parallelograms indicate data and normal rectangles indicate process.*

Essentially, the following method was used in the project: record videos of infants with markers attached to the limbs, detect these markers using image analysis, track the markers' movement to model trajectories in 3D space. The method was well suited for this project since the method could be tested with limited resources. In addition to a sufficient amount of infants to perform measurements on, testing of the method required only a few simple tools since most of the work can be done with a personal computer.

All programming in the project was executed in MATLAB© due to the ease of use and the group members' experience of the programing environment. MATLAB© has a great advantage over other environments because it includes toolboxes, which could be used for complex tasks if needed. These tasks would have required a lot more experience and time if they were to be done in another programing environment.

4.1 Detecting Markers with SIFT

One of the main problems to be considered in the project was how to recognize and detect the markers on the baby. In the field of computer vision this is called feature detection. Because feature detection is a common problem in computer vision, there are a wide variety of existing methods to solve this problem. In this project the SIFT algorithm was considered. The algorithm is widely proven and commonly used in a wide range of applications, such as object recognition, 3D modeling, and video tracking. The SIFT algorithm is robust and invariant to the features' scale, rotation, translation, and partially invariant to illumination-conditions. SIFT is also robust against local geometric distortion such as change in 3D-viewpoint[7]. These

properties make a great starting point for detecting and tracking the markers, which is an essential part of the project.

Given a gray scale image, the SIFT algorithm returns a set of interest points called keypoints in the SIFT framework, and descriptors which are distinctive representations of the keypoints or features. By using this algorithm the markers were expected to be found as keypoints to enable further operations like tracking the markers. For an example of SIFT, see Figure 5.2 on page 35.

One of the advantages of using SIFT is the accessibility. We chose to use an open source library called VLFeat[15] for MATLAB© which provided an extensive set of tools for computer vision, where SIFT was one of them. The ease of use and unlimited access made the option of using existing open source software very appealing for the project since writing this kind of application would require higher skills and be very time consuming.

The algorithm produces a large set of feature representations. Since the focus was detecting the markers, some parameters were changed within the algorithm to trim the result and get a smaller set of keypoints without discarding any of the keypoints representing markers. Some examples of parameters that can be changed in the algorithm are:

- **Levels** Specifies the number of levels for each octave in the DoG scale-space. Levels of a DoG scale-space is illustrated in Figure 2.2.
- **Octaves** Specifies the total number of octaves of the DoG scale-space. This is not limited by default to ensure true scale invariance. Octaves of a DoG scale-space is illustrated in Figure 2.2.
- **Peak Threshold** Filters peaks of the DoG scale-space which are too small in absolute value. Scale-space extrema detection is illustrated in Figure 2.3. The default of this parameter is 0, meaning all peaks are initially allowed.

As discussed in SIFT paper [7], some of these parameters such as *Levels* have optimal values with regard to stability and computational load. All parameters that have optimal values are set by default in the VLFeat functions. In this set-up, focus has been on trimming the Octaves and Peak Threshold parameters since they have the largest impact on the keypoint set while having a small chance of removing any keypoints for the markers. By limiting the Octaves, features of larger scale are neglected and Peak Threshold can be used to eliminate vague keypoints.

4.2 Enhancement of Candidate-Set

To get a more accurate representation of the markers than the representation SIFT produces, some methods must be applied to extract the keypoints representing markers from the keypoint candidate-set. Many methods were considered for this purpose. What was common for the derivation of all methods was looking for typical data in close proximity to the markers in order to successfully distinguish them from other keypoints. Characteristics specifically taken into consideration were the typical intensity or colour of the keypoint, and the shape.

4.2.1 Intensity and HSV Colour Filter

One of the most distinguishable characteristics of the markers were their colour: red, blue or black. To determine if a keypoint is localized on a marker or not, the mean pixel value in a circular area with radius equal to the keypoint's scale is calculated and compared to typical values of a marker. This calculation is illustrated in Figure 4.2. To detect the colour of a marker, the HSV representation of the current image is calculated. The HSV image provides a better parameter representation for detecting colour since the colour is mainly determined by the Hue parameter. The other two parameters, Saturation and Value mainly depend on the conditions of the recording, such as illumination. In comparison to the RGB representation, the HSV representation makes it easier to determine colour of markers regardless of other conditions. The black markers are detected by examining pixel values in the gray scale image. The keypoints whose HSV or intensity values do not the meet the criteria are simply discarded from the candidate set. The algorithm allows a certain range of the HSV and intensity parameter values in which the keypoint values are allowed to be within. If the keypoint values are outside this range, the keypoints are discarded.

In practice the RGB to HSV transformation is done by a simple command in MATLAB© called `rgb2hsv` which returns the HSV representation given an RGB image. The formulas for the transformation are shown below.

$$
\begin{aligned}
R' &= R/255 \\
G' &= G/255 \\
B' &= B/255 \\
C_{max} &= max(R',G',B') \\
C_{min} &= min(R',G',B') \\
\Delta &= C_{max} - C_{min}
\end{aligned}
$$

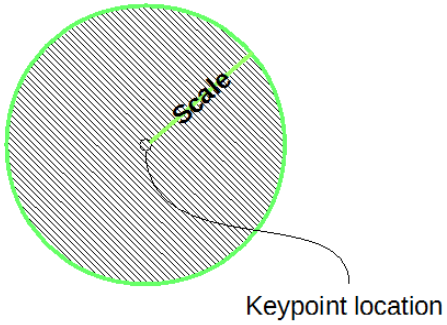


Figure 4.2: *The mean pixel value of all keypoints in the hatched area is compared to a typical value to determine if a certain keypoint is a possible marker.*

$$H = \begin{cases} 0^\circ & ,\Delta = 0 \\ 60^\circ \times (\frac{G'-B'}{\Delta} mod 6) & ,C_{max} = R' \\ 60^\circ \times (\frac{B'-R'}{\Delta} + 2) & ,C_{max} = G' \\ 60^\circ \times (\frac{R'-G'}{\Delta} + 4) & ,C_{max} = B' \end{cases}$$

$$S = \begin{cases} 0 & ,C_{max} = 0 \\ \frac{\Delta}{C_{max}} & ,C_{max} \neq 0 \end{cases}$$

$$V = C_{max}$$

4.2.2 Intensity Difference Filter

The Intensity difference filter calculates the difference between the mean intensity of the area shown in Figure 4.2, and a pixel value at a distance d from the perimeter of the circle in eight different directions evenly distributed over 360 degrees. An illustration of the eight directions where differences of intensity are calculated is shown in Figure 4.3. Since the markers used were dots on white wristbands, the intensity was expected to be higher in a majority of the directions from the keypoint area, indicating a round shaped object. Thresholds were set for the difference magnitudes and the number of directions that must give positive differences (increasing intensity moving from keypoint area) between the keypoint area and the proximity. Keypoints which did not meet the criteria were discarded. Appropriate values for the parameters are discussed in section 5.1.2.

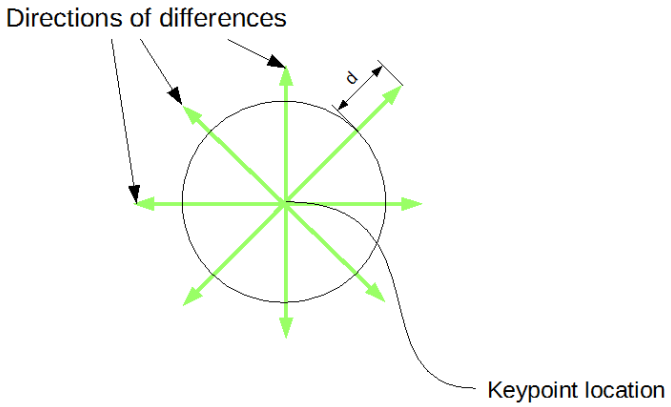


Figure 4.3: *Illustration of the eight directions where differences of intensity are calculated to detect an object's possible round shape.*

4.3 Tracking Multi-Trajectories

The basic principle of tracking the markers through the video is matching the keypoint-descriptors between consecutive frames, illustrated in Figure 4.4. This method is a simple way of utilizing the SIFT descriptors to link keypoints between frames. As long as the frame rate is sufficiently high, the difference between consecutive frames is expected to be small. If the change of the image is small, the changes in keypoints and descriptors are small as well, thus enabling the possibility to perform continuous matching. Because this method heavily relies on the results of matching, some possible problems which prevent matching from being successful needed to be considered:

- **Occlusion and instability:** Keypoints could be lost briefly due to obstruction of the marker or keypoint instability. This breaks the chain of consecutive matches making tracks unstable even if the keypoint only disappears for one frame. This problem was to some extent solved by matching the descriptor from the lost or occluded keypoint to frames later in the video.

- **False match:** Even if the descriptors are highly distinctive, a match does not guarantee the keypoints represent the same feature. False matches make sudden "jumps" appear in the trajectory and therefore do not reflect reality. By setting a threshold on the distance between matched keypoints, false matches with improbable lengths between them can be eliminated.

An algorithm to implement in MATLAB© was derived from the principal of matching with respect to the concerns mentioned above. The sequence of operations in the tracking algorithm is illustrated in Figure 4.5. The algorithm produces new tracks when new matches are found and discontinues tracks which fail to match, resulting in an increasing number of tracks. The number of frames a keypoint is allowed to be lost (the number of frames a keypoint is allowed to lack a match) before the track is

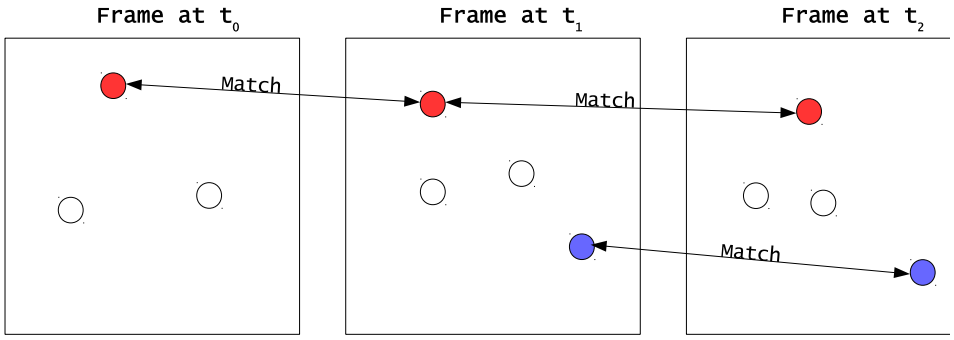


Figure 4.4: *The circles represent keypoints. All consecutive matches for a certain keypoint are assigned a particular track. The figure illustrates how the red circles have been assigned one track and the blue circles another track.*

discontinued does not have a given value and must be set by the user. When all the keypoints in one frame are checked for matches with vl_ubcmatch from the VLFeat toolbox, the set of keypoints from the next frame is considered and so on. All frames to be considered are looped through this way creating trajectories represented by three vectors: x-coordinate, y-coordinate, and frame. The frame vector contains the number of frames in which each coordinate pair (x and y) are present. A track could be represented as follows:

$$\mathbf{x} = \begin{bmatrix} 18.1 & 18.5 & 18.9 & 19.2 \end{bmatrix}$$
$$\mathbf{y} = \begin{bmatrix} 201.1 & 202.2 & 200.1 & 199.5 \end{bmatrix}$$
$$\mathbf{frame} = \begin{bmatrix} 51 & 52 & 53 & 54 \end{bmatrix}$$

Where x and y coordinates are stored keypoint locations from different frames. This trajectory is present in frames 51 through 54 in the sequence of tracking.

In summary, there were two parameters to set in this algorithm: Maximum distance between matched keypoints and the number of frames a keypoint is allowed to lack matches before the track is discontinued. The selection of these parameter values is discussed in section 5.2.3.

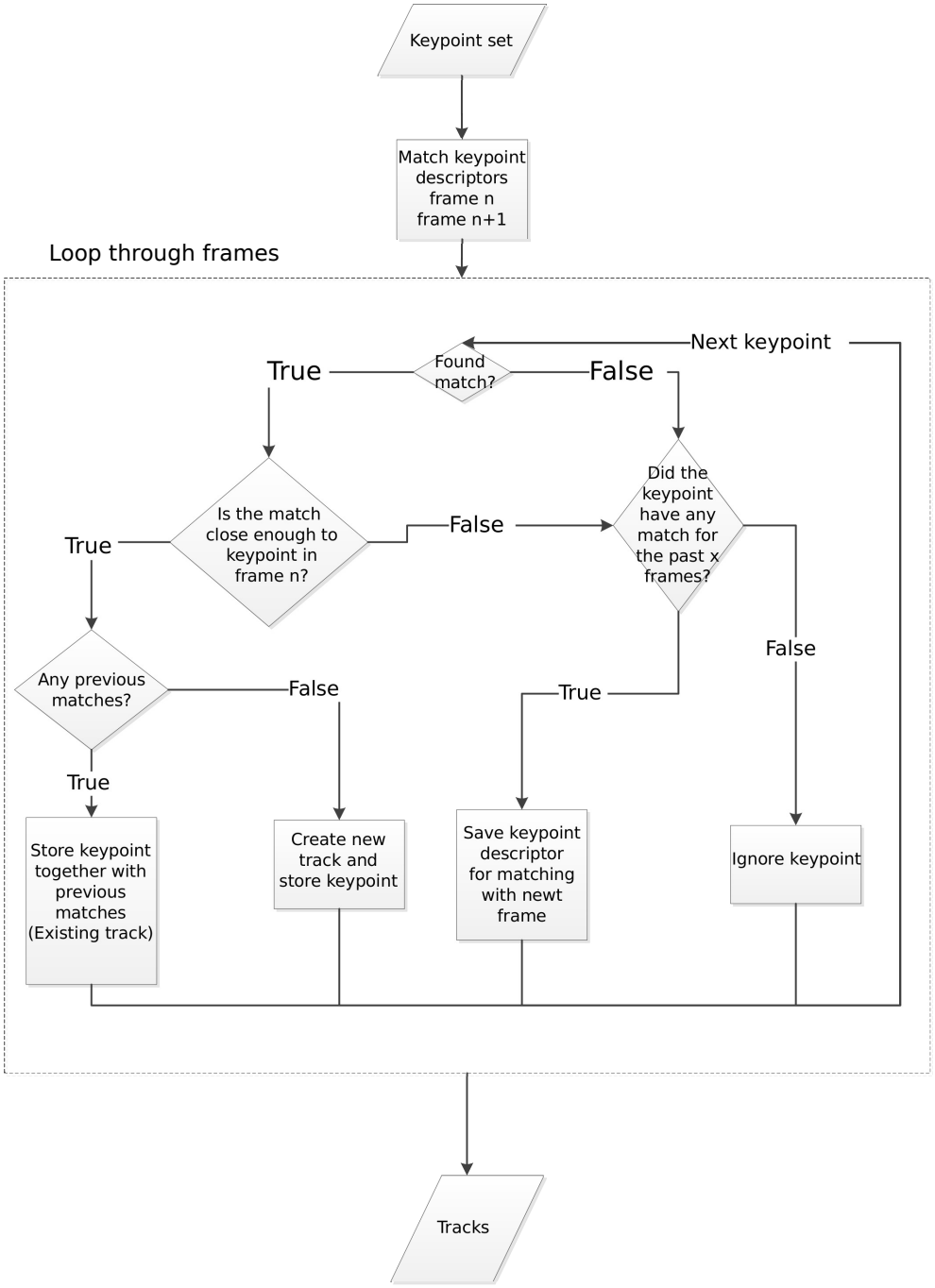


Figure 4.5: *Flowchart describing the sequence of operations in the tracking algorithm.*

4.4 Enhancement of Trajectory-Set

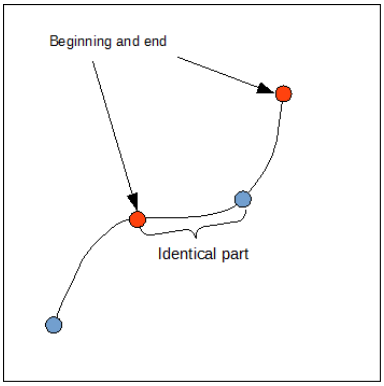
To increase the quality of the trajectories, they are further processed. Since the enhanced candidate set is not perfect, there are many tracks created which are of little or no value for modelling the movement. Therefore, efforts were made to narrow down the set of trajectories in order to create a more accurate model.

In the enhanced candidate-set there are keypoints in the background which are stable and successfully matched in the tracking algorithm, creating trajectories with no connection to the infants' movement. These were easily distinguished by their non-existent movement. Misjudgments could arise when examining the total movement of trajectories because of slight change of keypoint location even if there is no movement in the actual video. This is possibly due to slight changes in illumination and the fact the keypoint locations are a fit to surrounding data (see section 2.1). This results in long trajectories for keypoints, which are stationary as long as they are stable through many frames. Therefore, instead of examining the total movement, the total movement divided by the number of frames of the trajectory was examined, giving an estimation of the average step size of the trajectory. Trajectories with a considerably low step size are discarded from the set.

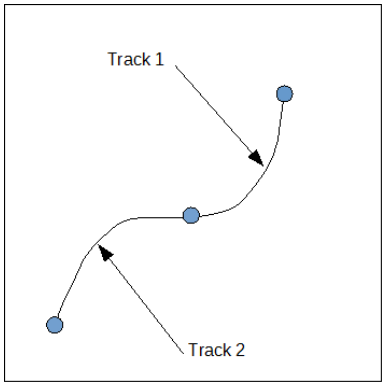
Trajectories which only exist for a small number of frames are also of little value for the model. These trajectories occur when unstable keypoints are tracked, therefore a limit was set for the minimum number of frames a trajectory must exist.

To refine the trajectory set even further, the phenomenon of keypoints with multiple orientations and descriptors was addressed. The consequence of multiple descriptors when tracking is multiple trajectories for one keypoint, in other words, identical or partially identical trajectories. An example result of this phenomenon is illustrated in Figure 4.6. Furthermore, it is possible for the trajectories to be connected due to sudden descriptor change as illustrated in Figure 4.6. An effort was made to merge these trajectories that share start or ending point, which should result in longer trajectories. This was done by checking all trajectories for common beginnings or ends and connecting them into one trajectory.

In summary there were two parameters to set values for in this method: minimum value of movement (*movement=length/number of frames*) and minimum number of frames a trajectory must exist. These values are further discussed in section 5.2.4.



(a) Partially identical tracks



(b) Connecting tracks

Figure 4.6: *(a) and (b) shows two different types of possible tracks due to the assignment of multiple descriptors to one keypoint.*

4.5 Multiple View Geometry

The camera calibration was performed with MATLAB©. The toolbox used was TOOLBOX_calib, which is a free-to-use release of a camera calibration toolbox for MATLAB© retrieved from [9]. The majority of the used functions were modified to better suit the needs of the project.

4.5.1 Camera Calibration

The intrinsic parameters are important to determine the camera matrix used to convert coordinate systems between the camera and world perspectives. To estimate these parameters one camera needs to capture a chessboard in multiple angles. This type of procedure was only performed once and the data was stored and reused during each session onwards. The extrinsic parameters however, needed to be redefined for each camera setup. These could be determined by capturing the chessboard from one angel.

The calibration images were extracted from the videos using MATLAB©. From watching the film on an external media player, the film sections of interest could be determined.

To calibrate, the corners of the chessboard pattern were clicked to create a rectangle with four points. The first clicked point on the chessboard would be the origin of our coordinate system. This creates the 3D room, from the cameras, relative to this origin.

4.5.2 Finding 3D Coordinates

After successfully calibrating the intrinsic and extrinsic parameters of each camera, the projection of pixel coordinates into world coordinates needed to be calculated. Two functions were created for this. These functions enable transformation between pixel, camera, and world coordinates. The functions were created by using the projection matrices in Equations 2.1 and 2.2.

The closest distance between two projections was estimated using the MATLAB© function lsqlin. This was set as the best estimation for a 3D coordinate, given pixel coordinates from two cameras.

4.5.3 Finding Corresponding Trajectories

To convert the 2D trajectories to 3D trajectories, a function was needed to find the 2D trajectories in the different camera views for the same point, in other words the corresponding trajectories in different cameras. The function takes all the trajectories and compares two cameras' trajectories at the same time. To calculate which trajectories' exist in the same frames, the function creates an $N \times N$ matrix, where N is the number of trajectories from the camera which created most tracks. This matrix is given values of one for trajectories which exist at the same time and zero for the trajectories which do not. For example, if row 1 column 3 has the value one, it means trajectory one from the first video exists at the same time as trajectory three from the second video. Since a lot of trajectories exist at the same time, which trajectories correspond to which has to be determined. This is done by converting all the trajectories which exist at the same time to 3D space and saving their highest residual value (the highest distance between the projection lines squared) into another $N \times N$ matrix. The best corresponding trajectories are then selected by finding the lowest value in each row. This means all trajectories from each camera only correspond to one other trajectory from another camera. Two parameters were used to filter out additional wrong trajectories. One parameter was used to limit the highest residual allowed in a match and one parameter set a limit on the amount of frames the trajectories at least had to exist at the same time. These last remaining corresponding trajectories are then plotted in 3D to represent the movement of the points on the baby.

5 Results

This section is divided into three main parts: Experimental Setup, Test Results, and Performance Evaluation. The Experimental Setup describes the final method setup used for creating the model. In the second part, Test Results, the results obtained from the chosen methods are presented and visualized. In the final section, Performance Evaluation, the accuracy of the main parts of the project, detecting & tracking markers, and mapping corresponding trajectories is evaluated.

5.1 Experimental Setup

In this section the parameter setup for producing our results are presented. Worth noting is that the values are not optimised for best result but are set so that a fair evaluation of the tested method can be done.

5.1.1 SIFT Algorithm

When deriving the used parameter values of Octaves and Peak Threshold, a conservative approach is applied to make sure no valuable data is lost in the first step of producing the candidate-set. The parameters are set as high as possible while still maintaining a high certainty no valuable keypoints are discarded. The final parameter setup is:

$$Octaves = 3$$

$$PeakThreshold = 2$$

By changing the parameters, the keypoint-set can be reduced by 50% - 60% without discarding any of the keypoints representing markers. An example of this is shown in Figure 5.2 on page 35 where red circles indicate the marker keypoints.

5.1.2 Enhancement of Keypoint-Candidate-Set

To obtain a robust and sufficiently accurate set of keypoints, there are a few parameters in the filter methods to consider. The current parameter setup is based on

evaluation of manually examined data from different videos. From this examination the parameters that guaranteed the inclusion of all markers, regardless of lighting conditions and other irregularities, were found.

Intensity and HSV Colour Filter

To set the range of values in which a keypoint is considered a possible marker, mean values of the different coloured markers are manually examined in different camera angles. The different camera angles give different lighting conditions which occasionally produce a wide spread of marker values. The purpose of setting the range in this manner is to obtain a robust filter. The range used is based on the values in Table 5.1 on page 33 where mean values of red and blue markers are presented. From those values the range of allowed values for the HSV parameters are set.

The range of values for the HSV parameters for blue markers are:

$$0.55 < H < 0.6$$

$$0.45 < S < 1$$

$$0.4 < V < 1$$

The range of values for the HSV parameters for red markers are:

$$0.65 < H < 1$$

$$0.4 < S < 1$$

$$0.4 < V < 1$$

The intensity values for black markers are estimated from testing. Since the intensity in the gray scale image is low in dark areas it is more intuitive to set an appropriate value for that parameter. The performed tests show a limit of 75 for the intensity value is suitable in order to both maintain keypoints on black markers and considerably reduce the candidate set. An example of an area from which pixel values are obtained is shown in Figure 5.1.

Intensity Difference Filter

The parameters d (distance from perimeter), number of positive differences, and difference magnitude are also set by manually examining typical marker data to estimate parameter values which are verified by testing. An example of an examined marker is shown in Figure 5.1. The parameters are set to allow deviation from the ideal case to include all markers. The parameters that are used and provide a robust result are set as follows:

$$d = 4$$

$$\textit{Number of positive differences} = 6$$

$$\textit{Difference magnitude} = 20$$

Camera angle	H	S	V
1	0.7344	0.5548	0.4882
2	0.9828	0.7722	0.9020
3	0.8597	0.8259	0.5142

Camera angle	H	S	V
1	0.5706	0.6541	0.9150
2	0.5764	0.5573	0.7575
3	0.5988	0.5053	0.4719

Table 5.1: *Mean values of HSV parameters for red and blue markers shown for the three camera angles. The top table includes values typical for red markers and the bottom table values typical for blue markers.*

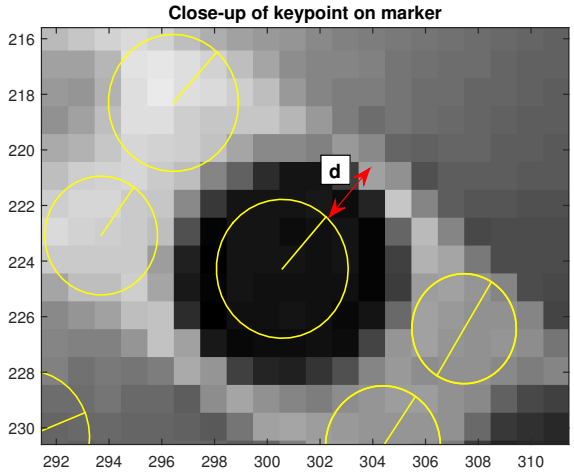


Figure 5.1: *Close-up image of keypoint on marker showing how intensity drastically changes around 3 pixels from the marker perimeter. The difference magnitude is 137 and appears positive in all predefined directions.*

5.1.3 Tracking Algorithm

The parameters to consider for the tracking algorithm are threshold for maximum length between matched keypoints and tolerance for how many frames a descriptor can lack matches without being discarded. These parameters are set by examining false matches and from these deciding appropriate values. For the maximum distance threshold the desired value must allow fast movement and discard false matches. A larger distance threshold is set for keypoints which match between non-consecutive frames since movement may have been larger between matches in that case. The tolerance of number of frames a descriptor may miss matches has been set as large as possible while still maintaining a probability of finding a true match. The parameters used are:

$$Maximum\ distance = 10\ \text{pixels (For consecutive matches)}$$

$$Maximum\ distance = 25\ \text{pixels (For non-consecutive matches)}$$

$$Tolerance = 6\ \text{frames}$$

5.1.4 Enhancement of Trajectory-Set

The parameters used in the Trajecotry-Set algorithm enhancement are:

$$Minimum\ number\ of\ frames = 4$$

$$Minimum\ movement = 0.5\ \text{(pixels per frame)}$$

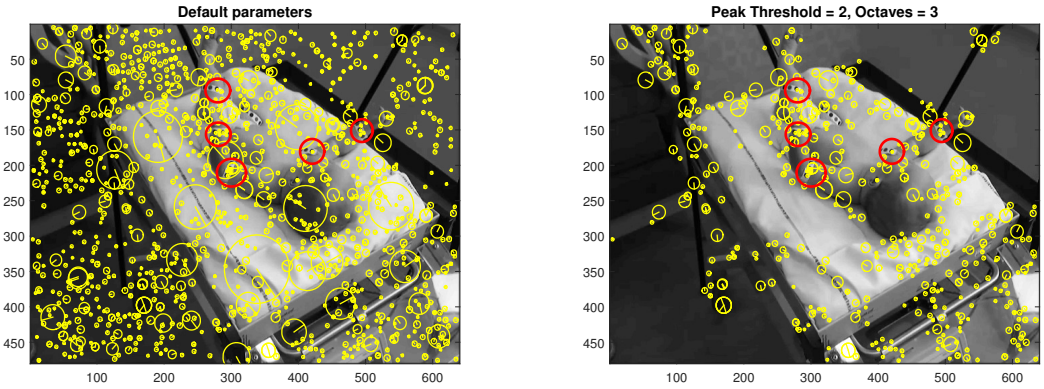
It is probable trajectories which exist in more than four frames provide little or no useful information about the movement. But since the analysis method of the trajectories and the model is not known, it is safer to preserve trajectories rather than discard them. Perhaps a large set of very short trajectories could provide a useful model. To set an appropriate movement limit, some trajectories have been examined. With the current camera setup, a limit of 0.5 pixels per frame appears to exclude background trajectories.

5.2 Test Results

This section visualizes and presents the results of using the above mentioned methods. The methods appear in the chronological order to improve the final model, step by step.

5.2.1 SIFT

In Figure 5.2 the result of using SIFT on a image is shown. It is notable the keypoint-set is very large (approximately 1000 keypoints) using the default parameters. The majority of the keypoints are points in the background, which are of no value when modelling the movement. The keypoint-set can be heavily reduced by changing the parameters (shown in Figure 5.2b) but still maintain valuable keypoints which are localized on markers, indicated by red circles. The result is the candidate-set.



(a) (b)

Figure 5.2: *Demonstration of the impact of parameter change in the SIFT algorithm. Keypoints are represented as yellow circles with radius equal to their scale. In the figure the candidate set is reduced by 58.8%. Evidently, keypoints representing markers are indicated by red circles.*

5.2.2 Enhancement of Candidate-Set

A comparison of using SIFT with the mentioned parameter setup, and using SIFT and enhancing the candidate-set with filters is shown in Figure 5.3. Upon close examination of the result, the parameter setup successfully removes a large portion of keypoints and preserve keypoints on markers. This method generally reduces the candidate-set by 80%-90%.

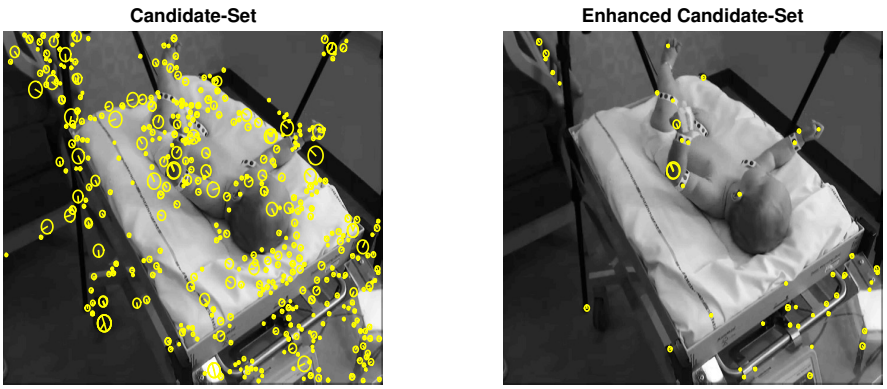
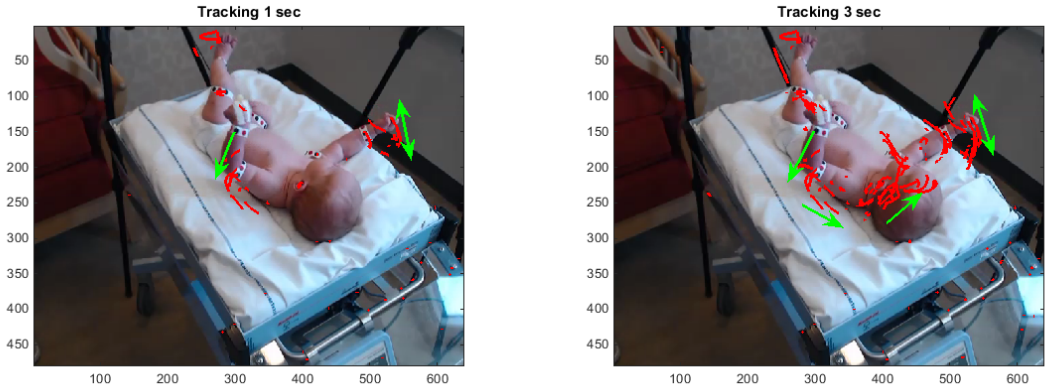


Figure 5.3: *Illustration of the significant reduction of keypoints in the candidate set as a result of enhancement methods. The figure shows a removal of approximately 90% of the keypoints.*

5.2.3 Tracking Multi-Trajectories

A visualisation of how the trajectories' movements are represented is presented in Figure 5.4. The representation in Figure 5.4 show how the movement, to a certain extent, is modelled. In the lower right corner of Figure 5.5b, stationary trajectories in the background are clearly visible. Trajectories that do not model any movement are unnecessary which makes the trajectory-set subject to refinement.

The number of trajectories increases with the length of the video. Approximately, four new trajectories are created per frame. Figure 5.4a shows 153 plotted trajectories and Figure 5.4b 403.



(a) (b)

Figure 5.4: *Plotted tracks on the first image in the tracking sequence after one second and three seconds respectively. Red curves are trajectories, and green arrows show the direction of movement.*

5.2.4 Enhancement of Trajectory-Set

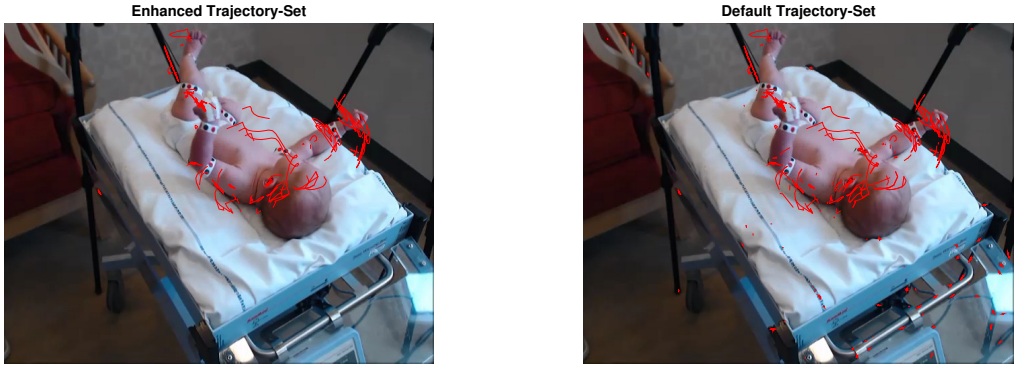
In Figure 5.5 the visual change due to enhancing the trajectory-set is shown. Almost all background trajectories in the default trajectory-set are removed in the enhanced trajectory-set. Although enhancement of the trajectory-set reduces the number of trajectories by about 80%, the trajectory movement is clearly visible. In Table 5.2 the proportions of the removed tracks for each step in the algorithm (merging, minimum number of frames, minimum length) are presented.

5.2.5 Camera Calibration

As previously mentioned, some of the camera calibration toolbox functions are modified. These modifications led to a self-sustaining program without much need of user input.

Length of sequence (frames)	Merging (%)	Min-frames (%)	Min-length (%)
90	16.1	64.1	19.7
150	15.7	65.2	17.7
400	16.8	68.8	14.4

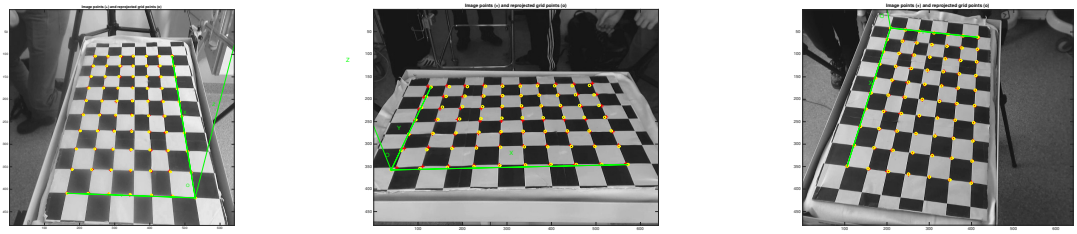
Table 5.2: *Proportions of trajectory removal by different criterion for different lengths of tracking sequence*



(a) (b)

Figure 5.5: *Comparison of processed trajectory-set(a) and default trajecotry-set(b). In both cases the tracking represents a four second sequence equivalent to 120 frames. In this case, the amount of trajectories is reduced by 81.7 % by processing.*

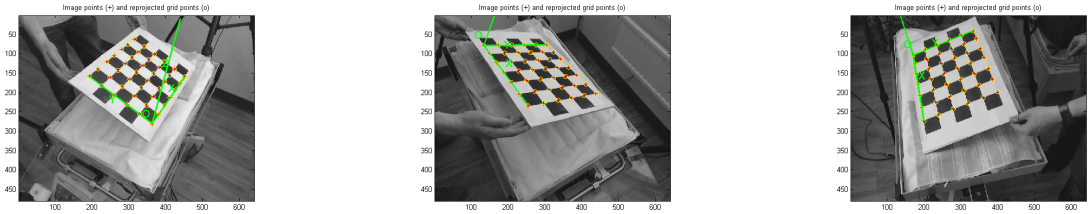
The toolbox estimates the world coordinate axis from the extracted pictures of the chessboard. The result appears to be good judging from the plots. In the beginning of the project a bigger, rougher chessboard was used. The squares were slightly uneven in area meaning the calibration was not ideal. The three camera views are shown in Figure 5.6 together with the calibration result.



(a) Camera one (b) Camera two (c) Camera three

Figure 5.6: *Images showing the calibration result for the first chessboard for the three cameras. The lines and dots are meant to match the chessboards' pattern.*

The uneven chessboard was replaced with a new chessboard with equally spaced squares, which improved the accuracy of the calibration. The improved calibration result is shown in Figure 5.7 and is based on six different angles.



(a) Camera one (b) Camera two (c) Camera three

Figure 5.7: *Calibration result for the second chessboard. The lines and dots are meant to match the chessboards' pattern.*

The camera calibration can be calculated with a pixel error of $err = [1.88516, 0.67679]$ using one angel of the chessboard. Pixel error means the number of pixels the calibration differs from the correct result. If the angels are increased, the accuracy of the calibration also increases. For example, the pixel error of the intrinsic parameters will decrease from $err = [1.88516, 0.67679]$ to $err = [0.22691, 0.30936]$ when increasing the number of calibration angels from one to six. One camera's position after the parameters have been calculated can be visualized as in Figure 5.8.

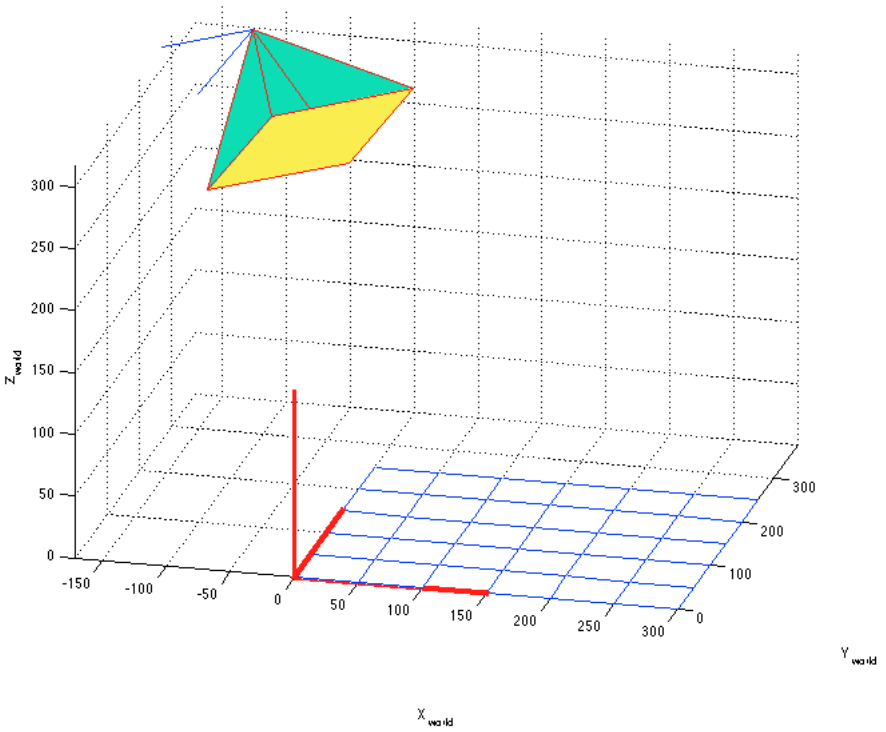
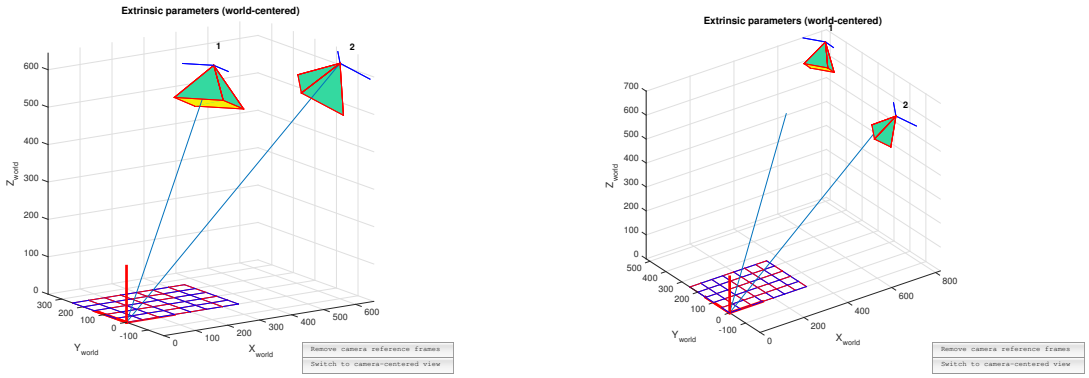


Figure 5.8: *The picture shows the position of the camera in the room after calibration.*

5.2.6 Mapping from 2D to 3D

The 3D coordinate of a point can be calculated by projecting two pixel points in the different camera views, originating from the same marker. The 3D point is set

as the midpoint of the closest distance between these projection lines. This can be visualized as in Figure 5.9a, where the projection lines from two cameras, and intersection has been plotted. The lines range between the cameras' nodal point and the origin. This result proves the conversion from camera space to world space is correct. If the calibration is not correct, the projections will not be either. This a higher residual is obtained when matching correct trajectories depending on the calibration error. Figure 5.9b visualizes how what the projections will look like for a poor calibration.



(a) Ideal conversion (b) Worse conversion

Figure 5.9: *Plot of 2 lines in 3D, ranging between the cameras' nodal point and the origin. The two tetrahedrons are representing camera one and two.*

The program created to map trajectories in 3D, will find the best match for tracks existing simultaneously in two cameras. The program will then map these trajectories to 3D coordinates. The result from one frame is visible in Figure 5.10. In the figure the points, mapped in 3D and the reverse projection of these onto camera frames, is visible. The reverse projection visualizes the 3D position found by the program. The figure shows how the program successfully has managed to map several corresponding points in 3D. It also shows one faulty 3D mapping which has arisen from the fact two markers originate from the same epipolar plane. This problem can be overcome by looking at the residual over several frames. Two points from different markers will seldom stay in the same epipolar plane over several frames, thus the residual will eventually be big and the match can be excluded.

When trajectories from 2549 consecutive frames were extracted and then matched through our program we could plot them in 3D as shown in Figure 5.11a. However, this was with trajectories existing in as few as ten frames. If the trajectories instead were calculated for those existing in more than 50 frames we received the plot shown in Figure 5.11b. Here the results of the trajectories are mostly localized around the feet and the baby's right arm. When any of these points, which existed in 50 or more frames, were projected onto the 2D images, as in Figure 5.13, it is clear it follows a point accurately. In the figure with all the trajectories (Figure 5.11a) there are also more trajectories following the left arm than the right arm, but these were too short and therefore do not exist in Figure 5.11b.

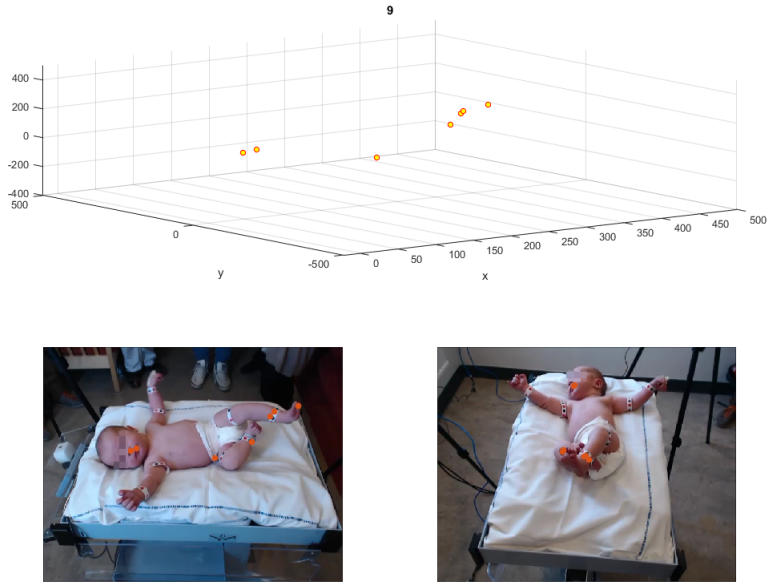
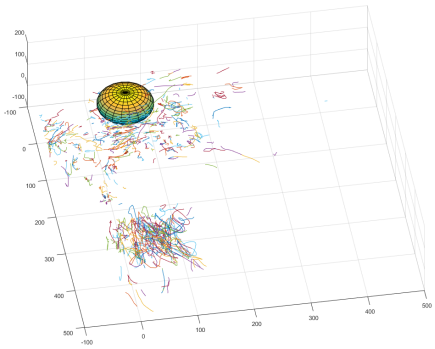
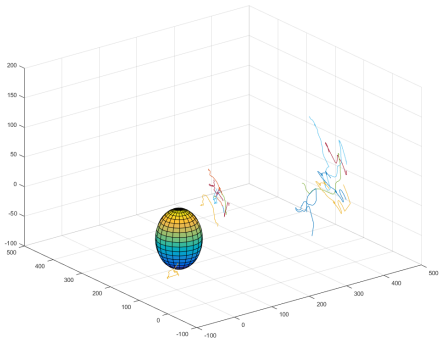


Figure 5.10: *This pictures shows trajectories in one frame mapped in the 3D room. The two subpictures show the reverse mapping of these points back to 2D. Many correct mappings are shown, as well as one incorrect seen in the bottom left picture on the infants hip.*



(a) All matching trajectories plotted in 3D. In this figure the trajectories are found around the arms and legs mostly.



(b) All matching trajectories which exists in more than 50 consecutive frames. Only a few trajectories of the right arm and trajectories of the legs are remaining.

Figure 5.11: *Trajectories calculated from evaluating 2549 frames, plotted in 3D. The sphere represents the approximate position of the baby's head.*

5.3 Performance Evaluation

This section evaluates the two main parts of this project, marker tracking and 2D-3D mapping. The two parts are evaluated separately, since the methods can be used one by one. The resulting model are dependant of the combined result, though, and in this section are the evaluation described for one video.

5.3.1 Detecting Markers & Tracking

The first steps of detecting markers are crucial for the final result of the movement modelling. Evaluation of the completeness of detecting markers is done by examining the proportion of detected markers to visible markers. In Table 5.3 some mean values of marker detection and clearly visible markers in each frame are presented for the different camera angles. It shows the proportion of detected markers to visible markers is rather small. This is due to inherent properties of the SIFT algorithm or measurements since the filters very rarely discard a keypoint on a marker (only one case during data examination).

Camera	Visible markers	Detected	Proportion (%)
1	9.75	2.75	28.2
2	9.5	4	42.1
3	10	2.75	27.5

Table 5.3: *Mean values of visible markers and proportion of detected markers to visible markers with SIFT. The values are calculated for 6 random frames.*

Since the keypoint-set is not perfect, there is a chance to model movement by tracking keypoints which do not represent markers but are located on the infant.

To measure the performance of the tracking algorithm and the trajectory-set refinement method, mean, and median values of track length in pixels and frames (lifespan) have been calculated, see Figure 5.12. Doing this should give some sense of how much movement is actually captured by the trajectories. Also taking into account the total number of trajectories, shown in Figure 5.12c , gives an overview of the performance. Looking at the lifespan of trajectories it seems it stabilizes, meaning the number of discontinued and created trajectories are approximately equal. The length of the trajectories shows movement is captured and the average length of trajectories stabilises and picks up movement continuously throughout the video. Some differences can be distinguished between camera views, it seems camera view two, in this case, is somewhat favorable in terms of picking up trajectories and maintaining continuous matches. Although similar tendencies can be spotted between camera views.

5.3.2 Connecting Corresponding Trajectories

When the corresponding trajectories are found their accuracy needs to be validated. A new set of manually created trajectories are created by simply clicking at the right marker in a number of consecutive frames, for all three cameras. These trajectories will be considered as the ground truth and so close to the ideal tracking as possible, although it is not possible to click on the exact pixel coordinate at all times. This ground truth is the reference trajectory used to validate the SIFT generated trajectories which was successfully converted in to the world coordinate system by our written program. The ground truth and the evaluated SIFT trajectory are both plotted in 3D in Figure 5.14 to visualize the similarities.

The validation is performed by evaluating the euclidean distance between the SIFT trajectory and the reference trajectory converted into 3D coordinates. The distance is given by the Pythagorean formula for the Euclidean 3-dimensional space:

$$\mathbf{d}(\mathbf{p},\mathbf{q}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + (q_3 - p_3)^2}$$

Where $\mathbf{p}$ and $\mathbf{q}$ are the matched point vectors from the SIFT trajectory and reference trajectory, respectively.

Another way to visualize the result is by projecting the 3D trajectory back to 2D and plotting it on the original video. The accuracy of the 3D model can be determined by following the trajectory in each frame. In Figure 5.13 the same 3D trajectory as in 5.14 can be seen projected onto the images. By analysing the video, it is clear the point follows the foot accurately. This is represented by the green line in the sub figures in Figure 5.13.

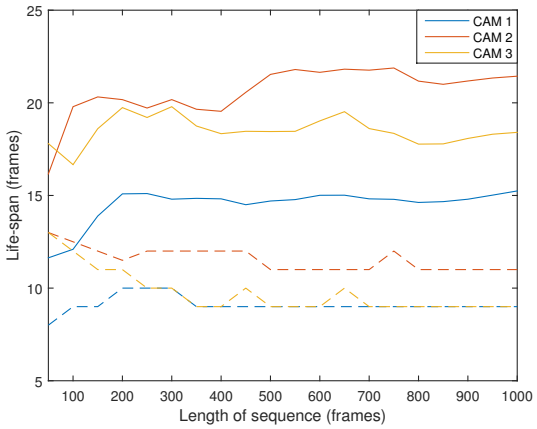
A selection of the euclidean distance between the reference trajectory for the left ankle's marker and the SIFT evaluated trajectory is displayed in Table 5.4.

Euclidean Distance (mm)
1.53
0.97
⋮
4.41
4.16
⋮
16.49
1.15
Mean Value = 16.30

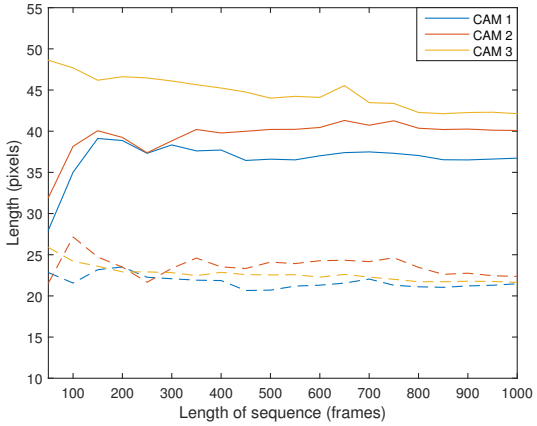
Table 5.4: *A selection of the Euclidean distance of the ground truth and the 111 element long trajectory in 3D space.*

The unit of distance is millimetre, and a low value is pursued. The high mean value is probably obtained by the sometimes imprecise reference trajectory, when one might have clicked on the wrong marker or on the edge, as can be visualized in Figure 5.14, where the two trajectories diverge at some points. The distance between the trajectories are probably affected a lot by human error but it validates that we are following the right point in space.

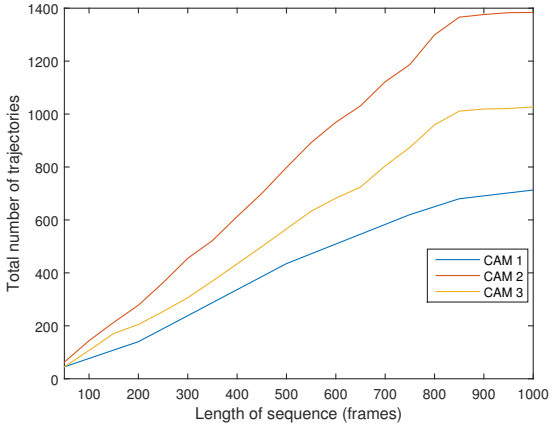
The trajectories' coordinates are plotted versus a time sequence in one video, in Figure 5.15, to visualize the error for each axis.



(a) Measure of trajectories' lifespan



(b) Measured length of trajectories



(c) Total number of trajectories

Figure 5.12: *Measurements of trajectories' characteristics to give a picture of how much movement is tracked. Dashed lines represent median values and solid lines mean values. The low median compared to the mean suggests the data is skewed to larger values. The values are calculated from the enhanced trajectory-set.*

(a) The start of trajectory from camera 3 is represented by the red dot.

(b) The start of trajectory from camera 2 is represented by the red dot.

(c) The end of trajectory from camera 3 is represented by the red dot.

(d) The end of trajectory from camera 2 is represented by the red dot.

Figure 5.13: *A trajectory consisting of 111 frames matched from two cameras in 2D and converted into 3D, projected back onto the 2D images to view the result. The trajectory's movement is marked by the green line.*

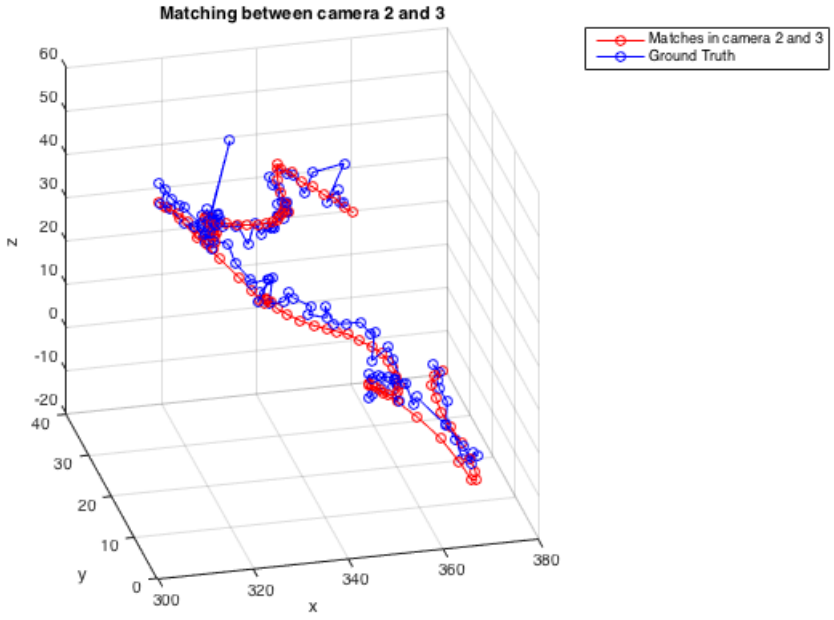


Figure 5.14: *The ground truth and the evaluated SIFT trajectory in 3D space.*

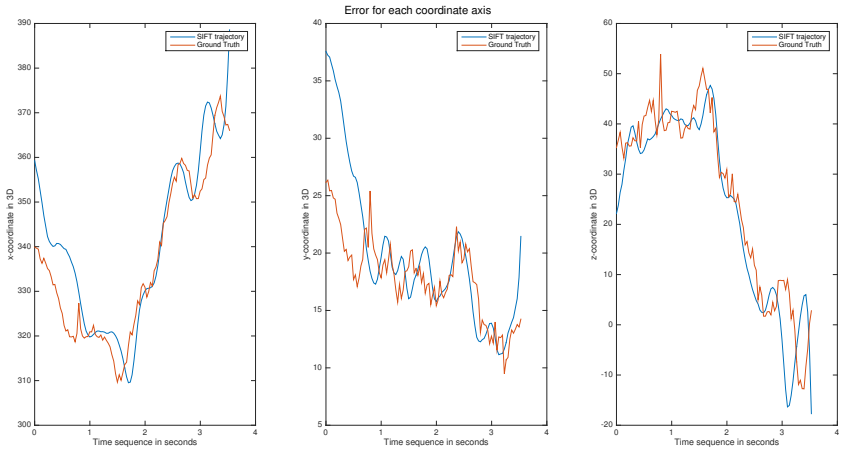


Figure 5.15: *Error of each coordinate axis in 3D space.*

6 Discussion

6.1 2D-3D mapping

The purpose of using multiple camera angles is to increase the chance of capturing a marker from multiple views in a certain frame. With the described camera set-up the problem still persists. The chance of seeing the same marker continuously over a large amount of frames is low with the cameras set up in wide angles. The markers need to be visible in multiple cameras in the same frame to be matched, which creates problems with the current camera set-up. By using wristbands with multiple markers instead of single markers on the limbs, the probability of finding markers corresponding to a certain limb in multiple camera views in the same frame increases. The keypoints do not need to be from the exact same marker for the program to match them, meaning it is sufficient if the tracking software can find at least one keypoint from the wristband. To accurately match two different markers from the same wristband, the 2D trajectories need to exist simultaneously in the same frame for at least 50 frames. If the program matches two markers next to each other (for example, a blue marker from camera 2 and a red marker next to the blue in camera 3) the result is a good 3D-trajectory.

Another adjustment that could have been made, to the function matching trajectories, is to instead of using the maximum residual to filter out wrong trajectories, it could use the mean value of all the residuals in the trajectory. This would not filter a trajectory in case only a few of the points were misleading but the rest of the trajectory had a small residual on average. These few points which had too high residual could then be filtered out from the trajectories after the match was made. The difference in the results in filtering depending on the maximum value or the mean value did not show any noticeable results. This might be due to the fact the trajectories are not long enough to give a big enough difference to the mean value.

The larger the number of frames the 2D-trajectories exist simultaneously in, the more accurate the matching result is. However, if the baby lies still it is possible for a marker on, for example, the hip and the foot to be in the same epipolar plane for multiple consecutive frames. This phenomenon is illustrated in Figure 5.10. Thus, 50 frames is not necessarily enough to match 2D-trajectories to 3D. If the 2D-trajectories exist for more frames, there is a high possibility either the hip or the foot moves in a different pattern. The majority of trajectories do not exist for that many consecutive frames, thus after filtering out the short trajectories there are

few trajectories left to match. Therefore, the probability of false matching is low. However, this is problematic since the resulting trajectories in 3D will be dependant on the number of long trajectories in 2D.

6.2 Detecting markers & Tracking

The set of candidate points produced by the SIFT algorithm is highly dependant on the video quality and the recording set-up. Insufficient or inhomogeneous lighting conditions could create difficulties when detecting markers, which yield difficulties when trying to produce continuous matches. In Figure 5.11b it is visible there are more long trajectories on the right side. This could possibly be due to lightning or the resolution. This kind of effects has not been taken into consideration and could have negative effects on modelling the movement. Since the analysis method of the model is not decided upon it is hard to determine if it will cause any problems or not. It is not uncommon for SIFT to fail to detect visible markers, regardless of parameter settings (see Table 5.3 on page 41). This could be because of video quality, such as lighting or resolution, or because of the nature of the SIFT algorithm. No formal studies have been done on how lighting conditions or resolution affects the candidate-set, more time needs to be put on investigating how the marker detection can be improved.

The video recordings took place in the hospital where the environment was stressful and there was a lack of space. The recordings were performed in different rooms each session, resulting in different lighting in videos from different sessions. Moreover, the paper tape with markers has not been a satisfying solution. The markers are attached with an adhesive tape, which often comes off. In some cases, the paper strips have been too short to reach around the infants' limbs resulting in temporary extensions having to be used. In other words, conditions for doing the measurements are not ideal. Having a stationary set-up where measurements can be done quick and easy would allow experimentation with the measurement set-up to improve performance and acquisition of more measurements. This is difficult to arrange since the recordings must be held at the hospital where keeping a room for the recording equipment is not possible. The paper strips could be replaced by more lasting bands, such as adjustable silicone bands. To improve measurements further, an experimental set-up at Chalmers should be considered, where mock-up measurements could be done to test different set-ups.

In the first example videos the baby had only black markers, while in the latter videos bands with blue and red markers were used. The reasoning behind using multiple colour markers was to distinguish the limbs' rotation. After using the SIFT algorithm on the new videos it was found the black markers yield a better result than the coloured markers. The reason is most likely due to higher intensity of the coloured markers in the gray scale image making them harder to detect. Using blue and red markers gives lower contrast than black markers in the gray scale image.

Since rotations are not taken into account, black markers on light colored strips would probably be the best choice.

The implementation of merging trajectories in the enhancement method is a very time consuming process, making tests inefficient and tedious. One solution to this problem would probably be to address the problem in the tracking algorithm. If the tracking algorithm could determine which trajectories are identical, in other words, keypoints with multiple orientations, the problem could be solved before post-processing. In addition to this, another improvement should be considered in the tracking algorithm. It is possible for two keypoints to match to the same keypoint in the next frame, meaning two trajectories are equal from a certain frame and onwards. One solution to this could be to use more than one form of tracking. Since vl_ubcmatch is the best form of tracking it can be used as the main tracking source. However, other methods such as "nearest neighbor" and motion models could probably be used to determine the best match and discard others, neglecting unnecessary trajectories.

In summary, detection and tracking methods seem to provide useful information about movement but further evaluation must be done to determine if the result is sufficient or needs to be improved to create useful models.

7 Conclusion

This thesis aimed to investigate the possibility of creating 3D models of infants by using image analysis. With good enough measurements it seems very plausible the written program could create good 3D-models. With the current measurements the software creates promising results. Previous research indicates that the trained human eye and computer-based video analysis can distinguish between normal and abnormal movements. Therefore, it seems possible to develop this project into a technique to create adequate 3D models that can be used for diagnosing and predicting neurological diseases in infants.

The 3D movement-models have to be further improved. The lack of corresponding 2D tracks is one of the main restricting factors in the computation of the 3D model. Tracks are found in 2D but tracks corresponding to the same marker are rarely found simultaneously in the different camera views. This makes the creation of a complete 3D model hard to create. In the studied video of this project, the infant's right arm and both legs have been successfully modelled. The left arm is hard to model due to the lack of long trajectories describing its movement, which are needed when filtering the bad matches out. An alternative set-up of the recording cameras was considered as one possible method for increasing the frequency of finding corresponding tracks. In the current recording set-up, angles between the camera views are large. This increases the risk of a marker being obstructed in one of the cameras. A set-up where the recording cameras are placed closer together would likely increase the amount of corresponding tracks found. In this case, vl_ubcmatch could possibly be used to identify corresponding tracks. Points originating from the same object could be matched by vl_ubcmatch, but the algorithm requires a smaller angle to perform the matching. Using vl_ubcmatch to identify corresponding trajectories would further improve the 3D mapping.

Also optimisation of the parameter values for the different algorithms and programs could improve the result with longer trajectories for example. This is not believed to change the result fundamentally but all improvements are a step in the right direction.

Suggestions for further improvement of the model:

- Investigate other possible measurement set-ups.
- Investigate other detection methods to increase the number of markers detected.
- Parameter optimisation

It is possible to make computer-based 3D models of an infant's movements using video-recordings. More work needs to be done before evaluating whether the method developed in this project can be used to predict neurological diseases. Nevertheless, the future of this method in clinical use seems promising. The results indicate that with improved models and more data, the method could be a useful tool to evaluate the functionality of an infant's nervous system.

Bibliography

[1] Einspieler C, Prechtl HFR. *Prechtl's assessment of general movements: a diagnostic tool for the functional assessment of the young nervous system.* Ment Retard Dev Disabil Res Rev 2005; 11:61-67

[2] Cerebral Palsy, introduction, NHS Choises, United Kingdom, retrieved 24.03.2015, http://www.nhs.uk/conditions/Cerebral-palsy/Pages/Introduction.aspx

[3] Zamore M, *Så utvecklas din nyfödda bebis*, publiced 2010-02-09, retrieved from the homepage "Vi föräldrar" 2015-03-26, http://www.viforaldrar.se/Forsta-aret/sa-utvecklas-din-nyfodda-b

[4] General movements trust, learning about Prechtl's Method on the qualitative assessment of general movements. http://www.general-movements-trust.info/5/home

[5] Adde L, Helbostad J.L, Refsum Jensenius A, Taraldsen G, Støen R, *Using computer-based video analysis in the study of fidgety movements.* Early Human Development 85 (2009-05-01), s.541-547.

[6] David G. Lowe. 1999. *Object Recognition from Local Scale-Invariant Features.* In *In International Conference on Computer Vision*, Corfu, Greece, pp. 1150-1157. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=790410

[7] David G. Lowe. 2004. *Distinctive Image Features from Scale-Invariant Keypoints.* http://www.cs.berkeley.edu/~malik/cs294/lowe-ijcv04.pdf

[8] Mikolajczyk, K. 2002. *Detection of local features invariant to affine transformations*, Ph.D. thesis, Institut National Polytechnique de Grenoble, France

[9] Camera Calibration Toolbox for Matlab, written by ean-Yves Bouguet, collected April 2015, http://www.vision.caltech.edu/bouguetj/calib_doc/

[10] Z.Zhang. 1998. *Determining the Epipolar Geometry and its Uncertainty:A Review.* http://research.microsoft.com/~zhang/Papers/IJCV-Review.pdf

[11] Zhang Z., Xu G.*Epipolar Geometry in Stereo, Motion and Object Recognition: A Unified Approach*, Doordrecht, The Netherlands:Kluwer Academic Publishers,

1996

[12] Hartley R., Zisserman A.*Multiple View Geometry in Computer Vision*,Cambridge University Press, 2004, chapter 8

[13] *Preterm Birth*, World Health Organization. Updated November 2014, retrieved May 11, 2015, `http://www.who.int/mediacentre/factsheets/fs363/en/`

[14] *Three-Dimensional Computer Vision: a Geometric Viewpoint*, MIT Press. 1993

[15] *SIFT toolbox* `http://www.vlfeat.org/index.html`

[16] Original picture made by Arne Nordmann, Created may 2015, the yellow ball was added. CC BY-SA 3.0