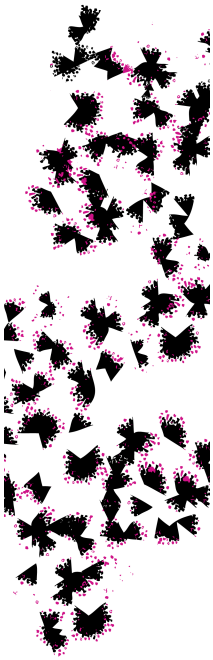

ROBOTICS AND MECHATRONICS

Pose estimation and motion set-point generation for robotics

N.E.W. (Nils) van Schooten

MSc Report

Committee:
Prof.dr.ir. S. Stramigioli
Dr.ir. T.J.A. de Vries
Dr.ir. F. van der Heijden
Dr.ir. J. van Dijk

February 2015

003RAM2015
Robotics and Mechatronics
EE-Math-CS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

UNIVERSITY OF TWENTE.

MIRA CTIT
BIOMEDICAL TECHNOLOGY AND TECHNICAL MEDICINE

Acknowledgements

During my master assignment I sometimes encountered challenging aspects. With the help of a number of people these challenges were overcome. This thesis would not have been possible without them. I want to thank Ferdi van der Heijden for guiding me in the field of computer vision and for reserving time, in his busy schedule. I want to express thanks to Tadele, I appreciate the effort he put into the project. I would like to thank Tim de Jager for helping me realise the implementation of the method. I thank Stefan and Jan Jaap for their input, and also other students and members of RaM. I further want to thank Niels, Bart, and Ingo for their help. My parents I want to thank for supporting me throughout my master assignment. Finally I want to thank my supervisor Theo de Vries for his positive input, it is greatly appreciated.

Abstract

Through developments in different disciplines, robotic applications become more autonomous. For the Bobbie robot, this development takes shape in the task of autonomously manipulating objects with the Philips Robotic Arm (PRA). For this task it is required that the controller of the PRA receives an input for the pose of the end-effector. The input is generated with the motion set-point generator. The end-effector of the robotic arm is guided in an appropriate manner to the desired destinations. A design and implementation of the motion set-point generator are presented. In the application of autonomous care robotics pose estimation is an essential task for the manipulation of objects. For the manipulation of objects, object recognition is required and the estimation of the pose of objects. Linemod is a template matching method with which objects can be recognized. Experiments were performed to verify if the method of Linemod is suited for the application of pose estimation for autonomous care robotics. It is concluded that it is plausible that pose estimation of objects can be performed by Linemod.

Contents

Chapter 1

Introduction

The group of Robotics and Mechatronics (RaM) of the University of Twente works on different robotic applications. One of the aims of the group is the development of robots that can function safely around humans. A project that focuses on this goal is project Bobbie. On the website of the Bobbie project[1] the goals of project are formulated: "The overall goal of the Bobbie project is to bootstrap a Dutch industry for personal robots for the elderly care [...]. This project will result in new methods to design a robot system, using standardized architectures, which can safely work in a care situation. As a proof of these methods, a specific realization in the form of a safely working prototype will be shown as an end result. These types of personal robots are expected to become a market with a similar influence as the personal computer market. Comparing the two markets, we are now in the era before the design of the IBM standard PC architecture. There are very few personal robotic systems on the market, they are expensive, and components are not interchangeable. Identical to how the IBM architecture revolutionized the PC market, the creation of design standards for personal robots will open up the great potential of the personal robotics market, and the Dutch Industry can play a key role". For this purpose a robot is built in the lab of RaM. The Philips Robotic Arm (PRA) is attached to the body of the Bobbie robot. This robotic arm has approximately the same dimensions as the human arm. Like humans, the arm has seven degrees of freedom. In figure 1.1 an image of the Bobbie robot is depicted. Other projects that were implemented in the Bobbie robot include head which consist of a neck with which the robot can aim its head. With this the perception of the robot becomes more versatile. Other applications were salience controlled eyes [13] and a mobile platform which can navigate autonomously through a floor plan with an application of SLAM. Robotic systems are application of different disciplines, in this project a goal is to provide a coupling between two disciplines namely path planning and pose estimation of objects. The

[1] www.bobbierobotics


Figure 1.1: The Philips robotic arm (left) and the platform and body of the Bobbie robot (right)

coupling of these two disciplines is needed to allow for a system that is (more) autonomous. The goal for this project is to provide a design that estimates a pose of objects and to create a path on basis of this pose. To clarify the goals of the project are listed underneath:

1. Recognise an object
2. Estimate the position and orientation of objects using prior knowledge of that object (i.e. there is a model of the object)
3. Plan paths to grab an arbitrary object.
4. Move the object to a desired position following an acceptable motion.

The first goal is not part of this assignment, however it is in many cases important. The first two goal gives a very specific separation between pose estimation and object recognition. This is further elaborated in the chapter about pose estimation. The organization of this report is as follows. It starts with background information in chapter 2. The purpose of this is to inform readers who are unfamiliar with the subject, with some additional information so that the other parts of the report can be better understood. In Chapter 3. starts with the architecture of the vision system of the robot is. An analyses is made about different ways of estimating poses of objects based on a method as described by Hinterstoisser et all. [1]. In chapter 4 a design of the motion set-point generator is presented. The set-point generator provides a homogeneous matrix to the impedance controller of the PRA. The thesis ends with chapter 6. Conclusions.

Chapter 2

Technical Background

In this chapter a number of subjects are presented. These subjects are used in this thesis. A short explanation is given for purpose of clarity; in [2] a more elaborate description is given.

2.1 Frame of reference

Frames of reference are coordinate systems. These frames express a certain pose (position and orientation) with respect to a global reference frame. Frames can be used to express the pose of an object. The standard basis for a three-dimensional Euclidean space is the set of unit vectors that point in the direction of the axes of a Cartesian coordinate system. The bases for the global reference frame are:

$$e_x = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \qquad e_y = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \qquad e_z = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{2.1}$$

This standard basis set can be written in a matrix:

$$\begin{pmatrix} e_x & e_y & e_z \end{pmatrix} = \left(\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \quad \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \quad \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \right) \tag{2.2}$$

The position of the origin of the global reference frame is:

$$p_O = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \tag{2.3}$$

Other frames can be defined in the coordinates of the global reference frame. In the figure 2.1 below 3 frames are depicted.

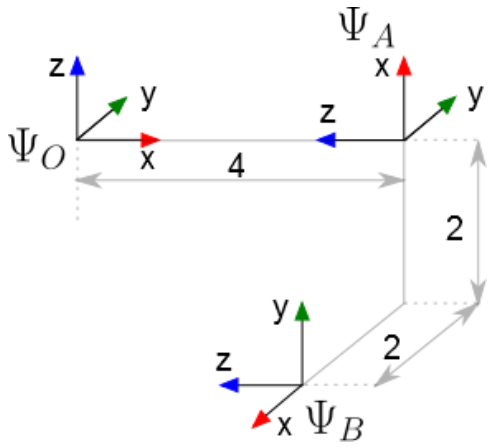

Figure 2.1: Frames of reference. A global reference frame Ψ_O , reference frame A; Ψ_A and reference frame OB; Ψ_B

In the figure, three reference frames are depicted namely Ψ_A, Ψ_B and Ψ_O which is the global reference frame. The basis set and position of Ψ_A and Ψ_B are given in coordinates of Ψ_O in respectively equation 2.4 and 2.5.

$$\begin{pmatrix} b_x & b_y & b_z \end{pmatrix} = \left(\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \quad \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \quad \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix} \right) \qquad p_O = \begin{pmatrix} 4 \\ 0 \\ 0 \end{pmatrix} \tag{2.4}$$

$$\begin{pmatrix} b_x & b_y & b_z \end{pmatrix} = \left(\begin{pmatrix} 0 \\ -1 \\ 0 \end{pmatrix} \quad \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \quad \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix} \right) \qquad p_O = \begin{pmatrix} 4 \\ -2 \\ -2 \end{pmatrix} \tag{2.5}$$

This is a simple example in which the bases of the frames stay aligned to the global reference frame. However, any orientation can be expressed with the basis set. In the definition of frames an option can be made for left handed and right handed frames. In this thesis a choice was made for a right handed reference frame. In figure 2.2 the difference between the left handed and right handed system is illustrated.

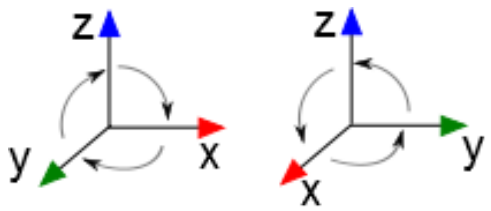

Figure 2.2: Left handed coordinate system (left) and a right handed coordinate system (right)

In the figure the left handed and right handed axes of the frames have colored arrows to indicate which is left handed and right handed. The arrow of the

x-axis is red, for the y-axis is green and z is blue. For a right handed system the arrows should read rgb or xyz. If the frames are defined and a choice for either a left handed or a right handed system has been made, the orientation of the frame is defined if two of the three bases are known. The third basis can than be calculated. This property is used in the thesis.

2.2 Transformation of coordinates

Reference frames can express the position and orientation of objects with respect to a global reference frame. The origin of the frame can express the position of the object. The orientation of the frame can be expressed by the basis set of the frame. To express frames with respect to each other homogeneous matrices can be used. A homogeneous matrix can be expressed by a 3D Cartesian coordinate and a rotation matrix. The rotation matrices express the orientation of a frame with respect to another frame. In this section a description of rotation matrices and homogeneous matrices follow, the example illustrated in figure 2.1 is used for this purpose.

2.2.1 Rotation matrix.

Change of orientation of objects by rotating the object from one frame to another frame can be expressed in a rotation matrix. The notation of the rotation matrix follows:

$$R_b^{a,c} \tag{2.6}$$

The letters (a,b,c) represent frames of reference. The rotation matrix expresses the orientation of frame a (Ψ_a), with respect to frame b (Ψ_b) expressed in coordinates of frame c (Ψ_c). Because in most applications all frames are expressed in the global reference frame, the notation of frame in which it is expressed is not required. The notation of the rotation is expressed by:

$$R_b^a \tag{2.7}$$

An interesting property of the rotation matrix is that the orientation of Ψ_b with respect to Ψ_a is the inverse of the orientation of Ψ_a with respect to Ψ_b. This is expressed in equation 2.8.

$$R_a^{b\,-1} = R_b^a \tag{2.8}$$

Another property of the rotation matrices is that they can be multiplied with each other. Calculations of orientation of one frame with respect to another frame can be calculated by multiplication of rotation matrices. An example is equation 2.9

$$R_a^b = R_c^b * R_a^c \tag{2.9}$$

This can be expressed as equation 2.10

$$R_a^b = R_c^b * R_c^{a-1} \tag{2.10}$$

The interpretation of this is that by rotating from frame Ψ_a to frame Ψ_b and then rotating from frame Ψ_b to frame Ψ_c is equal to a rotation from Ψ_a to Ψ_c. If this is applied to the example of the previous section as depicted in 2.1 The rotation matrix R_A^B can be calculated calculated:

$$R_A^B = R_O^B * R_A^O \tag{2.11}$$

R_A^O can be calculated with the inverse of R_O^A:

$$R_A^B = R_O^B * R_O^{A^{-1}} \tag{2.12}$$

For the calculation the rotation matrices of R_O^B and R_O^A are required. Because the orientation of Ψ_B is expressed with respect to the origin, R_O^B can by be found by taking the basis set of Ψ_B as expressed in equation 2.5. The equation follows:

$$R_O^B = \begin{pmatrix} b_x & b_y & b_z \end{pmatrix} = \left(\begin{pmatrix} 0 \\ -1 \\ 0 \end{pmatrix} \quad \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \quad \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix} \right) = \begin{pmatrix} 0 & 0 & -1 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \tag{2.13}$$

The values for R_O^A can be obtained in similar fashion, this is expressed in equation 2.13:

$$R_O^A = \begin{pmatrix} b_x & b_y & b_z \end{pmatrix} = \left(\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \quad \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \quad \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix} \right) = \begin{pmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \tag{2.14}$$

With this the rotation matrix R_A^B can be calculated

$$R_A^B = R_O^B * R_O^{A^{-1}} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}^{-1} = \begin{pmatrix} 0 & 0 & -1 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \tag{2.15}$$

To verify that this is the rotation matrix the following calculation can be performed.

$$R_O^B = R_A^B * R_O^A \tag{2.16}$$

The orientation of Ψ_B with respect to Ψ_O (expressed in R_O^B) can be calculated with the orientation of Ψ_B (expressed in R_O^A) and the change of orientation between Ψ_A and Ψ_B (expressed in R_A^B).

2.2.2 Homogeneous matrix.

The position of a frame with respect to another frame expressed in a global reference frame can be expressed in a similar fashion as the rotation matrix. The expression is:

$$p_b^{a,c} \tag{2.17}$$

In which p expresses the position of frame a, with respect to frame b in the global coordinate of frame c. The homogeneous matrix is expresses both the orientation and the position of a frame with respect to another frame. The homogeneous matrix can be expressed as rotation matrix and a Cartesian coordinate. This is expressed in the following equation:

$$H_c^{a,b} = \begin{pmatrix} R_b^{a,c} & p_b^{a,c} \\ 0 \quad 0 \quad 0 & 1 \end{pmatrix} \tag{2.18}$$

The homogeneous matrix expresses the pose of a reference frame with respect to another reference frame expressed in a global reference frame. As for the rotation matrices, often it is not necessary to indicate the global reference frame of the homogeneous matrix. For the homogeneous matrices the similar transformation can be performed as was presented for the rotation matrices. The equation that expresses the calculation of the frames in figure 2.1 follow:

$$H_A^B = H_O^B * H_A^O \tag{2.19}$$

$$H_A^B = H_O^B * H_O^{A^{-1}} \tag{2.20}$$

The change of pose between Ψ_A and Ψ_B can be expressed as a change of pose from Ψ_A to Ψ_O and from Ψ_O to Ψ_B . To obtain H_A^O the inverse of H_O^A is calculated. For the rotation matrices this was calculated for the example from figure 2.1. Similar calculation can be performed for the homogeneous matrix as for the rotation matrix. However, this is not further elaborated here.

2.3 Axis-angle representation and Rodriguez rotation formula

The rotations described by rotation matrices can be achieved by rotating around an axis with a certain angle. With the axis-angle representation it is possible to calculate the axis around which is rotated, and the angle with which is rotated from a rotation matrix. With the Rodrigues rotation formula it is possible to calculate the rotation matrix from an axis and angle with which is rotated around the axis. The Rodrigues rotation formula and the axis-angle representation can transform the representation of a rotation respectively from the axis and angle to a rotation matrix and vice versa.

This is illustrated in the figure 2.3.The methods are presented in the next subsections.

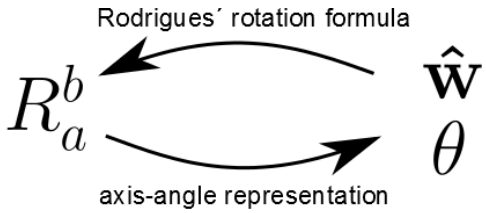

Figure 2.3: The relation of the axis and angle with the rotation matrix. The Rodrigues' rotation formula and the axis-angle representation are ways to transform the two representations into each other.

The methods are presented in the next subsections.

2.3.1 Rodrigues' rotation formula

With the Rodrigues' rotation formula it is possible to calculate the rotation matrix from a rotation axis and angle with which it is rotated. The unit vector, of the rotation axis $\mathbf{w}$ is $\hat{\mathbf{w}}$, which is used in the calculation. The Rodrigues' rotation is expressed in equation 3.9

$$R_a^b = I_{3\times 3} + (\sin\theta)\tilde{\hat{\mathbf{w}}} + (1-\cos\theta)\tilde{\hat{\mathbf{w}}}^2 \tag{2.21}$$

The angle with which is rotated is θ. For the calculation of the rotation matrix, the skew symmetric of $\hat{\mathbf{w}}$ is required. The skew symmetric matrix has a one to one relationship with $\hat{\mathbf{w}}$ and is the unit vector of the rotation, this is notated as $\tilde{\hat{\mathbf{w}}}$. Skew symmetric matrice have the following property: $-\hat{\mathbf{w}} = \hat{\mathbf{w}}^T$. If the elements of the unit vector are written as $\hat{\mathbf{w}} = (w_1, w_2, w_3)^T$, the skew symmetric matrix $\tilde{\hat{\mathbf{w}}}$ can be written as:

$$\tilde{\hat{\mathbf{w}}} = \begin{pmatrix} 0 & -w_3 & w_2 \\ w_3 & 0 & -w_1 \\ -w_2 & w_1 & 0 \end{pmatrix} \tag{2.22}$$

2.3.2 Axis-angle representation

From the rotation matrix both the angle θ and the unit vector $\hat{\mathbf{w}}$ can be calculated. The rotation through the axis-angle representation is depicted in figure 2.4.

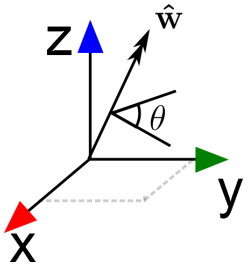

Figure 2.4: Axis-angle representation of rotation.

For the calculation the elements of the rotation matrix are used. The elements of the rotation matrix are expressed in equation 2.23.

$$R_a^b = \begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{pmatrix} \tag{2.23}$$

The parameters are calculated by using the axis-angle representation, as described before. The angle θ is calculated by:

$$\theta = \arccos(\frac{1}{2}(R_{11} + R_{22} + R_{33} - 1)) \tag{2.24}$$

The unit rotation vector ($\hat{\mathbf{w}}$) is calculated by:

$$\hat{\mathbf{w}} = \frac{1}{2\sin\theta} \begin{pmatrix} R_{32} - R_{23} \\ R_{13} - R_{31} \\ R_{21} - R_{12} \end{pmatrix} \tag{2.25}$$

For the calculation one rotation matrix is required as input. With this, the axis around which is rotated ($\hat{\mathbf{w}}$) and the traveled angle(θ) are calculated.

2.4 Twist

A twist is a vector that expresses velocity in six directions. Twists are used to describe the motion of rigid bodies. The twist consist of a rotational velocity vector and a linear velocity vector. The rotation velocity expresses the rotational velocity around 3 axes of a reference frame. The linear velocity expresses the velocity of in the three directions of the reference frame. This is expressed in equation 2.26

$$\mathbf{T} = \begin{pmatrix} \mathbf{w} \\ \\ \mathbf{v} \end{pmatrix} = \begin{pmatrix} w_x \\ w_y \\ w_z \\ v_x \\ v_y \\ v_z \end{pmatrix} \tag{2.26}$$

In this $\mathbf{w}$ is the vector which consists of the rotational velocity in three directions, $\mathbf{v}$ is the vector which consists of the linear velocity in three directions. The twist can be written as a unit twist multiplied with a scalar. In this the normalization can be applied to the rotational velocity:

$$\mathbf{T} = \begin{pmatrix} \mathbf{w} \\ \\ \mathbf{v} \end{pmatrix} = w * \hat{\mathbf{T}} = w * \begin{pmatrix} \hat{\mathbf{w}} \\ \\ \bullet \end{pmatrix} \tag{2.27}$$

In this case the linear velocity ($\mathbf{v}$) is dependent on the rotation vector, that is why the linear velocity is not represented the equation. However, there is a linear velocity, this is indicated as a dot ($\bullet$). If the twist represents a pure linear motion the linear velocity is normalised:

$$\mathbf{T} = v * \hat{\mathbf{T}} = v * \begin{pmatrix} 0 \\ 0 \\ 0 \\ \\ \hat{\mathbf{v}} \end{pmatrix} \tag{2.28}$$

In the next section the twist of a screw motion is expressed.

2.5 Chasles theorem

Chasles theorem states that the most general rigid body displacement can be produced by a translation along an axis and by a rotation about that axis. This motion can be interpreted as a screw. The screw motion can be described by a number of parameters.

The pitch (λ) is defined as the ratio between the rotation and the translation along the rotation axis.

The vector $\mathbf{r}$ is a vector from the initial frame of the motion, which covers the shortest distance to the rotation axis. The direction of rotation is expressed by the unit vector $\hat{\mathbf{w}}$. By obtaining the correct values for the parameters the screw-motion can describe a path between two arbitrary frames. To clarify in the figure below a screw is depicted with the parameters.

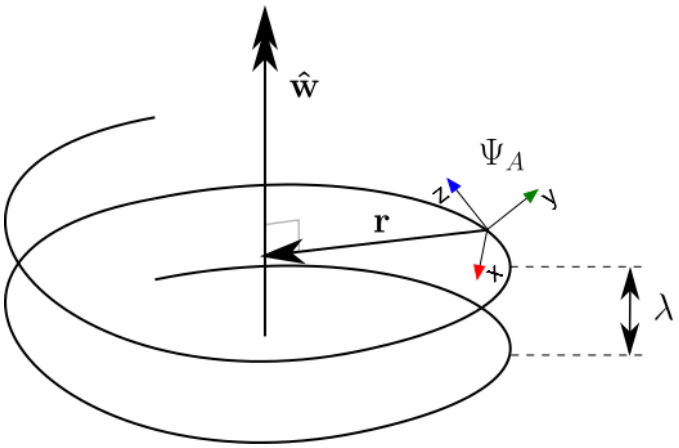

Figure 2.5: Image of a screw with the parameters λ, r and $\hat{\mathbf{w}}$

The screw-motion can be expressed by a twist. The expression for the twist is:

$$\mathbf{T} = \begin{pmatrix} \mathbf{w} \\ \\ \mathbf{r} \wedge \mathbf{w} + \lambda \mathbf{w} \end{pmatrix} = w * \hat{\mathbf{T}} = w * \begin{pmatrix} \hat{\mathbf{w}} \\ \\ \mathbf{r} \wedge \hat{\mathbf{w}} + \lambda \hat{\mathbf{w}} \end{pmatrix} \tag{2.29}$$

The twist can be expressed in terms of a unit twist, this is shown in the two right members of equation 2.29. In the calculation of the velocity vector a wedge product ($\wedge$) is used. The wedge product is a generalization of a cross product. With screw theory the motion from any pose to any other pose can be expressed. If a straight line is described $\mathbf{r}$ in equation 2.5 becomes infinitely large. For practical applications the twist is written as a linear velocity. In case of a linear motion the twist is expressed as:

$$\mathbf{T} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \\ \mathbf{v} \end{pmatrix} = v * \hat{\mathbf{T}} = v * \begin{pmatrix} 0 \\ 0 \\ 0 \\ \\ \hat{\mathbf{v}} \end{pmatrix} \tag{2.30}$$

Chapter 3

Pose estimation

3.1 Introduction

Robots or Robotic elements are designed with the intention of executing tasks that humans are not able to accomplish, have no time for, or do not want to do. Due to developments in computer vision algorithms, camera technologies and the increasing computer speed and memory capacities, robots are becoming more autonomous. An important subject for autonomous robots is object recognition. Object recognition is a technology in which a computer analyses an input from the environment through an algorithm with the goal to classify or identify objects. The input of sensors is compared to data about the objects of which the the computer has a model. The input for these algorithms can come from a wide set of different sensors or combination of sensors. A common sensor which is used for this task is a camera For the Bobbie robot it is important to recognise objects, because the task of the robot is to manipulate objects autonomously. It is desired that the robot can estimate the pose of objects. To navigate the robotic arm, it is important to know where objects are positioned and how to approach objects in order to manipulate the objects properly.

3.1.1 Context

In the field of computer vision different terms are used that can have an ambiguous meaning. To explain what is understood by object recognition different meanings of the words classification and identification are described in table (3.1.1).
The multiple interpretations make the described terms ambiguous. Moreover the description of identification II. and classification I. are very similar. In the description of classification II. and III. a subtle difference is made. In the examples the color can be classified in both ways. For object recognition in this thesis an implementation is used that recognises objects by identifying them in the narrow sense. To manipulate objects, the orientation and

Term	Description
I. Identification (Narrow sense)	To establish as being a particular person or object. An example is face recognition.
II. Identification (Broad sense)	To establish to what group a person or object belongs.
I. Classification	To establish to what group an object belongs. Groups are defined based on intrinsic properties. For example fruits; apple, pear, oranges.
II. Classification	To establish to what group an object belongs. Groups are defined based on properties that are defined by the user. The color of an object is green, red or blue, there is no highest or lowest color. The classification is performed with nominal or categorical variables.
III. Classification	To establish to what group an object belongs by using an ordinal variable. The object's color can be classified in this way, by associating colors with their electromagnetic spectrum. Red has a relative low frequency, green has a frequency in the middle of the visible light and violet color has the highest possible frequency.

position of the objects with respect to the robot are required. Pose estimation is the estimation of the position and orientation of an object in an 3D Euclidean space with respect to the camera device, expressed in coordinates of a (global) reference frame. The goal of object recognition and pose estimation differ. However for the Bobbie robot it is necessary to recognise objects before the pose can be estimated. In a number of methods the task of object recognition and pose estimation is a joint task. In this chapter such a method is described.

3.1.2 Problem Statement.

For autonomous robots the recognition of objects and pose estimation of those objects is necessary for more flexible application of robots and executing more complicated tasks autonomously. The problem statement can be formulated as finding a method that is suitable for an autonomous robots for the task of recognising objects and estimating their poses. The poses can be expressed in rotation matrices for the rotation, and 3-dimensional coordinates for the position. An assessment of accuracy is required to indicate if the method is suitable for the task of pose estimation of rigid objects.

3.1.3 Outline of the chapter.

This chapter starts with the architecture of the vision system of the robot, in section 3.2. In this section, different classes of methods are considered and

arguments are given for the chosen architecture. In section 3.3 the chosen method is described. The chosen method is Linemod, this is a template matching method. Section 3.4 explains how template matching can be used for the estimation of pose of objects. In this section an estimation of the accuracy of the method is made. An experiment is performed to asses the suitability of the method to estimate the pose, this is described in section 3.5. Based on the results of the experiments, the suitability of the method for pose estimation in robotics is discussed. The chapter ends with Conclusions and Recommendations.

3.2 Architecture

In this section the architecture of the vision system of the robot is described. Three aspects of the architecture are considered: the method of recognition and pose estimation, the camera that is used, and the positioning of the camera on the robot. The choices for the different aspects are interdependent. However, each aspect is considered separately, for reasons of clarity. These choices for the aspects determine the architecture of the robot.

3.2.1 Method of pose estimation

In the introduction, the difference between pose estimation and object recognition was mentioned. Some methods encompass the joint task of object recognition and pose estimation. The aim of care robots is to grab objects. For this purpose the recognition of objects and the estimation of pose of objects is required. For an efficient care robot it is desired to execute tasks in a limited amount of time. A method is desired with which the robot can recognise objects and estimate the pose of the object in a short amount of time. While both the tasks of object recognition and pose estimation are important for autonomous robots, in this subsection pose estimation is considered.

Overview of available methods

For pose estimation, a number of classes of methods is available. In this section a short overview is given as described by [6]. One separation of methods is made between pose estimation of that of rigid objects and non-rigid objects. There are a wide range of different methods for pose estimation of non-rigid objects an example of which is active contours. Both rigid and non-rigid objects occur for the tasks that are described for the Bobbie robot. Humans are in the materialistic sense, non-rigid objects. However, rigid objects are the most common class of objects that need to be manipulated. For this reason and the reason of simplicity, the non-rigid objects are not

considered. The rigid pose estimation methods are divided in a number of classes, per class a short overview is given of what the class entails.

Pose estimation methods based on explicit feature matching

This class of methods is based on image features that do not significantly change while changing the viewpoint of the camera. Methods such as edge-based approaches are included in this class of methods.

Appearance based pose estimation

In [6] this class is defined as "A different class of methods consist of the appearance-based approach, which directly compare the observed image with the appearance of the object at different poses with explicitly establishing correspondence between model features and parts of the image.". This is a statistical analysis of features, a set of features is associated with a pose. This can be achieved with a statistical analysis like Principle Component Analyses.

Template-based pose estimation

With template matching, it is possible to associate a certain template with a certain pose of the camera with respect to the object. The template consist of features. Each template is associated with a pose of the camera with respect to the object.

Point cloud segmentation approaches

Pose estimation through point cloud segmentation is a method in which an observed point cloud is fitted to a registered point cloud. This is achieved by minimizing a criterion. The rotation and translation required to fit the observed depth image to the point cloud are the orientation and position of the object with respect to a standard orientation and position of the registered object. With this the orientation and position of the observed point cloud can be calculated, and as such the pose of the object.

Selected class and method: Template matching

The selected method is a template matching method named Linemod. The method that is chosen can be used for estimating poses and object recognition. The choice of the method is explained. The Linemod method as described in [1] is a method in which a collection of templates represent a model of the object. In Linemod, both 3D and 2D features are used to create templates. The usage of these different modalities results in a better recognition of the object. An argument to use the method is that it is fast.

The algorithm can detect an object with a frame rate of 10 fps. The frame rate can be achieved with a set of 25 objects of which templates are created. It is reported that the detection with this method is also robust. Learning new templates is easy. The Linemod method works well with non-textured objects. Non-textured objects are a class of objects that are hard to recognise with other methods. This method is interesting for that reason, because many household objects often do not have much texture. Input required for Linemod is a color image (RGB) and a point cloud. Linemod is a template matching method in which an object is recognised if the templates that belong to the object have the highest match when comparing it to the input. The image location of the recognised object is given as an output.

3.2.2 Camera

A method is chosen that requires a 3D point cloud as input. While this limits the choice of camera devices, there is still a variety of camera's available for this purpose. Here a number camera's are compared to each other, of which one is chosen. For every camera a short description is given. In [11] a comparison of camera's is made, this is used for the comparison in this section. A choice is made on the basis of these properties.

Leap motion

The leap motion is a camera that senses the depth of its direct environment. The camera uses infrared and has a high accuracy. However, the range of the Leap motion is a box with the dimension of $(30 \times 30 \times 60)$ cm^3. This limits its capabilities for the application of a household robot, however, for close range high precision robotic manipulation this device is an interesting option. For close range sensing, the leap motion could be used to make a more precise point cloud segmentation of an object and a high precision pose estimation. This could be interesting for the Bobbie robot, but it is probably not necessary to have such high accuracy for the regular tasks the Bobbie robot is intended for.

PMD CamCube

The PMD CamCube is a time of flight camera. This type of camera produces a depth image by measuring the time that the light travels. The PMD Camtube has a good precision. However, the camera's resolution is low. It does not provide color channels, and is expensive in comparison to other cameras. In [11], it is reported that in the future, it is expected that time of flight camera's improve, and that these camara's are interesting because they are relatively easy to calibrate.

PointGrey Bumblebee XB3

The Bumblebee is a high precision camera with point cloud capabilities that features stereo vision. The Bumblebee works according to the principles of triangulation. An advantage of the Bumblebee is that the camera has a high resolution. In principle there are no limit's to the maximum range of triangulation. However, in [11] it is mention that range is limited.

Microsoft Kinect

The Kinect is a point cloud camera, which produces a depth image and a color map. The Kinect has a project with which a pattern (structured light) is projected on the environment. With which it is possible to reconstruct a map of the depth of the environment. An advantage of the Kinect is that it produces a point cloud and color map of the whole scene. The other camera's produce a point cloud of a region of interest. For the depth perception, the Kinect is equipped with an infrared projector and the infrared camera, which are placed next to each other. The images are aligned by means of a transformation. The Kinect is calibrated during production. The camera matrix K is known as well as the translation between the cameras of the Kinect[1].

Choice of camera

For the purpose of comparing the camera's some properties of the camera's have been listed in a table, see below. The minimal range of the PMD

camera	accuracy [cm]	range [m]	frame rate [fps]	resolution [px]
Kinect	0.6	0.5-3 or 0.8-4.5	30	640 x 480
Bumblebee	0.5	0.5 - 4.5	15	1280 x 960
PMD	0.3	? - 7	25	200 x 200
Leap motion	0.00001	0.01 - 0.3	200	?

Table 3.1: Properties of depth camera's

Camtube and the resolution of the Leap motion are not known. However for the Leap motion it should be possible to obtain point cloud information coupled to color images. The choice is made for the Kinect. The Kinect has a reasonable range, and an accuracy that is acceptable, in comparison to the other camera's. Advantages of the Kinect are, that it is fast, cheap, and it delivers a depth map of the whole image frame. This is in contrast to other camera's that only have a region of interest in which a depth image is

[1] A more elaborate explanation can be found at, for example http://pille.iwr.uni-heidelberg.de/ kinect01/doc/index.html

available. Furthermore the for the Kinect no calibration is required and the depth values are calculated on board.

3.2.3 Placement of the camera

Robots have moving parts. While for humans the position of the eyes fixed, choices for robots can be different. The choice is divided in two general options:

- Hand to eye. In this case the camera device is on a fixed position with respect to the moving part of the robot.
- Eye-in-hand. The camera is fixed to the end-effector of the robotic arm. This can result in a more close (and accurate) observation of the object which is the aim of the end-effector

The choice for a point cloud camera with a large minimal range, like the Kinect has the effect that the choice for the eye-in-hand option is not an advantage. This is because the work area of the arm is a half hemisphere with a radius of 65 cm, while the minimal range of the Kinect is 50 cm. Because of this the choice is made for a Hand to eye setup. The minimal range of the Kinect is still a big distance in comparison with the work area of the robotic arm. The distance of the Kinect to the work area can be increased by placing the camera further away from the robotic arm. For example on a plateau, or in case of the Bobbie robot on top of a neck. If close range devices are used, like the leap motion, the Eye-in-hand setup can be an interesting option, because it allows for accurate actuation. A hybrid system is an option, where one camera determines globally the pose of the object. When the end-effector is close enough to the object the eye-in-hand configuration takes over.

3.3 Template matching: Linemod

In this section a short overview of the Linemod method is given. In [1, 19] the method is described in detail. The global principles of template matching method Linemod are explained in this section. In order to understand the application of template matching for pose estimation, it is helpful to understand the properties of the method of Linemod. During the learning phase models of the object are created. The models of the objects consist of registered templates. A model of an object consists of a set registered templates. During the detection phase the registered templates of the models of objects are compared to an observed templates. The comparison is achieved by calculating the similarity measure between observed templates and registered templates. The highest match of an observed templates with

a registered templates is used to identify the observed templates with the registered templates. If the similarity measure is above a certain threshold the templates are recognised and as such the object is recognised. The similarity measure might also be used for the estimation of the pose of objects. In this section a description of the learning phase and the detection phase is given. The expression for the similarity measure is given in the detection phase. The similarity measure is used to determine which registered templates are observed. Experiments were performed in which the similarity measure is recorded.

3.3.1 Learning phase

Templates consist of features on certain locations in an image. The templates are obtained by processing input from the camera. For the Linemod method, a RGBD (acronym for Red Green Blue Depth) input is required. The colors (RGB) represent the regular color channels of a camera. The measurement of depth (D) results in a depth image which has the same resolution as the color channels. From the depth image a point cloud is created, which is used as an input. Several steps are performed to extract features from the input and create templates. These steps are, feature extraction, filtering, quantization and pooling. In the following list the steps are explained:

- Feature extraction. There are different kinds of features in Linemod, these are called modalities. The modalities of Linemod are color gradients and normals. The color gradients are obtained from the color channels, the color gradient is approximated with a Sobel filter. For the extraction of normals, the point cloud of the Kinect is used. In the implementation as described by [19], the normals are calculated by taking the cross product. The cross product is taken vectors that are spanned between the selected point and the two points in closest proximity of the selected point. In [1] the the normal is calculated with the Taylor series. In the implementation that is used the similarity measure is calculated with an additional modality. The modality is called color names. Color names is based on the theory that humans give the color of the objects a name, for example yellow. The colors names are calculated from RGB values. The extracted features for the modalities color gradients and normals have a pixel location, a direction and the feature of the color gradient modality has a size.

- Filtering. After the feature extraction, the modalities are filtered according to a number of criteria. The size of the color gradient is such a criteria. The normals are grouped and averaged and the normals near to the edge (which is detected from the color gradient) are ignored, because the depth at the edge are not reliable for the measurements of

depth. The Sobel filter gives features a certain magnitude and a direction. After filtering steps the dominant features are selected. While the magnitude of the unfiltered features is used to obtain the dominant features, the direction of these features is used to compare the features of the templates. The modalities used in Linemod are scale-invariant. The size of the object on the image does not affect the appearance of the template, in principle. In [1] it is reported that in practice the recognition of objects works well between the scale of 1.0 and 2.0.

- Quantization. The quantization of the normal and the color gradient is depicted in respectively in figure 3.2 and 3.1

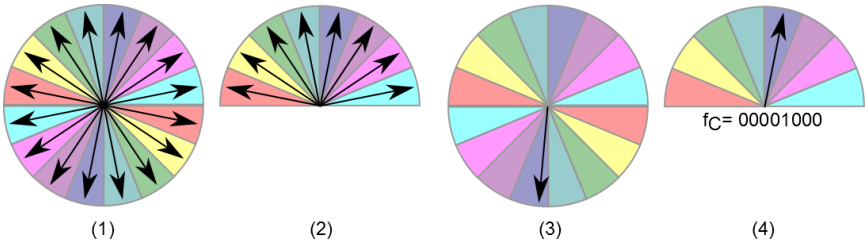

Figure 3.1: Quantization and rectification of the color gradient

The color gradient quantized in 16 parts. However, half of the directions is rectified, this is depicted in figure 3.1 (4)

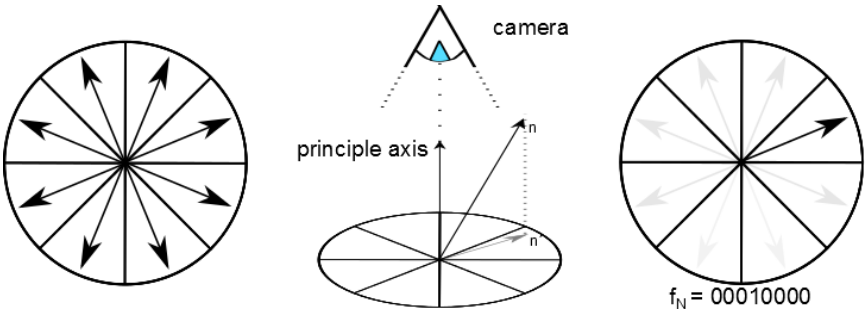

Figure 3.2: Quantization and rectification of the color gradient

In figure 3.2 (2) It can be seen how a normal is mapped onto a plane perpendicular to the camera's principle axis. With the mapping of the normal, it can be quantified. The normal is quantified in 8 directions, without being rectified. Both modalities have features that can be described by 8 bit.

- Pooling. Templates are created as an image such as produced in the in the previous steps of preprocessing. The features are listed as well, a certain way of ordering the list is used, for the color gradients the

listing is performed by starting with the most dominant feature and than following the contour of the object. The members of the list points to a certain image location. By listing the features an arbitrary shape can be compared efficiently. A template is defined as an image $\mathfrak{O}$ with a certain list which is denoted as P. The notation of the template is:

$$T(\mathfrak{O}, P) \tag{3.1}$$

In this $\mathfrak{O}$ is the preprocessed image containing the dominant feature, P is the list of features organised according to some criteria.

During the detection phase these steps are performed to generate observed templates.

Object models and template sets.

The steps described above create templates for all modalities described in Linemod. In [19] the templates registered from different perspectives and with different modalities are organised into an object model. The object model is denoted as:

$$O = \begin{pmatrix} T_1^1 & \cdots & T_n^1 \\ \vdots & \ddots & \vdots \\ T_1^m & \cdots & T_n^m \end{pmatrix} \tag{3.2}$$

In which the T is a template, m is the number of modalities, in case of [1] that is two, in [19] the number of modalities is three. The number of templates is indicated by n. A template set is defined as a collection of templates, which consist of a number of templates, for each modality one. These templates are created from the same input image. The pose of the camera with respect to the object is the same for each template in the template set. The i^{th} template set is defined as:

$$T_i = \begin{pmatrix} T_i^m \\ \vdots \\ T_i^m \end{pmatrix} \tag{3.3}$$

Such that the Object model O can be written as:

$$O = \begin{pmatrix} T_1 \cdots T_i \cdots T_n \end{pmatrix} \tag{3.4}$$

Each registered template set can be associated with a pose of the camera with respect to object. A pose of a camera with respect to an object is used to calculate the orientation of the object with respect to the camera in pose estimation. This is further elaborated in the next section about pose estimation.

3.3.2 Detection phase

During the detection phase the output of the Kinect is preprocessed in a similar way as during the learning phase. After some additional steps, the observed template is compared to the registered templates with the similarity measure. In the previous section the magnitude of the color gradients is used to recognise the dominant features. For the detection of templates the orientations of features are compared. Because of this the color gradients are scale-invariant. The normals do not have a magnitude, only a direction. The direction of both the color gradient and normals are used as features to compare templates. With quantified features a fast comparison is possible. The similarity measure for the modalities presented in [1] are derived from the similarity measure as presented by Steger [9]:

$$\mathcal{E}_{Steger}(I, T, c) = \sum_{r \in P} |cos(ori(\mathfrak{O}, r) - ori(I, c + r))| \tag{3.5}$$

In equation 3.5 $\mathcal{E}_{Steger}$(I,T,c) is the similarity measure for the modalities of the quantified color gradient and normal. The orientation of features is indicated by $Ori(A, B)$. The meaning of $Ori(A, B)$ is the orientation of the feature at location B in image A. The orientation is expressed in radians. A is an image with features in it, with a certain direction. B is a list item which points to a location in image A. The features of the registered template (expressed in image $\mathfrak{O}$) and observed template (expressed in image I) have a certain orientation (angle) in radians. The difference between the orientations of I and $\mathfrak{O}$ is an angle, which can obtained by subtraction of the two orientations. The co-sinus of the angle is calculated. This calculation is performed for each place in list P. These results are added, this gives the similarity measure for an modality. The lists of image I and image $\mathfrak{O}$ can be shifted with respect to each other. By adjusting c the lists be compared at different places with respect to each other. In [1] an adaptation is made to Steger's similarity measure which results in a similarity measure for robust and efficient recognition of objects through template matching. The adaptations are not further addressed here. The method results in the similarity measure for the recognition of objects. The value for the similarity measure of the used algorithm typically had value between 400 and 650. The fact that the features are scale-invariant is used for pose estimation. Because the features are scale-invariant the distance between the camera and the object does not influence the detection of the template in principle. In the estimation of the pose of an object with respect to the camera, the quantization of features play a role. By quantization of features, the orientation of the template is quantized as well. In the next section a method of estimating pose trough template matching is described. The effect of quantization of features on accuracy of the pose estimation is addressed there.

3.4 Pose estimation with template matching

In the previous section, a description of the Linemod method was given. In [10] it is mentioned that Linemod can be used for estimating pose, by means of the Itterative Clossest Point method (ICP) Different versions of the ICP method exist, for example [18] and [17]. In ICP a point cloud model of an object is compared to a point cloud created from the input of the Kinect. With ICP it is possible to rotate and translate the observed point cloud to the pose of the registered point cloud. The ICP method consists of an optimization in which the point cloud converges to a local minimum error. For the ICP method an initial estimation of pose is required, which is sufficiently accurate. The accuracy of the initial point cloud is required, in order to ascertain that the ICP method converges the input to the global minimum. Template matching methods like Linemod can be used to acquire an initial pose estimation. In this section a way to calculate the pose of the object with respect to the camera is presented, using templates with scale-invariant features. The orientation of the object with respect to camera can be estimated from any distance in principle, because the modalities are scale-invariant. The method of pose estimation is divided in orientation estimation and position estimation. An estimation of the accuracy of the method is made for both the position estimation and the orientation estimation.

3.4.1 Method for estimating the orientation of the objects

In this section a description is given of how Linemod can be used for the calculation of the orientation of an object with respect to the camera. It is possible to calculate the pose (orientation and position) of a camera with respect to a calibration field. An example of calculating the pose of a 2D surface with respect to a camera is described in [12]. If an object is placed on a platform with a calibration field, the pose of the object with respect to the camera can be defined. During the registration of the templates, the pose of the object can be recorded and associated with a template set. If during the detection the observed template set is recognised as a registered template set, the associated orientation of camera with respect to the object is known. To represent the orientation of a template set, use can be made of spherical polar coordinates. In this representation, the orientation of the template with respect to object is expressed in of two angels. A certain point on a hemisphere can be associated with a certain pose of the camera with respect to the object. This can be represented in figure 3.3.

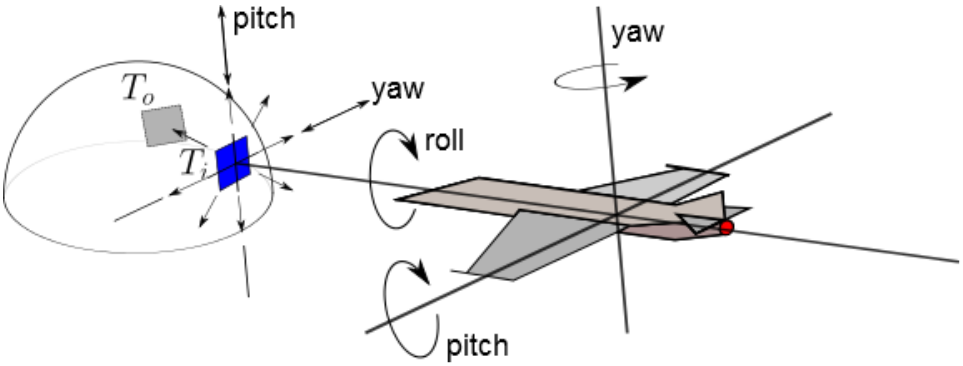

Figure 3.3: Change of orientation of the template expressed in roll, pitch and yaw.

In the figure above, the orientation of template set T_i is associated with a point on the hemisphere. During detection an observed template set is associate with an registered template set. If an observed template set is recognised as the correct registered template set, the orientation of template is known. However, if the orientation of the observed template set and the recognised registered template set differ, an error in the orientation estimation occurs. This type of error is expressed in figure 3.3 by the direction of the yaw and pitch. Because spherical polar coordinates express the orientation by two angles, one degree of freedom remains for orientation of the template. This degree of freedom is expressed by the roll. This can be seen in figure 3.3. In this subsection a number of expressions are formulated to calculate the orientation of the camera with respect to an object. Later in this section the accuracy of the orientation estimation using template matching is addressed. The template sets of the object are registered from different orientations, this can be represented by a hemisphere as was explained before. This is also expressed in figure 3.4. In the figure the lines from the cameras to the object represent the principle axes (also known as optical axes). The camera registers templates from different sides. The image plane of the template is depicted in the figure. The image plane is perpendicular to the principle axis. The principle axes in the figure are also perpendicular to the hemisphere. Part of the hemisphere can be associated with the registered template sets. A solid angle can express such a part of a hemisphere.

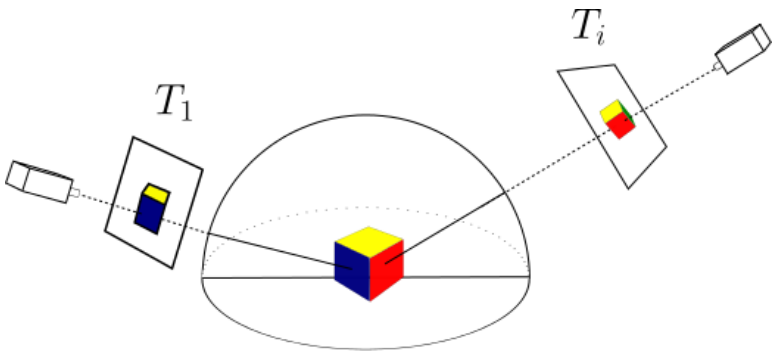

Figure 3.4: Object, template sets and cameras

In figure 3.5 three frames are shown. The frame of the object is defined as Ψ_O, the frame of the i^{th} registered template set(T_i) is defined as Ψ_{T_i}.The frame of the camera is defined as Ψ_C. During the learning phase, the frame of the camera and the frame of the template have the same orientation. During detection, the orientation of Ψ_{T_i} and Ψ_C can differ.

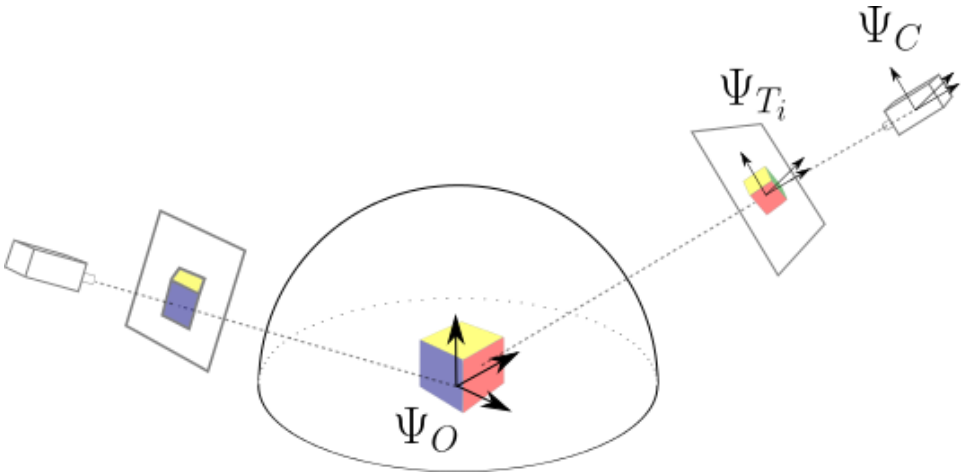

Figure 3.5: Frame of the camera(Ψ_C), frame the i^{th} template set(Ψ_{T_i}), and frame of the object (Ψ_O)

For the calculation of the object's orientation with respect to the orientation of the camera, different rotation matrices are defined. The rotation matrix R_C^O expresses the orientation of frame Ψ_O with respect to the orientation of Ψ_C. The rotation matrix $R_{T_i}^O$ expresses the orientation of frame Ψ_O with respect to the orientation of frame Ψ_{T_i}. The rotation matrix $R_C^{T_i}$ expresses the orientation of frame Ψ_{T_i} with respect to the orientation of frame Ψ_C The rotation matrix $R_{T_i}^O$ can be obtained by use of a calibration field, as described previously. The matrix $R_C^{T_i}$ is the rotation matrix expressing the orientation of the frame of the registered template set T_i with respect to the orientation

of the frame of the camera. The orientation of the object with respect to the camera can be expressed with the rotation matrix R_C^O. To calculate R_C^O, first some other calculation are performed. With template matching the orientation of the object with respect to the camera is expressed by R_C^O. However this is only true if the camera is at the same orientation with respect to the object as during the learning phase. In figure 3.6 the camera is in the same orientation as during the learning phase. If the camera, or the object is rotated along the principle axis the situation changes, this is represented in figure 3.7 and 3.8. The rotation along the principle axis changes the orientation of the object with respect to the camera. The calculation of R_C^O that are required are elaborated in the remainder of this subsection. In the figures 3.6 and 3.7 the vector $\hat{\mathbf{w}}_p$ is depicted, $\hat{\mathbf{w}}_p$ is an unit vector which is in the same direction as the principle axis. The rotation axis around which the template is rotated is the principle axis. It is possible to define direction of principle axis of the template from the rotation matrix $R_{T_i}^O$ which is recorded during registration. During the registration, for each template set T_i, an orientation can be assigned. In this case the pose of the template is equal to the pose of the camera. This can be expressed as:

$$R_{T_i}^O = R_C^O \tag{3.6}$$

This situation is depicted in figure 3.6.

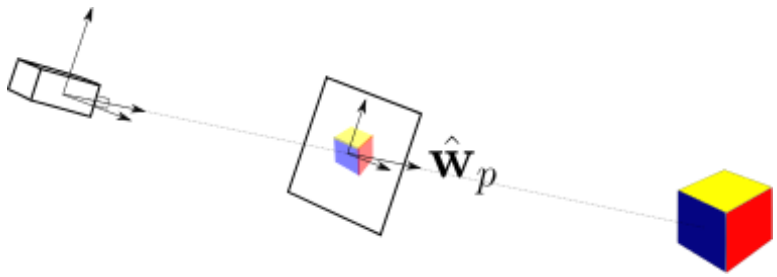

Figure 3.6: Standard orientation of the registered template

The distinction between the orientation of the template and the camera is made because during detection the orientation can differ. This is illustrated by comparing figure 3.6 and 3.7. In figure 3.8 the cameras views is illustrated.

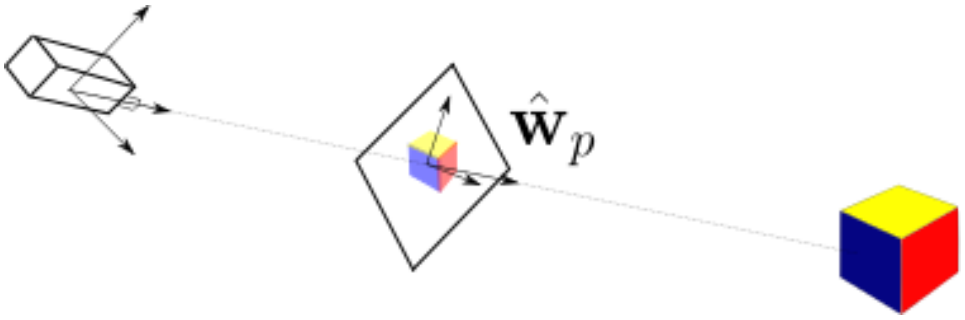

Figure 3.7: Rotated camera and orientation of the observed template, with respect to the registered template

From the rotation matrix R^O_{Ti} the orientation of the principle axis can be determined. The angle between the observed template and the registered template θ_p, which is depicted in figure 3.8.

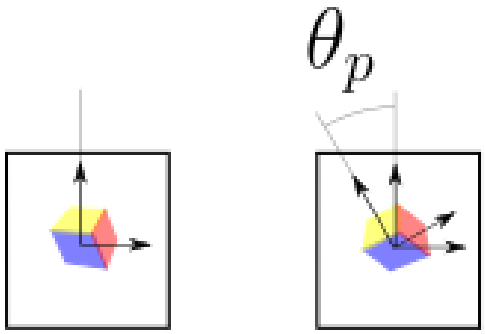

Figure 3.8: Template in standard orientation (left) and the rotated orientation (right)

The rotation around the principle axis can be expressed in a rotation matrix $R^{T_i}_C$, which expresses the rotation from the orientation of the the template set T_i with respect to the orientation of the camera. The calculation of the R^O_C can be achieved by:

$$R^O_C = R^O_{T_i} * R^{T_i}_C \tag{3.7}$$

The rotation of the template with respect to the camera is dependent on the angle of the observed template set with respect to the registered template set. In the case that the angle θ_p between the registered and observed template is zero, $R^{T_i}_C$ becomes the identity matrix. Equation 3.7 becomes: equation

$$R^O_C = R^O_{T_i} * R^{T_i}_C = R^O_{T_i} * I_{3\times 3} = R^O_{T_i} \tag{3.8}$$

Which is in accordance with expression 3.6. The orientation of the template can be calculated and is expressed in the rotation matrix $R^{T_i}_C$. The rotation matrix $R^{T_i}_C$ can be calculated by use of the Rodrigues' rotation formula. For

this the axis around which the template set is rotated is required, and the angle with which is rotated. These are respectively $\hat{\mathbf{w}}_p$ and θ_p. The angle θ_p between the registered template set and the observed template set was depicted in figure 3.8. If θ_p and $(\hat{\mathbf{w}}_p)$ can be obtained, the rotation matrix of the orientation of the camera with respect to the template set can be calculated with the Rodrigues' formula[5]:

$$R_C^{T_i} = I_{3\times3} + (\sin\theta_p)\tilde{\hat{\mathbf{w}}}_p + (1-\cos\theta_p)\tilde{\hat{\mathbf{w}}}_p^{\,2} \tag{3.9}$$

Where $\tilde{\hat{\mathbf{w}}}_p$ is the skew matrix of the unit vector of the rotation axis. If the $R_C^{T_i}$ from equation 3.9 is applied to equation 3.7, R_C^O can be calculated. With the calculation of R_C^O, the orientation of the object with respect to the camera is calculated. The orientation can be calculated by recognising the correct template and obtaining the angle θ_p. The rotation matrix is calculated in 3.9. With template matching an observed template is associated with a registered template with which a pose can be assigned. The registered template is recognised by the observed template with the highest similarity measure. The rotation of the observed template set with respect to the recognised registered template set, determines the orientation of the template set. To quantify the orientation that applies to a template, a solid angle is used. This is elaborated in the following subsection.

3.4.2 Solid angles

The accuracy of the orientation estimation depends on a number of things. A solid angle is introduced to quantify the error of orientation. The error of orientation is defined as the difference between the orientation of the observed template and the registered template. The accuracy of orientation estimation can be expressed by an angle. The angle can be calculated by using solid angles. Solid angles are expressed in steradians. Steradians are the SI unit of the solid angle. The solid angle is an expression of the ratio between the area covered by a part of a sphere and the square of the radius of the sphere. An example of a solid is illustrated in figure 3.9.

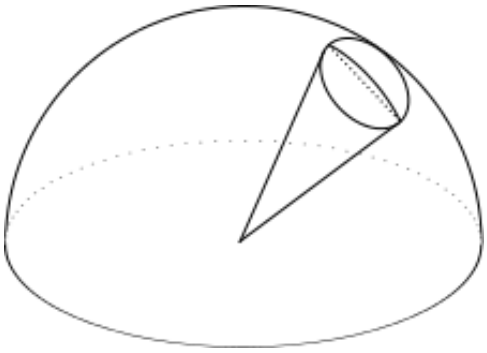

Figure 3.9: Solid in a hemisphere.

The template sets that are registered cover one side of all possible orientation of the object on a platform. This can be represented with a hemisphere. The representation of the object through a hemisphere is used to calculate the accuracy of the method. In [10] a model of an object consists of template sets that cover one hemisphere of an object. The object is on a calibration platform. As such the object can only be observed from one side of the platform. In figure 3.4 the principle axis, which is the line from the camera to the object, is orthogonal to the image plane. With respect to the hemisphere the principle axis crosses the hemisphere perpendicular to the surface of the hemisphere. The solid angle is expressed in equation 3.10.

$$\Omega = \frac{A}{r^2} \tag{3.10}$$

In which A is the surface of the part of the sphere that is covered, and r is the radius of the sphere The unit steradians $[sr]$ is dimensionless.

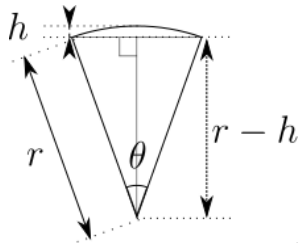

Figure 3.10: cross section of the solid as depicted in 3.9

The part of the sphere that is covered with solid, is a spherical cap. The spherical cap is depicted in figure 3.9. A cross section of the solid angle is depicted in figure 3.10. The surface area of a spherical cap can be calculated as:

$$A_{cap} = 2\pi rh \tag{3.11}$$

For a hemisphere the height is equal to the radius ($h = r$). The surface area of a hemisphere can be written as:

$$A_{hemisphere} = 2\pi r^2 \tag{3.12}$$

The solid angle of a hemisphere is $2\pi[sr]$:

$$\Omega_{hemisphere} = \frac{A_{hemisphere}}{r^2} = \frac{2\pi r^2}{r^2} = 2\pi \tag{3.13}$$

The solid angle of a spherical cap can be written as:

$$\Omega_{cap} = \frac{A_{cap}}{r^2} = \frac{2\pi rh}{r^2} \tag{3.14}$$

The angle can be calculated by the following equation:

$$\cos(\frac{\theta}{2}) = \frac{r-h}{r} \tag{3.15}$$

Multiplying equation 3.15 with (r):

$$r - h = r\cos(\frac{\theta}{2}) \tag{3.16}$$

The equation can be written as:

$$h = r(1 - \cos(\frac{\theta}{2})) \tag{3.17}$$

Substitute h in equation 3.15 and Ω_{cap} becomes:

$$\Omega_{cap} = 2\pi r^2(1 - cos(\frac{\theta}{2})) \tag{3.18}$$

Ω_{cap} is the expression for a solid angle, depending on angle θ as depicted in the figure. To calculate the number of templates as a function of the angle, the solid angle of the hemisphere is divided by the solid angle of the spherical cap:

$$N = \frac{\Omega_{hemisphere}}{\Omega_{cap}} \tag{3.19}$$

With this the equation becomes:

$$N = \frac{2\pi r^2}{2\pi r^2(1 - \cos(\frac{\theta}{2}))} = \frac{1}{1 - cos(\frac{\theta}{2})} \tag{3.20}$$

In 3.20 the number of templates is a function of the solid angle. Because the number of template sets is a natural number, the ceiling of the real number is taken:

$$N = ceil(\frac{1}{1 - cos(\frac{\theta}{2})}) \tag{3.21}$$

Equation 3.20 can be rewritten to:

$$\theta = 2\arccos(1 - \frac{1}{N}) \tag{3.22}$$

With 3.22 an expression for the angle covered by the template set T_i is obtained. With this the accuracy of the method can be calculated.

3.4.3 Accuracy of the orientation estimation

In the previous section two different rotation were identified. One is rotation was defined of the orientation of the registered template set with respect to the object (pitch and yaw). The other rotation is the rotation of the template around the principle axis of the camera (roll). In this subsection an estimation of the accuracy is made for both. First the accuracy of the yaw and pitch is expressed, after that the roll of the template is addressed.

Accuracy of the orientation estimation (yaw and pitch)

In the previous subsection, steradians where introduced. An expression for the angle covered by one template is given in equation 3.22. The angle is a function of the number of templates. With this an assumption is made that the templates are distributed uniformly. Another assumption is, that if an observed template is compared with the registered templates, the template will have a highest similarity measure with the closest registered template. The accuracy of the orientation can be expressed by an angle. For this the angle from 3.22 is divided by two:

$$\theta_{error,max} = \frac{\theta}{2} \tag{3.23}$$

A question that can be asked is, if the assumptions hold. For this a short description is given about thresholds which are applied during the registration of templates. During the learning phase two thresholds (t_{low} and t_{high}) are used. During the learning phase a new template set can be created. An input from the camera is used to create new templates. These templates are compared by means of the similarity measure, with the templates that belong already to registered templates. If the similarity measure is higher than t_{high} the new template set is rejected as a registered template set. This is done to ensure that not to many templates are registered. The low threshold is used to reject new template sets that have a very low similarity with it self. This is to keep the model with a relative low number of templates. The thresholds ensure a maximum to the number of templates, which in term ensures the speed of the template matching method.
It might be reasonable from the context to assume that if templates are registered with the thresholds, the templates are recognised by the closest template. The assumption is made because, if the goal is recognition of the object, a well chosen threshold would allow for the recognition of the object with a minimal number of templates. However, if the method is robust by having more templates than necessary, the number of templates other behavior might occur. The calculation of the accuaracy of the orientation estimation of the observed template with respect to the object follows. In [1] a number of N=2000 templates was mentioned. Linemod has two modalities. If the number of templates is 2000, the number of template sets is 1000. The number of template sets can be used to calculate the solid angle of a template in a uniformly distributed hemisphere model. The angle is calculated with 3.22. The angle becomes:

$$\theta = 2\arccos(1 - \frac{1}{N}) = 2 * \arccos(1 - \frac{1}{1000}) \tag{3.24}$$

This can be substituted in 3.23, that equation than becomes:

$$\theta_{errortemplate,max} = \frac{\theta}{2} = \frac{2 * \arccos(1 - \frac{1}{1000})}{2}$$

$$= \arccos(1 - \frac{1}{1000}) \approx 0,045[rad] \tag{3.25}$$

This is an angle of approximately 2.6°. However, for the implementation of [19], this might differ. In experiments a test is performed to see if the assumptions made in this section are correct.

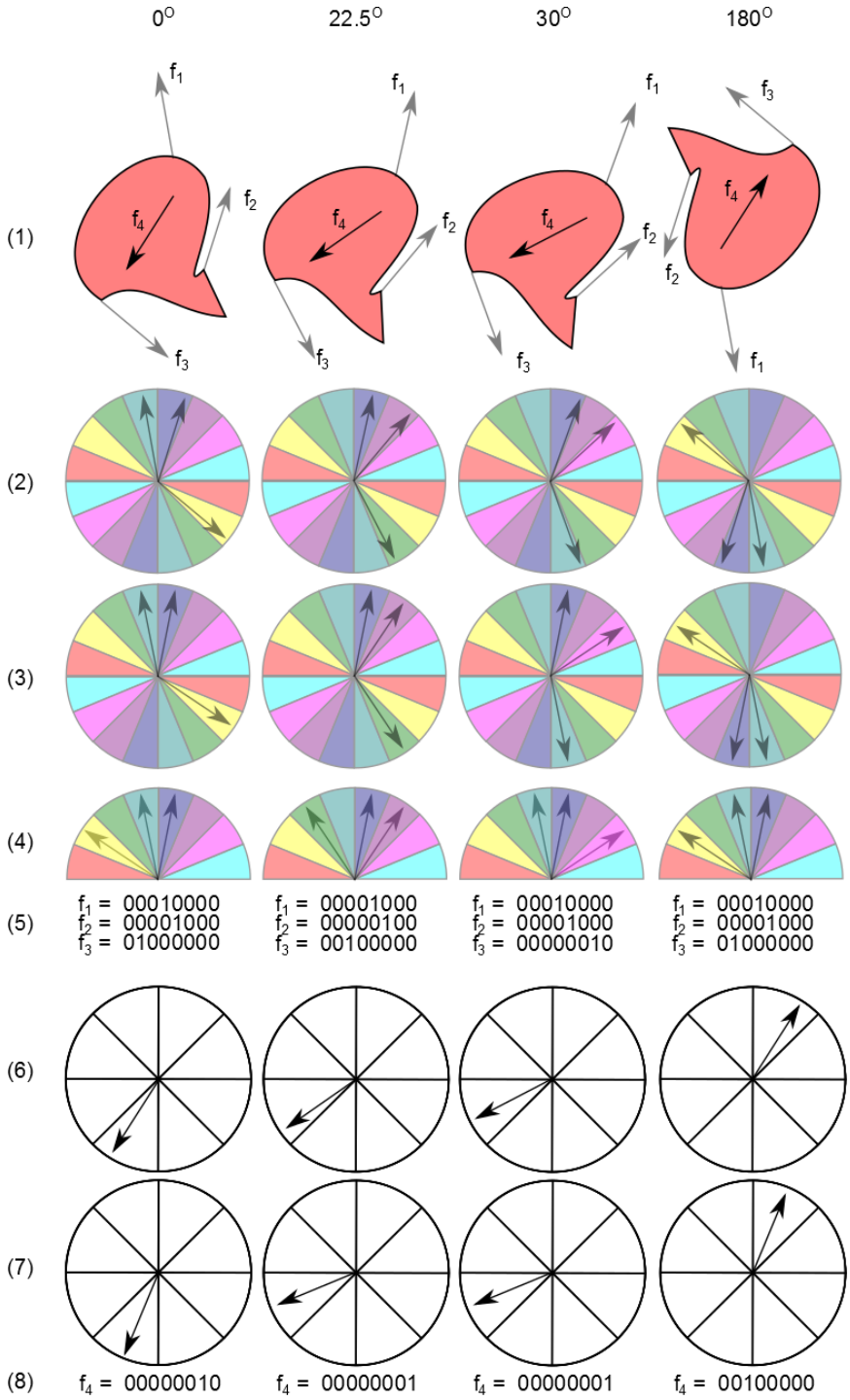

Figure 3.11: Quantization of features as a function of orientation. In this figure 4 orientations are depicted; every column has its own associated orientation (0°,22, 5°,30°,180°). The angle is indicated above the figure. The figure shows the steps of quantization. For the color gradient these steps are; (2) before quantization (3), after quantization, and after rectification of the features in (4). In (5) the 8 bit descriptors of the features are depicted. For the normals, the features are depicted before quantization (6), and after quantization (7). In (8) the 8 bit descriptor of the features are depicted.

Accuracy of the orientation estimation (roll)

In this section the accuracy of estimation of roll of an template is addressed. The change of orientation caused by roll of the camera can be calculated with 3.9. For the calculation of the roll, angle θ_p is required. In the estimation of θ_p the quantization and rectification of the features play a role. In figure 3.11 a template of an object (1) is rotated with the angles, 0°,22, 5°,30° and 180°. These rotations show that the features are quantized in different ways. The quantization is for the color gradient performed by dividing the features in one of 16 directions. Because the features are rectified, the orientation of the features of the color gradient at 180° are equal to the features of the color gradient at 0°, as can be seen in figure 3.11 (4) and (5). However, the normal is not rectified. As such the orientation of the template can still be estimated. The accuracy of the estimation of θ_p depends on the quantization of the features. The angle for the features of the color gradient is $\pi/8 = 22, 5°$. It might be possible to estimate the orientation more accurate. Figure 3.11 (2) at 30° depicts features in which quantization is different from the other orientations of the template. In that case feature 2 and 3 are not quantified next to each other. If the true orientation of template can be coupled to this phenomena the orientation can be estimated more accurate. Another possibility to estimate the orientation more accurate, would be to save the unquantified features. After recognition of the (regular) observed template set with the (regular) registered template set the comparison of the two unquantified template sets can be performed. In this way only two templates with unquantified features are compared, which allows the method to perform fast. On the other hand the storage of quantified templates would require much more memory than quantified templates.

3.4.4 Position estimation

In this subsection a suggestion is made on how the position of the object with respect to the Kinect might be measured. The estimation of the position is not validated in any way in this thesis. The emphasis is here that this is a suggestion for the position estimation. If the rotation of the the object is known, the estimation of the position can be found by relating the features to a position. During the registration of the template sets, the depth of the location of the features in the image plane can be measured, with the Kinect. A virtual position for the origin can be defined, for example in the middle of the calibration platform. The distances of the features to the virtual origin can be calculated and saved. During the detection the distance of these features can be measured and per feature the distance to the origin can be subtracted. The distance of the origin is calculated by taking the mean distance of the distance of the features subtracted by the saved distances

during the registration. This can be expressed in a summation:

$$\mathbf{d} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{d}_{i_{feature}} - \mathbf{d}_{i_{origin}} \tag{3.26}$$

One assumption that is made is that the features are approximately at the same image locations in the registered template as they are to the observed template. It might be useful to select features that are strong, in the sense that they are always observed and always occur on the same image position.

3.4.5 Accuracy of the position estimation.

The accuracy of the position estimation by using the depth image depends on the accuracy of the Kinect. However, the error of position estimation also depends on the error of the orientation. The error of the orientation can occur due to the recognition of a registered template that does not represent the true orientation of the observed template. The theoretical accuracy was calculated previously. The error that occurs by identifying the wrong template is an error of orientation. The error of orientation can affect the estimation of the position, because the features have a position that differs from the one during the registration. Because of this, an error of position estimation can occur. No experiment was performed for the estimation of the position and how it depends on the estimation of orientation. However, it is expected that error in position estimation are small if the error of the orientation estimation are small. With more significant errors in rotations, the estimation of position with the proposed method as expressed in equation 3.26, becomes less reliable.

3.5 Experiments

In section 3.3 it was explained how similarity measure is obtained by comparing registered template sets of a model to observed template sets. In section 3.4 a calculation was presented that indicates the accuracy of the estimation of the orientation with Linemod. The accuracy of the orientation estimation was calculated as a function of the number of registered template sets. In this section the performed experiment to validate if the accuracy of the orientation estimation holds. The section starts with the description of the experimental setup. After that the results of the experiments are given. The section ends with a discussion about the results.

3.5.1 Experimental setup

The experimental setup was designed to determine the similarity measure of an observed template set compared to a registered template set. In the experiment the object is rotated. The rotation of the object is expressed in degrees. The angle is the difference in orientation between the registered template set and the observed template set. For the purpose of rotating the object, a turntable is used. The object is positioned on a turntable that consists of a large bearing with a width of 30 cm and a plate on top of it. With the width of the bearing it is easy to rotate the object with high precision. It is possible to rotate the object with a precision better than a half degree. The measurement interval is one degree. A laser is positioned next to the turn table. The laser points to the scale of the turn table. If the turn table is rotated, the laser points to the angle with which it is rotated. In the setup the rotation is measured in degrees. The Kinect is positioned at a height of 40 cm above the turn table with a distance of the 100 cm to the object. The Kinect's view has an angle pointing down. In the experimental setup the Kinect is positioned above the object. The object is positioned on a turn table. An schematic drawing of the experimental setup is depicted in figure 3.12

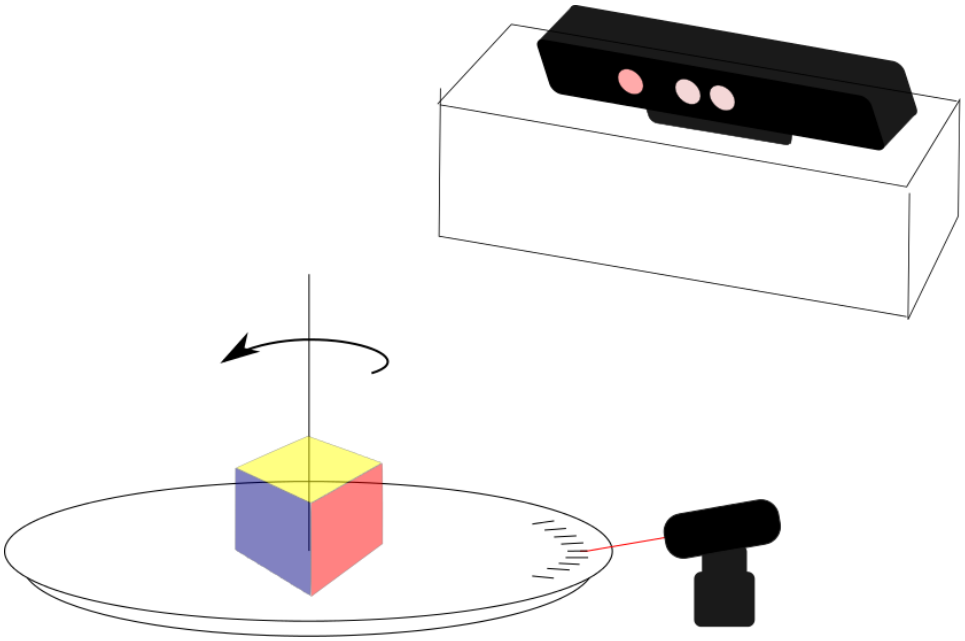

Figure 3.12: Schematic depiction of the experimental setup

During the learning phase, one template set is registered. During the detection the similarity measure is recorded. For each degree a measurement is performed. Because of the high frame rate of the method, approximately 100 measurements of similarity measure were recorded for each degree. The mean value and the standard deviation are depicted in the results. Measurements where taken between $-20°$ and $20°$ from the registered template set, resulting in 41 measurements. The process is repeated for the same object, with the registered template shifted $10°$. The registered template is in that case recorded on the $-10°$ mark. The experiment was performed on a number of objects of which three are depicted in the subsection Results. The three objects are a rolling pin, a washing brush and Rubik's Cube. These objects were not chosen with a particular purpose, however they may be encountered in a household. The objects lack a texture, the Linemod method is suited for the recognition of such objects.

3.5.2 Illumination

During the first experiment it was established that the similarity measure can differ due to difference in illumination. With direct sunlight the Kinect performs poorly, because the infrared structured light patterns cannot be recognised by the infrared camera. In that case no depth information is available. With artificial light was there were problems as well, during registration from one pose, many template sets were recorded. Why this occurred is not known, it might have something to do with the frequency of

the artificial light. The experiments were performed with indirect daylight conditions. Because light conditions changed during the experiments the illumination was measured.

3.5.3 Results

For the experiment plots are presented of the washing brush, the rolling pin, and a Rubik's Cube, in figure 3.13 the situation for the setup from the Kinect's view can be seen.

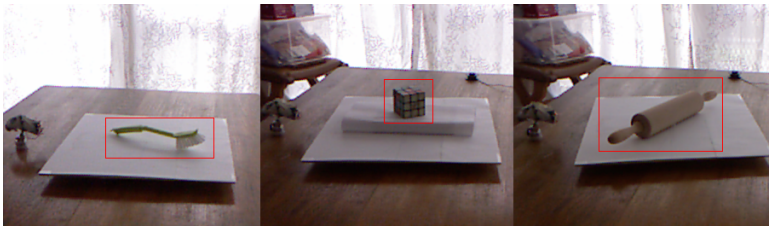

Figure 3.13: Images from the Kinect during the recording

The results of the experiments are depicted in graphs. For each object one graph of the similarity measure of the object is presented. In figure 3.14 the similarity measure of observed template set with the registered template set of the washing brush is given. Two plots are depicted, one is blue and the other is green. The blue plot represents a set of similarity measurements (y-axis) of observed template sets compared to the registered template set. On the x-axis an scale in degrees is depicted. For the blue plot the scale indicates the difference in orientation of the registered template set with the observed template set. For the green plot a similar measurement is performed, however now the registered template set is shifted by 10°. The registered template of the green plot is registered at $-10°$. The results of the experiments are depicted in graphs. For each object one graph of the similarity measure of the object is presented. Figure 3.14 is the similarity measure of the washing brush. One blue, and one green plot are depicted. The blue plot represents the similarity measurements (y-axis) of the observed template sets compared to the registered template set. On the x-axis an scale in degrees is depicted. For the blue plot the scale indicates the difference in orientation of the observed template set with the registered template set. For the green plot a similar measurement is performed, however now the registered template set is shifted by 10° The registered template of the green plot is registered at $-10°$.

The illumination for the measurement of the rolling pin and the Rubik's cube are presented. The results of the experiments are given per object, underneath.

Washing brush

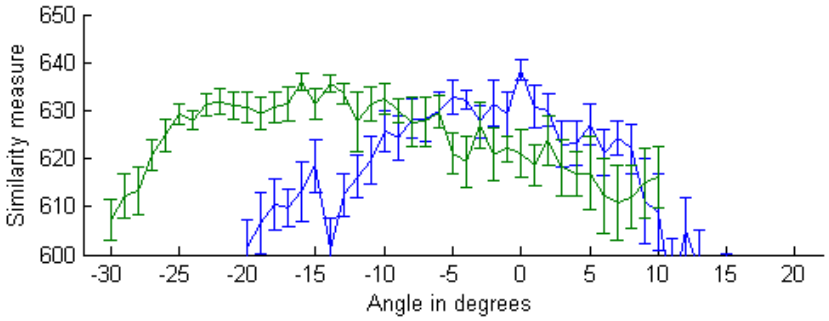

Figure 3.14: Similarity measure of the Washing brush recorded at different orientations.

Rubik's Cube

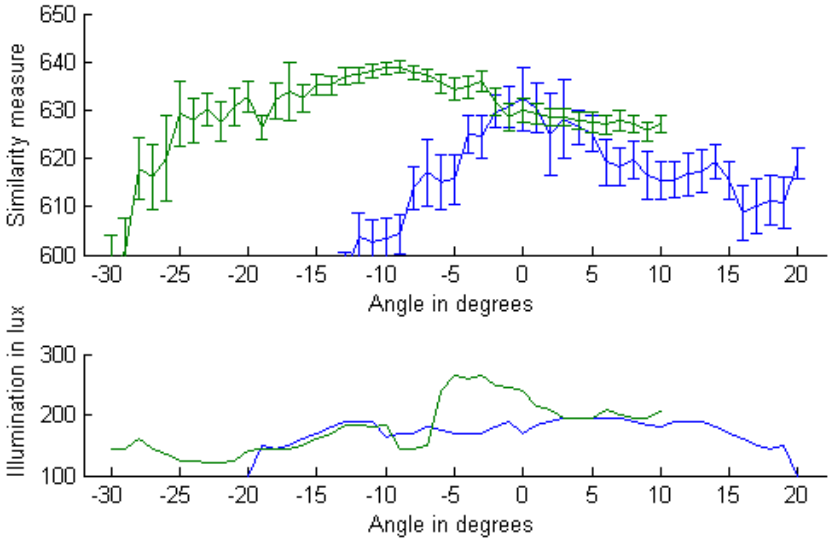

Figure 3.15: Similarity measure of the Rubik's Cube and the illumination recorded at different orientations.

Rolling pin

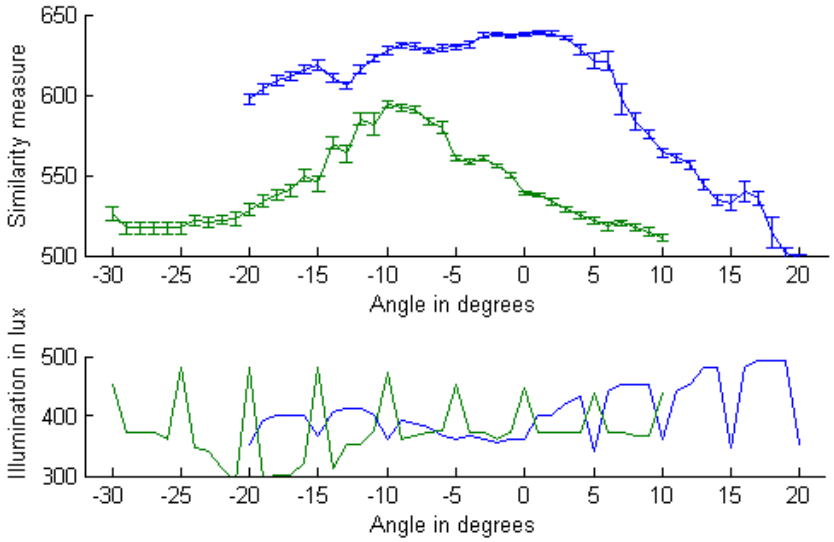

Figure 3.16: Similarity measure of the Rolling pin and the illumination recorded at different orientations.

3.5.4 Discussion

In this subsection a number of different aspects are presented. The aspects are object recognition, illumination and accuracy of the orientation estimation. For each of the aspects, the observations made are from the results presented in previous subsection.

Object recognition

During the experiments the objects were detected in a robust way. If the distance between the Kinect and the object changed, the objects were in most cases still recognised. In the experiments the similarity measure was recorded. For the purpose of detection a threshold is used to determine if an object is recognised or not. The value of this threshold can be set at different levels. However, normally the threshold for recognition is set at 576. This is mentioned because it helps to interpreter what the measurements that are presented in the results mean. For example the registered template set at $-10°$(the green plot) of the Rolling pin (3.16) would not be recognised at $-20°$, because the similarity measure at that point is less than 550. In this case the registered template set would not have been recognised anyway, because the similarity measure of the observed template set at that angle is higher for the registered template set at $0°$ (blue plot) than for the registered

template set at $-10°$ (green plot). For the recognition of templates, and objects for that matter, the recognition is based on the highest match. In the case of the Rubik's cube (3.15) the registered template set at $0°$ (blue plot) would have been recognised only from $-1°$ to $1°$ because the similarity with the registered template set at $-10°$ (green) has a higher similarity measure with the observed template set from $-30°$ to $-2°$ and from $2°$ $-10°$. The standard deviation of the results is indicated with an error bar, in which the error bar represents 2 standard deviations. The indicated standard deviation differs per object. No clear connection is found in the standard deviation of the similarity measure and the angle between the observed template set and the registered template set.

Illumination

The illumination of two object is measured at the different angles. The measurement of which was performed as a reference measurement. Because the measurements were performed with daylight, the illumination varied. A coincident occurred in the measurement of the Roling pin (figure 3.16). First a measurement was performed with an interval of $5°$. This was done to limit the change of daylight in the measurements. After that the measurements between the $5°$ marks were performed, by now the light conditions had changed. If a look is taken at figure 3.16 it can be seen that while the illumination almost doubled in some cases, no change in the measure of similarity seems to occur. This means that while the similarity measure under influence of direct sunlight and artificial light have a significant effect, changes in illumination without these factors do not seem to cause a change in the similarity measure.

Accuracy of orientation estimation

The theoretical accuracy of the Linemod method was calculated by the use of solid angles in section 3.4. For the calculation of the accuracy of the estimation, two assumptions were made. The first assumption is that the template sets are uniformly distributed. From the results it cannot be established how the templates are distributed over the hemisphere. However, some registered template sets have a wider range in which the observed template sets are recognised (as that registered template set) than other registered template sets. A clear example of this is depicted in figure 3.15, in which the green plot covers a much wider range than the blue plot. In more regular shapes (rotational symmetric for example) it is not expected that the template sets are uniformly distributed. An example is a cup, if it is rotated along its rotation axis very little change occurred. The second assumption is that the observed template set is associated with registered template set that is the closest orientation with respect to orientation of the

ground truth (the observed template set). The results from the experiments indicate that this assumption is not generally true. Some templates are more dominant than others. While this means that in some cases the error of orientation estimation becomes more significant, an orientation can still be associated with the observed template set. In the section 3.4 calculations for accuracy were performed to estimate the orientation. It was calculated that with 1000 template sets the accuracy of the Linemod method would be 2.6° The measurements presented in the results are not near that accuracy. Depending on the object some objects have a far wider range than other objects. Reasons for this could be that the implementation used is not exactly the same as Linemod. The templates are probably not uniformly distributed over the the hemisphere. Another reason could be that to have a robust object recognition many template sets are registered, which have only a very small orientation in which they have the highest match with the observed template sets.
In most of the plots the similarity measure drops significantly before ±20°. The method can be used for estimation of pose because the point cloud segmentation methods such as ICP, have a wide range of convergence. It is reported that the range of convergence for orientation in [17] is between 40° and 50° and in [18] it is up to 70°. It is plausible that the implemented method is accurate enough for an initial estimation for the ICP methods. This is because the ICP methods have a wide convergence. Concerns remain about how the similarity measure varies with a rotation of the camera (roll) and change of distance between the object and the camera.

3.6 Conclusions and Recommendations

In this section, conclusions and recommendations of the application of Linemod as pose estimator are given.

3.6.1 Conclusions

For flexible autonomous, robotic application it is desired that objects can be manipulated. For this task information about the pose of objects is desired. If objects are in standard orientation the estimation of the pose is equal to the estimation of position of objects. However, objects are not always in a standard positions. In that case the estimation of pose of an object becomes a more complicated task. For many objects a limited number of ways are available to correctly manipulate them. Therefor the estimation of orientation and position (pose) is important. In object recognition through template matching, the task for the template matching entails the recognition of one of the registered template sets of the (correct) object model. For pose estimation it is required to recognise the correct template set within

the object model. The difference between object recognition and pose estimation using template matching can be an conflict of interest. Because a fast recognition of objects is desired, it is desired to have as few as possible template sets with which a robust recognition is possible. This is because the fewer the number of template sets is, the faster the template matching can be performed. In section 3.4 an expression was given with which the theoretical accuracy of the method can be calculated. In the calculation the accuracy depends on the number of template sets. The higher the number of template sets, the higher the accuracy of the orientation estimation is. For the estimation of pose it is desired that features are used that discriminate against change of orientation, while for object recognition such features are not desired. Considering the results of the experiments that were performed, it is plausible that the estimation of orientation with Linemod is possible. However there are questions about how the similarity measure changes with rotation of the template. Other factors that may contribute to a different similarity measure are the distance of the object with respect to the camera and surrounding of the object. If the changes in similarity measure due to these factors do not change the recognition of the correct registered template set, the method can be used for the estimation of orientation of objects. If change of the similarity measure occurs due to these factors it does not have to result in failure to estimate the orientation. The method works as long as the correct registered template set has the highest similarity measure with the observed template set.

3.6.2 Recommendations

In the conclusions it was mentioned that changes in orientation of the observed template set (roll) and changes of distance between the camera and the object might have an effect on the similarity measure. For a robust orientation estimation it is important to understand what effect these factors have. For this reason it is recommended that experiments are performed to measure the effects that these factors have. An experiment is desired in which the similarity measure is a function of the distance, without changing the orientation of the object with respect to the camera. Another experiment which would be interesting, is recording the similarity measure while rotating the camera around the principle axis. With this experiment, effects of changing orientation of the template set on the similarity measure can be measured. In the implemented method three modalities are used. It is possible to use different combinations of modalities, for template matching. What the effects are for pose estimation might be interesting. It is recommended to perform experiments with different combinations of modalities.

Chapter 4

Motion set-point generator

4.1 Introduction

In this chapter a design is presented to provide input for controller of the Philips Robotic Arm (PRA) of the Bobbie robot. The purpose of the Bobbie project is developing a robot that is capable of serving in a home environment. For this purpose it is desired that the Bobbie robot is able to execute activities such as fetching objects, opening/closing doors and drawers, operating switches etc[1]. These activities involve grabbing particular objects. For the task of grabbing objects autonomously, the robot needs an input with which it identifies objects and estimates where the gripper of the robotic arm should steer toward. In the previous section a method was described to estimate the pose of objects. Object recognition and pose estimation are important for autonomous robots because it allows robots to search, find and locate a recognized object. If the poses of objects are known, this information can be used as an input to steer the end-effector to a certain pose.
For the task of grabbing objects, the Bobbie robot is equipped with the PRA with a gripper as end-effector. The arm consists of 7 joints which have approximately the same dimensions and joint space as a human arm. In previous work, an impedance controller was developed for the task of controlling the movement of the PRA [3, 14]. The controller requires a series of set-points as input. The set-point is expressed as a homogeneous matrix as a function of time (H(t)). The name motion set-point generator is derived from the fact that the set-point generator generates a moving set-point. The velocity of the end-effector is limited by limiting the velocity of the set-point. This is important because for safety it is required to limit the velocity of the PRA. The goal of this chapter is to present a design of an algorithm to generate a time series of desired poses (set-points), with an acceptable motion profile.

[1] http://www.Bobbierobotics.nl/

4.1.1 Problem statement.

The problem statement can be formulated as a design task. The goal is to steer the end-effectors of the PRA. The PRA is equipped with a controller which requires a homogeneous matrix as an input. The design task is to create an algorithm that generates a homogeneous matrix $H(t)$. The homogeneous matrix $H(t)$ should guide the end-effector from the start pose H_{n} to the end pose $H_{\mathrm{n+1}}$. Furthermore, the end-effector should follow an acceptable motion profile. An acceptable motion profile entails a maximum velocity, acceleration and jerk, reasons for this are elaborated in this chapter. The end-effector does not exactly follow the set-point because of intrinsic properties of the controller, however it can follow it near precise. It is assumed that the impedance controller ensures that the end-effector follows the set-point properly.

4.1.2 Outline of the chapter

This chapter starts with section 4.2 in which methods to calculate a homogeneous matrix are presented and compared. In section 4.3 a soft motion profile is described. The soft motion profile ensures that the motion of the set-point has maximum values for the velocity, acceleration and jerk. In section 4.4 the results of a number of simulations are presented that were performed to validate the motion set-point generator. The limitation of the motion set-point generator are considered in section 4.5. The chapter ends with conclusions and recommendations.

4.2 Calculation of the homogeneous matrix

The input that is required for the impedance controller is the homogeneous matrix, that varies over time. In this section two methods are presented to calculate the homogeneous matrix $H(t)$. The methods that are used, are descriptions in which no singularities can occur. This fact is important because this ensurers safety. If singularities occur the robotic arm can behave unpredictable, with high velocities for example. Especially in human environments it is important to ensure intrinsic stable system. That is why these methods are chosen. The reason for presenting two methods is that they are related to each other and both methods have some advantage. The first method is described by screw theory, which consists of a description that is compact ergo efficient. The second method calculates parameters with which the soft-motion profile can be calculated, which allows to set maximum values for the velocity, acceleration and jerk of the end-effector, for both linear as well as rotational movement. The inputs for both methods are the start pose and the end pose both of the end-effector expressed in homogeneous matrices which describe a pose with respect to a global ref-

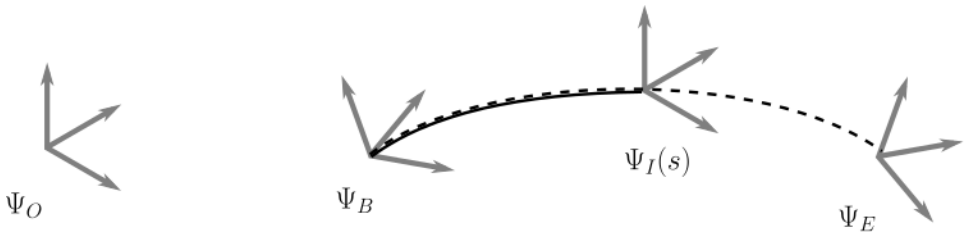

Figure 4.1: A path from Ψ_B to Ψ_E

erence frame. For the second method the axis-angle representation is used, this representation is described has been chapter 2.

4.2.1 Frames and homogeneous matrices

The problem that is described can be interpreted as a function that describes a path from one frame to another frame. If both frames are either left- or right-handed, it is possible to calculate a path from one frame to the next. For the purpose of describing a path, 4 frames are defined. Ψ_O is the global reference frame, Ψ_B is the initial frame from which the path starts. Ψ_E is the end frame towards which a path is planned. $\Psi_I(s)$ is the frame that describes the path and is a function of variable s. The 4 frames are depicted in figure 4.1. To describe the pose of the frames with respect to each other, homogeneous matrices are used. In chapter 2 a description of the homogeneous matrix is given.

4.2.2 Screw theory

This method describes how the homogeneous matrix is calculated that describes a path $\Psi_I(s)$ between the initial frame Ψ_B and Ψ_E. The frames are respectively expressed as homogeneous matrices $H_O^I(s)$, H_O^B and H_O^E, as depicted in figure 4.1. From the frames presented here the expression for the homogeneous matrices are of the frame (B,I or E) with respect to the global frame (O). The homogeneous matrix H_O^B is the pose with respect to the global frame Ψ_O. The $H_I^O(s)$ is the homogeneous matrix that is required as input for the impedance controller.

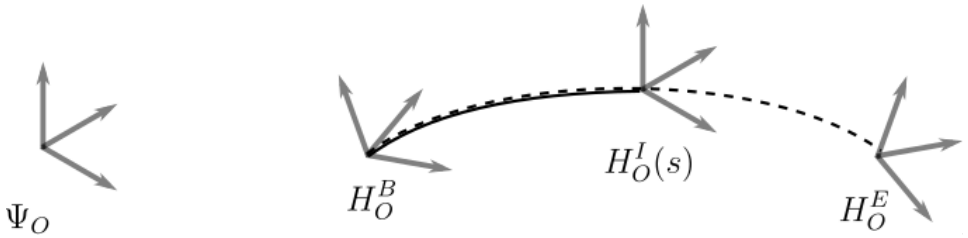

Figure 4.2: A path from Ψ_B to Ψ_E, expressed in homogeneous matrices.

The homogeneous matrix $H_0^I(s)$ is desired, because it describes the path of the set-point with respect to a global reference frame Ψ_O. $H_O^I(s)$ can be calculated in terms of the begin pose and end pose respectively H_O^B, H_O^E, the pose of the moving frame $\Psi_I(s)$ with respect to frame B is expressed with $H_B^I(s)$. This is in [2] with the following equations:

$$H_B^I(s) * H_O^B = H_O^I(s) \tag{4.1}$$

From equation (4.1) it can be seen that if $H_B^I(s)$ is known $H_O^I(s)$ can be calculated. To calculate the $H_B^I(s)$, first the two known poses are used, the begin and end pose expressed in the homogeneous matrices. This is the information required to calculate a path. The difference of the pose is expressed as a homogeneous matrix:

$$H_B^E = H_O^E * H_B^O \tag{4.2}$$

The inverse of H_B^O is H_O^B, substituted in the equation (4.2) the expression becomes (4.3).

$$H_B^E = H_O^E * H_O^{B^{-1}} \tag{4.3}$$

The expression for the homogeneous matrix that is the change of pose can be written as an exponent of a twist in (4.4).

$$H_B^E = e^{\tilde{T}} \tag{4.4}$$

which can be rewritten as (4.5):

$$\tilde{T} = \log(H_B^E) \tag{4.5}$$

The function H(s) is defined as (4.6):

$$H_B^I(s) := e^{\tilde{T}s} \tag{4.6}$$

For the path of the homogeneous matrix $H_B^I(s)$, with s:

$$0 \leq s(t) \leq 1 \tag{4.7}$$

For s = 0 the homogeneous matrix is the identity matrix:

$$H_B^I(0) = I \tag{4.8}$$

This is in accordance with the fact that $H(0) = H_O^B$ is the begin pose of the path. Furthermore for $H(1)$:

$$H_B^I(1) = H_B^E \tag{4.9}$$

The expression for the homogeneous matrix $H_O^I(s(t))$ becomes

$$H_O^I(s(t)) = H_B^I(s(t)) * H_O^B \tag{4.10}$$

The value of expression (4.10) for $s(t) = 0$ and $s(t) = 1$ are respectively:

$$H_O^I(0) = I_{4\times4} * H_O^B = H_O^B \tag{4.11}$$

$$H_O^I(1) = H_B^E H_O^B = H_O^E \tag{4.12}$$

Screw theory presents a concise and clear description for a path. The velocity of the motion is determined by the profile of $s(t)$. In the next subsection another method is described to relate the twist of the movement to the calculation of $H(t)$. In this method a set of parameters is calculated, these parameters can be used to calculate a soft-motion profile with limited velocity, accelerations and jerks for both rotational as linear movement. This soft-motion profile can be used for the problem in this chapter by normalizing the distance of the soft motion profile such that the function becomes suitable for s(t).

4.2.3 Calculation of the homogeneous matrix with an alternative interpretation

In the previous section screw theory was used to calculate a homogeneous matrix. In this section Chasles theorem and the axis-angle representation are used to calculate twists. The twist is used to calculate the homogeneous matrix $H(t)$. The method described in this subsection is less compact, however the parameters of a twist for a screw motion or linear motion can be used to construct a motion profile in which maximum values can be given for the velocity, acceleration and jerk of the set-point. The calculation of the homogeneous matrices is achieved by first describing the twist. Two cases of twist are calculated, namely a screw motion and a linear motion. If the rotation from one frame to another is zero a linear motion of the set-point occurs, in all other cases the motion can be described by a screw motion. In this subsection, first the calculation of twist is presented for both the screw motion and the linear motion. Next a general algorithm is presented including both motions. After that a calculation of the homogeneous matrix as a function of twist is presented.

Twist of a screw motion: an alternative interpretation

The path of a screw motion such as described in Chasles theorem [2, 5],can be described by a number of parameters these are, the unit vector $\hat{\mathbf{w}}$, the vector $\mathbf{r}$, and λ. Another parameter can be defined as the angle θ. The unit vector $\hat{\mathbf{w}}$ is the direction of the axis around which the the set-point is rotated. The vector $\mathbf{r}$ is a vector pointing to perpendicular to the rotation axis and can be interpreted as a radius. The parameter λ is the pitch of the screw motion and θ is the angle along which the set-point is rotated.
The values for the parameters of the screw motion can be calculated with information of the begin pose expressed in (H_O^B) and the end pose expressed

in (H_O^E). For this purpose the axis-angle representation [5] is used. The axis-angle representation describes any rotation, with one rotation vector around which the set-point is rotated ($\hat{\mathbf{w}}$) and an angle (θ) with which the set-point is rotated around the rotation axis. The calculation of rotation matrix from orientation $R_O^B = H_O^B(1{:}3,1{:}3)$ to orientation $R_O^B = H_O^B(1{:}3,1{:}3)$ is described in [2], the calculation of R_E^B can be expressed as (4.13):

$$R_E^B = R_O^B * R_E^O \tag{4.13}$$

By using the axis-angle representation the unit rotation axis $\hat{\mathbf{w}}$ and the travelled angle θ can be calculated from R_E^B. A description of the axis-angle representation can be found in chapter 2. The unit vector $\hat{\mathbf{w}}$ and traveled angle θ are depicted in figure 4.3.

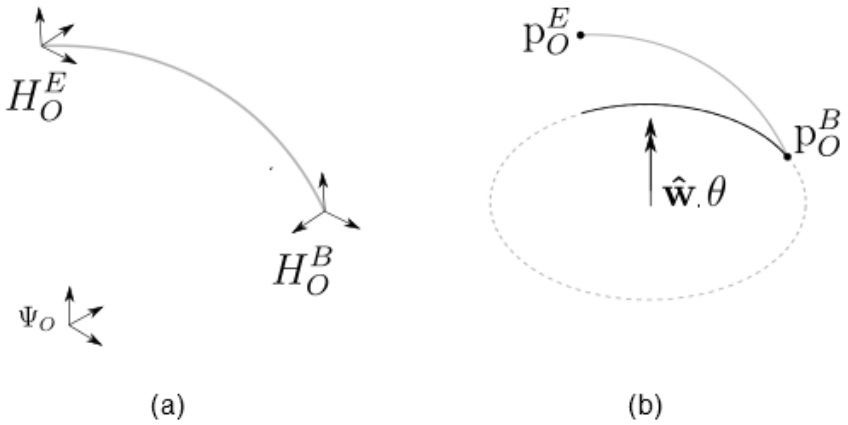

Figure 4.3: Homogeneous matrices (a) and the axis-angle representation (b)

To calculate the homogeneous matrix with this method, first the twist is required, Chasles theorem is used for this purpose. According to Chasles theorem a twist of a screw motion can be written as:

$$\mathbf{T} = \begin{pmatrix} w_x \\ w_y \\ w_z \\ v_x \\ v_y \\ v_z \end{pmatrix} = \begin{pmatrix} \mathbf{w} \\ \\ \mathbf{v} \end{pmatrix} = \begin{pmatrix} \mathbf{w} \\ \\ \mathbf{r} \wedge \mathbf{w} + \lambda * \mathbf{w} \end{pmatrix} \tag{4.14}$$

In these equations $\mathbf{r}$ is the vector that spans the shortest distance from the start pose to the rotation vector. The parameter λ is in $[m/rad]$ which represents the pitch of the screw. The twist is described as a function of vector $\mathbf{w}$, which is the rotational velocity. The rotational velocity w can be expressed as:

$$w = |\mathbf{w}| \tag{4.15}$$

The rotational velocity w which is a scalar. The rotation vector $\mathbf{w}$ can be expressed in terms of the rotational velocity w and unit vector of the rotation vector ($\hat{\mathbf{w}}$):

$$\mathbf{w} = w * \hat{\mathbf{w}} \tag{4.16}$$

The twist can be expressed as an unit twist multiplied by rotational velocity (w):

$$T = w * \hat{\mathbf{T}} = w * \begin{pmatrix} \hat{\mathbf{w}} \\ \\ \mathbf{r} \wedge \hat{\mathbf{w}} + \lambda * \hat{\mathbf{w}} \end{pmatrix} \tag{4.17}$$

During the motion a rotation around $\hat{\mathbf{w}}$ is perfomed. When the movement is completed the traveled angle is θ is reached. During the motion the traveled angle can be expressed as $\theta(t)$. The integral of the rotational velocity w is the traveled angle $\theta(t)$. This is expressed in equation (4.18), ith t_0 the start time:

$$\theta(t) = \int_{t_0}^{t} w(\tau) d\tau, \tag{4.18}$$

If an object rotates around a certain vector ($\hat{\mathbf{w}}$) with an angle θ it makes no difference where the vector is positioned. Therefor ($\hat{\mathbf{w}}$) is a free vector, which implicates that it can be placed anywhere to achieve the desired rotation. The fact that ($\hat{\mathbf{w}}$) is a free vector can be used to obtain a certain change in position. The vector $\mathbf{r}$ spans the shortest distance between the origin of the initial frame of the object and the vector it rotates around, which is $\hat{\mathbf{w}}$. The vector $\mathbf{r}$ is perpendicular to $\hat{\mathbf{w}}$. These facts can be used to calculate $\mathbf{r}$. However first λ is calculated, after that $\mathbf{r}$ follows. The parameter λ and the variable $\mathbf{r}$ can be calculated from $\hat{\mathbf{w}}$, θ and the poses from the start of the path to the end of the path, respectively H_O^B and H_O^E. Figure 4.4 below is depicted to visualize the calculations that follow.

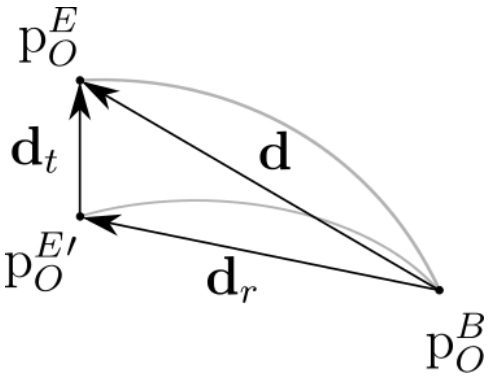

Figure 4.4: Distances between two points. $\mathbf{d}_t$ represents the translation distance, $\mathbf{d}_r$ represent the rotational distance , $\mathbf{d} = \mathbf{d}_t + \mathbf{d}_r$. Vector $\mathbf{d}$ represents the total distance.

In figure 4.4 the Euclidean distance between H_O^B and H_O^E can be represented as a vector $\mathbf{d}$, which can be calculated with equation (4.19):

$$\mathbf{d} = p_O^E - p_O^B \tag{4.19}$$

Vector $\mathbf{d}$ represents the distance from the begin pose (expressed in H_O^B) to the end pose (expressed in H_O^E). Vector $\mathbf{d}$ can be expressed as a combination of two vectors which are perpendicular to each other. These two vectors are $\mathbf{d}_t$, and $\mathbf{d}_r$ both are depicted in figure 4.4. It is possible to calculate the two vectors because $\mathbf{d}_t$ is parallel to $\hat{\mathbf{w}}$ and $\mathbf{d}_r$ is perpendicular to $\hat{\mathbf{w}}$. Because of this ($\mathbf{d}_t$) can be expressed as the the dot product of $\mathbf{d}$ and the vector $\hat{\mathbf{w}}$:

$$\mathbf{d}_t = \mathbf{d} \cdot \hat{\mathbf{w}} \tag{4.20}$$

The vector $\mathbf{d}_r$ can be calculated in two ways. Because $\mathbf{d}_r$ is perpendicular to $\hat{\mathbf{w}}$ the cross product can be taken. However, from figure 4.4 it can be seen that:

$$\mathbf{d} = \mathbf{d}_t + \mathbf{d}_r \tag{4.21}$$

Equation (4.21) can be rewritten to:

$$\mathbf{d}_r = \mathbf{d} - \mathbf{d}_t \tag{4.22}$$

With (4.22) vector $\mathbf{d}_r$ is derived.
The pitch is a parameter which expresses the relation between the rotation and the translation along the direction of the (unit) rotation axis ($\hat{\mathbf{w}}$) . The pitch is notated as λ and can be expressed by dividing the length of the translation parallel to $\hat{\mathbf{w}}$ by the traveled angle θ The length of the translation parallel to $\hat{\mathbf{w}}$ is the norm of $\mathbf{d}_t$. The expression becomes:

$$\lambda = \frac{|\, \mathbf{d}_t \,|}{\theta} \tag{4.23}$$

The pitch expressed in λ can assume negative values, negative values for λ mean that the translation parallel to $\hat{\mathbf{w}}$ is in opposite direction of rotation vector $\hat{\mathbf{w}}$.

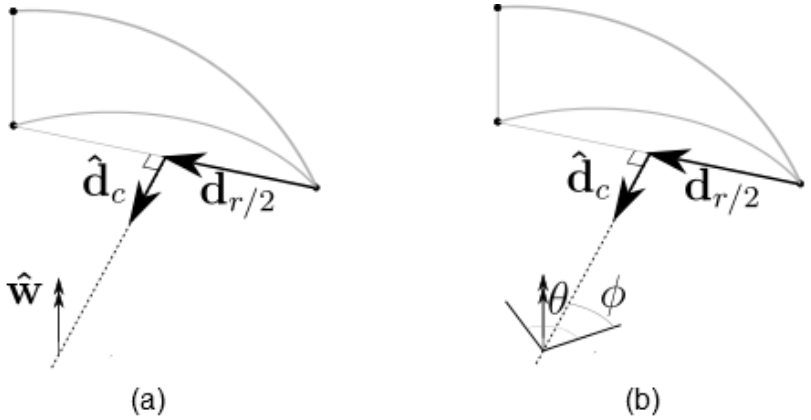

Figure 4.5: In (a) the vectors $\hat{\mathbf{d}}_c$, $\mathbf{d}_{r/2}$, $\hat{\mathbf{w}}$ are depicted. In (b) an illustration the vectors $\hat{\mathbf{d}}_c$, $\mathbf{d}_{r/2}$, and angles θ and ϕ

In figure 4.5 two illustrations are depicted. In both illustrations ((a) and (b)) two curved lines are depicted. The long curved line is the path, described by a screw. The short curved line is an arc which is on a plane that is perpendicular to the unit rotation axis ($\hat{\mathbf{w}}$). This arc is part of a circle. The rotation axis is situated in the centre of that circle. It is possible write an expression which expresses the centre of the circle.
In figure 4.4 it can be seen that vector $\mathbf{d}_r$ spans the chord of the circle. This vector can be seen as a part of a special triangle which is called a isosceles. The isosceles has 2 sides that have the same length. The chord spanned by $\mathbf{d}_r$ is the third side of the isosceles.
If in a circle a triangle is drawn by drawing two lines from the centre of the circle to the ends of one chord of that circle a triangle is drawn that is a isosceles, in figure 4.6 two illustrations are depicted, in both illustration an which isosceles can be seen.
The centre of the circle is somewhere on a line on the plane of the circle perpendicular to the chord. That line crosses the chord in the middle. This line can be expressed as a vector. In figure 4.5 the vector $\hat{\mathbf{d}}_c$ is depicted, an extension of $\hat{\mathbf{d}}_c$ is the line that crosses the chord perpendicular in the middle. The direction of the line and vector $\hat{\mathbf{d}}_c$ can be expressed because $\hat{\mathbf{d}}_c$ is perpendicular to $\mathbf{d}_r$ and $\hat{\mathbf{w}}$, which are in term perpendicular to each other. As such $\hat{\mathbf{d}}_c$ can be expressed in the following expression:

$$\hat{\mathbf{d}}_c = \frac{\hat{\mathbf{w}} \wedge \mathbf{d}_r}{|\hat{\mathbf{w}} \wedge \mathbf{d}_r|} \tag{4.24}$$

The angle between the two sides of the isosceles that have the same length

is θ. This is depicted in the right illustration of figure 4.5. In figure 4.6 it is shown that the line that is the extension of vector $\hat{\mathbf{d}}_c$ splits the isosceles in two symmetric parts. Because of this the an other angle can be expressed that is the angle θ divided by 2, ϕ is depicted in figure 4.6. The expression for ϕ is:

$$\phi = \frac{\theta}{2} \tag{4.25}$$

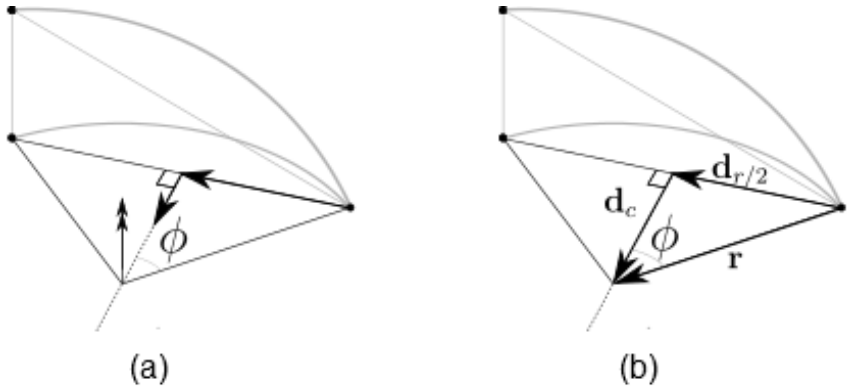

Figure 4.6: The calculation of $\mathbf{d}_c$ and $\mathbf{r}$

The vector $\mathbf{d}_{r/2}$ and $\mathbf{d}_c$ are depicted in figure 4.6. The calculation of $\mathbf{d}_{r/2}$ is expressed in equation (4.26):

$$\mathbf{d}_{r/2} = \frac{\mathbf{d}_r}{2} \tag{4.26}$$

The calculation of $\mathbf{d}_c$ can be performed by finding an expression for the length of $\mathbf{d}_{r/2}$ this is expressed as:

$$d_{r/2} = |\mathbf{d}_{r/2}| \tag{4.27}$$

With ϕ from (4.25) and $d_{r/2}$ from (4.27) the length of $\mathbf{d}_c$ can be expressed:

$$d_c = \frac{d_{r/2}}{tan(\phi)} \tag{4.28}$$

With this the vector $\mathbf{d}_c$ can be expressed as:

$$\mathbf{d}_c = d_c * \hat{\mathbf{d}}_c \tag{4.29}$$

The vector $\mathbf{r}$ can be calculated by adding $\mathbf{d}_c$ to $\mathbf{d}_{r/2}$:

$$\mathbf{r} = \mathbf{d}_c + \mathbf{d}_{r/2} \tag{4.30}$$

As is depicted in figure 4.6 (b). The parameters that are required to use equations (4.17) are expressed. In order to calculate the twist used to calculate the homogeneous matrix some additional expressions follow.

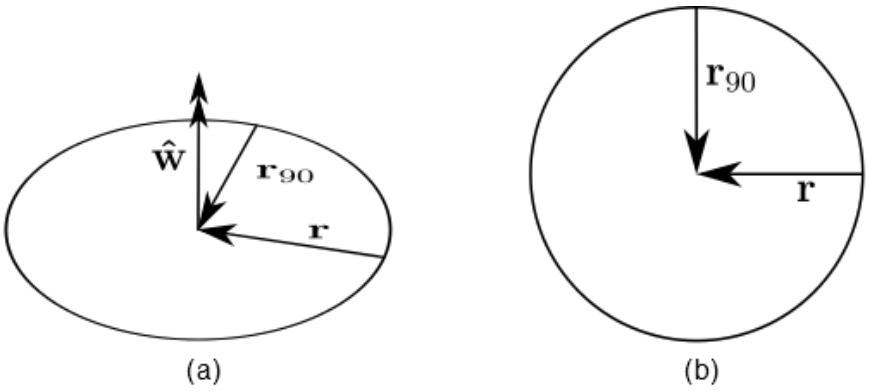

Figure 4.7: Interpretation of $\mathbf{r}$ and $\mathbf{r}_{90}$

In figure 4.7 a circle is depicted. Two views are given to clarify the illustration. From the border of the circle two vectors ($\mathbf{r}$ and $\mathbf{r}_{90}$) point to the centre of the circle. Earlier in this section vector $\mathbf{r}$ was calculated. To calculate $\mathbf{r}_{90}$ the wedge product can be taken between $\hat{\mathbf{w}}$ and $\mathbf{r}$.

$$\mathbf{r}_{90} = \hat{\mathbf{w}} \wedge \mathbf{r} \tag{4.31}$$

Earlier it was described that $\mathbf{r}$ is the vector of the frame perpendicular to the (unit) rotation axis $\hat{\mathbf{w}}$. For the calculation of the twist with screw theory only the initial $\mathbf{r}$ is required to described the twist. In the calculation of the twist in the alternative interpretation a $\mathbf{r}$ is required that calculates this for every moment of the motion of the set-point. The function is a time dependent function $r(t)$ that is a function of the orientation of the set point. The orientation can be expressed with $\theta(t)$ For that purpose an expression for $\mathbf{r}(\theta(t))$ is required. This expression can be found by using the parameters $\mathbf{r}$ and $\mathbf{r}_{90}$. The function $\mathbf{r}(\theta(t))$ can be calculated, by multiplying $\mathbf{r}$ with $\cos\theta(t)$ and adding it with $\mathbf{r}_{90}$ multiplied by $\sin\theta(t)$. This results in the following expression:

$$\mathbf{r}(\theta(t)) = \cos\theta(t)\mathbf{r} + \sin\theta(t)\mathbf{r}_{90} \tag{4.32}$$

To interpreter the calculation of $\mathbf{r}$(t) figure 4.8 is depicted.

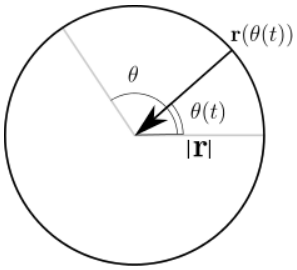

Figure 4.8: $\mathbf{r}(\theta(t))$ as a function of θ(t), with θ the travelled angle. The figure can be interpreted as the top view of a screw.

The twist is now a time dependent function depending on $w(t)$ only. Because $r(t)$ is dependent only on $w(t)$. The twist can be written as 4.33:

$$\mathbf{T}(t) = w(t)\hat{\mathbf{T}}(t) = w(t)\begin{pmatrix} \hat{\mathbf{w}} \\ \\ \mathbf{r}(t) \wedge \hat{\mathbf{w}} + \lambda * \hat{\mathbf{w}} \end{pmatrix} \tag{4.33}$$

The goal of the next section is to find appropriate motion profile for $w(t)$. To recapitulate, the following steps are taken in this description to calculate **T(t)**. The only input that is required are the homogeneous matrices H_O^A, H_O^B and the motion profile $w(t)$. In this description the dependency on the previous steps are indicated between parenthesis (H_O^B) and the end pose (H_O^E).

1. From H_O^B and H_O^E the rotation matrix R_B^E can be calculated. From this rotation matrix the unit rotation vector $\hat{\mathbf{w}}$ and traveled angle θ can be calculated. With θ, the angle ϕ can be calculated by dividing θ by two; $\phi = \frac{\theta}{2}$.

2. From H_O^B and H_O^E, p_O^B and p_O^E can be extracted. From this $\mathbf{d}$ is calculated: $\mathbf{d} = p_O^E - p_O^B$.

3. The vector $\mathbf{d}_t$ can be calculated by taking the dot product of $\hat{\mathbf{w}}$ (1) and $\mathbf{d}$ (2); $\mathbf{d}_t = \hat{\mathbf{w}} \cdot \mathbf{d}$.

4. The pitch λ is calculated from θ (1) and $\mathbf{d}_t$ (3); $\lambda = \frac{|\mathbf{d}_t|}{\theta}$

5. The vector $\mathbf{d}_r$ can be calculated from $\mathbf{d}$ (1) and $\mathbf{d}_t$ (3); $\mathbf{d}_r = \mathbf{d} - \mathbf{d}_t$. From the vector $\mathbf{d}_r$ the vector $\mathbf{d}_{r/2}$, can be calculated by $\mathbf{d}_{r/2} = \frac{\mathbf{d}_r}{2}$. From $\mathbf{d}_r$ also the scalar d_r can be calculated, by calculating the norm: $d_r = |\mathbf{d}_r|$.

6. The vector $\hat{\mathbf{d}}_c$ can be calculated with $\hat{\mathbf{w}}$ (1) and $\hat{\mathbf{d}}_r$ (5): $\hat{\mathbf{d}}_c = \frac{\hat{\mathbf{w}} \wedge \mathbf{d}_r}{|\hat{\mathbf{w}} \wedge \mathbf{d}_r|}$

7. The length d_c can be calculated with $\phi(1)$ and $d_{r/2}(5)$: $d_c = \frac{d_{r/2}}{tan(\phi)}$.

8. The vector $\mathbf{d}_c$ is calculate from the unit vector $\hat{\mathbf{d}}_c$ (6) and d_c (7); $\mathbf{d}_c = d_c * \hat{\mathbf{d}}_c$

9. Now $\mathbf{r}$ can be calculated, with $\mathbf{d}_c$ (8) and $\mathbf{d}_{r/2}$ (5): $\mathbf{r} = \mathbf{d}_c + \mathbf{d}_{r/2}$.

10. The vector $\mathbf{r}_{90}$ can be calculated with wedge product of $\hat{\mathbf{w}}$ (1) and $\mathbf{r}$ (9): $\mathbf{r}_{90} = \hat{\mathbf{w}} \wedge \mathbf{r}$.

11. To find an expression for $\mathbf{r}(\theta(t))$, $\theta(t)$ is required. The expression of $\theta(t)$ can be found by integrating the rotational velocity: $\theta(t) = \int_{t_0}^{t} w(\tau)d\tau$. With t_0 as the start time.

12. The vector $\mathbf{r}(\theta(t))$ can be expressed with $\mathbf{r}$ (9) and $\mathbf{r}_{90}$ (10): $\mathbf{r}(\theta(t)) = \cos\theta(t)\mathbf{r} + \sin\theta(t)\mathbf{r}_{90}$.

Steps 1 to 10 are algebraic expressions, for every screw motion these parameters are calculated. Step 11 and 12 are numerical expressions. With these steps values for parameters (1 to 10) and variables (11 and 12) are obtained with which the twist (4.33) be calculated. In the next section the description of $\mathbf{T}(t)$ is not required to calculate the soft motion profile. This is because the calculation of the homogeneous matrix is more efficient with the previous method: screw theory and therefore the calculation of $\mathbf{T}(t)$ is redundant. However the remainder of this section is about the calculation of the twist $\mathbf{T}(t)$ of the linear motion and the calculation of a homogeneous matrix $H(t)$ as a function of $\mathbf{T}(t)$. With the representation of the twist that is described above the linear velocities of the twist in different directions can be interpreted.

4.2.4 Twist of a linear motion

In this section a linear motion is expressed in a twist. In contrast to the twist of a screw motion, a linear motion can be described in a twist without an extensive calculations. For the purpose of describing a path figure 4.9 three homogeneous matrices (H_O^B,H_O^E,$H_O^I(s)$) are considered. These homogeneous matrices express respectively the pose of three frames (Ψ_B,Ψ_E,$\Psi_I(s)$), with respect to the global frame Ψ_O. In which $H_O^I(s)$ is the expression for the pose of the frame that describes the desired path.

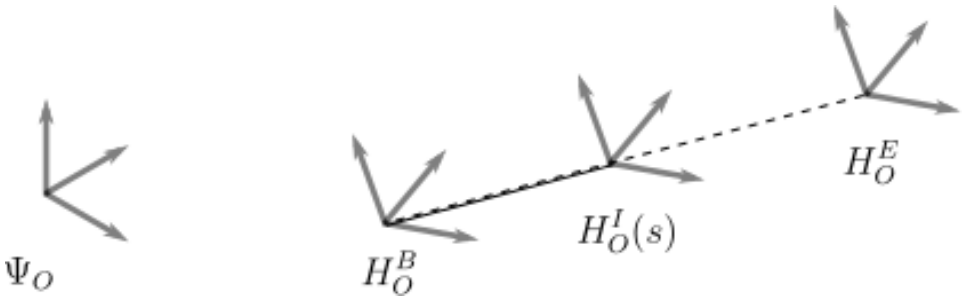

Figure 4.9: Path of a linear motion

In this case the twist can be written as a linear velocity. The expression of the twist becomes:

$$\mathbf{T}(t) = \begin{pmatrix} w_x \\ w_y \\ w_z \\ v_x \\ v_y \\ v_z \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \\ \mathbf{v} \end{pmatrix} = v(t) * \hat{\mathbf{T}} = v(t) * \begin{pmatrix} \mathbf{0} \\ \\ \hat{\mathbf{v}} \end{pmatrix} \tag{4.34}$$

The unit vector $\hat{\mathbf{v}}$ expresses the direction of the path. The absolute velocity is given as a function $v(t)$. The calculation of the twist in a linear motion is relatively simple, because the direction of the twist is fixed. The direction of the velocity is expressed in right member of equation 4.34 by the unit twist, which is written as an unit velocity vector. The direction of the velocity can be expressed by calculating the vector of the distance:

$$\mathbf{d} = p_2 - p_1 \tag{4.35}$$

With this the unit vector of the velocity can be calculated.

$$\hat{\mathbf{v}} = \frac{\mathbf{d}}{|d|} \tag{4.36}$$

The velocity vector can be calculated by multiplying the unit vector with the desired velocity.

$$\mathbf{v}(t) = v(t)\hat{\mathbf{v}} \tag{4.37}$$

With $v(t)$ as the value of the motion profile.

4.2.5 Algorithm for a general calculation of twists.

In the previous subsections two twists were described. For the calculation of the homogeneous matrix $H(t)$ it is necessary that the correct twist is selected. To ensure that the appropriate twist is used an algorithm is used. The algorithm calculates the twist as is described in the previous sections.

The appropriate motion is selected, which is either a screw motion or a linear motion. The decision to choose one of the motions is based on the the begin pose and end pose, which are expressed in H_O^B and H_O^E. Before a path is created a check is performed to see if the pose of the two homogeneous matrices differ. If the two homogeneous matrices are the same no movement is required. The check is performed by subtracting the homogeneous matrix expressing the begin pose H_O^B from the homogeneous matrix expressing the end pose H_O^E. If the norm of the difference is of the homogeneous matrix is zero no movement occurs. The algorithm follows in algorithm 1.

Algorithm 1 Detecting if set-point is at the requested pose.

procedure
  if $\left|H_O^E - H_O^B\right| = 0$ **then**
    No movement occurs;
    $m = 0$
  else
    Movement occurs;
    $m = 1$

If movement is confirmed, the next check is to recognise if the motion can be described by a linear motion or a screw motion. For this purpose it is checked if there is a change of orientation between H_O^B and H_O^E. This is done by subtracting the two rotation matrices, which can be obtained from the homogeneous matrices. The rotation matrices can be expressed as the first 3 rows and the first 3 columns of the homogeneous matrix, $R_O^B = H_O^B(1:3, 1:3)$, $R_O^E = H_O^E(1:3, 1:3)$. For the purpose of saving the recognised movement two Boolean variables are used: wv, vv. The algorithm for recognising the movement is expressed as:

Algorithm 2 Procedure for deciding the sort of twist

procedure
  if $|(R_1 - R_2)| = 0$ **then**
    Linear motion
    wv = 0
    vv = 1
  else
    Screw motion
    wv = 1
    vv = 0

If the Boolean vv is true ($vv = 1$), the motion is linear, the Boolean wv is in that case false ($wv = 0$). If the motion is not linear, the motion can be described by a screw motion. In that case the Boolean variables become ($wv = 1$) and ($vv = 0$). From the previous sections methods to calculate

the parameters and variables $\hat{\mathbf{w}}$, $\hat{\mathbf{v}}$, $r(t)$ and λ. These inputs along with the Boolean variables wv and vv can be used to calculate the twist $\mathbf{T(t)}$. However, the motion profiles $v(t)$ and $w(t)$ are also required. The expression for the twist is expressed in equation (4.40):

$$\mathbf{w}(t) = w(t) * wv * \hat{\mathbf{w}} \tag{4.38}$$

$$\mathbf{v}(t) = \mathbf{r}(t) \wedge \mathbf{w} + \lambda * \mathbf{w} = A(t) * \mathbf{w} \tag{4.39}$$

With A(t) as the sum of the skew matrix r(t) with the addition of the the pitch (λ):

$$A = \widetilde{\mathbf{r}(t)} + \lambda * I_{3\times 3} \tag{4.40}$$

The values of matrix A are of a skewmatrix of vector $r(t)$ plus values of *lambda* multiplied by a 3×3 identity matrix. With this the $\mathbf{T}$(t) becomes:

$$\mathbf{T}(t) = m * \begin{pmatrix} wv * w(t) * \hat{\mathbf{w}} \\ \\ wv * w(t) * A * \hat{\mathbf{w}} + vv * v(t) * \hat{\mathbf{v}} \end{pmatrix} \tag{4.41}$$

In equation (4.41) two main checks are in place. The first one is m, this check prevents a twist from generating a motion if there is no desire for a motion. The second check are the two Boolean variables that allow the switch of twist in the expression of the twist. The motion profiles $w(t)$ and $v(t)$ determine the magnitude of the twists, the direction of the twist is determined by the unit twist. The next section elaborates on how the motion profile $v(t)$ and $w(t)$ are calculated.

4.2.6 Calculation of the homogeneous matrix as a function of twist

While calculating the homogeneous matrix $H(t)$ the homogeneous matrix is divided into a rotation component as can be expressed by equation (4.42).

$$H(t) = \begin{bmatrix} R(t) & p(t) \\ {0_3}^T & 1 \end{bmatrix} \tag{4.42}$$

According to [5], the calculation of the rotation matrix can be expressed as an exponential of a rotation. The expression for the rotation matrix is given as:

$$R(t) = e^{\widetilde{\mathbf{w}(t)}} \tag{4.43}$$

In which $\widetilde{\mathbf{w}}(t)$ is the skew symmetric matrix ($\widetilde{\mathbf{w}(t)}^T = -\widetilde{\mathbf{w}(t)}$) of the unit vector $\tilde{\mathbf{w}}$ multiplied with the motion profile $w(t)$ By differentiating both

sides the following expression is obtained.

$$\dot{R}(t) = \widetilde{\mathbf{w}(t)} e^{\widetilde{\mathbf{w}(t)}} \quad \dot{R} = \tilde{\mathbf{w}} e^{\tilde{\mathbf{w}}} \tag{4.44}$$

Here the right equation is the short version. After substituting (4.44) in (4.43) the equation for the rotation matrix becomes:

$$\dot{R} = \tilde{\mathbf{w}} R \tag{4.45}$$

By integrating the equation becomes:

$$R(t) = \int_{t_0}^{t} \widetilde{w(\tau)} R(\tau) \, \mathrm{d}\tau \tag{4.46}$$

With the initial value of $R(t_0) = R_O^B$ (from the homogeneous matrix H_O^B). In this equation the initial Rotation matrix is the initial pose of the path H_O^B. Previously, the calculation of $\mathbf{T}(t)$ consisted of the the velocity vector which was expressed as $\mathbf{v}$(t). To obtain the position vector required for the homogeneous matrix $H(t)$, the the velocity vector is integrated. The integration is component wise, so three integrals are taken:

$$p_i(t) = \int_{t_0}^{t} v_i(\tau) \, \mathrm{d}\tau + p_i(t_0) \tag{4.47}$$

with $i = 1, 2, 3$ the expression for $\mathbf{p}(t)$ and $\mathbf{v}(t)$ is given as:

$$\mathbf{p}(t) = \begin{bmatrix} p_1(t) \\ p_2(t) \\ p_3(t) \end{bmatrix} \tag{4.48}$$

$$\mathbf{v}(t) = \begin{bmatrix} v_1(t) \\ v_2(t) \\ v_3(t) \end{bmatrix} \tag{4.49}$$

The homogeneous matrix can be calculated by taking the exponent of the twist $\tilde{\mathbf{T}}$ according to [2]. A description of obtaining the exponential of a twist can be found in [4]. The calculation of the position is different in this method, because $\mathbf{T}(t)$ is a time variable twist. That is the reason that the position $\mathbf{p}(t)$ can be calculated by integrating the velocity part of the twist. If the homogeneous matrix was described by a normal twist $\mathbf{T}$ (which contains constant values at least for the unit twist), the homogeneous matrix can be calculated by taking the exponent of the twist. The rotation matrix $R(t)$ can with this description of a twist still be calculated in this way. This is because the direction of the the rotation part (which can be expressed in a unit twist) does not change during the motion. The direction of the rotation in a screw motion is constant, while the linear velocities change direction. This is why the rotation can still be calculated as indicated in equation 4.46, while the position can be calculated as an integral of the linear velocity.

4.2.7 Comparison of screw theory and the alternative interpretation.

Screw theory and the alternative interpretation both describe a screw motion. The different methods to calculate the homogeneous matrix were presented in this section. In this subsection, a comparison is made between the two methods described. Both descriptions describe a screw motion of a set-point which require two homogeneous matrices. The path of both descriptions is the same. Screw theory is a description where the parameters of the twist are not explicitly calculated. For the calculation of the homogeneous matrix by screw theory, no explicit expression for the parameters is necessary, though the parameters can easily be calculated. With the alternative interpretation, the parameters of the twist and the twist of a screw motion are calculated, without using a logarithmic function. In [4] an expression can be found that calculates the exponent of a twist. In the alternative interpretation a similar expression is used. There is no reason to use the alternative interpretation instead of screw theory for the goals stated at the start of this chapter motion, however this expression was used in this thesis. One interesting possibility that might be useful is that an additional expression of a time dependent twist $\mathbf{T}(t)$ is derived. With the expression of the twist as presented in this section, some special cases of motions can be described. An example is an elliptical motion by calculating r(t), from expression (4.32), in a different way, by scaling one of the vectors.

4.3 Soft motion profile

For a safe movement of the robotic arm, it is required to follow a certain motion profile that adheres to a predefined limit of velocity of the robotic arm. In the previous section, a method was presented to calculate twists by multiplying the unit twist with a motion profile. With the twists, a path can be calculated which is expressed in the form of a homogeneous matrix. In this section, a motion profile is presented for the purpose of calculating an appropriate twist. The motion profile can be interpreted as a function that expresses a velocity as a function of time. The unit twist can be considered as the direction of the velocities. The motion profile can be considered as the magnitude of the velocities. An acceptable motion profile is required for the motion, of the set-point, this is for the six dimensions of the twist. For the motion the position, velocity, acceleration and jerk are considered. There are multiple reasons for this. The maximum jerk relates to the wear of joints. Acceleration and velocity can be important for safety reasons. The object that is in the gripper of the robot can have properties of which limitations on velocity and acceleration are imposed. For example, while pouring water from a bottle into a glass the rotation of the bottle should be sufficiently slow to avoid spillage of water. Another example can be the movement of a cup of coffee and the limited accelerations, to avoid spillage. Or a screwdriver should have a high rotational velocity along the screwdriver, however it is not desired that screw driver has a high linear velocity, because the object could, unintentionally, become a stabbing weapon. A suitable motion profile that limits the velocity, acceleration and jerk is described by a soft motion profile. The soft motion profile as described in [7] limits the values of jerk, acceleration and velocity. Within these requirements, a path is generated that covers the distance of the path in the minimal amount of time. The distance is expressed in an angle (θ) or a euclidean distance (d), depending if the twist is expressed as a screw motion or a linear motion. The change of the velocities expressed in the twist can be expressed by an acceleration vector. The change of the acceleration vector, in turn, with a vector representing the jerk. The expression of the twist and the functions derived from it, are used for the calculation of a motion profile that can be used to create a twist with maximum values for velocity, acceleration and jerk. The profile of the velocity determines values for acceleration and jerk. It is possible to calculate a velocity profile with maximum values for velocity, acceleration and jerk. For this purpose, in the next subsection the motion profile is described.

4.3.1 Motion profile

The soft motion profile can be described by the function of the jerk of the motion. Jerk $[m/s^3]$ is a time derivative of the acceleration $[m/s^2]$.

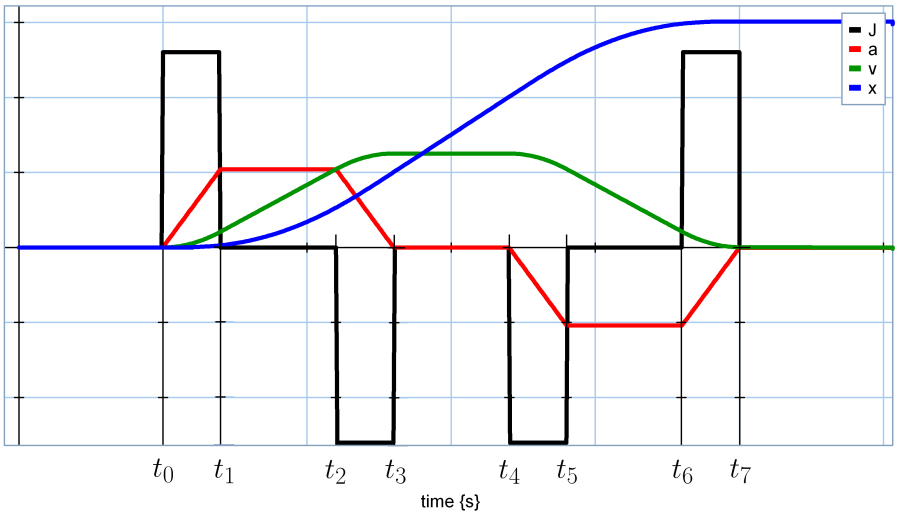

Figure 4.10: Soft motion: functions of jerk, acceleration, velocity and position.

The acceleration, velocity and position can be calculated by integrating the function of the jerk. The function of jerk consists of three constant values. The maximum jerk is characterized by a positive constant, J_{max}; if jerk is applied it is either as J_{max} or $-J_{max}$. The function of the jerk can also be zero. The jerk, acceleration, velocity and position are depicted in figure 4.10. In this figure the jerk is represented by the black profile. The acceleration is depicted with the red profile, the green profile represents the velocity and the position is depicted with the blue profile. The motion profile that is depicted in green above is divided in 7 parts. For this purpose a time period δ is introduced, which is defined as:

$$\delta_{t_{k+1}t_k} = t_{k+1} - t_k \tag{4.50}$$

For example the first time period is:

$$\delta_{t_0t_1} = t_1 - t_0; \tag{4.51}$$

Assuming that the maximum and minimum jerk are equal, and the acceleration and deceleration are equal, the time period of maximum and minimum jerk is:

$$\delta_{t_0t_1} = \delta_{t_2t_3} = \delta_{t_4t_5} = \delta_{t_6t_7} = T_j \tag{4.52}$$

In the time periods $\delta_{t_0t_1}$ and $\delta_{t_6t_7}$ positive maximum jerk is applied. In the time periods $\delta_{t_2t_3}$ and $\delta_{t_4t_5}$ negative maximum jerk is applied. After the first time period $\delta_{t_0t_1}$ the maximum acceleration is reached. This time period is expressed as:

$$\delta_{t_1t_2} = \delta_{t_5t_6} = T_a \tag{4.53}$$

The maximum acceleration is maintained throughout the time period $\delta_{t_1 t_2}$. After the fifth time period $\delta_{t_4 t_5}$ the maximum deceleration is reached, the maximum deceleration is maintained throughout the time period $\delta_{t_5 t_6}$. At time t_3 the maximum velocity is reached which is maintained through the time period $\delta_{t_3 t_4}$:

$$\delta_{t_4 t_5} = T_v \tag{4.54}$$

The functions for jerk, acceleration and velocity are described by the following mathematical equations for the seven parts of the function. The functions express the jerk, acceleration and velocity of the soft motion profile. To indicate which part of the part of the function is described, the seven parts are indicated by a subscript in the equations underneath.

$$\begin{aligned}
J_1(t) &= J_{max}; & v_4(t) &= v_3(t_3) \\
a_1(t) &= \int_{t_0}^{t} J_1(\tau)d\tau & J_5(t) &= -J_{max} \\
v_1(t) &= \int_{t_0}^{t} a_1(\tau)d\tau & a_5(t) &= a_4(t_4) + \int_{t_4}^{t} J_5(\tau)d\tau \\
J_2(t) &= 0 & v_5(t) &= v_4(t_4) + \int_{t_4}^{t} a_5(\tau)d\tau \\
a_2(t) &= a_1(t_1) & J_6(t) &= 0 \\
v_2(t) &= v_1(t_1) + \int_{t_1}^{t} a_2(\tau)d\tau & a_6(t) &= a_5(t_5) \\
J_3(t) &= -J_{max} & v_6(t) &= v_5(t_5) + \int_{t_5}^{t} a_6(\tau)d\tau \\
a_3(t) &= a_2(t_2) + \int_{t_2}^{t} J_3(\tau)d\tau & J_7(t) &= J_{max} \\
v_3(t) &= v_2(t_2) + \int_{t_2}^{t} a_3(\tau)d\tau & a_7(t) &= a_6(t_6) + \int_{t_6}^{t} J_7(\tau)d\tau; \\
J_4(t) &= 0 & v_7(t) &= v_6(t_6) + \int_{t_6}^{t} a_7(\tau)\mathrm{d}\tau; \\
a_4(t) &= 0
\end{aligned} \tag{4.55}$$

For reasons of brevity the functions for jerk acceleration and velocity are not elaborated here. The time periods T_j, T_a, T_v can be calculated through an algorithm presented in the next subsection. The maximum values for jerk, acceleration and velocity as well as the distance are used to calculate the soft motion profile.

4.3.2 Algorithm for soft motion profile.

For the soft motion profile time periods, T_v, T_a and T_j are required. To calculate the time periods T_j, T_a and T_v, three cases are considered. The choice between the cases depend on the traversed distances. The traversed distances can be calculated with (J_{max}, A_{max} and V_{max}). In [7] limiting conditions are given namely:

- Condition 1: V_{max} is reached, this means that A_{max} is reached as well. This is because if there is a limited jerk, and there is a limited velocity the acceleration cannot be infinitely high. The traversed distance is given by:

$$T_j = T_{jmax} \qquad T_a = T_{amax} \qquad T_v = 0 \tag{4.56}$$

  With:

$$D_{thr1} = \frac{A_{max}V_{max}}{J_{max}} + \frac{V_{max}^2}{A_{max}} \tag{4.57}$$

- Condition 2: here only A_{max} is reached

$$T_j = T_{jmax} \qquad T_a = 0 \qquad T_v = 0 \tag{4.58}$$

$$D_{thr2} = 2\frac{A_{max}^3}{J_{max}^2} \tag{4.59}$$

The values for T_{jmax} and T_{amax} can be calculated by:

$$T_{jmax} = \frac{A_{max}}{J_{max}} \qquad T_{amax} = \frac{V_{max}}{A_{max}} - \frac{A_{max}}{J_{max}} \tag{4.60}$$

In the first condition it is mentioned that if V_{max} is reached A_{max} is reached as well. In this it is implicitly assumed that the given values for A_{max} is reached and that it is limited. The calculation of an appropriate value for A_{max} is elaborated in the next section Adaptation of the algorithm. The calculated distances D_{thr1} and D_{thr2} are used in the algorithm to calculate the time periods T_v, T_a and T_v. The algorithm follows:

Algorithm 3 Time periods for a soft motion profile

procedure
  if $D >= D_{thr1}$ **then**
    $T_j = T_{jmax}$
    $T_a = T_{amax}$
    $T_v = \frac{D - D_{thr1}}{V_{max}}$
  else
    if $D_v >= D_{thr2v}$ **then**
      $T_v = 0$
      $T_j = T_{jmax}$
      $T_a = \sqrt{\frac{{A_{max}}^2}{4 * J_{max}} + \frac{D}{A_{max}}} - \frac{3A_{max}}{2J_{max}}$

    else
      $T_v = 0$
      $T_a = 0$
      $T_j = \sqrt[3]{\frac{D}{2J_{max}}}$
  $T_f = 4 * T_j + 2 * T_a + T_v$
=0

The three cases are determined by comparing the distance of the path with the two traversed distances. These distances determine the calculation of the time periods required for the motion profile. The parameters required for the algorithm are a distance (D) and values for the maximum velocity, acceleration and jerk; V_{max}, A_{max} and J_{max}. For the application of calculating the twist the distance is expressed as an angle in case of a screw motion and a euclidean distance in case of a linear motion.

4.3.3 Adaptation of the algorithm.

The time periods of the jerk T_{jmax} and the acceleration T_{amax} are calculated with the maximum values of the velocity, acceleration and jerk. In the first condition it is mentioned that if the maximum velocity is reached, the maximum acceleration is reached as well. If the values of the acceleration exceeds the maximum value implicated in the first condition, the values of T_{jmax} and T_{amax} can assume incorrect values. This can result in a motion with undesired behavior. Observed behavior include a discontinuous motion profiles and other irregularities. There is a maximum value of acceleration, which is expressed as a limit of A_{max} notated as $A_{max,limit}$. In this subsection this limit is calculated and an algorithm is presented to determine the value of $A_{max,limit}$.

$A_{max,limit}$ can be calculated by using the values for maximum jerk and maximum velocity. If the maximum values for J_{max} and V_{max} are known, the limit of A_{max} can be calculated. The minimal time to achieve the maximum

constant velocity is by applying a period of maximum jerk directly followed by a period of negative maximum jerk, without a constant acceleration. The time period for both periods for the jerk have the same length. If no constant acceleration occurs from the soft motion profile it can be seen that the second part of the function does not exist. This means that the time period is $\delta_{t_{12}} = 0$. The expression in terms of $\delta_{t_{12}}$ is $t2 = t1 + \delta_{t_{12}}$ which results in $t2 = t1$. At that point the velocity can be expressed in term of the maximum velocity:

$$v(t_1) = v(t_2) = 0.5 * V_{max} \tag{4.61}$$

Where V_{max} is the maximum velocity of the profile, and $v(t1)$ and $v(t2)$ are the velocity at that time point, such as was presented in figure 4.10. The equation for $v(t)$ can be expressed as:

$$v(t_1) = 0.5 * c * {t_1}^2 \tag{4.62}$$

With c being the maximum value of the jerk. For simplicity $t_0 = 0$, than the period of maximal applied jerk, $t_1 - t_0 = t_1 - 0 = t_1$. The value of c and the value of t_1 can be be given as:

$$c = J_{max} \qquad t_1 = T_j = A_{max}/J_{max} \tag{4.63}$$

Equation (4.61) and (4.62) can be used to derive equation (4.64):

$$0.5 * V_{max} = 0.5 * c * {t_1}^2 \tag{4.64}$$

The values of c and t_1 are substituted in the right member of equation (4.64), with which equation (4.65) is obtained:

$$0.5 * V_{max} = 0.5 * J_{max} * \left(\frac{A_{max}}{J_{max}}\right)^2 \tag{4.65}$$

Which can be simplified to:

$$V_{max} = J_{max} * \left(\frac{A_{max}}{J_{max}}\right)^2 \tag{4.66}$$

(4.66) can be rewritten to:

$$A_{max} = J_{max} * \sqrt{\frac{V_{max}}{J_{max}}} = \sqrt{J_{max} V_{max}} \tag{4.67}$$

Which is the maximum value for A_{max}.

$$A_{max,limit} = \sqrt{J_{max} V_{max}} \tag{4.68}$$

If $A_{max,limit}$ is smaller than the given $A_{max,initial}$, the initial value for A_{max} is changed into the calculated maximum $A_{max,limit)}$. In terms of algorithm this is written as: The calculation of the maximum acceleration After implementation of the calculation of the maximum value of A_{max} the undesired behavior did not occur anymore.

Algorithm 4 Calculation of A_{max}

procedure
$A_{max,limit} = \sqrt{J_{max} V_{max}}$
if $A_{max,initial} > A_{max,limit}$ **then**
$A_{max} = A_{max,limit}$
else
$A_{max} = A_{max,initial}$

4.3.4 Implementation of the algorithm in the software

The parameters calculated in this subsection are used to generate a motion profile. For the implementation in 20-sim a motion profile exists that is called a partial cubical. This profile is similair to a soft motion profile. The parameters are not the same, therefore a calculation is performed to calculate the parameters that are required for the 20-sim software. The algorithm described before can obtain the correct time periods of the constant velocity(T_v), constant acceleration (T_a), and constant jerk(T_j). For the algorithm the parameters are given as:

- Start time: t_{start}. This is the moment in time at which the motion starts.
- Stop time: t_{stop}. The time at which the motion ends, calculated by adding the start time to the the final time at which the motion ends. $t_{stop} = t_{start} + Tf$
- Stroke: (D); The distance that is traveled. In case of a linear motion this is the Euclidian distance between $p1$ and $p2$, in case a screw motion this is the angle θ.
- $CV = Tv/Tf$. The time ratio between the time that the velocity is constant and the total time.
- $CA = Ta/Tf$. The time ratio between the time that the acceleration is constant and the total time.

4.3.5 Parameter values for the soft motion profile algorithm

Previously it was mentioned that the parameters required for the calculation of the soft motion profile are, the distance and, the maximum values for velocity, acceleration, and jerk. In case of the of a linear motion, the distance is the Euclidean distance between the initial pose and the final pose of the path. In case of a screw motion the distance is represented by the traveled angle θ. The subject of this section is the calculation of the parameters for the motion profile. The required parameters are V_{max},A_{max} and J_{max}.

For the calculation of the twist two cases were formulated previously. If no change in orientation occurs between one pose and the next pose, a linear path is created.For all motions in which a rotation occurs, the motion is described by a screw. A special case of a screw motion is the pure rotation. In this motion the position of the set-point does not change. The pure rotation can be described by an unit rotation vector ($\hat{\mathbf{w}}$) and a motion profile ($w(t)$). A linear motion can be described by an unit vector ($\hat{\mathbf{w}}$) and a motion profile ($v(t)$). The twist of the linear motion and the pure rotation are different descriptions. However the two motions have in common that they can be described by a unit vector and a motion profile. The maximum values of velocity, acceleration and jerk can be calculated in three directions. This why the parameters for these motions can be calculated similar fashion.
The parameters of the screw motion are calculated in a different fashion. For the screw motion only the maximum linear velocity and the maximum rotational velocity are considered.

Calculation of maximum values for the twist: pure rotations and linear motion

A pure rotation is a rotation without translation, linear motion is a motion without rotation. In these cases of motion, three directions can be considered, in a simple way. The direction of the linear motion can be considered in a velocity vector v. The direction of the rotation is described by the rotational velocity vector w. This is represented as:

$$\mathbf{w} = w(t) \cdot \hat{\mathbf{w}} \tag{4.69}$$

For the calculation of the maximum value of the motion profile w(t) the expression is rewritten as:

$$\mathbf{w}_{max} = \alpha \cdot \hat{\mathbf{w}} \tag{4.70}$$

which can be expressed as:

$$\begin{pmatrix} w_{max_x} \\ w_{max_y} \\ w_{max_z} \end{pmatrix} = \alpha \cdot \begin{pmatrix} \hat{w}_x \\ \hat{w}_y \\ \hat{w}_z \end{pmatrix} \tag{4.71}$$

In this, $w_{max_x}, w_{max_y}, w_{max_z}, \hat{w}_x, \hat{w}_y, \hat{w}_z$ are scalar values. The maximum scalar values are the given maximum rotational velocity in three directions. The parameter α represents the maximum value for the motion profile. For the purpose of calculating the value of α, it is expanded to a vector. Such that the equation becomes:

$$\begin{pmatrix} w_{max_x} \\ w_{max_y} \\ w_{max_z} \end{pmatrix} = \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix} \cdot \begin{pmatrix} \hat{w}_x \\ \hat{w}_y \\ \hat{w}_z \end{pmatrix} \tag{4.72}$$

Here the maximum rotational velocities are taken as the inproduct of the α vector and the unit vector $\hat{\mathbf{w}}$. Because it is a inproduct α_1, α_2 and α_3 can be calculated as:

$$\begin{pmatrix}\alpha_1\\ \alpha_2\\ \alpha_3\end{pmatrix} = \begin{pmatrix}\frac{w_{max_x}}{\hat{w}_x}\\ \frac{w_{max_y}}{\hat{w}_y}\\ \frac{w_{max_z}}{\hat{w}_z}\end{pmatrix} \tag{4.73}$$

The minimal value of α_1, α_2 and α_3 determines the value for α, because the velocity, acceleration and jerk in one direction determines the values in other directions. The vector $\hat{\mathbf{w}}$ is unit vector, this means that $|\mathbf{w}_{max}| = \alpha$

$$\alpha = max(\alpha_1, \alpha_2, \alpha_3) \tag{4.74}$$

The value of α is the maximum rotational velocity. This procedure can be performed for every linear motion and pure rotational motion, for velocity, acceleration and jerk.

Calculation of maximum values for the twist: screw motion

For a screw motion the parameters for the soft motion profile can be derived by calculating the maximum rotational velocity. The linear velocity v is calculated through w as can be seen in the description of a twist. In vectors this can be expressed as:

$$\mathbf{v} = \mathbf{v}_t + \mathbf{v}_r \tag{4.75}$$

$$\mathbf{v} = \mathbf{r} \wedge \mathbf{w} + \lambda * \mathbf{w} \tag{4.76}$$

In order to calculate the absolute velocity of the motion the absolute values are calculated.

$$|\mathbf{v}| = |\mathbf{v}_t + \mathbf{v}_r| \tag{4.77}$$

$$|\mathbf{v}| = |\mathbf{r} \wedge \mathbf{w} + \lambda * \mathbf{w}| \tag{4.78}$$

The length of the vectors v_t and v_r expressed in $|\mathbf{v}_t|$ and $|\mathbf{v}_r|$ are can be written as scalar multiplications:

$$v_t = |\mathrm{r} \wedge \mathbf{w}| = r * w \tag{4.79}$$

$$v_r = |\lambda * \mathbf{w}| = \lambda * w \tag{4.80}$$

With v_t, v_r, r, w scalar values. The vectors $\mathbf{v}_t$ and $\mathbf{v}_r$ are perpendicular vectors, becaus of this the total velocity can be calculated with Pythagoras:

$$v^2 = {v_t}^2 + {v_r}^2 = (\lambda * w)^2 + (r * w)^2 \tag{4.81}$$

$$v^2 = \lambda^2 * w^2 + r^2 * w^2 \tag{4.82}$$

$$v = \sqrt{\lambda^2 w^2 + r^2 w^2} = \sqrt{(\lambda^2 + r^2) w^2} = \sqrt{\lambda^2 + r^2} * w \tag{4.83}$$

With this the linear velocity is related to the rotational velocity. To calculate the maximum rotational velocity as a function of the linear velocity.

$$w = \frac{v}{\sqrt{\lambda^2 + r^2}} \tag{4.84}$$

$$w_{max,vmax} = \frac{v_{max}}{\sqrt{\lambda^2 + r^2}} \tag{4.85}$$

The minimal value for the rotational velocity is obtained by taking the minimum of both.

$$w_{max} = min(w_{max,v_{max}}, w_{max,w_{max}}) \tag{4.86}$$

The minimal value is taken as value for the maximum rotational velocity. For the maximum values for acceleration and jerk similar calculations are performed. These calculations guarantee maximum values for both rotational velocity and linear velocity of the set-point.

4.3.6 Coupling paths

In the previous section a way to calculate the homogeneous matrix $H(t)$ was presented. In this section a description of soft motion profile was given. The soft motion profile generates a path that ensures an acceptable motion of the set-point. With this the end-effector can be steered to any pose in space, if the joint space allows it. For a useful application of this implementation, it is desired that the robotic arm is able to steer from pose (H_O^n) to the next pose (H_O^{n+1}) and from (H_O^{n+1}) to (H_O^{n+2}) and so on. An application is desired in which paths are coupled according to a list of homogeneous matrices. For this purpose the paths are coupled. The coupling of paths is achieved by multiplying the calculated homogeneous matrices $H_i(t)$ with a step functions. A more detailed description more can be found in appendix A.

4.4 Validation of the motion set-point generator

The homogeneous matrix $H(t)$ is created as an input for the impedance controller. The homogeneous matrix represents a set-point which follows a path, with certain maximum velocities, accelerations and jerks both in rotational and linear direction. In this section the motion set-point generator is tested, through simulations. In the simulations, the paths are generated and the behavior of the set-point is recorded. This tests the design as described in this chapter.Three general cases for calculating parameters for the motion profile were indicated in the previous section. These were, linear motion, screw motion and pure rotation. For these motions, simulations were performed. In the simulation, the position and velocity are recorded.

The expression of the rotation is recorded by interpreting the rotation matrix with the axis-angle representation. The representation consists of the unit vector $\hat{\mathbf{w}}$ which represents the direction of the rotation. And $\theta(t)$, which is the traveled angle. By differentiating $\theta(t)$, the rotational velocity $w(t)$ is obtained. The rotation matrix $R(t)$ can be expressed in the axis-angle representation by comparing $R(t)$ with another rotation matrix on its path. The start orientation of the path expressed in rotation matrix R_O^B is used for this purpose. In this way the rotation and rotation axis can be calculated. With the angle θ and rotation axis expressed as unit vector $\hat{\mathbf{w}}$ and the start of the path these expressions represent the rotation matrix. A plot of the rotational velocity $w(t)$ is calculated through the differentiation of the angle θ . For the rotation in the screw motion the expectation is that the vector $\hat{\mathbf{w}}$ consists of constant values, because the direction of the rotation is fixed. The position of the set-point is plotted. The velocity of the set-point is calculated by differentiating the position. The homogeneous matrix that is the output of the motion set-point generator is represented by the rotation matrix $R(t)$ and the Cartesian coordinates expressed in $\mathbf{p}$. The choice to express the rotation matrix in the axis angle representation is that it is a more compact and insightful representation. For the simulations, an example is used with different motions connected to each other as described in Apendix A. In the next subsection a description is given of the different poses and paths, for which simulations were performed.

4.4.1 Coupled paths

In this section different motions are described. All motions are calculated by using the begin pose and end pose which are expressed in homogeneous matrices. An example was used in which six poses were chosen, expressed in homogeneous matrices. Between the six poses 5 paths were generated. The chosen example consists of a series of motions that could represent a robotic arm navigating toward a bottle of water, grabbing it and pouring it's content. The example contains the different motions that were described in this chapter. These motions were, linear motion, pure rotation and screw motion. The homogeneous matrices are for simplicity numbered in chronological order. The homogeneous matrices are with respect to the global reference frame and expressed in the coordinates of the global reference frame, which is the base (shoulder) of the robotic arm.

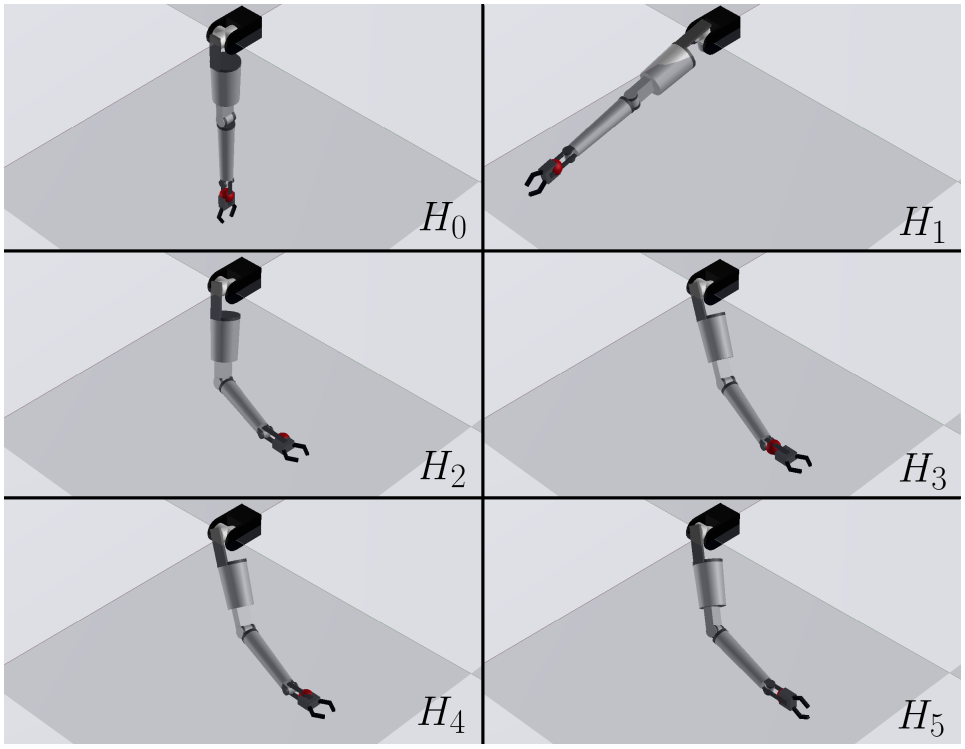

Figure 4.11: Poses of the PRA, for a motion grabbing a bottle.

The six poses that are used to generate the paths are depicted in figure 4.11 above. The 5 motions are described by the poses listed underneath:

- Motion 1: $H_0 - H_1$ is a screw motion. This screw motion is special, because the pitch λ is zero. In this case, values for the rotational and linear velocity were chosen to illustrate that the rotational velocity is limited by the linear velocity.

- Motion 2: $H_1 - H_2$ is a screw motion from pose H_1 to H_2. In this case the rotational velocity is the limiting factor.
- Motion 3: $H_2 - H_3$ a linear motion from H_2 to H_3. This motion is toward the supposed pose of the bottle. After this motion, the open gripper of the end-effector would close, effectively grabbing the bottle.
- motion 4: $H_3 - H_4$ The bottle is lifted and moved in a linear motion toward a pose where it will be rotated. The bottle is lifted a small distance, so that it will not touch the table it is standing on.
- motion 5: $H_4 - H_5$ This motion is the rotation of the bottle. If the bottle is open, effectively pouring its content. This is a pure rotation, such that the rotational motion from H_4 to 5 is limited with the maximum values of for the rotation velocity in multiple directions.

In the next subsection the generated path for the motion 1,2,4 and 5 are validated by simulating the generation of these paths. The expression of the homogeneous matrices is given as:

$$H_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -0.65 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad H_1 = \begin{pmatrix} 0.2334 & 0 & -0.9724 & 0.632 \\ 0 & 1 & 0 & 0 \\ 0.9724 & 0 & 0.2334 & -0.1517 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$H_2 = \begin{pmatrix} 1 & 0 & 0 & 0.15 \\ 0 & 0 & -1 & 0.4 \\ 0 & 1 & 0 & -0.32 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad H_3 = \begin{pmatrix} 1 & 0 & 0 & 0.15 \\ 0 & 0 & -1 & 0.46 \\ 0 & 1 & 0 & -0.32 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$H_4 = \begin{pmatrix} 1 & 0 & 0 & 0.15 \\ 0 & 0 & -1 & 0.5 \\ 0 & 1 & 0 & -0.3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad H_5 = \begin{pmatrix} 0 & -1 & 0 & 0.15 \\ 0 & 0 & -1 & 0.5 \\ 1 & 0 & 0 & -0.3 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

4.4.2 Simulations for the validation of the motion set-point generator

The motions described by the motion set-point generator, guarantees two points:

- The path of the set-point starts at a pose expressed in the initial homogeneous matrix. The path ends at the requested pose expressed in the final homogeneous matrix. The motion is either described as a linear motion or a screw-motion.
- The motion profile of the set-point is guaranteed to not exceed given values for, jerk, acceleration and velocity.

The values for the maximum velocity and rotational velocity are calculated. The maximum values of the rotation of the screw motion, determine the velocity as was established in section 4.2. Therefor, the velocities are calculated as was presented in the calculation of the parameters for the motion profile section 4.3. In the simulations, the values of the homogeneous matrices can be differentiated only once, because software could not perform differentiation of not smooth functions. Because of this, only the position and velocity are recorded. However, the calculation of the maximum values of the jerk and acceleration are performed in similar fashion. With this it can be assumed that if the velocity behaves as expected, the acceleration and jerk do as well. Now follow the different simulations of the motions. The simulations are represented in graphs depicting the different variables as mentioned before. Per simulation a calculation is performed to determine the maximum values, and to check if they hold.

Motion 1: Screw-motion.

The motion from H_0 to H_1 is a screw motion. The set values for the maximum rotational velocity and the maximum value for the linear motion are respectively $w_{max} = 1[rad/s]$ and $v_{max} = 0.2[m/s]$. According to the calculation as performed in section 4.2 the rotational maximum velocity is given by equation (4.85)

$$w_{max,v_{max}} = \frac{v_{max}}{\sqrt{\lambda^2 + r^2}} \tag{4.87}$$

In this case $\lambda = 0$ and $r = 0.65$ the values of the parameters of the screw were calculated in the section 4.2. This results in the following value for the maximum value for the rotational velocity as a function of the linear maximum velocity, $w_{max,v_{max}}$

$$w_{max,v_{max}} = \frac{v_{max}}{\sqrt{0.65^2}} = \frac{v_{max}}{0.65} = \frac{0.2}{0.65} \approx 0.3077 \tag{4.88}$$

Now taking the minimum value for the rotational velocity from (4.88)

$$w_{max} = min(w_{max,v_{max}}, w_{max,w_{max}}) = min(0.3077, 1) = 0.3077 \tag{4.89}$$

Which is the maximum value for the rotational velocity, which guarantees that the linear velocity is not exceeding the given maximum.

Results

The results of the simultion are illustrated in the 4 figures that are depicted in this section. The 4 figures, illustrate the position, the rotation, the velocity and the rotational velocity of the set-point. The position over time of the set-point is depicted in figure 4.12.

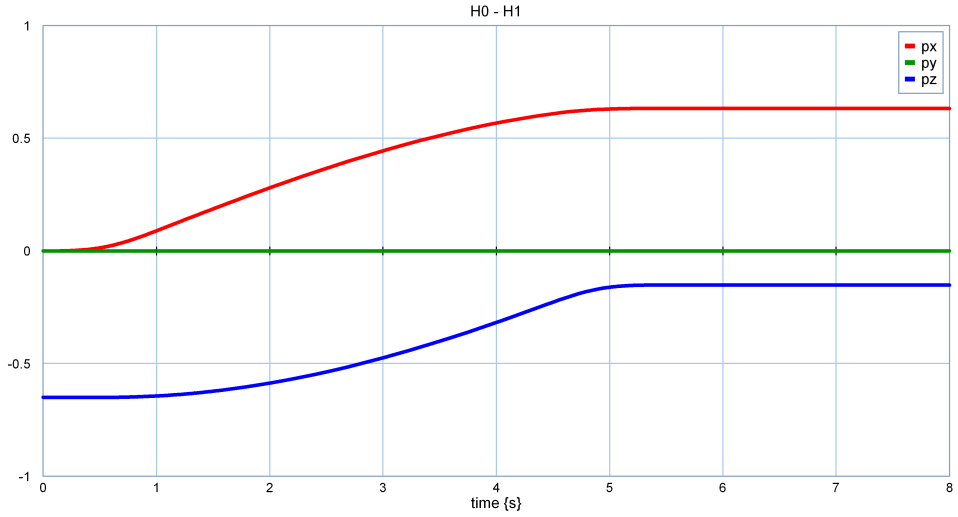

Figure 4.12: Positions of the set-point.

The begin and end values of the coordinate expressing the position in figure 4.12 are according to the homogeneous matrices H_0 and H_1 which were previously expressed. The rotation of the set point is depicted in figure 4.13.

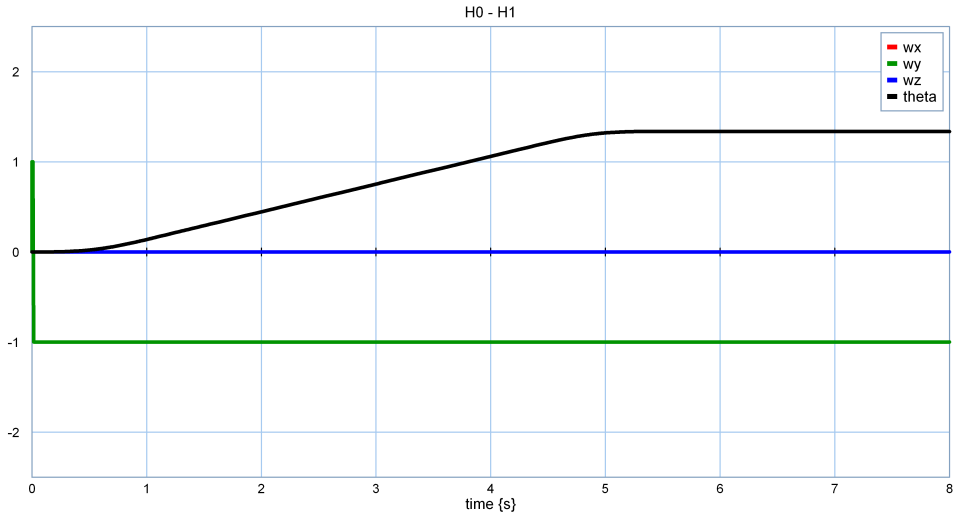

Figure 4.13: Direction of the rotation and angle θ

In this figure the angle θ is given as a function of time. The rotation axes are constant, as expected. Though due to start phenomena the value at zero is not -1. The rotation of the set-point is around the y direction, this is the reason that the values for x and z are zero. The norm of the three directions is always one, as it is in this case. The negative value indicates that the arm

does rotate around y from x axis to z axis. Because the positive value would be a rotation from z axis to x axis. As such, the value of -1 is correct.

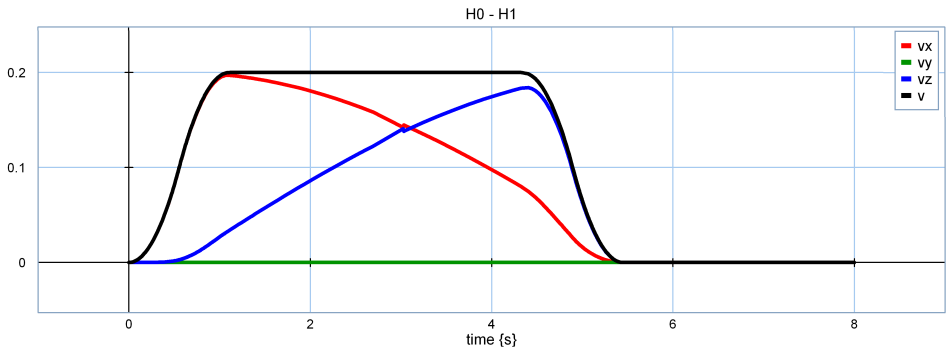

Figure 4.14: Velocity of the set-point.

In figure 4.14 the linear maximum velocity was reached but not exceeded, this is in accordance with previous calculations. A small irregularity can be noted at the crossing of velocity vx and vy in figure 4.14 Another irregularity can be noted in figure 4.15 at the beginning of the plot. Because the positions of this motion do not appear to have these errors, it is assumed that the irregularities are due to differentiation errors.

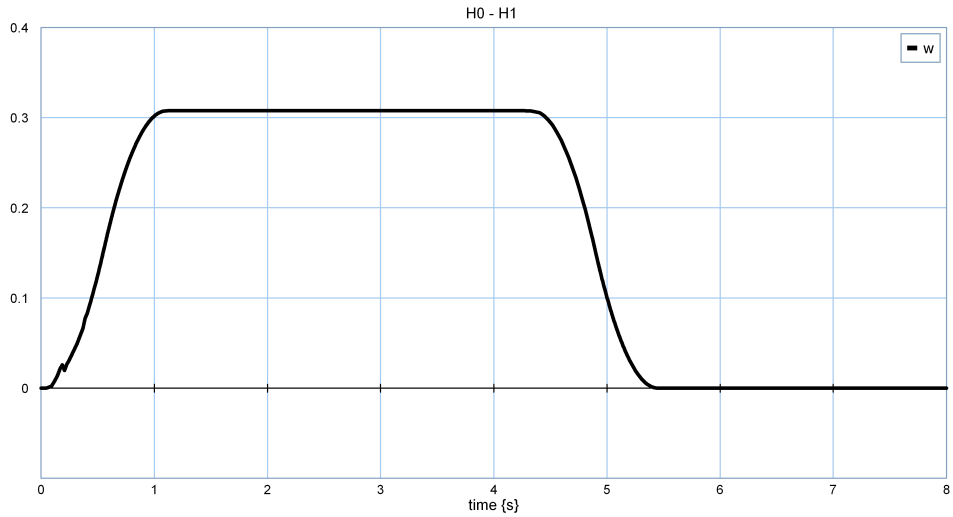

Figure 4.15: Rotational velocity of the screw motion.

The maximum rotational velocity $w_{max} = 1$ that was set, was not reached. The maximum rotational velocity was instead calculated to be 0.3077 $[rad/s]$, this rotational velocity holds.

Motion 2: Screw-motion.

The maximum value for the rotational velocity for this motion is calculated in similar fashion as the first motion. However in this motion values for the set-point are chosen in which the rotational velocity is determined by the maximum of the rotational velocity. The chosen values are $w_{max} = 0.5[rad/s]$ for the maximum value for the rotational velocity and $v_max = 2[m/s]$ for the maximum value for the linear velocity. The rotational maximum velocity as a function of the linear velocity is given by:

$$w_{max,vmax} = \frac{v_{max}}{\sqrt{\lambda^2 + r^2}}$$

In this case $\lambda = -0.1018$ and $r = 0.371$ the values of the parameters of the screw were calculated as is described in section 4.2. This results in the following value for the maximum value for the rotational velocity as a function of the linear maximum velocity, that is $w_{max,v_{max}}$.

$$w_{max,v_{max}} = \frac{v_{max}}{\sqrt{-0.1018^2 + 0.371^2}}$$

$$= \frac{2}{\sqrt{-0.1018^2 + 0.371^2}} = \frac{2}{0.1480} \approx 5.1987 \tag{4.90}$$

Now taking the minimum value of the calculated rotational velocities.

$$w_{max} = min(w_{max,v_{max}}, w_{max,w_{max}}) = min(5.1987, 0.5) = 0.5 \tag{4.91}$$

The maximum value for rotational velocity of the screw is calculated. The results are given in the following section.

Results

The results are given in 4 figures. First follows the result of the position of the set-point. In figure 4.16

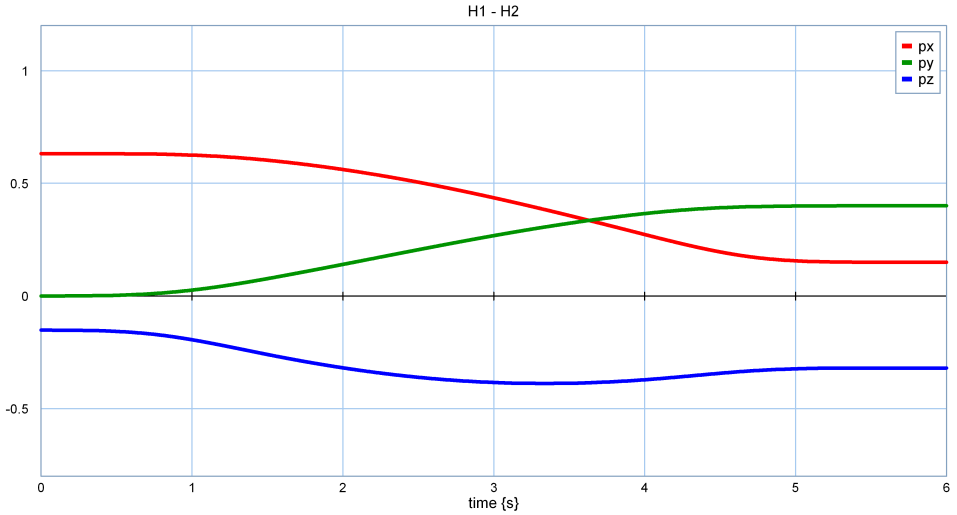

Figure 4.16: Positions of the set-point.

The begin and final values of the x,y and z coordinates are in accordance with H_1 and H_2. Now follows the result of the rotation matrix, expressed with the axis-angle representation.

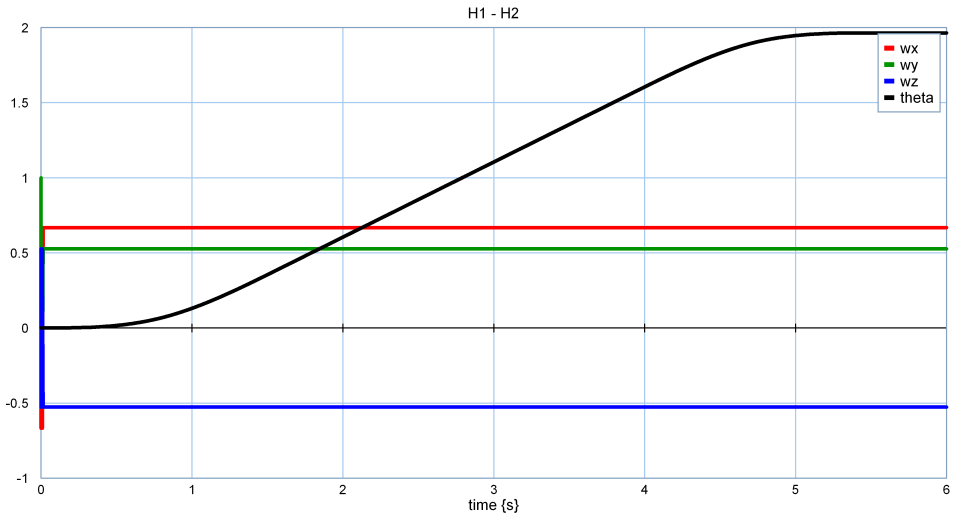

Figure 4.17: Direction of the rotation and angle θ.

The direction of the rotation is expressed in unit vector which are constant as expected. In comparison to the previous rotation the rotation axis points in a direction that is not aligned with one of the standard bases of the global reference frame. The angle is a little over $\pi/2$ which is correct. In the calculations it followed that the linear maximum velocity was not reached. In figure 4.18 the value of the velocity is clearly less then 2 $[m/s]$ that was

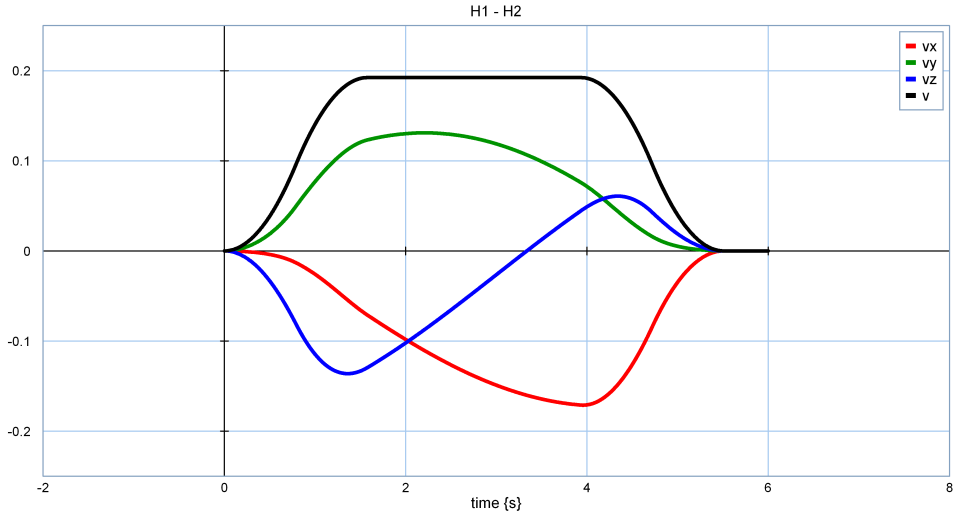

Figure 4.18: Velocity of the set-point.

given. The maximum velocity was not reached, because it was limited by the rotational velocity. The maximum rotational velocity is the calculated 0.5 $[rad/s]$. In figure 4.19 the profile of the rotational velocity is depicted. The maximum velocity and the maximum rotational velocity are not exceeded. The maximum rotational velocity is reached, because in this path the maximum rotational velocity determined from the maximum value for the rotational velocity.

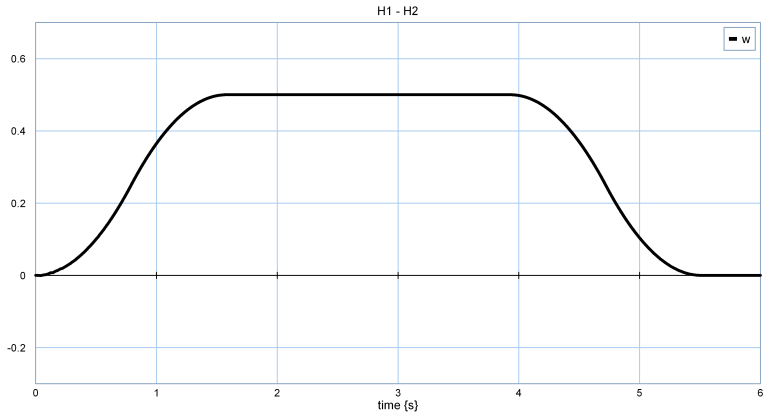

Figure 4.19: Rotational velocity

Motion 3: Linear motion

The third motion is a linear motion in one dimension. The calculation of the maximum value of the velocity is straightforward, that's why its not elaborated here. Motion 4 illustrates an example involving the calculation of maximum value for the linear motion.

Motion 4: Linear motion.

The linear motion above was a motion in the y-direction. The motion in the direction from H_3 to H_4 is a linear motion in two directions namely the y-direction and the z-direction. This motion illustrates how the maximum value for the linear velocity is calculated. To this end the values for the velocity in three directions are given. And the direction of the velocity as the unit vector

$$v_{max} = \begin{pmatrix} 0.03 \\ 0.06 \\ 0.02 \end{pmatrix} \qquad \hat{\mathbf{v}} = \begin{pmatrix} 0 \\ 0.8944 \\ 0.4472 \end{pmatrix} \tag{4.92}$$

The calculation of the maximum value for the velocity is performed considering the direction of the velocity. To this end equation 4.72 is used, however instead of rotational velocity w the linear velocity v is used:

$$\begin{pmatrix} v_{max_x} \\ v_{max_y} \\ v_{max_z} \end{pmatrix} = \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix} \cdot \begin{pmatrix} \hat{v}_x \\ \hat{v}_y \\ \hat{v}_z \end{pmatrix} \tag{4.93}$$

Here the maximum rotational velocities are taken as the inproduct of the α vector and the unit vector $\hat{\mathbf{w}}$. Because it is a inproduct α_1, α_2 and α_3 can be calculated as: Calculating the values for α_1, α_2, α_3.

$$\begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix} = \begin{pmatrix} \frac{v_{max_x}}{\hat{v}_x} \\ \frac{v_{max_y}}{\hat{v}_y} \\ \frac{v_{max_z}}{\hat{v}_z} \end{pmatrix} = \begin{pmatrix} \frac{0.03}{0} \\ \frac{0.06}{0.8944} \\ \frac{0.02}{0.4472} \end{pmatrix} = \begin{pmatrix} \infty \\ 0.0671 \\ 0.0447 \end{pmatrix} \tag{4.94}$$

The minimal value of α_1, α_2 and α_3 determines the value for α, because the velocity, acceleration and jerk in one direction determines the values in other directions. In the final equation alpha is calculated. Recall that $\hat{\mathbf{v}}$ is a normalised vector this means that $|\mathbf{v}_{max}| = \alpha$. And as such $v_{max} = \alpha$

$$v_{max} = \alpha = min(\alpha_1, \alpha_2, \alpha_3) = min(\infty, 0.0671, 0.0447) = 0.0447 \tag{4.95}$$

This is the maximum value for the velocity. The velocity in the y direction is limited by the velocity in the z direction. This can be seen in the results.

Results

The function of position of the set-point in this motion is depicted in figure 4.20.

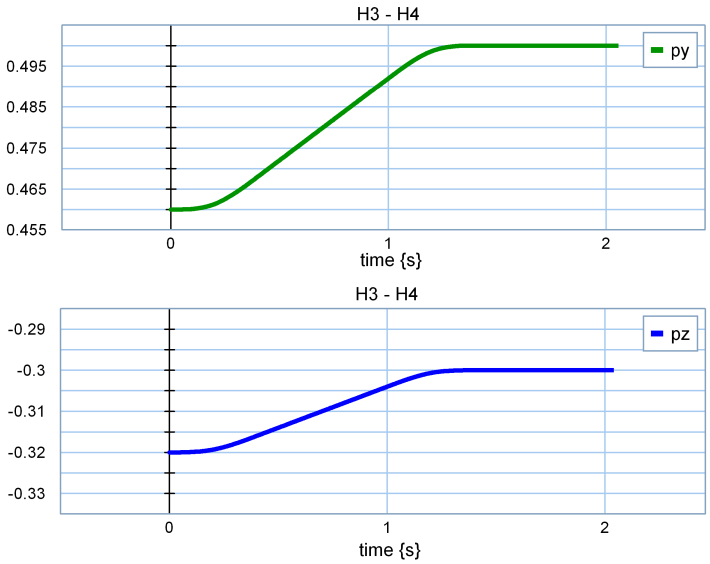

Figure 4.20: Positions of the set-point.

The positions are in accordance with the given homogeneous matrices. In figure 4.21 the velocity is depicted.

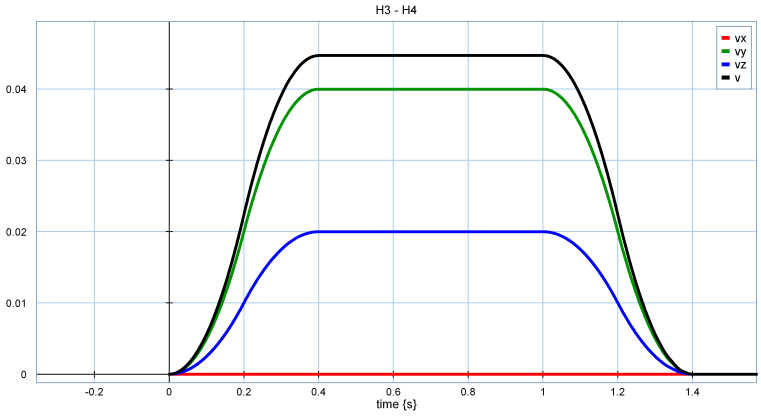

Figure 4.21: Velocity of the set-point.

The direction of the velocity is expressed in the unit vector $\hat{\mathbf{v}}$. The absolute maximum velocity is $v \approx 0.0447$. As mentioned the velocity of the y direction is limited by the maximum velocity in the z-direction. This can clearly be observed from figure 4.21. The maximum velocity in z direction reached

the maximum velocity in that direction, which was 0.02. The maximum velocity in the y direction was not reached, which was 0.06. Instead the maximum velocity is 0.04, twice as much as the z direction. This can easily be explained by the fact that the unit vector is twice the size in y-direction, with respect to the z-direction.

Motion 5: Pure rotation.

The rotational velocity is in similar fashion calculated as the linear motion (motion 4). The calculation follows:

$$w_{max} = \begin{pmatrix} 0.2 \\ 0.3 \\ 0.4 \end{pmatrix} \qquad \hat{\mathbf{w}} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \tag{4.96}$$

This results in the following value for the rotation velocity:

$$\begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix} = \begin{pmatrix} \frac{w_{max_x}}{\hat{w}_x} \\ \frac{w_{max_y}}{\hat{w}_y} \\ \frac{w_{max_z}}{\hat{w}_z} \end{pmatrix} = \begin{pmatrix} \frac{0.2}{0} \\ \frac{0.3}{1} \\ \frac{0.4}{0} \end{pmatrix} = \begin{pmatrix} \infty \\ 0.3 \\ \infty \end{pmatrix} \tag{4.97}$$

Which is used to calculate the maximum value for the rotational velocity.

$$w_{max} = min(\alpha_1, \alpha_2, \alpha_3) = (\infty, 0.3, \infty) = 0.3 \tag{4.98}$$

Results

The results of the simulations are presented in two figures. The first figure illustrates the rotation angle θ and the rotation axis during the path. This figure is depicted underneath. The second figure illustrates the rotatonal velocity of the set-point.

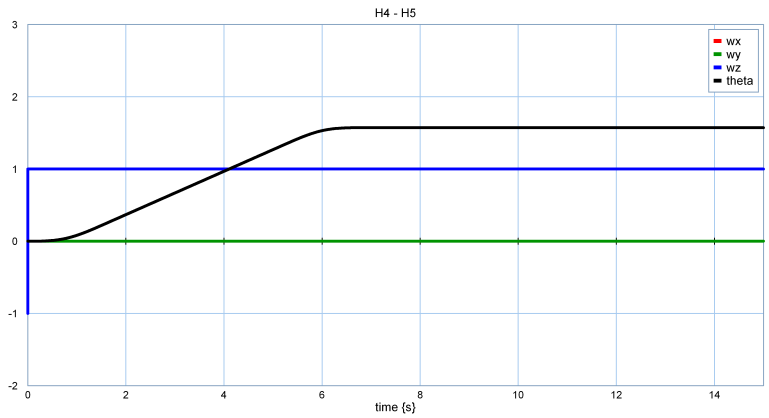

Figure 4.22: Direction of the rotation and angle θ

In this case the rotation vector points in the y direction. The rotation is in the y-direction, from z axis to the x axis. The rotation is in the opposite direction as the first motion, hence the value is 1.

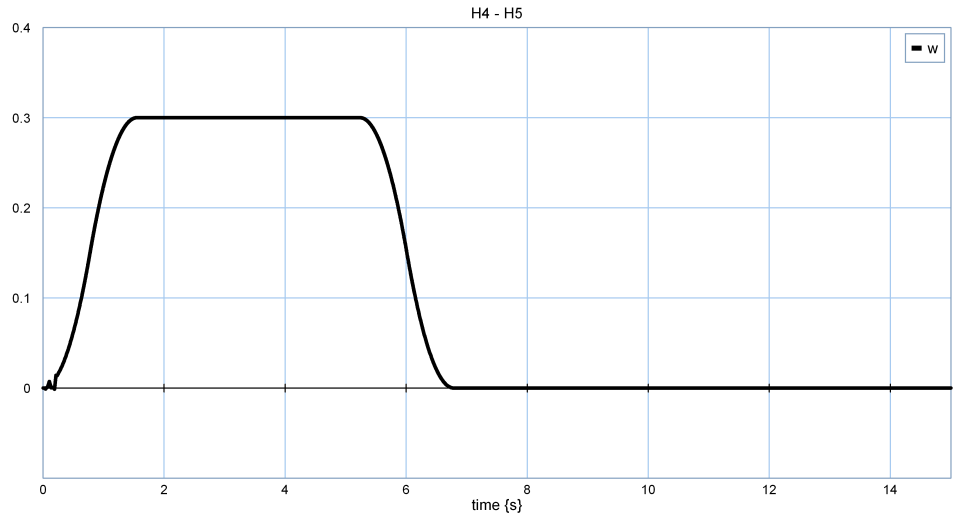

Figure 4.23: Rotational velocity of the screw motion.

The values of the 4.22 show that this motion has a fixed value for the rotation axis, as expected. The total angle given as 90 degrees or $\pi/2 \approx 1.57$ which is the value at the end of the path expressed in θ, indicated as "theta" in the graph in figure The value of the rotational velocity does not exceed the maximum value of $w_{max} = 0.3$, according to result depicted in figure 4.23.

4.4.3 Discussion

The values of the velocities and rotational velocities are in accordance with the given values. Some unexpected behavior was observed in the velocity profiles. The cause for this is not found, however the errors are minor. The cause is probably the result of errors in due to differentiation of the position. The results shown in this section are based on values to illustrate the three cases. In principle any value can be admitted for the calculation of maximum values. The tests indicate that limitations of given velocity of the set-point hold in the simulations. The accelerations and jerk were not tested. However for the motion set-point generator to return a coherent profile the jerk and acceleration have to be the set values. In case of the linear motion and the pure rotation the values of the velocity,acceleration and jerk are limited in different directions. The use of maximum values might be useful for more potential dangerous tasks like cutting bread. In this case the maximum acceleration and jerk can be necessary to cut the bread, however this is only allowed in certain directions. The limitation of

velocity, acceleration and jerk is in the direction of global coordinates. For some objects like knives a local limitation of these values can be useful. The limitation in a local reference frame can be achieved by a transformation of the direction. In most applications a general limitation of the velocity, acceleration and jerk is sufficient and is the limitation in multiple directions is redundant. However in some case has an useful application.

4.5 Limitations of the motion set-point generator

The paths of this chapter behave in the desired way. The paths that are described can in principle describe a motion from an arbitrary pose to any other arbitrary pose (expressed in homogeneous matrices), with desired values of jerk, acceleration and velocity of the set-point. The motion set-point generator generates a path from one pose to another pose. However, path planning can be a more complicated task, because it requires a notion of where other parts of the the arm are positioned and what specific position of the arm can be achieved. Not every movement from an arbitrary pose to another arbitrary pose can be achieved. In the simulation, the movement of the arm could sometimes not be achieved because of the limited joint-space. In some cases the difference between the set-point and the end-effector increased to high values. Such behavior is not desired. Solutions to predicting if a path is possible are described in [15, 16]. Every movement for the PRA was first entered in the 20-sim model before the real arm would be set up for the task. For the task of autonomous path planning, the Bobbie robot is required to check if such motion is possible. If simulations are efficient, these could be a solution. However, the methods cited are efficient and therefor more practical in a real-time situation such as is desired in robotics. The path ensures maximum values for v/a/j (velocity/acceleration/jerk) of the set-point of the end-effector, however the values for the v/a/j for the joints are not ensured. Often the end-effector is the point that is the furthest away from the base of the PRA. In such cases it can be assumed that in general the joints of the robotic arm have a limited value for v/a/j as well. However this is not guaranteed, there are possible situation where the other joints have a higher v/a/j. In that case additional methods need to be used to guarantee that other parts of the robotic arm are not exceeding the values of v/a/j.

4.6 Conclusions and recommendations

4.6.1 Conclusions.

The impedance controller requires a homogeneous matrix as an input. To calculate the homogeneous matrix with screw theory, the begin and end pose

are required as input. For an appropriate motion profile, the values of the velocity, acceleration and jerk are required to be limited. With these inputs, a path can be generated as an input for the controller.
A motion profile with maximum velocity, acceleration and jerk is called a soft motion profile. An algorithm is used which calculates the time periods of the different parts of the soft motion profile. The soft motion algorithm uses the distance, and the maximum velocity, acceleration and jerk as an input. It generates the fastest path within these maximum values. For each element of the twist, the maximum value for it is required as an input. To set these values in a meaningful way a calculation is needed, because the velocities are coupled. The calculation delivers values that are used to calculate the soft motion profile.
With this the motion, from one homogeneous matrix to another, is achieved. To use the robotic arm in a meaningful way, it is desired to move the end-effector from one pose to the next pose. And from that pose to the next, and so on. To this end the paths as described in this chapter are coupled by multiplying each path with a step function. This results in a set of motions which can be generated by putting in a set of desired homogeneous matrices and the maximum values for velocity, acceleration and jerk for the motions.
In this chapter a method was presented that describes the motion of a set-point as a screw motion or linear motion. The method of screw theory and on alternative interpretation were presented. Screw theory is compact and therefor preferable to the alternative interpretation. The alternative interpretation is somewhat redundant in the sense that the path is fully described by screw theory. In this chapter the appropriate parameters for the soft motion profile were obtained by means of the alternative interpretation. However, the calculation of the motion profile can be achieved by screw theory as well. The alternative interpretation may be considered insightful, because it delivers the parameters of the twist, which are used for calculating the maximum values of velocity, acceleration and jerk. One other interesting thing about the alternative interpretation, is that it delivers a continuous twist $\mathbf{T}(t)$. This is a function that indicates the velocity and rotational velocity at any given time.

4.6.2 Recommendations.

The end-effector of the robotic arm can assume poses in a certain space. This space is dependent on the joint space and the length of the parts of the robotic arm and the pose of the base of the robotic arm. In [16] it is indicated that some parts of the space in which the robotic arm can reach, can be reached from more previous poses. This subspace is therefor a preferable work area. To achieve that the base of the arm moves in such a way that the end-effector operates in this the preferable work area an number of parts can move. On way to move the base of the arm is by driving the robots

wheelbase. Another manner in which the end-effector can be moved into this are is by a joint in the middle of the robot. This would allow the robot to navigate the robotic arm to an work area that is favourable.
Until now the motions that were generated, were checked by the 20-sim simulation before they were implemented in the robotic arm. This was done to ensure that the robotic arm would perform in a proper way. In case undesired behavior was observed, another path would be chosen. For an autonomous robot it is useful to have built in capabilities, to check if a motion is possible. One way to let the robot check if a motion is possible is by using a simulation program such as 20-sim. However in [15] a way to calculate if such motions are possible is proposed. This is a more efficient way to figure out if a motion is possible, and as such it can be recommended to be implemented for the PRA.
If a pose cannot be reached because the end of the joint space is reached it is useful to have a limitation on the size of error between the set-point and the end-effector. In some simulations the set-point could not be reached. The error between the set-point and the end-effector became large. At a certain point in time the robotic arm started to move again, this resulted in a very high values for velocity and acceleration. This is an example of an unsafe situation of the robotic arm. The work previously cited can predict if a motion is possible and can prevent such situations. However a fail save for this situation can be the limitation of the maximum error between the set-point and the end-effector. Limiting the error can avoid too high velocities and accelerations. After a limitation is exceeded, the robotic arm could stop and reset, after which an alternative path is chosen. While such situations ought to be prevented, such safety systems can help to achieve higher levels of safety of the robotic arm.
In this chapter an elaborate description was given to calculate the homogeneous matrix. In section 2 an alternative interpretation was given to calculate the homogeneous matrix. The calculations for the parameters of the motion profile could also have been performed by calculating the unit twist. It should be possible to limit the motion in similar way, because the relation between the velocity and the rotational velocity is the same. It might be interesting to calculate it in this fashion, because it is more compact. The motion profile can still be calculated in similar way, as well as the coupling of paths. To adjust the motion profile to the twist, it is to be normalised, so that the distance becomes 1. If this adaptation is made the homogeneous matrix can easily be calculated with similar values for the velocity, acceleration and jerk.

Chapter 5

Conclusions and Recommendations

5.1 Conclusions

The useful application of care robots like Bobbie is becoming more realistic. The goal of care robots is to serve the elderly population. The tasks of the Bobbie robot include autonomously executing tasks that otherwise would be performed by humans. The application of care robots in a human environment requires that the Bobbie robot can perform tasks safely. The design of care robots depends on a number of disciplines such as control engineering, mechanics and computer vision. It is important that these disciplines are aware of each others tasks and goals, in order to design a useful robot. For the work presented in this thesis, goals were formulated in the introduction. Here the goals are compared to the work that is presented in this thesis.

The controller of the Philips Robotic Arm (PRA) requires an input. The motion set-point generator creates an input for the impedance controller of the PRA. The input is a set-point expressed in a homogeneous matrix $H(t)$. However it in turn requires input information, this information is the desired end pose of the path (H_O^E). For the purpose of obtaining the desired pose, the current pose of an object to be grabbed is required. This can be achieved by recognising the object, and estimating the pose of the object. For the controller of the robotic arm, a motion set-point generator was built. The motion set-point generator provides a suitable input for the impedance controller of the PRA. The path generated by the generator ensures that the maximum values for velocity, acceleration and jerk are limited. The robotic arm can be steered from an arbitrary pose to any other arbitrary pose in principle, if the joint space allows it. The paths generated by the motion set-point generator can be coupled. With this the end-effector can move through a series of poses. With the coupling of paths, different tasks can be executed. If an object's pose is known, a standard path can be calculated.

Path planning for this implementation means that a number of desired poses are given with the desired maximum values for velocity, acceleration and jerk for each path. The end-effector then moves from pose to pose, achieving the desired motions. For the task of object recognition and pose estimation, Linemod was used. The implementation of this method worked well. Different objects can be detected fast and robustly. Linemod is a template matching method, which uses scale-invariant features to compare registered templates with observed templates. Because the features are scale-invariant templates can be recognised from different distances. An orientation of the object with respect to the camera can be coupled to a registered template set. If during detection the observed template set is recognised as the registered template, the orientation is obtained. In this way, the orientation of the object with respect to the camera can be recognised. The depth image of the Kinect can be used to obtain a position of the object. Experiments were performed to test the accuracy of Linemod as a pose estimator. The results of the experiments indicate that it plausible that Linemod can be used to estimate the pose of objects. The experiments also indicate that the accuracy is sufficient for use as an initial estimation for ICP.

5.2 Recommendations

For autonomous actions of the Bobbie robot it would be useful to implement the method of Linemod as a pose estimator. Furthermore, it is recommended to device experiments to test pose estimation of objects. For the implementation of the motion set-point generator it is recommended that a number of standard paths for different kinds of tasks are created. Such that if the pose of an object is known and the PRA is in reach of that object, the robot can grab it by executing the standard path associated with that object.

Appendix A

In the previous chapter, a method was presented to create a motion set-point generator. The chosen method guides the end-effector of the robotic arm from a begin pose to an end pose, in the shortest amount of time, while adhering to predefined maximum values for velocity, acceleration and jerk. To move the PRA in useful way, it is desired to move the robotic arm from pose 1 to pose 2, and from pose 2 to pose 3 and so on. For this purpose, the paths are coupled, which is the subject of this appendix. The coupling of the motions is achieved by multiplying each path with a step function. The paths are generated as homogeneous matrices as described in chapter 4. Each path is multiplied with a step function. To obtain the correct step functions the start and end time of each step functions are required. In this appendix a method to calculate the begin and end times for each step function is presented. A path can be expressed in a homogeneous matrix. In section 4.2 the expression for the homogeneous matrix was:

$$H_O^I(t) \tag{5.1}$$

The paths are coupled, to indicate the order of the paths the number of a path can be indicated by n:

$$H_1(t) \tag{5.2}$$

The initial pose is H_0, The path from H_0 to H_1 is the first path, $H_1(t)$. In section 4.3 the different time steps and time periods of in a soft motion profile were calculated. The expression for the total time of the motion is T_f. The time period T_f of the motion of the set-point can be expressed in terms of start time t_s and the end time t_e of the motion profile:

$$T_f = t_e - t_s \tag{5.3}$$

In the section 4.3 the start time t_e was expressed as t_0, the end time t_e was expressed as t_7. The equation 5.3 can be rewritten to express the end time of the path:

$$t_e = t_s + T_f \tag{5.4}$$

The end time (t_e) is calculated by adding the total time period of the motion profile (T_f) to the given start time (t_s). In the coupling of paths a rest period

was added. The rest period is expressed in (T_r). A reason for a rest period is that the movement of the end-effector is guided by the set-point, however the end-effector lags behind. Because of this some extra time is required to guarantee that the end-effector reaches the set-point. In this case a rest period of two seconds is chosen, this seems to be sufficient. However the rest period can be changed to requirements of the robot. The start time of the next path can be calculated by:

$$t_{s_2} = t_{s_1} + T_{f_1} + T_{r_1} \tag{5.5}$$

The start time of the second path t_{s_2} can be calculated by adding the start time of the first path t_{s_1} to the time period of the first path T_{f_1} and the rest time T_{r_1}. This can be generalised to:

$$t_{s_n} = t_{s_{n-1}} + T_{f_{n-1}} + T_{r_{n-1}} \tag{5.6}$$

With this the a general way of calculating the start time of the paths is obtained. The start time is dependent on the previous paths. Equation 5.6 is a general way to calculate the start time of a path, however an exception is the start time of the first path. The first start time is given. The step function with which the different paths are multiplied have a start and stop time. The end time of the step function of the n^{th} path, is the start time of the step function of the $(n+1)^{th}$ path. The start and stop time are at the middle of the rest period T_r, this can be seen in in figure 5.1 . The next step function starts at this time. The homogeneous matrix of the end of first path is equal to the homogeneous matrix of the second path. This is true for every path following the previous path. Such that: If the notation for the end time of the n^{th} path is t_{e_n} the pose of the end of the path is expressed as:

$$H_O^{I_n}(t_{e_n}) \tag{5.7}$$

The notion of the start time of the $(n+1)^{th}$ path is $t_{s_{n+1}}$. The next path starts at the same pose as the previous path starts. This can be expressed as:

$$H_n(t_{e_n}) = H_{n+1}(t_{s_{n+1}}) \tag{5.8}$$

Between the end time of the n^{th} path and the begin of the $(n+1)^{th}$ path the function of the homogeneous matrix is constant. In equation 5.8 the expression is given for the fact that the end of the n^{th} path is equal to the beginning of the $(n+1)^{th}$ path With the coupling of the paths a set of homogeneous matrices can be inserted, with for each path the maximum values for velocity, acceleration and jerk. The motion set-point generator provides a moving set-point that passes trough the different homogeneous matrices. The coordinates expressed in the homogeneous matrix have a $T_r = 2$. In the middle of the rest period the n^{th} step function ends, and at that point in time the $(n+1)^{th}$ step function starts. The positions expressed in the

homogeneous matrix contain constant values from 1 second before to 1 second after the start of the step function. This results in a continuous smooth function that is differentiable twice. An example of a coupled function is depicted in the figure 5.1.

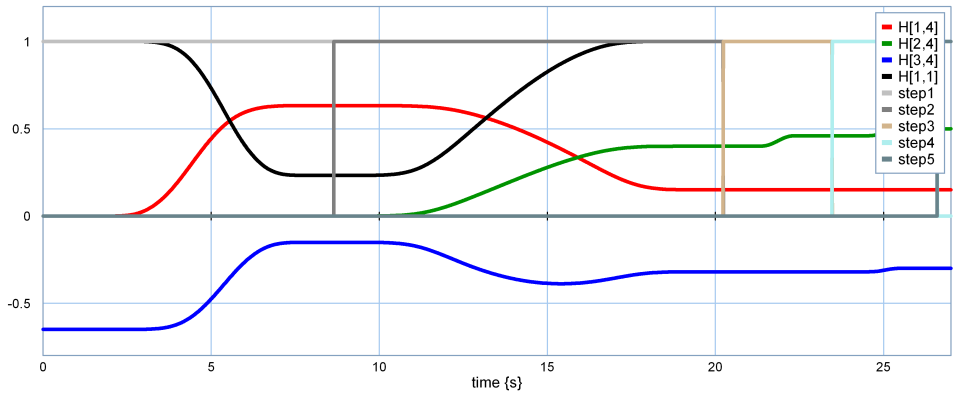

Figure 5.1: Coupling of paths; values for the homogeneous matrix and step functions.

In the figure, four elements of the homogeneous matrix are depicted. The elements $[1,4],[2,4],[3,4]$ are respectively the coordinates of the position x, y, z of the set-point. The element $[1,1]$ is an element of the rotation matrix. Five step functions are depicted, for each end of a step function the next step function starts. Here it can be seen that no changes of the elements occur one second before and until one second after the end and start of the step functions. This is in accordance to the rest time T_r.

Bibliography

[1] Stefan Hinterstoisser, Member, IEEE, Cedric Cagniart, Slobodan Ilic, Member, IEEE, Peter Sturm, Member, IEEE, Nassir Navab, Member IEEE, Pascal Fua, Fellow, IEEE, and Vincent Lepetit.
Gradient Response Maps for Real-Time Detection of Textureless Objects. IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL.34, NO.5
MAY 2012.

[2] Stefano Stramigioli Tutorial (T9). Geometry and Screw Theory for Robotics. Stefano Stramigioli, Senior Member, IEEE Robotics and Mechatronics, University of Twente, The Netherlands

[3] Hans de Boer. *Modeling and Control of the Philips Robot Arm* Hans de Boer, Tadele Shiferaw Tadele, Theo J.A. de Vries, Member, IEEE and Stefano Stramigioli, Senior Member, IEEE Robotics and Mechatronics, University of Twente, The Netherlands

[4] Martijn Visser, Stefano Stramigioli, Cock Heemskerk Cayley-Hamilton for roboticists, Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robot and Systems October 9-15, 2006, Beijing, China

[5] Richard M. Murray *A Mathematical Introduction to Robotic Manipulation* Richard M. Murray California Institute of Technology. Zexiang Li, Hong Kong University of Science and Technology. S. Shankar Sastry, University of California, Berkeley.

[6] Christian Wöhler *3D Computer Vision, Efficient Methods and Applications* 2nd ed. 2013, XVIII, 382 p. ISBN 978-1-4471-4150-1

[7] Ignacio Herrera-Aguilar *Soft Motion Trajectory Planning and Control for Service Manipulator Robot* Ignacio Herrera-Aguilar, Daniel Sidobre, ∗ LAAS - CNRS 7, avenue du Colonel Roche 31077 Toulouse, France Universite Paul Sabatier 118, route de Narbonne 31062, Toulouse, France Instituto Tecnologico de Orizaba Av. Oriente 9 No.852 94330 Orizaba, Mexico

[8] Model Based Training, Detection and Pose *Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes* Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, Nassir Navab, CAMP, Technische UniversitÃ¤t MÃ¼nchen (TUM), Germany, Industrial Perception, Palo Alto, CA, USA, CV-Lab, Â´ Ecole Polytechnique FÂ´ edÂ´ erale de Lausanne (EPFL), Switzerland

[9] C. Steger, "Occlusion Clutter, and Illumination Invariant Object Recognition", Int'l Archives of Photogrammetry and Remote Sensing, vol. 34, 2002

[10] Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes, Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, Nassir Navab, CAMP, Technische UniversitÃ¤t MÃ¼nchen (TUM), Germany, Industrial Perception, Palo Alto, CA, USA CV-Lab, Â´Ecole Polytechnique FÂ´ edÂ´ erale de Lausanne (EPFL), Switzerland

[11] Three Depth-Camera Technologies Compared Fabrizio Pece, Jan Kautz, Tim Weyrich Department of Computer Science, University College London, UK

[12] Zhang. A *Flexible New Technique for Camera Calibration.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(11):1330-1334, 2000.

[13] R. Reilink, S. Stramigioli, F. van der Heijden and G. van Oort Focus of attention distance estimation in a saliency-controlled stereo moving-camera setup

[14] Tadele Shiferaw Tadele Human-friendly Robotic Manipulators: Safety and Performance Issues in Controller Design. 2014

[15] *This paper is not (yet) published at this moment, but will probably be published sometime in 2015* *J.A.J. Huttenhuis, D.J. Borgerink, D.M. Brouwer, S. Stramigioli* *Kinematic Design Method for a Rail Mounted Inspection Robot Arm* *December 11, 2014*

[16] Using a Model of the Reachable Workspace to Position Mobile Manipulators for 3-d Trajectories Franziska Zacharias, Wolfgang Sepp, Christoph Borst and Gerd Hirzinger

[17] Marco A. Chavarria, Gerald Sommer, STRUCTURAL ICP ALGORITHM FOR POSE ESTIMATION BASED ON LOCAL FEATURES

[18] Andrew W. Fitzgibbon, Robust Registration of 2D and 3D Point Sets, Department of Engineering Science, University of Oxford.

[19] Robust Object Detection For Service Robots, Tim de Jager. ICA-3698823. Master thesis. Multimedia and Geometry, Computer Science, Utrecht University, October 17, 2013