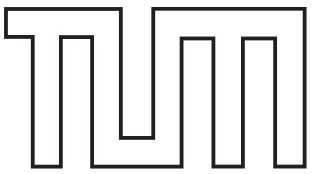

Technische Universität München
Institute for Media Technology
Prof. Dr.-Ing. Eckehard Steinbach

Diplomarbeit

Methods for Hand-Eye Coordination of a serial Robot from partial Observations

Author:	Fanzhen Kong
Matriculation Number:	2803959
Address:	Waldfriedhofstr.115
	81377 München
Advisor:	Dr.rer.nat. Wolfgang Sepp
	M.Sc. Nicolas Alt
Begin:	22.02.2010
End:	30.09.2010

Kurzfassung

Eine präzise Manipulation von Objekten mit einem Roboter erfordert die genaue Kenntnis der Lage des Endeffektors relativ zum Objekt. Mit die Annahme, dass die Lage des Objekts schon bestimmt ist, die Lage des Roboterendeffektors wird durch die vorher bestimmte Lage der Basis und anhand der Vorwärtskinematik mit Messungen der Gelenkwinkel- und Positionssensoren errechnet. Aber bei Robotern, die im menschlichen Umfeld arbeiten, wird auf die Leichtbauweise geachtet, um die Sicherheit zu erhöhen. Durch die Leichtbauweise werden geringere Steifigkeiten erziehlt als bei Industrierobotern. Somit weicht die über die Vorwärtskinematik errechnete Lage von der tatsächtlichen Lage des Endeffektors ab. Im Rahmen der Arbeit werden Methoden der Bildverarbeitung entwickelt, die es ermöglichen, die Lage eines seriellen Roboters in Kamerabildern in Echtzeit zu bestimmen und somit auch die Abweichungen zu minimieren. Somit soll gewährleistet werden, dass in Bildern lokalisierte Objekte mit dem Roboter präzise gegriffen werden können.

Abstract

Precise object manipulation by a robot requires precise knowledge of the position of the robot endeffector relative to the object. By the so-called eye-to-hand coordination, both the position of the object and the position of the robot relative to the camera are determined. In practice, usually the position of the robot base to camera is calibrated in advanced and the position of the robot endeffector relative to the base is calculated by forward kinematics with joint angle configurations. For the robots working in the human environment, they are constructed with lightweight in order to increase security, which achieves lower stiffness than industrial robots. Thus, the reached position of robot-effector deviates from its commanded position. The work of this thsis is to develop a method based on the image processing to minimize deviations and thus to estimate the real position of the robot endeffector in real time. Thus, the robot end-effector can be guaranteed to precisely grip the target object.

Contents

Chapter 1

Introduction

Humanoid robots are a group of robots that have overall appearance designed in basis of human body and can mimic some human behaviors [KFH+08]. As autonomous robots, humanoid robots can detect the confuguration transformation of their bodies, perceive the alteration of their surrouding environments, adjust their strategies to adapt environments and continue their behaviors to reach goals. Due to their human-like performance ability, they may be employed to proceed many physical and mental tasks that human undergo daily, such as object manipulation. In addition, they can also be used to finish many difficult tasks that are beyond human's ability (heavy physical works, etc.) or work in some extreme conditions (low and high temperature, etc.).

In the past years, many efforts have been made in the development of humanoid-robot technology that has become an interdisciplinary field involving electric engineering, mechanics, coginitive science, linguistics etc.. It is certainly important to create robots that are as human-like as possible, while the development of humanoid robots also helps us to understant human body's biological and mental processes.

One of the most important human capabilities in daily life is to manipulate objects, while such ability is still quite limited for robots. The key technologies as human-friendly are required to reach those capabilities, i.e. light-weight, kinematically redundant and „soft-controllable“ arms on mobile platforms, preferably equipped with articulated hands, and consuming minimal power only [HSAS+02].

A mobile humanoid robot named „*Rollin'Justin*“ with two-arms-hands system has been developed in the institute of Robotics and Mechatronics from German Aerospace Center (DLR). The two-arms-hands system is composed of two subsystems. The upper subsystem is a manlike upper-body system „*Justin*“ with two arms and two hands. The lower subsystem is a newly developed mobile platform attached to the upper system. The design of humanoid manipulator „*Justin*“ is based on the modular 7-DoF DLR-Lightweight-Robot-III (DLR-LWR-III) [HSAS+02] and the four-fingered DLR-Hand-II [BGLH01]. Two cameras („eyes“) are mounted on the head of „*Rollin'Justin*“ for its image processing.

In particular, the system are built symmetrically with a right-handed and a left-handed subsystem. It is designed for researches on sophisticated control algorithms for complex kinematic chains, as well as mobile two-handed manipulation and navigation in typical human environments [BWS+09]. Besides two-hands manipulation on the table, the movable torso also enables the system to reach objects on the floor and to grasp highly elevated objects [OEF+06].

For the tasks of object manipulation, the precise judgements on the relative position and orientation from the end-effector of the robot to an object is required, before the robot can manipulate the object. In the praxis, the relative position between the robot basis and the camera is calibrated in advanced. In operations, the relative position and orientation of the objects in camera coordinate system can be obtained by image processing in camera. The relative position and orientation between the robot end-effector and camera could be calculated with forward kinematic according to the calibrated DH-Parameter.

In versus, as the robot „$Rollin'Justin$“ are linked with revolute joints, the position and orientation of robot end-effector can be in principal commanded according to the joint angles between each links and oberserved by joint angle sensors. When both the target position and orientation of the objects and the current position and orientation of the robot end-effector in camera coordinate system are known, the next procedure is to make the robot end-effector reach and grap the objects.

In the robotics community, two main different procedures, the so-called “Open-Loop-Control System” and “Closed-Loop-Control System”, have been often used to make the robot end-effector reach and grap the target objects. By “Open-Loop-Control System”, the target position and orientation of robot end-effector is only to be commanded without monitoring the actual reached position and orientation of the robot end-effector. In contrast, by “Closed-Loop-Control System”, the actual position and orientation of the robot end-effector should be monitored and analyzed in camera. This makes sure that it really reached the target position and orientation.

By developing robots as „$Rollin'Justin$“ which work in the human environments, ideas are emphasized in light-weight construction to increase the security. But the light-weight robots can reach less stiffness compared to the robots, which work in the industries. Under such a condition, the from forward kinematics calculated positions and orientations of end-effector are proved to differ from the actual positions and orientations, respectively. Under this consideration, the standard “Open-Loop-Control System” is not suit for our case as it doesn't know if the target positions and orientations are reached.

Compared to the “Open-Loop-Control System”, the “Closed-Loop-Control System” has the advantage in verifying the actual reached positions and orientations of the robot end-effector. But in our case, as the robot end-effector is considered as the last joint of robot arm, which is actuated and concealed with a robot hand, it makes the real position of the end-effector invisible. By the help of landmarks on robot end-effector, the position and orientation of landmarks in camear coordinate system can be obtained by image processing

in camera, but the relative position and orientation between the landmarks and robot end-effector is uncertain in advanced.

To solve this problem, there is another option. As our cameras are mounted in the head of robot, it is assumed to use the so-called "Eye-To-Hand Coordination ". Another option of coordination by tracking problems is the so-called "Eye-In-Hand Coordination ", by which the cameras are mounted directly on the robot end-effector. It could solve the problem by "Closed-Loop-Control System", but also has the disadvantage in rather limited view to observe the object and its surrounding environments.

In this thesis, a combined system of both "Open-Loop-Control System" and "Closed-Loop-Control System" is developed. The idea of this method, is to command the robot end-effector with arbitrary joint angle configurations (the so-called "Open-Loop-Control System"), observe the actual position and orientation of the landmarks on robot end-effector in camera (the so-called "Closed-Loop-Control System"), determine the relative position and orientation between landmarks and robot end-effector, compare the actual positions and orientations of the robot end-effector with the commanded positions and orientations.

As known, there are deviations from the actual positions and orientations to the commanded positions and orientations, respectively. To minimize these deviations, a method of online calibration based on this combined system is developed. The target is to find out, if the deviations can be converged to stable values with learning of plenty of observations, and therewith estimate the more accurate position and orientation of robot end-effector for each new observations.

In this thesis, some foundamental knowledges and the details of problem definition will be described in chapter 2. In chapter 3, some state-of-the-art methods to solve the similiar problems will be introduced. Then in chapter 4, it will be followed by the concrete methods used in the thesis with 3 landmarks or 1 landmarks and also methods of online calibration. In chapter 5, two different experiments with details of implementation details and results not only in simulation environments but also with the real robot for this thesis will be introduced. Last but not least, the chapter of conclusion and outlook are refered in chapter 6.

Chapter 2

Foundations and Problem definition

In following chapter, the foundamental knowledge of robotics and problems by hand-eye-coordination of „*Rollin'Justin*“ will be introduced.

2.1 Spatial Representation

In the study of robotics, the locations of objects in three dimensional world are concerned. In three dimensional world, a rigid body can be described by six parameters. Three of them determines the position in the room, and the other three determines the orientation. These parameters are also called degrees of freedom (DoF). In order to describe the position an orientation of a rigid body in three dimensional space, it will be attached to a coordination system, or **frame**. An object in a coordination system **A** can be described in transformation matrix

$$^{\mathrm{A}}T_{\mathrm{O}} = \begin{bmatrix} ^{\mathrm{A}}R_{\mathrm{O}} & ^{\mathrm{A}}\mathbf{t}_{\mathrm{O}} \\ \mathbf{0} & 1 \end{bmatrix}, \tag{2.1}$$

with $^{\mathrm{A}}\mathbf{t}_{\mathrm{O}} \in \mathbb{R}$ representing the translation vector from origin of coordination system **A** to position of object **O** and $^{\mathrm{A}}\mathbf{R}_{\mathrm{O}} \in \mathbb{R}^3$ representing the rotation matrix from the orientation of coordination system **A** to the orientation of object **O**. The vector (0,0,0,1) in fourth row extends the matrix to homogenious coordination.

If the Object **O** is observed in other coordination system **B**, the representation can be expressed in formel

$$^{\mathrm{B}}T_{\mathrm{O}} = ^{\mathrm{B}}T_{\mathrm{A}} \cdot ^{\mathrm{A}}T_{\mathrm{O}}, \tag{2.2}$$

in which the expression of $^{\mathrm{B}}T_{\mathrm{A}}$ represents the position and orientation of coordination system **B** in coordination system **A**.

2.1.1 *XYZ*-Moving

Instead of ${}^{\mathrm{A}}R_{\mathrm{O}}$, the orientation of object **O** can also be seen as three sequential rotations. There are four different rotations in spatial room:

Rotation as **Kardan**, **Euler**, **Andoyer**, and around a vector.

In this thesis, the XYZ-fixed-angles convention according to **Kardan** [Cra05] is considered. Each rotation will be excecuted around one fixed axis of coordination system **A**. The rotation matrix ${}^{\mathrm{A}}R_{\mathrm{O}}$ can be constructed as the product of three part rotations

$$ {}^{\mathrm{A}}R_{\mathrm{O}} = R_{XYZ}(\alpha, \beta, \gamma) = R_Z(\gamma) \cdot R_Y(\beta) \cdot R_X(\alpha), \tag{2.3} $$

where the $R_X(\alpha)$ represents the rotation around axis $\boldsymbol{X}$ for angle α, the $R_Y(\beta)$ represents the rotation around axis $\boldsymbol{Y}$ for angle β, the $R_Z(\alpha)$ represents the rotation around axis $\boldsymbol{Z}$ for angle γ.

According to this convention the orthonormal matrix R with Determinant det=1

$$ {}^{\mathrm{A}}R_{\mathrm{O}} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{2.4} $$

can be expressed as:

$$ {}^{\mathrm{A}}R_{\mathrm{O}} = \begin{bmatrix} \cos\beta \cdot \cos\gamma & -\cos\beta \cdot \sin\gamma & \sin\beta \\ \cos\alpha \cdot \sin\gamma + \cos\gamma \cdot \sin\beta \cdot \sin\alpha & \cos\gamma \cdot \cos\alpha - \sin\beta \cdot \sin\gamma \cdot \sin\alpha & -\cos\beta \cdot \sin\alpha \\ -\cos\gamma \cdot \cos\alpha \cdot \sin\beta + \sin\gamma \cdot \sin\alpha & \cos\alpha \cdot \sin\beta \cdot \sin\gamma + \cos\gamma \cdot \sin\alpha & \cos\beta \cdot \cos\alpha \end{bmatrix} \tag{2.5} $$

The inverse calculation for the three rotation parameters (α,β,γ) can be obtained from 2.4 and 2.5 with the formels

$$ \alpha = atan2\left(-r_{23}, r_{33}\right), \tag{2.6} $$

$$ \beta = atan2\left(r_{13}, sqrt\left(\frac{r_{11}^2 + r_{12}^2 + r_{23}^2 + r_{33}^2}{2}\right)\right), \tag{2.7} $$

$$ \gamma = atan2\left(-r_{12}, r_{11}\right), \tag{2.8} $$

with

$$ atan2(x, y) = \arctan\left(\frac{x}{y}\right). $$

Consider the limits of calculation by programming, for the special cases of that $\beta \leq -90°$ or $\beta \geq 90°$, the angles can be calculated in other ways. $\alpha = 0$,$\gamma = atan2(r_{21}, r_{22})$, and $\beta = -\frac{\pi}{2}$ if $\beta \leq -90°$ or $\beta = \frac{\pi}{2}$ if $\beta \geq 90°$.

2.2 Robotics

In the robotics the position and the orientation of frame **O** will be described in a different way.

2.2.1 Coordination System

In this thesis , there are four important coordination systems. They are basis coordination system **Basis**, object coordination system **Object**, end-effector coordination system named as **TCP**(Tool Center Point), and camera coordination system **Camera**.

The **Basis** coordination system is positioned at the fixed location of robot. The **Camera** coordination system is fixed at the left camera. The **TCP** coordination system is considered at the last joint of left or right arm (depending on which arm is used to grap the object). The **Object** coordination system is used to define the position and orientation of landmarks to be extracted for estimating the position and orientation of end-effector.

The position and the orientation of end-effector in robot basis coordination system **Basis** can be expressed in transformation matrix ${}^{Basis}T_{TCP}$, which can be also called as „**Pose**“ . The end-effector can also be observed in other coordination system, e.g. in camera coordination system. The transformation matrix can be calculated as

$$ {}^{Camera}T_{TCP} = ({}^{Basis}T_{Camera})^{-1} \cdot {}^{Basis}\, T_{TCP}, \tag{2.9} $$

in which both the ${}^{Basis}T_{Camera}$ and ${}^{Basis}T_{TCP}$ are calculated using „forward kinematic model“.

2.2.2 Forward Kinematics

Foward kinematics is the very basic problem in study of mechanical manipulation. This is the static geometrical problem of computing the position and orientation of the end-effector of the robot. Specifically, given a set of joint angles, the forward kinematic problem is to compute the position and orientation of the TCP frame relative to the base frame. [Cra05]

The position and the orientation of **TCP** can be expressed in the way of:

$$ {}^{Basis}T_{TCP} = {}^{Basis}\, T_0 \cdot \prod_{i=1}^{n} ({}^{Link,i-1}T_{Link,i}) \cdot {}^{Link,n}\, T_{TCP}, \tag{2.10} $$

in which the transformation matrix ${}^{Link,i-1}T_{Link,i} \quad i \in [1,n]$ means the relative position between the link i-1 and link i, which consist the manipulator from robot basis to end-effector. Different from normal rigid body transformation, the kinematics model can be constructed by Denavit-Hartenberg-Transformation (DH).

2.2.3 Denavit-Hartenberg-Transformation (DH)

In the robotics community, the kinamatics model can be described by giving the values of four quantities for each link. Two describe the link itself, and two describe the link's connection to a neighboring link. In the usual case of a revolute joint, θ_i is calles the **joint variable**, and the other three quantities would be fixed **link parameters**. The definition of mechanisms by means of these quantities is a convention usually called the **Denavit-Hartenberg notation**[Cra88].

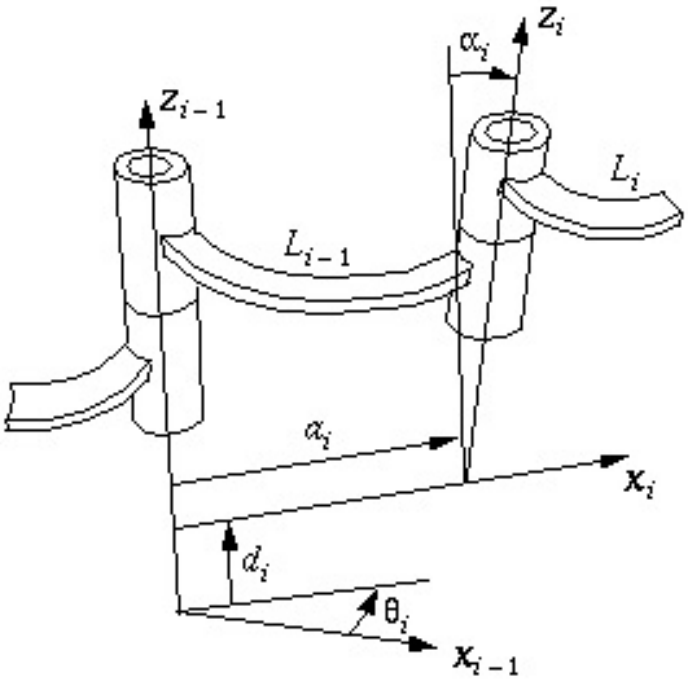

Figure 2.1: Denavit-Hartenberg frame assignment.

According to the **DH** convention, a frame is defined to be attached to each link, in order to describe the relative position between neighbor links (see figure 2.1). The convention to locate frames on the links is used as follows [Cra05]:

- The $\hat{Z}$-axis of frame i, called $\hat{Z}_i$, is coincident with the joint axis i;
- The origin of frame i is located where the a_i perpendicular intersects the joint i axis;
- $\hat{X}_i$ points along a_i in the direction from joint i to joint i+1;
- $\hat{Y}_i$ is defined according to the right-hand rule.

If the link frames have been attached to the links according to this convention, the following definitions of the link parameters are valid [Cra05]:

- a_i = the distance from $\hat{Z}_i$ to $\hat{Z}_{i+1}$ measured along $\hat{X}_i$;

- α_i = the angle from $\hat{Z}_i$ to $\hat{Z}_{i+1}$ measured about $\hat{X}_i$;
- d_i = the distance from $\hat{X}_{i-1}$ to $\hat{X}_i$ measured along $\hat{Z}_i$;and
- θ_i = the angle from $\hat{X}_{i-1}$ to $\hat{X}_i$ measured about $\hat{Z}_i$.

The transformation matrix between each links according to this convention can be calculated as follows:

$$^{i-1}T_i = R_{\mathrm{X}}(\alpha_{i-1}) \cdot T_{\mathrm{X}}(a_{i-1}) \cdot R_{\mathrm{Z}}(\theta_i) \cdot T_{\mathrm{Z}}(d_i), \tag{2.11}$$

where R_{X} and R_{Z} describe the rotations about the relevant x- or z-axis, meanwhile the T_{X} and T_{Z} describe the translations along the relevant x- or z-axis.

2.3 Rollin' Justin

Figure 2.2: DLR's *Rollin' Justin*.

Humanoid robots are developed for the applications not only in human environment but also in space environments. The capabilities of humanoid robots to deal with complex object manipulation tasks is a key problem. For this consideration the development of robust control strategies and intelligent manipulation planners for two-hands manipulation is currently an important research in the robotics community. Besides, to realize human

performace for robots in tool usage and object manipulation, strong but small arms and flexible gripping devices like dexterous hands seem necessary on the hardware side.

In the institute of Robotics and Mechatronics from German Aerospace Center (DLR), a mobile humanoid robot named „*Rollin'Justin*“ (see figure 2.2) with two-arms-hands system has been developed, which combines a manlike upper-body system „*Justin*“ with two arms and two hands and a newly developed mobile platform. The mobile robotic system „*Justin*“ with its 7-DoF DLR-Lightweight-Robot-III (DLR-LWR-III) [HSAS+02] arms and its two four fingered DLR-Hand-II [BGLH01] is an ideal experimental platform for these research issues.

In particular, the modular concepts of the DLR-LWR-III and the DLR-Hand-II are exploited by building the system symmetrically with a right-handed and a left-handed subsystem. The newly developed mobile platform enables the system to reach long ranged objects on the floor and to grasp highly elevated objects. PMD sensors and cameras allow the 3D reconstruction of the robot's environment and therefore enable Justin to perform given tasks autonomously. The upper body mounted on the mobile platform with a total of 51 DoF represents a highly complex kinematic structure [BWS+09]. In this thesis, only the joints on the upper body with 43 DoF are interested. Table 2.1 gives an overview of the 43 actuated DoF.

Subsystem	Torso	Arms	Hands	Head & Neck	$\sum$
DoF	3	2 X 7	2 X 12	2	43

Table 2.1: Upper Body Overview.

The details of the structure and DH-Parameters of Upper Body of „*Justin*“ will be presented in Appendix A.1.

2.3.1 DLR-Light-Weight-Robot-III

In the last years, a third generation of torque-controlled light weight robots has been developed in DLR's robotics and mechatronics lab. The main concept of developing this Light-Weight-Robot LWR III is aiming at reaching the limits of what seems achievable with present day technologies not only with respect to light-weight, but also with respect to minimal power consumption and losses [HSAS+02].

Classical industrial robots, which is compared to the light weight robots, with heavier weight, their typical load: weight ratio of 1:10 or worse and their pure position control philosophy are not adequate candidates for the human-friendly daily manipulation.

The new DLR Light Weight Robot has an outstanding ratio of payload to total mass. Though it weights only 14kg, it is able to handle payloads of 14kg over the whole dynamic range. Very light gears, powerful motors and weight optimized brakes have been integrated

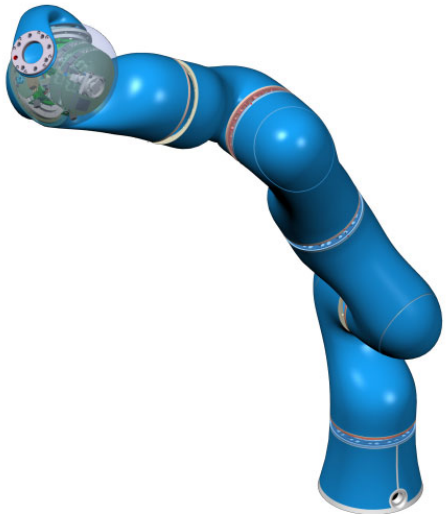

Figure 2.3: New DLR Light-Weight-Robot LWR III

into the robot. Similar to the human arm, the robot has seven degrees of freedom which results in advanced flexibility in comparison to standard industrial robots. The integrated sensors are most progressive - each of the Light Weight Robot joints has a motor position sensor and a sensor for joint position and joint torque. Thus the robot can be operated position, velocity and torque controlled [Lig].

2.4 Problem

For the tasks of object manipulation, we need to know not only the position and orientation of the object relative to basis coordinate system, but also the position and orientation of the end-effector (**TCP**) relative to basis coordinate system. According to the posiiton and orientation of the object, we can command the end-effector to move towards the object and grab it. For these kind of object manipulation tasks, there are two kind of standart control system:

1. Open-Loop-Control
2. Close-Loop-Control

The structure of these two control system can be coarsely illustrated in figure 2.4:

Open-Loop-Control

The idea of **Open-Loop-Control** system is just to command the end-effector to the position of the object. With the given position and orientation of the object from camera,

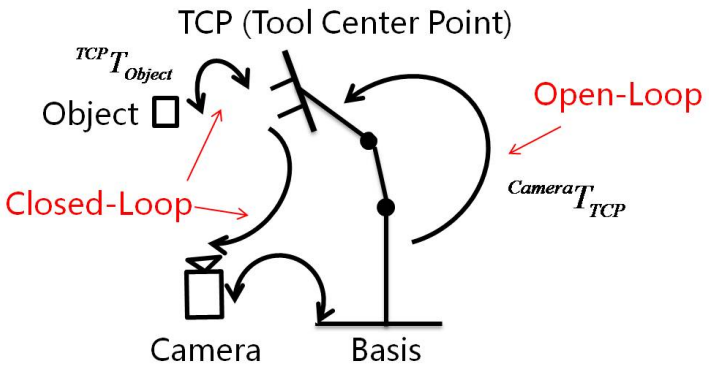

Figure 2.4: Open-Loop-Control System vs. Close-Loop-Control System

the joint angles configuration of the robot can be calculated by an inverser kinematic model to directly reach the position of the object or step by step towards it.

But as introduced before, the elasticity of robot segments of the DLR-Light-Weight-Robot-III has the nature with less stiffness compared to the industriell robot arms. Or the DH parameters for each link may not calibrated accurately. Therefore, the actual reached position and orientation of the **TCP** derivates from the commanded position and orientation of the **TCP**.

The derivations could be varianted according to different poses. The more straightforward the robot arms moved, the less stiffness they can get. The derivations could be ranged from mili meter even to some centi meters depending on the actual weight and the gravity.

By a large derivation, the end-effector may miss the target of the object. Therefore, the **Open-Loop-Control** system is not reliable for our case.

Closed-Loop-Control

As known that, the **Open-Loop-Control** system only commands the robot end-effector to the new position without considering the actual position it reaches. The **Closed-Loop-Control** system is not only command the robot end-effector with joint angles configuration according to the position and orientation of the object. It observes also the current position and orientation of the robot end-effector for each new pose from camera. By means of that, it can determine if the robot end-effector has the target position and orientation reached.

But here shows up another problem. The robot end-effector of the „*Rollin'Justin*“ is considered as the last joint of the robot arm. As the robot arms are also mounted with two artificial hands DLR-Hand-II, it makes it impossible to observe the position and orientation of the **TCP** from the surface.

For this consideration, there are also two possibilities to solve the problem. As illustrated in figure 2.5,

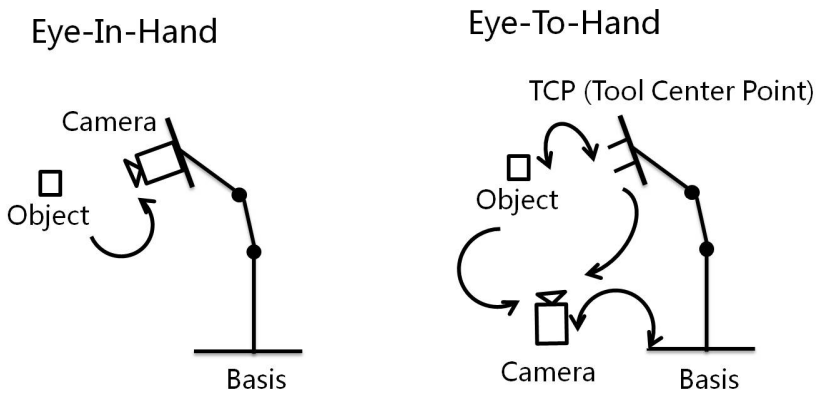

Figure 2.5: Eye-In-Hand vs. Eye-To-Hand

with an so-called Eye-In-Hand Coordination (the camera is mounted directly on the end-effector), the relative position and orientation from object to camera coordinate system can be easily determined and checked if the end-effector has the target position and orientation reached.

But with this coordination, the view of the camera would be much smaller than the Eye-To-Hand Coordination (the camera is mounted on the head of the robot). As the end-effector moves towards the object step by step, the view of the camera will not only get smaller and smaller, but also lose the global information of the environment.

Therefore, we will still choose the Eye-To-Hand Coordination in a Closed-Loop Visual Servoing system. The idea of the method can be seperated into two aspects:

On one side, according the position and orientation of the target object in camera coodirnate system, the control loop system still commands the end-effector to move towards the target object position as **Open-Loop-Control** system does.

But on the other side, it doesn't observe the position and orientation of the end-effector, but a marker-based object, which has the fixed relative position and orientation to **TCP** coordinate system (on **TCP** or actuated with **TCP**).

This relative position and orientation from marker-based object to **TCP** is still unknown in advanced. But by more and more commanded moves of the robot end-effector, it can be calibrated. For each new moves, the control loop system should be able to estimate the deviations between the commanded and the actual position and orientation of the marker-based object. The details of this method will be introduced in chapter 4.

Chapter 3

State-Of-The-Art

As mentioned in the previous chapter, the main purpose of this thesis is to tracking the actual position of the end-effector relative to object for the tasks of object manipulation. As known, the end-effector of the robot „*Rollin'Justin*“ are the two four fingered DLR-Hand-II [BGLH01]. In this thesis, the TCP (Tool Center Point) is considered as the last joint of the arm, which is connected with the hand.

As mentioned before, the actual reached position and orientation of the TCP deviates from the commanded position and orientation of the TCP. By commading the end-effector to reach the target object, the actual position and orientation of the TCP can not be certain with servoing, as the TCP is not visible from Camera. For this consideration, it is needed to tracking the position and orientation of TCP with some methods.

In the following paragraphs, there are several developed methods from other authors. They estimate or track the position and orientation of an object in space solely by methods of image processing or that estimate the position and orientation of the target in combination with the actuated structures.

Real-Time Visual Tracking of Complex Structures

In the paper of [Dru02], a novel framework for 3D model-based tracking is presented. It combines the graphical rendering technology with constrained active contour tracking to create a robust wire-frame tracking system. An internal CAD model of the object is used for tracking. Thus the visible edge features are compared and identified online at each frame in the video feed in order to find the correspondences.

A Lie group formalism is used to deal with the motion computation problem. It makes it simple to a geometric terms and the tracking problem becomes a simple optimization problem by solving the iterative reweighted least squares.

In order to track the edges of the model as lines in the image, it is necessary to determine

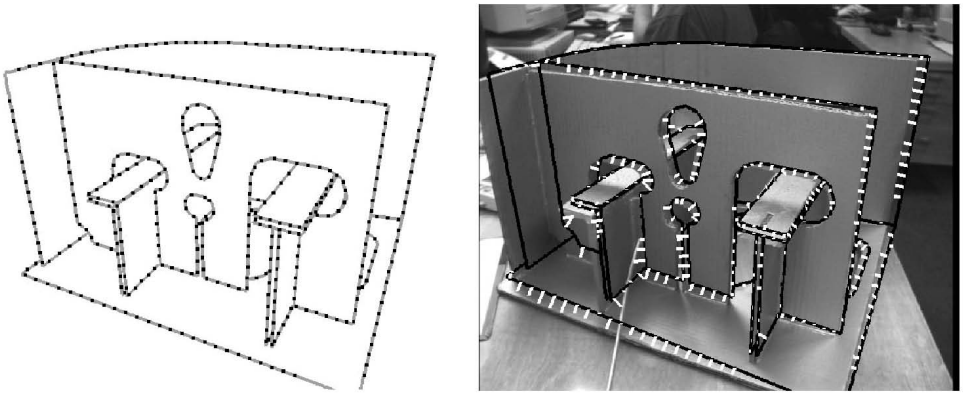

Figure 3.1: Sample points are assigned and distances measured.

which lines are visible at each frame and where they are located relative to the camera. Where the line is visible, the sample points are assigned to search for the nearest intensity discontinuity in the video along the edge normal (see Figure 3.1).

In the figure 3.2, the tracking system operation is shown. At each cycle, the system renders the expected view of the object by step a using its current estimate of the projection matrix, P. At step b, the sample points of the identified edges are assigned in image coordinates. Then, the edge normal is searched in the video image for a nearby edge at step c. At step d, the least-squares estimate of the motion M is given from the projected measurement vector. Then the euclidean part E is updated at step e. Finally, the new projection matrix P is obtained with camera parameter K and E to give a new estimate for the current position.

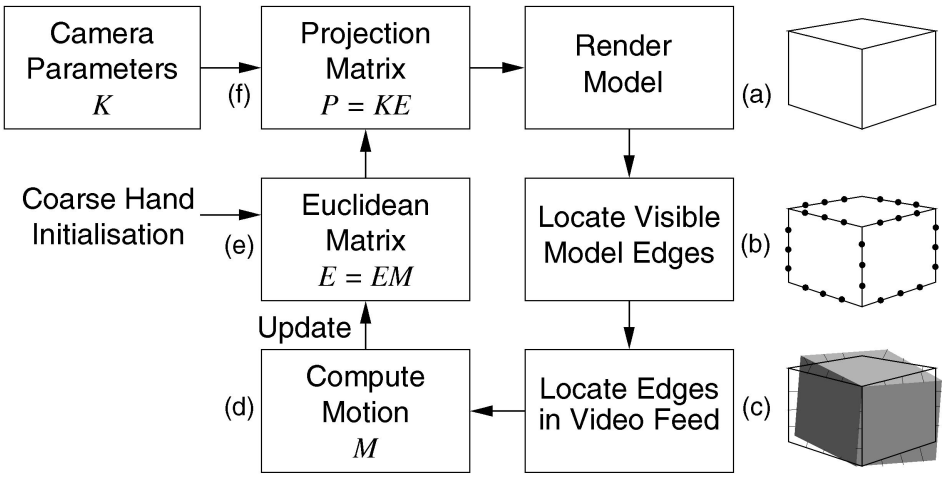

Figure 3.2: Tracking system operation.

Besides, a visual servoing system is presented to show the accuracy of the results from

the tracker. The system also incorporates real-time online calibration of internal camera parameters.

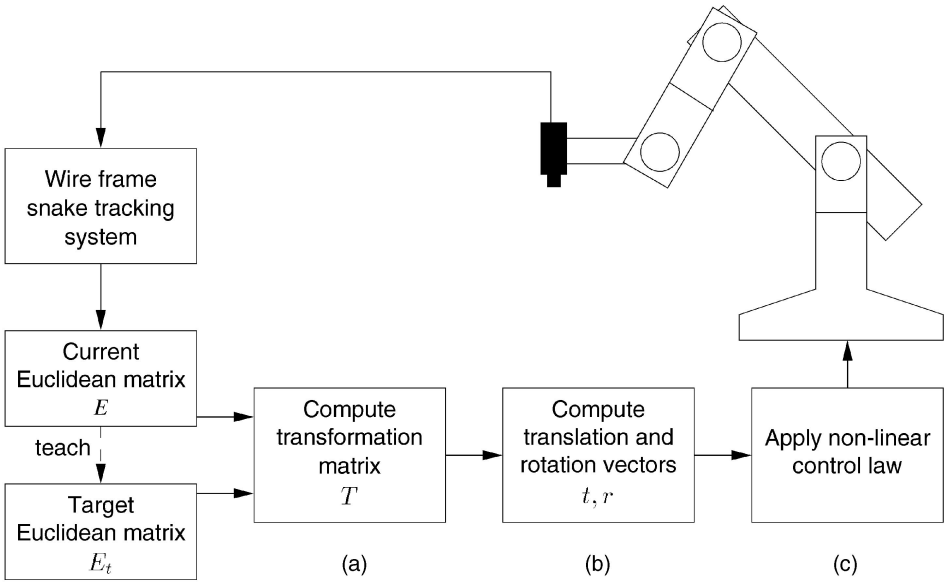

Figure 3.3: Visual servoing system operation.

This system takes the Euclidean matrix,E,from the output of the tracking system and within a nonlinear control loop to provide feedback to servo the robot to a target pose. The product of current Euclidean matrix E with target Euclidean matrix E_t^{-1} ($T = EE_t^{-1}$) represents the transformation from the target position to the current position. The translation and rotation vectors of T are then multiplied by a gain factor and sent to the robot as velocities.

This paper also describes how this tracking system has been extended to complex configurations. For examples, by the using of multiple cameras, tracking of articulated structures or of multiple structures with constraints.

This system has two main limitations. It depends on a coarse hand localization and can only handle piecewise rigid polyhedral structures. It's difficult to extract features from curved objects such as sculpture and smooth textured surface.

Real-Time Object Tracking without Feature Extraction

Different from the previous approach with model-based tracking system, the paper [MHS06] propose an appearance-based tracking method without any feature extraction from the images. As the model-based tracking system has the limitation on curved objects, this paper uses shape and color information of the object.

In advance, a range-finder or stereo cameras are used to measure the shape of the given object to prepare a CG-model of the object. Many CG images are needed to be rendered with varied motion parameters to this model. The motion parameters of the CG images are adequated to minimize the difference between these CG images and captured images. And the motion tracking of the real object is achieved when the minimization is converged.

The images of a moving object change as the pose and position of the object. To model the motion parameter for the CG images, it is processed as follows. A gradient constraint equation from [HW88] is described as

$$I(x, y, t) = I(x + \delta x, y + \delta y, t + \delta t). \tag{3.1}$$

But different from it, the equation of the image in this paper is denoted by the posture parameters as

$$I = I(p_1, \cdots, p_n). \tag{3.2}$$

Here n is degrees of freedom of the object motion. To solve this equation for determining optical flow, it can be rewritten as

$$\frac{dI}{dt} \simeq \sum_{i=1}^{n} \frac{\partial I}{\partial p_i} \frac{dp_i}{dt}. \tag{3.3}$$

The motion parameters p_i are unique if the object is rigig body. The number of the pixels of the object in image is much larger than the degrees of freedom n, therefore this equation can be solved by using least squares method. In this paper, the described method is to track the motion of the rigid object. Therefore, the number of the parameters is 6. When the shape of the object is already measured, CG images whose pose and position is identical to the real object must completely coincide with the real images.

However, when the motion is towarding the optical axis of the camera, it is difficult to measured with single camera. When some different motion look similar from the camera, it is not so stable when the number of the motion parameter increases. Besides, it is also hard to track multiple objects, which are occluded with each other. Therefore, a multiple viewpoint camera system is proposed to conquer this weakness.

It is assumed that the system consists of m cameras. Each image from the camera is captured by a PC, which are connected via LAN together. To estimate the object pose and position, the summation of least squares error from all images must be minimized. However, this method still has the disadvantage that the change of the environment light condition affects.

Tracking Humans Interacting with the Environment Using Efficient Hierarchical Sampling and Layered Observation Models

In the paper from [HW88], a markerless tracking system for unconstrained human motions is presented, which are typical for everyday manipulation tasks. The system from [HW88] is capable of tracking a high dimensinal human model (51 DOF) without constricting the type of motion and the need for training sequences.

Different from the approaches introduced before, the approach from this paper does not incorporates purely with model-based tracking or appearance-based tracking. It can be composed of two key componets that substantially contribute to the accurancy and reliability of the system.

First, a sophisticated hierarchical sampling strategy for recursive Bayesian estimation that combines partioning with annealing strategies to enable efficient search in the presence of many local maxima. Second, an appearance model that allows for implicitly dealing with the environmental occlusions.

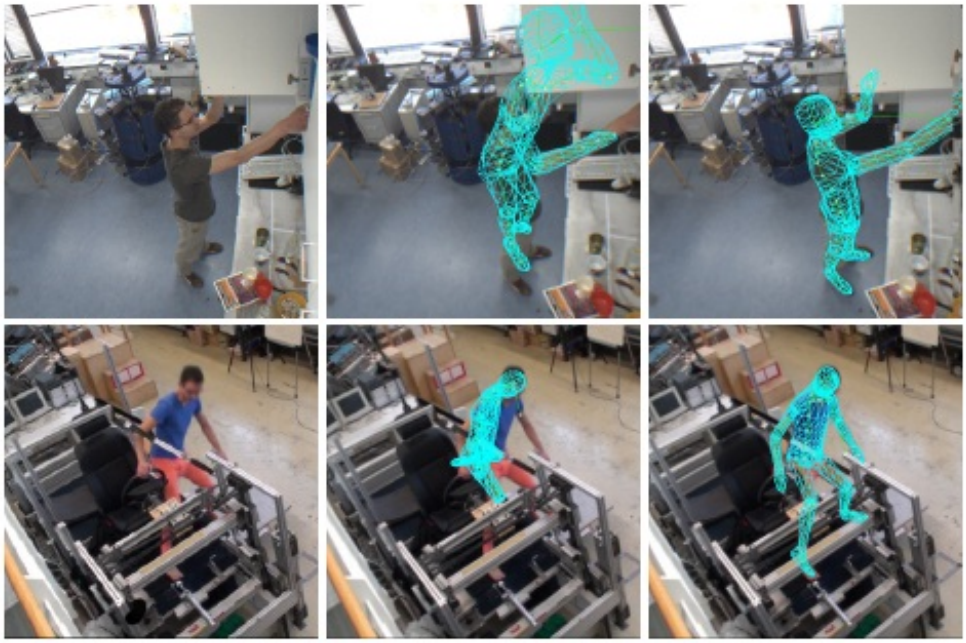

Figure 3.4: Two challenging setups for HMC featuring dynamic environments and occlusions. The center column shows the failure of shape-based methods. The right column shows the results using our appearance model with implicit environment modeling.

A working system is presented in this paper to extracting high-dimensional human motion representations from challenging sequences (see Figure 3.4). It exhibit not only the reliable performance in a large variety of scenarios with arbitrary types of motions but also the ability at segmentation of manipulated objects from the moving human and occlusion effects.

Although there are still steps for manual initialization for the first pose of human, the necessary amount of user input for both pre- and postprocessing is smaller than the marker-based tracking systems.

Concluded from the three different approaches from the previous paragraphs, they all have the advantages and disadvantes for different situation.

For the approach from [Dru02], it has the advantage in accuracy and ease of programming and implementation. But in turn, it also has the limitation that it depends on a coarse initialization of hand localization and can only deals with piecewise rigid polyhedral structures.

In the meanwhile, the approach from [MHS06] can handle curved objects and smooth texture with a appearance-based tracking system. But it also has the weakness that the change of environmental light affects on.

Compared these two approaches, the approach from [BB09] present a markerless tracking system, which is aim at tracking a high-dimensional human model (51 DOF) without constricting the type of motion and the need for training sequences. It deals also with the manipulation activities involving dynamic objects and frequent occlusions with a shape-based tracking method.

But for our tasks, the object manipulation by the robot considerates only the tracking of the object and the robot end-effector. It is not necessary to model a complex structures as [BB09]. For our cases, a simple method for the pose estimation with hand-eye coordination is developed. The details of the method will be introduced in Chapter 4.

Chapter 4

Methods of Hand-Eye-Coordination for Pose Estimation

As discussed in chapter 2, the actual position and orientation of the **TCP** frame may differ from the position and orientation obtained with forward kinematics. Several causes of this problem are possible. For instance, the DH parameters for each link may not calibrated accurately. Another possibility is the elasticity of robot segments due to the light-weight nature of the construction or the (missing) stiffness of the joint. Accordingly, the actual position of **TCP** tends downwards by some milimeters to even some centimeters depending on the actual weight and the gravity.

For object-manipulation tasks, not only the position and orientation of the object is needed but also the position and orientation of the **TCP**. The **TCP** frame is the reference frame used for manipulation with a gripper that is attached to the **TCP**. Deviations of the actual from the commanded frame may lead to inaccurate navigation or even the failure of the manipulation task. One solution to this kind of problems is a more accurate knowledge of the actual **TCP** frame.

In chapter 3, several methods have been presented that estimate or track the position and orientation of an object in space solely by methods of image processing or that estimate the position and orientation of the target in combination with the actuated structures. In this thesis, a so called hand eye coordination method was developed. In contrast to the methods introduced in chapter 3, the method in this thesis focuses on the specific problem for the robot „$Rollin'Justin$“.

Despite complex methods of extracting the texture and shape of the **TCP**, the method in this thesis is kept simple. It relies on the localization of landmarks, which are glued upon the surface of the gripper. The landmarks used in this thesis are in a 2D chess-board pattern with black and white colours (see figure 4.1).

By using the stereo cameras, which are mounted in the head of the robot „$Rollin'Justin$“,

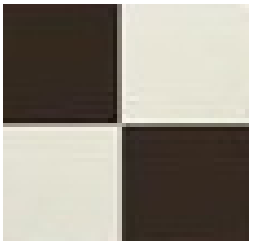

Figure 4.1: Landmark: chess board shape

the positions of the Body (the landmarks are interested in this case) in **Camera** frame can be obtained. In the case of three markers are extracted, the transformation matrix ${}^{\text{Camera}}T_{\text{Body}}$ can be calculated. Based on the knowledges of forward kinematic model and the DH parameters of the joints in „*Rollin'Justin*“, the transformation matrix ${}^{\text{Camera}}T_{\text{TCP}}$ can be also gained.

Under this conditions, we can get transformation matrix ${}^{\text{TCP}}T_{\text{Body}}$ by multiplication of the inverse transformation matrix ${}^{\text{Camera}}T_{\text{TCP}}$ and transformation matrix ${}^{\text{Camera}}T_{\text{Body}}$. As the relative position from **TCP** to **Body** is fixed, by many different observations we can optimize and get the relative accurate position of **TCP** for new observations. The details of the method in this case will be introduced in section 4.1.

Since the transformation matrix ${}^{\text{Camera}}T_{\text{Body}}$ can be only obtained in the case that, at least three markers are extracted in the stereo cameras, we must consider that, these are only the ideal conditions, which cannot be fulfilled for every poses. In the reality, as the landmarks are glued upon the surface of the hand of the robot, it is possible that only one marker can be extracted when the robot arm and hand moved or rorated.

In other case, even if all three markers are within the range of the view of both cameras, they may also not well extracted in images when they are oriented or positioned in the same ray from cameras. If there are not enough markers found in **Camera** frame, only the translation vector of ${}^{\text{Camera}}T_{\text{Body}}$ can be got, but the rotation matrix are under-determined.

For these considerations, a second method was also developed to solve this problem. Different from the first method, the transformation matrix of ${}^{\text{Camera}}T_{\text{Body}}$ is not interested, but only the position of the marker in **Camera** frame. By help of the property of the elastic joints of the robot, it was assumed that there is a strengh to pull the end-effector (**TCP**) towards the orientation of the actual positions, instead of calibrating the relative position from **Body** to **TCP**, it will try to optimize the needed mass value, which can minimize the deviations between the actual extracted positions of **Body** to commaned positions. The details will be declaired in section 4.2.

Different from the state of the art methods in the field of object manipulation, we are not focusing on that, how to control the **TCP** to reach the target position (the so-called **Open-Loop-Control** system), but considering about that, how to correct or minimize the errors in an automatic control loop system.

Based on the ideas of the method mentioned before, we can get the more accurate transformation matrix ${}^{\mathrm{TCPest}}T_{\mathrm{Body}}$, in which **TCPest** is the estimated position and orientation of end-effector. Instead of the origin with forward kinematic calculated position of **TCP**, this new position **TCPest** are estimated by multiplication of the quasi accurate transformation matrix of ${}^{\mathrm{Camera}}T_{\mathrm{Body}}$, the optimized transformation matrix ${}^{\mathrm{TCPest}}T_{\mathrm{Body}}$.

By the help of the estimated transformation matrix ${}^{\mathrm{TCP}}T_{\mathrm{TCPest}}$ from the more accurate estimated position and orientation of **TCPest** to the commanded position and orientation of **TCP**, the deviations in between can be obtained. It can be used in the framework of the robot control.

The more of the feasible observations are learned in the process, the more accurate of the new estimated position and orientation of **TCPest** for new observation can be obtained. That's why an online calibration method are developed in this thesis. But the data of the observations can be increased very bigger by the time, it is also a question of choosing new feasible data and delete old data. The concrete idea of the method for online calibration will be presented in section 4.3.

4.1 Pose Estimation with 3 Markers

As introduced before, in this section, it will be emphasized on the method of pose estimation with 3 markers. It was assumed that, for each observation, the very same 3 markers can be found in the stereo cameras mounted in the head of robot „*Rollin'Justin*“.

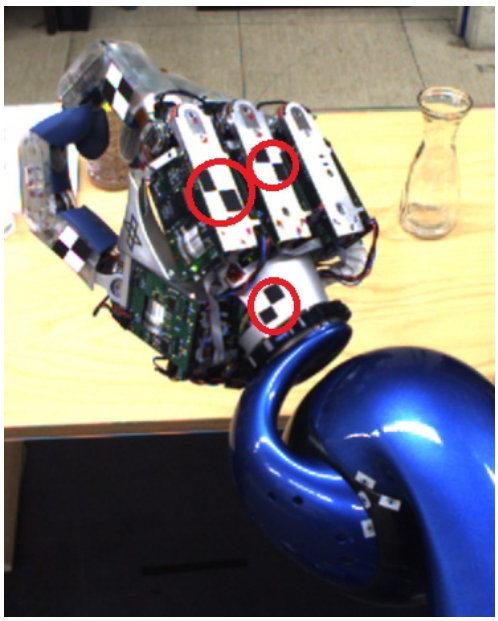

Figure 4.2: Markers on right hand

As mentioned, in order to know the position of the **Body**, we assume several marks to be glued on the right hand of the robot (see figure 4.2), in which only the 3 markers marked with red circles are interested in this thesis. For both cameras, we can get a picture like this, it leads to the first question of the method, which is how to extract the coordinates of the markers in images autonomously.

4.1.1 Corner Localization

However, traditional corner detection algorithm, do not address the specific appearance of the checkerboard pattern. In the following, an approach is outlined, that is based on point symmetries [WS10]. First of all, the color images have to be converted into grayscale with the following equation,

$$I(\mathbf{x}) = 0.299 I_{\text{red}}(\mathbf{x}) + 0.587 I_{\text{green}}(\mathbf{x}) + 0.114 I_{\text{blue}}(\mathbf{x}) \tag{4.1}$$

with I is the intersity at image points. After that, the gray scaled pictures have to be filtered in both x and y direction to have two new numerical gradient images I_X and I_Y with the equation

$$\nabla I((x,y)) = (\frac{\partial}{\partial x} I(x,y), \frac{\partial}{\partial y} I(x,y)) \tag{4.2}$$

in which I_X corresponds to dI/dx, the differences in x (horizontal) direction and I_Y corresponds to dI/dy, the differences in y (vertical) direction. The spacing between points in each direction is assumed to be one.

The algorithm here of extracting the corner points can be formulated as following:

$I(\mathbf{x})$: intersity at image coordinate $\mathbf{x}$

σ: standard deviation of 2D gaussian intergration Kernel

$G(\mathbf{x}, \sigma)$: gaussian intergration Kernel

$O(\mathbf{x})$: objective function(symmetry)

$$G(\mathbf{x}, \sigma) = \frac{1}{2\pi\sigma^2} \cdot e^{\frac{-\|\mathbf{x}\|^2}{2\sigma^2}} \tag{4.3}$$

$$O(\mathbf{x}) = \sum_{y \in \mathcal{N}(\mathcal{O})} G(\mathbf{y}, \sigma) \left\| I(\mathbf{x}+\mathbf{y}) - I(\mathbf{x}-\mathbf{y}) \right\|^2 \tag{4.4}$$

$\mathcal{N}(\mathcal{O})$: neighbourhood of point $\mathbf{x}$
e.g. $y \in (-3\sigma, -3\sigma), (-3\sigma, -3\sigma+1), \cdots (+3\sigma, +3\sigma)$ with integration radius 3.

$$\arg\min_{\mathbf{x}} O(\mathbf{x}) = \arg\min_{\mathbf{x}} \sum_{\mathbf{y}} G(\mathbf{y}, \sigma) \left\| I(\mathbf{x}+\mathbf{y}) - I(\mathbf{x}-\mathbf{y}) \right\|^2 \tag{4.5}$$

$$\Longleftrightarrow \frac{\partial}{\partial \mathbf{x}} O(\mathbf{x}+\delta\mathbf{x}) = 0 \tag{4.6}$$

This equation is solved by taking only the derivative of $I(\mathbf{x}+\mathbf{y}) - I(\mathbf{x}-\mathbf{y})$ into account, that is,

$$J(\mathbf{x}, \mathbf{y}) = \nabla I(\mathbf{x}+\mathbf{y}) - \nabla I(\mathbf{x}-\mathbf{y}). \tag{4.7}$$

$$J(\mathbf{x}) = \begin{bmatrix} J(\mathbf{x}, (-3\sigma, -3\sigma)) \\ J(\mathbf{x}, (-3\sigma, -3\sigma+1)) \\ \vdots \\ J(\mathbf{x}, (3\sigma, 0)) \\ \vdots \\ J(\mathbf{x}, (3\sigma, 3\sigma)) \end{bmatrix}, \tag{4.8}$$

and

$$G(\sigma) = \begin{bmatrix} G((-3\sigma, -3\sigma)) & 0 & \cdots & 0 \\ 0 & G((-3\sigma, -3\sigma+1)) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & G((3\sigma, 3\sigma)) \end{bmatrix} \tag{4.9}$$

We define the residual $\Delta I(\mathbf{x})$ to be,

$$\Delta I(\mathbf{x}) = (I(\mathbf{x}+(-3\sigma,-3\sigma)) - I(\mathbf{x}-(-3\sigma,-3\sigma)), I(\mathbf{x}+(-3\sigma,-3\sigma+1)) - I(\mathbf{x}-(-3\sigma,-3\sigma+1)), \cdots)^T \tag{4.10}$$

The objective function is minimized with a Gauss-Newton approximation of the hessian matrix. Finally,

$$\underbrace{(\sqrt{G} \cdot J(x_{est}))^T \cdot (\sqrt{G} \cdot J(x_{est}))}_{H_{2x2} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}, \quad where \quad c=b} \cdot \sigma_x = (\sqrt{G} \cdot J(x_{est}))^T \cdot \Delta I, \tag{4.11}$$

can be used for finding the step $\delta\mathbf{x}$ towards the location where symmetry gets optimal. Moreover, the condition of the approximated Hessian $H(\mathbf{x})$ can used as a performance index for symmetry. The better the condition, the stronger the local symmetry is. Following the philosophy of *good features to track* [Bra08], the smaller absolute Eigenvalues of the Hessian matrix is used as the indicator.

An image I_{ev} is generated, that contains the smaller absolute Eigenvalue at the corresponding location of the image I. The next step is to find the points with locally the brightest intensity in the picture I_{ev}. This is acchieved by repeatedly searching the maximum in the image and masking this point and the surrounding points for the following search.

4.1.2 Stereo Triangulation

When we got the image coordinate of markers in both cameras, we can use an algorithm of stereo triangulation from Bougunet [Bou98] to get the 3D coordinate in left camera frame. The details of algorithm are formulated below:

Let $\mathbf{X}_R = [X_R \quad Y_R \quad Z_R]$ and $\mathbf{X}_L = [X_L \quad Y_L \quad Z_L]$ are the 3D coordinates of a point in the stereo cameras. They can be related by a rigid motion equation:

$$\mathbf{X}_R = R \cdot \mathbf{X}_L + T, \tag{4.12}$$

where R and T are the rotation matrix and translation vector from left camera frame to right camera frame.

With $\mathbf{x}_R = [x_R \quad y_R \quad 1]^T = \frac{\mathbf{X}_R}{Z_R}$ and $\mathbf{x}_L = [x_L \quad y_L \quad 1]^T = \frac{\mathbf{X}_L}{Z_L}$ are the normalized coordinates from image coordinates with the projection parameters in both cameras, the equation 4.12 can be written as:

$$Z_R \cdot \mathbf{x}_R = Z_L \cdot R \cdot \mathbf{x}_L + T \tag{4.13}$$

leading to the following relation that:

$$\begin{bmatrix} -R \cdot \mathbf{x}_L & \mathbf{x}_R \end{bmatrix} \cdot \begin{bmatrix} Z_L \\ Z_R \end{bmatrix} = T \tag{4.14}$$

Let $\mathcal{A} = [-R \cdot \mathbf{x}_L \quad \mathbf{x}_R] = [-\mu \quad \mathbf{x}_R]$, where $\mu = R \cdot \mathbf{x}_L$. The least squares solution for 4.14 is then:

$$\begin{bmatrix} Z_L \\ Z_R \end{bmatrix} = (A^T \cdot A)^{-1} \cdot A^T \cdot T \tag{4.15}$$

By solving the equation 4.15, we can then get the 3D coordinate $\mathbf{X}$ in both cameras (here only the 3D coordinates in left camera frame $\mathbf{X}_L$ are interested).

4.1.3 Body Frame definition

After getting the 3D coordinates of 3 Markers in left camera frame, we have to define a new Body frame for the Markers. The idea of defining the coordinates of Markers in Body frame is to find the tranformation matrix ${}^{\text{Camera}}T_{\text{Body}}$ from camera frame to Body frame.

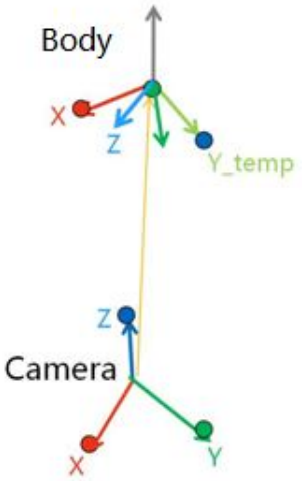

Figure 4.3: Body frame definition

As illustrated in figure 4.3, the algorithm of calculating the transformation matrix ${}^{\text{Camera}}T_{\text{Body}} = [R \quad T]$ can be formulated as following:

With the markers in **Camera** frame $\{\mathbf{o}_{1,\text{Camera}} \quad \mathbf{o}_{2,\text{Camera}} \quad \mathbf{o}_{3,\text{Camera}}\}$, the centroid point of 3 Markers $\mathbf{o}_{\bar{\text{Camera}}}$ can be used as the translation vector T from camera frame to Body frame. Moreover, the rotation matrix R can be calculated as:

$$\Delta O = [(\mathbf{o}_{2,\text{Camera}} - \mathbf{o}_{1,\text{Camera}}) \quad (\mathbf{o}_{3,\text{Camera}} - \mathbf{o}_{2,\text{Camera}}) \quad (\mathbf{o}_{3,\text{Camera}} - \mathbf{o}_{1,\text{Camera}})], \tag{4.16}$$

is the matrix composed from the difference vector of 3 Markers. Assume that the longest vector as $\mathbf{r}_{\text{x}}$ axis for Body frame, the second longest vector as temporary axis $\mathbf{r}_{\text{y,temp}}$, calculate the $\mathbf{r}_{\text{z}}$ axis by $\mathbf{r}_{\text{z}} = (\mathbf{r}_{\text{x}} \quad cross \quad \mathbf{r}_{\text{y,temp}})$ and the $\mathbf{r}_{\text{y}}$ axis by $\mathbf{r}_{\text{y}} = (\mathbf{r}_{\text{z}} \quad cross \quad \mathbf{r}_{\text{x}})$, the $\bar{\mathbf{o}_{\text{c}}}$ are seen as the origin point of the new Body frame. All the axes have to be normalized with length one. After this process, we can obtain the transformation matrix ${}^{\text{Camera}}T_{\text{Body}}$ as:

$${}^{\text{Camera}}T_{\text{Body}} = \begin{bmatrix} [\mathbf{r}_{\text{x}} & \mathbf{r}_{\text{y}} & \mathbf{r}_{\text{z}}] & \mathbf{o}_{\bar{\text{Camera}}} \\ \mathbf{0} & & & 1 \end{bmatrix}, \tag{4.17}$$

and subsequently

$$\mathbf{o}_{i,\text{Body}} = ({}^{\text{Camera}}T_{\text{Body}})^{-1} \cdot \mathbf{o}_{i,\text{Camera}} \tag{4.18}$$

4.1.4 Principal component analysis

When there are more poses are observed, it will be a problem that, for each observation, the $\mathbf{o}_{i,\text{Body}}$ in the Body frame could be different because of noisy symmetry localisation. But in reality, as always the same 3 markers are used for all observations, the position and orientation for itself doesn't change, the $\mathbf{o}_{i,\text{Body}}$ should also stay the same for the following process.

To get a more accurate or adjusted new $\mathbf{o}_{i,\text{Body}}$ for all observations, an algorithm of principle componet analysis (**PCA**) is used in this thesis. The idea of **PCA** involves a mathematical procedure that transforms a number of possibly correlated variables into a smaller number of uncorrelated variables called principal components. In this thesis, the process of this transformation are processed as follows:

Let $\mathbf{o}_{i,\text{Body},j}$ be the i-th marker point in the j-th observation and let

$$\bar{\mathbf{o}}'_{\text{Body}} = \sum_i \sum_j \mathbf{o}_{i,\text{Body},j} \tag{4.19}$$

be the centroid of all points in all observations. The covariance of all observations is given by

$$\text{Cov}(O) = \sum_i \sum_j (\mathbf{o}_{i,\text{Body},j} - \bar{\mathbf{o}}'_{\text{Body}})(\mathbf{o}_{i,\text{Body},j} - \bar{\mathbf{o}}'_{\text{Body}})^{\text{T}} \quad . \tag{4.20}$$

Let $U\Sigma V^T = \text{Cov}(O)$ denote the singular value decomposition then the principal components of all observations are given in the matrix V.

The observations $\mathbf{o}_{i,\text{Body},j}$ are transformed into the coordinate frame of the principal component analysis **BodyPCA** by

$$\mathbf{o}_{i,\text{pca},j} = V^T(\mathbf{o}_{i,\text{Body},j} - \bar{\mathbf{o}}'_{\text{Body}}) \quad . \tag{4.21}$$

The transformation matrix from **BodyPCA** to **Body** is as,

$${}^{\text{Body}}T_{\text{BodyPCA}} = \begin{bmatrix} V & \bar{\mathbf{o}}'_{\text{Body}} \\ \mathbf{0} & 1 \end{bmatrix}, \tag{4.22}$$

Extracting seperately three Markers from $\mathbf{o}_{i,\text{pca},j}$ by calculating the mean value for each marker from all the observations. The new transformation matrix ${}^{\text{Camera}}T_{\text{BodyPCA}}$ for each observations should also be formulated as

$${}^{\text{Camera}}T_{\text{BodyPCA}} = {}^{\text{Camera}}T_{\text{Body}} \cdot {}^{\text{Body}}T_{\text{BodyPCA}} \tag{4.23}$$

4.1.5 $^{\text{Camera}}T_{\text{Body}}$ Estimation

From the previous section, the defined Body frame $\{\mathbf{o}_{1,\text{Body}} \quad \mathbf{o}_{2,\text{Body}} \quad \mathbf{o}_{3,\text{Body}}\}$ should be substituted by the new frame PCA $\{\mathbf{o}_{1,\text{PCA}} \quad \mathbf{o}_{2,\text{PCA}} \quad \mathbf{o}_{3,\text{PCA}}\}$. The next step is to estimate the accurate transformation matrix $^{\text{Camera}}T_{\text{Body}}$ by an objective function.

The transformation matrix $^{\text{Camera}}T_{\text{Body}}$ is a rigid body motion transformation. According to the XYZ-Moving convention, it can be presented by six parameters $(\alpha_x, \alpha_y, \alpha_z, t_x, t_y, t_z)$, in which the parameter α_x, α_y and α_z represent the rotation parameter around the axis x, y and z, the paramter t_x, t_y and t_z represent the translation parameter along the axis x, y and z.

The idea of this estimation method is to minimize the pixel deviation in image coordinates. Here is to find the most appropriate value of these six parameters to reach the minimum of the pixel errors, with

$$\arg\min_{\alpha,\mathbf{t}} \sum_{i=1}^{\#Markers} \|\mathbf{o}_{i,\text{Image}} - f_{\text{p}}(^{\text{Camera}}T_{\text{Body}}(\alpha,\mathbf{t}) \cdot \mathbf{o}_{i,\text{Body}})\|^2 \tag{4.24}$$

$$\Longleftrightarrow \frac{\partial f_{\text{p}}(^{\text{Camera}}T_{\text{Body}} \cdot \mathbf{o}_{i,\text{Body}})}{\partial(\alpha,\mathbf{t})} = \mathbf{0} \tag{4.25}$$

where

$$f_{\text{p}}(^{\text{Camera}}T_{\text{Body}} \cdot \mathbf{o}_{i,\text{Body}}) \tag{4.26}$$

$$= f_{\text{p}}(\mathbf{o}_{i,\text{Camera}}) \tag{4.27}$$

$$\tag{4.28}$$

For each marker $\mathbf{o}_{i,\text{Camera}} = \mathbf{c} = [c_x, c_y, c_z]^T$, it has to be undistorted by a distorted camera as follows:

$$\mathbf{c}_n = \begin{bmatrix} \frac{c_x}{c_z} \\ \frac{c_y}{c_z} \end{bmatrix} \quad by \quad undistorted \quad camera \tag{4.29}$$

$$With \quad \varkappa = \sqrt{\mathbf{c}_{nx}^2 + \mathbf{c}_{ny}^2} \tag{4.30}$$

$$\mathbf{c}_d = \begin{bmatrix} \mathbf{c}_{nx} \cdot (1 + k_1 \cdot \varkappa^2 + k_2 \cdot \varkappa^4) \\ \mathbf{c}_{ny} \cdot (1 + k_1 \cdot \varkappa^2 + k_2 \cdot \varkappa^4) \end{bmatrix} \quad by \quad distorted \quad camera \tag{4.31}$$

The project function 4.26 can be followd as,

$$f_{\mathrm{p}}((c_x, c_y, c_z)) = A \cdot [c_{dx}\, c_{dy}\, 1]' \tag{4.32}$$

with

$$A = \begin{bmatrix} s_x & \gamma & u_0 \\ 0 & s_y & v_0 \end{bmatrix}, \tag{4.33}$$

in which the s_x and s_y represent the focal length in x and y coordinate, u_0 and v_0 are the principal points in x and y coordinate, and the γ is the skew factor.

After solving the function of 4.24 by the help of Nonlinear Least Squares Regression, we can have the optimized six parameters $(\alpha_x, \alpha_y, \alpha_z, t_x, t_y, t_z)$ and optimized transformation matrix ${}^{\mathrm{Camera}}T_{\mathrm{Body}}$ for each observations.

4.1.6 ${}^{\mathbf{Camera}}T_{\mathbf{TCP}}$ with forward kinematic

According to the forward kinematic modell, the transformation matrix ${}^{\mathrm{Camera}}T_{\mathrm{TCP}}$ can be calculated by multiplication of all the link transformation matrix from **Camera** to **TCP**.

The position of the **TCP** is on the last joint of the right or left arm, the position of the left **Camera** is on a fixed position to the last joint of the neck (the transformation matrix ${}^{\mathrm{Camera}}T_{\mathrm{head1}}$ is known), the position of the robot basis **Basis** is assumed to be the first joint of the torso.

According to the structure of the „*Rollin'Justin*“ from section 2.3, there are 3 joints for torso, 2 joints for head&neck, 7 joints for both arms. The transformation matrix ${}^{\mathrm{Camera}}T_{\mathrm{TCP}}$ can be calculated as follows:

$$
\begin{aligned}
{}^{\text{torso2}}T_{\text{Basis}} &= {}^{\text{torso2}}\,T_{\text{torso1}}(q_{\text{torso2}}) \cdot {}^{\text{torso1}}\,T_{\text{torso0}}(q_{\text{torso1}}) \cdot {}^{\text{torso0}}\,T_{\text{Basis}}(q_{\text{torso0}}) \\
{}^{\text{headBaseFrame}}T_{\text{torso2}} &= {}^{\text{headDockingFrame}}\,T_{\text{torso2}} \cdot {}^{\text{torso2}}\,T_{\text{Basis}} \\
{}^{\text{armBaseFrame}}T_{\text{torso2}} &= {}^{\text{armDockingFrame}}\,T_{\text{torso2}} \cdot {}^{\text{torso2}}\,T_{\text{Basis}} \\
{}^{\text{Camera}}T_{\text{Basis}} &= {}^{\text{Camera}}\,T_{\text{head1}} \cdot \quad \cdots \\
&\quad {}^{\text{head1}}T_{\text{head0}}(q_{\text{head1}}) \cdot \quad \cdots \\
&\quad {}^{\text{head0}}T_{\text{headBaseFrame}}(q_{\text{head0}}) \cdot \quad \cdots \\
&\quad {}^{\text{headBaseFrame}}T_{\text{torso2}} \\
{}^{\text{TCP}}T_{\text{Basis}} &= {}^{\text{arm6}}\,T_{\text{arm5}}(q_{\text{arm6}}) \cdot \quad \cdots \\
&\quad {}^{\text{arm5}}T_{\text{arm4}}(q_{\text{arm5}}) \cdot \quad \cdots \\
&\quad \cdots \\
&\quad {}^{\text{arm1}}T_{\text{arm0}}(q_{\text{arm1}}) \cdot \quad \cdots \\
&\quad {}^{\text{arm0}}T_{\text{armBaseFrame}}(q_{\text{arm0}}) \cdot \quad \cdots \\
&\quad {}^{\text{armBaseFrame}}T_{\text{torso2}}
\end{aligned}
$$

$$
\Longrightarrow {}^{\text{Camera}}\,T_{\text{TCP}} = {}^{\text{Camera}}\,T_{\text{Basis}} * ({}^{\text{TCP}}T_{\text{Basis}})^{-1}, \tag{4.34}
$$

in which the parameter $q_{...}$ represent the joint angle offset of each joint for current observation (**Pose**). The transformation matrix for each link can be calculated by DH-Transformation according to the current joint angle offset q (the same with θ, see subsection 2.2.3).

4.1.7 Estimation of ${}^{\text{TCP}}T_{\text{Body}}$

According to the previous sections, the position and orientation of the markers **Body** in **Camera** frame (transformation matrix ${}^{\text{Camera}}T_{\text{Body}}$) and the position and orientation of the end-effector **TCP** in **Camera** frame (transformation matrix ${}^{\text{Camera}}T_{\text{TCP}}$) for each observation can be obtained seperately by the means of image processing in stereo cameras and kinematic modell. It can be followed that the transformation matrix

$$
{}^{\text{TCP}}T_{\text{Body}} = ({}^{\text{Camera}}T_{\text{TCP}})^{-1} \cdot {}^{\text{Camera}}\,T_{\text{Body}}. \tag{4.35}
$$

In an ideal situation, the transformation matrix ${}^{\text{TCP}}T_{\text{Body}}$ (the position and orientation of the **Body** in **TCP** frame) is assumed to be stable and unchanged along the time as long as the same markers are used in same location. But since we know the property of the light-weight-robot of the robot arms from the previous chapter, it is known that the actual

reached position and orientation of the **TCP** in **Basis** frame (the transformation matrix ${}^{\text{Basis}}T_{\text{TCP}}$) deviates from the commanded position.

The deviations could be different in different **Pose**. This scenario is assumed to be caused by the property of the arms. While the transformation matrix ${}^{\text{Basis}}T_{\text{Camera}}$ is assumed to be sufficiently accurate, the position and orientation of **TCP** in Camera frame is not accurate enough. (that means, the transformation matrix ${}^{\text{Camera}}T_{\text{TCP}}$ is not reliable).

The target of this thesis is to estimate the accurate position and orientation of **TCPest**, and then therewith to minimize the deviations between the commanded position of **TCP** and the actual position of **TCPest**. That means, both the rotation and translation errors from the transformation matrix ${}^{\text{TCP}}T_{\text{TCPest}}$ for each new observation is to be minimized.

Here in the thesis, the position of the **Body** in **Camera** frame is assumed to be deviated in a small enough order of magnitude. Based on this precondition, the target of the method can be seen as finding the transformation matrix ${}^{\text{TCPest}}T_{\text{Body}}$.

As known, the transformation matrix ${}^{\text{TCPest}}T_{\text{Body}}$ is also a rigid body transformation. It can be represented by six parameters $(\alpha_x, \alpha_y, \alpha_z, t_x, t_y, t_z)$, which composed of three rotaion parameters $(\alpha_x, \alpha_y, \alpha_z)$ around the X, Y and Z axis, and three translation paramters (t_x, t_y, t_z) along the X, Y and Z axis. By means of optimization method, these six parameters are used to be optimized in different equations, to be analyzed and get the estimated transformation matrix ${}^{\text{TCPest}}T_{\text{Body}}$.

In this thesis, three different objective functions are evaluated that are minimized with respect to the parameters of ${}^{\text{TCPest}}T_{\text{Body}}$:

Objective function for angle-axis errors: minimize the angle-axis error based on the equation 4.35

Objective function for 3D marker-measurement errors: minimize the errors in 3D **Camera** coordinate system

Objective function for 2D image-measurement errors : minimize the errors in 2D **Image** coordinate system

Each method has different advantages or disadvantages in the reality. The details of the methods are going to be introduced in the following paragraphs.

Objective function for angle-axis errors

As illustrated in figure 4.4, the known transformation matrix ${}^{\text{Basis}}T_{\text{TCP}}$, ${}^{\text{Basis}}T_{\text{Camera}}$ and ${}^{\text{Camera}}T_{\text{Body}}$ are filled with color red. And with the equation

$$ {}^{\text{Camera}}T_{\text{TCP}} = {}^{\text{Basis}}T_{\text{Camera}}^{-1} \cdot {}^{\text{Basis}}T_{\text{TCP}}, \tag{4.36} $$

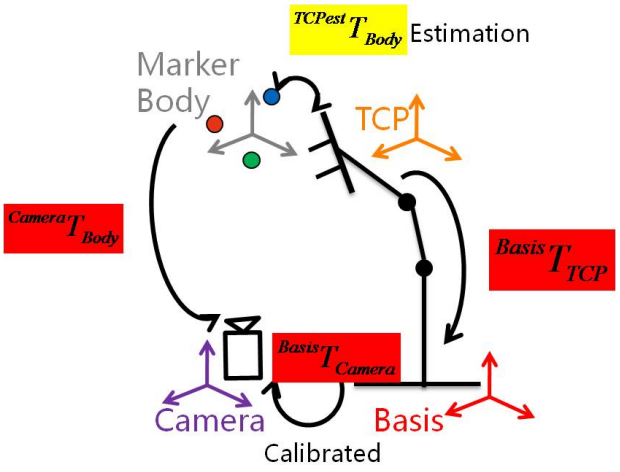

Figure 4.4: Objective function for angle-axis errors

the unknown transformation matrix ${}^{\text{TCPest}}T_{\text{Body}}$, which is filled with color yellow in figure 4.4, can be got for the beginning according to the equation 4.35, which can be reformulated as:

$$({}^{\text{TCPest}}T_{\text{TCP}})_{4x4} = ({}^{\text{TCPest}}T_{\text{Body}})_{4x4} \cdot ({}^{\text{Camera}}T_{\text{TCP}})^{-1}_{4x4} \cdot ({}^{\text{Camera}}T_{\text{Body}})_{4x4}. \tag{4.37}$$

With a single observation

$${}^{\text{TCPest}}T_{\text{TCP}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.38}$$

can be obtained. For more observations the coordinate frame **TCPest** is searched that minimizes an appropriate norm or error function of ${}^{\text{TCPest}}T_{\text{TCP}}$. Hereafter, the angle and norm of the translation vector of ${}^{\text{TCPest}}T_{\text{TCP}}$ are minimized as follows:

This resulting 4x4 matrix ${}^{\text{TCPest}}T_{\text{TCP}}$ can also be represented by a vector of three translation parameters (T_x, T_y, T_z) and a vector of rotation parameters. By the means of Angle-Axis convention (which means that the rotation is not around X, Y and Z axis, but around a determinant axis K), it can be represented by a vector of (K_x, K_y, K_z) and rotation angle θ.

As known that, the transformation matrix ${}^{\text{TCPest}}T_{\text{Body}}$ can be represented by three translation paramters (t_x, t_y, t_z) and three rotation parameters $(\alpha_x, \alpha_y, \alpha_z)$. Therefore it can be calibrated by optimizing these six parameters in equation 4.39.

$$\arg\min_{\alpha,\mathbf{t}} \sum_{i=1}^{\#observations} L_{\mathrm{AA}}({}^{\mathrm{Camera}}T_{\mathrm{TCP,i}} \cdot {}^{\mathrm{TCPest}}T_{\mathrm{Body}}(\alpha,\mathbf{t}) \cdot {}^{\mathrm{Camera}}T^{-1}_{\mathrm{Body,i}}), \tag{4.39}$$

to minimize the angle-axis and translation errors from **TCPest** frame to **Body** frame, in which

$$L_{\mathrm{AA}}((R \quad \mathbf{t})) = \|T\|^2 + \|\theta\|^2 \tag{4.40}$$

whereas, $R = \exp(\theta * (K_x, K_y, K_z))$

$\|T\|^2$: squared of norm of translation vector (T_x, T_y, T_z)

$\|\theta\|^2$: squared of rotation angle around axis (K_x, K_y, K_z)

There are two preconditions for this optimization method:

1. At least 3 markers are extracted for each observations to obtain the transformation matrix ${}^{\mathrm{Camera}}T_{\mathrm{Body}}$
2. At least 3 feasible Poses are observed, so that there will be enough values (2x#Observations) to optimize the six parameters of ${}^{\mathrm{TCPest}}T_{\mathrm{Body}}$

Objective function for 3D marker-measurement errors

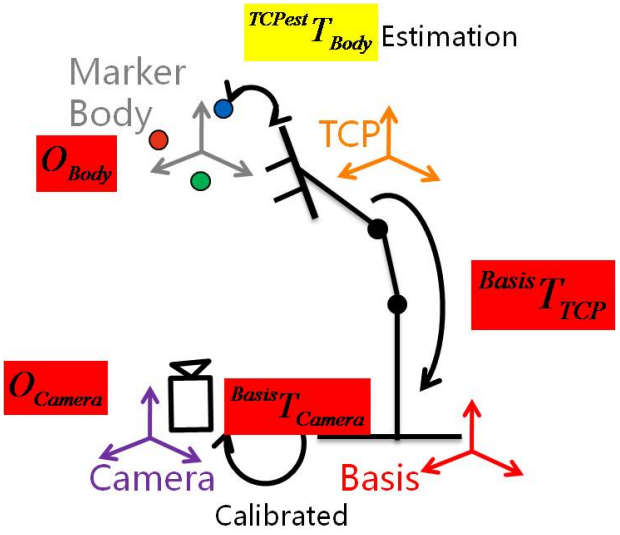

Figure 4.5: Objective function for 3D marker-measurement errors

Different from the Objective function for angle-axis errors, this method doesn't require the pre-knowledge of the transformation matrix ${}^{\text{Camera}}T_{\text{Body}}$, but the marker points in **Body** frame and correspondingly stereo triangulated points in 3D **Camera** frame.

As illustrated in figure 4.5, the marker points in Camera fram with $\mathbf{o}_{\text{Camera}}$ and the ones in Body frame with $\mathbf{o}_{\text{Body}}$ are known, which are marked with background color red. It's the same for the transformation matrix ${}^{\text{Basis}}T_{\text{TCP}}$ and ${}^{\text{Basis}}T_{\text{Camera}}$. The transformation matrix ${}^{\text{TCPest}}T_{\text{Body}}$ with color green is unknown.

The idea of this method, is to minimize the errors in **Camera** coordinate system. That means, the deviations between the extracted and stereo triangulated marker points in **Camera** coordinate system and the ones with forward kinematics (see below).

$$\arg\min_{\alpha,\mathbf{t}} \sum_{i=1}^{\#Markers} \sum_{j=1}^{\#Observations} \|\mathbf{o}_{i,\text{Camera},j} - {}^{\text{Camera}}T_{\text{TCP},j} \cdot {}^{\text{TCPest}}T_{\text{Body}}(\alpha,\mathbf{t}) \cdot \mathbf{o}_{i,\text{Body},j}\|^2, \quad (4.41)$$

in which $\mathbf{o}_{i,A,j}$ represents the i-th marker points in j-th observation for frame A. Compared to the optimization method in previous paragraph, this method has the advantages that, it doesn't need the transformation matrix ${}^{\text{Camera}}T_{\text{Body}}$ in advanced. Moreover, it has more values (#Markers X #Observations) to be used for the parameters optimization.

But it also has the disadvantage that, there could be errors in the the stereo triangulated values in 3D **Camera** coordinate system. That't why, an Objective function for 2D marker-measurement errors was also developed.

Objective function for 2D marker-measurement errors

In this optimization method, the markers points in 2D **Image** coordinate system are used. Similiar with the Objective function for 3D marker-measurement errors, the markers points in **Body** frame and the transformation matrix ${}^{\text{Camera}}T_{\text{TCP}}$ with forward kinematics for each observations are as known (see figure 4.6).

The idea of this method is to minimize the errors between the extracted marker points in 2D **Image** coordinate system and the projected ones with forward kinematics.

$$\arg\min_{\alpha,\mathbf{t}} \sum_{i=1}^{\#Markers} \sum_{j=1}^{\#Observations} \|\mathbf{o}_{i,\text{Image},j} - f_{\text{p}}({}^{\text{Camera}}T_{\text{TCP},j} \cdot {}^{\text{TCPest}}T_{\text{Body}}(\alpha,\mathbf{t}) \cdot \mathbf{o}_{i,\text{Body},j})\|^2, \quad (4.42)$$

in which, the function $f_{\text{p}}(x)$ can be got from equations 4.26 and 4.33. Compared to the optimization method of minimizing the errors in 3D **Camera** coordinate system, this method depends on the assumption, that the pixel errors in images are the dominant errors.

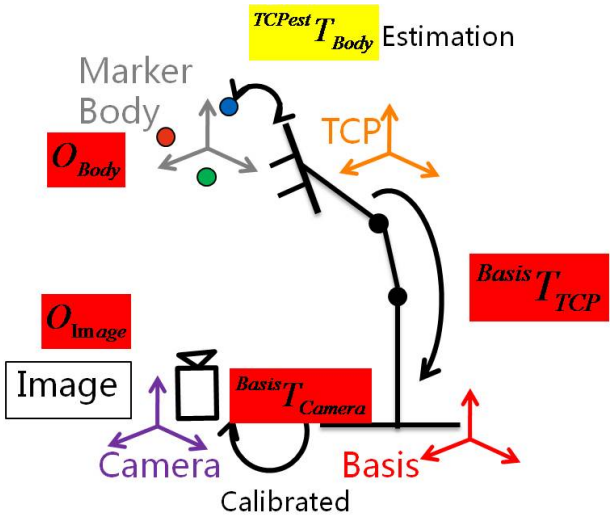

Figure 4.6: Objective function for 2D marker-measurement errors

But on the other side, the same with the Objective function for 3D marker-measurement errors, they all need the marker points in **Body** frame ($\mathbf{o}_{i,\text{Body},j}$) according the equation 4.41 and 4.42. As mentioned in previous sections, it describes the same problems for marker frame definition and ${}^{\text{Camera}}T_{\text{Body}}$ optimization.

From this point of view, it means that, at least three markers should be extracted as precondition. Otherwise, in the case that only one marker is extracted in advanced, only the translation vector $\mathbf{t}$ of transformation matrix ${}^{\text{TCPest}}T_{\text{Body}}$ could be optimized. ${}^{\text{TCP}}T_{\text{Body}}$ is considered as under-determined. For this consideration, a pose estimation method with one marker was developed for this problem, which will be introduced in 4.2.

4.2 Pose Estimation with 1 Marker

As mentioned in last section, we know that, there are not always three markers can be extracted for each observation. There are several possibilities. For example, because of the illumination in bad situation, or the markers oriented in a bad direction from **Camera**. As introduced, not only for the optimizatin method of transformation matrix ${}^{\text{Camera}}T_{\text{Body}}$ but also for the later optimization method for transformation matrix ${}^{\text{TCPest}}T_{\text{Body}}$, at least three extracted markers are needed. Due to such reasons, another thought of pose estimation with one marker was throwed out.

According to the elasiticity and flexible nature of the joints and links of „*Rollin'Justin*“, a modell based on the knowledge from [KB02] was developed. Before introducing the

detail of the method for this thesis, a few foundations of flexible joints and links will be introduced in advanced. The modeling is based on using lumped flexibility and assuming small deformation.

It is assumed, that we can apply a strength on material to cause the deformation. For the links, we consider bending and torsion effects, but we neglect the traction and compression effects. The joint flexibility is represented by a linear translational spring along the axis of a prismatic joint, and by a linear torsional spring about the joint axis for a revolute joint [KB02].

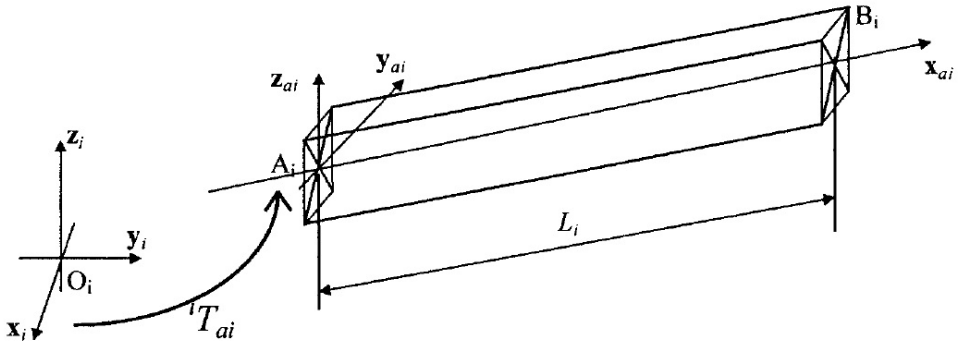

Figure 4.7: Shape of Link i

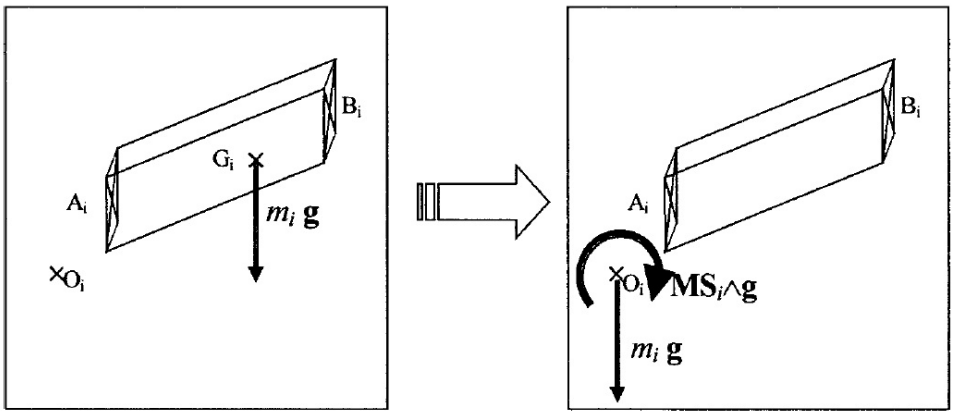

Figure 4.8: Gravity effect on link i

In order to compute the elastic deformation of the links and joints, we need to compute the forces and moments (wrenches) exerted on each link due to gravity and the surrounding links. The details of the equations to do the calculation is introduced in [KB02]. For the links deformation, it was considered that the elastic displacements are proportional to the forces and moments (Hook's Law).

The deformation at B_i can be decomposed into a bending about the y_{ai} axis (due to the force ${}^{ai}F_{z_{B_i}}$ and the moment ${}^{ai}M_{y_{B_i}}$), a bending about the z_{ai} axis (due to the force ${}^{ai}F_{y_{B_i}}$ and the moment ${}^{ai}M_{z_{B_i}}$), and a torsion about the x_{ai} axis (due to the moment ${}^{ai}M_{x_{B_i}}$) (see figure 4.8). The resulting displacement of the terminal frame is related with these forces and moments, the elasicity coefficients K_{ti}, the link length L_i.

On the other side, for joints deformation, a global elasticity coefficient K_{az_i} is defined for each joint i, which representing the elasticity of the position gain of the control loop and the elasticity of the transmission system. For a revolute joint, the rotational deformation of frame R_i is about axis z_i. It is taken proportional to K_{az_i} and to the moment ${}^{i}M_{z_{O_i}}$ exerted about the axis z_i :

$$ {}^{i}\delta_{a_{z_i}}(O_i) = \sigma_i K_{a_{z_i}} {}^{i}M_{z_{O_i}}. \tag{4.43} $$

Our method of pose estimation with one marker is based on the equation 4.43. As the joint deformation from equation 4.43 is due to the forces and moments exerted on each link from gravity effect. It was assumed that, a controllable force could also be exerted on the link of **TCP** pulling towards the direction of deviation vectors between commanded and actual position of the extracted Marker (see figure 4.9).

The resulting joint deformation could be transformed with forward kinematic into new more accurate transformation matrix ${}^{\text{Camera}}T_{\text{TCP}}$. The basic idea of this method is to find the required mass m (exerted on **TCP**) to cause these forces and moments, and so that can minimize the errors in translation vector from actual position of the extracted marker to commanded position of this marker.

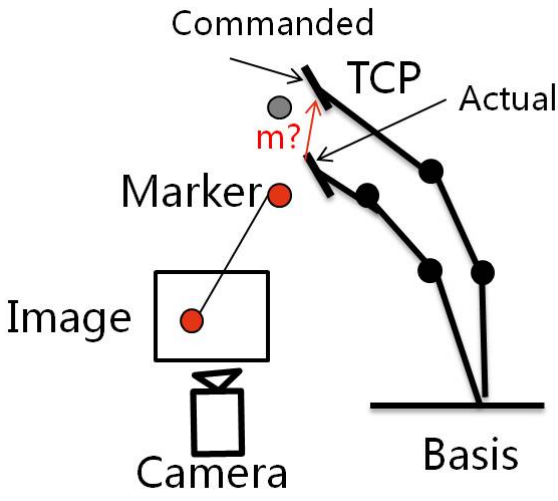

Figure 4.9: Pose Estimation with one Marker

According to the equations from [KB02], it was assumed that, the exerted forces can cause the torques τ as

$$\tau(m) = \text{torquesForSpring}(m\text{x}\mathbf{v}, \mathbf{q}), \tag{4.44}$$

in which, m represent the effective mass value of the force (in initial $m = 0$), $\mathbf{q}$ are the commanded joint angles configuration. Instead of gravity vector g, here the vector v is used, which represent the difference vector between commanded position of the marker and actual position of the extracted marker in **Camera** frame. It can be calculated as follows:

$$\mathbf{v} = {}^{\text{Camera}}T_{\text{Basis}}(q)\,{}^{\text{Basis}}T_{\text{TCP}}(q) \cdot {}^{\text{TCPest}}T_{\text{Body}} \cdot \mathbf{o}_{\text{Body}} - \cdot\mathbf{o}_{\text{Camera}}, \tag{4.45}$$

in which the transformation matrix ${}^{\text{TCP}}T_{\text{Body}}$ is an intial guess. As only the direction from the actual position of the marker to commanded position of this marker is interested, the vector $\mathbf{v}$ should be normalized by dividing the norm of $\mathbf{v}$: $\|\mathbf{v}\|$.

Since here only the translation vector from ${}^{\text{TCP}}T_{\text{Body}}$ is interested, it can be calculated from 4.35 with the under-determint transformation matrix ${}^{\text{Camera}}T_{\text{Body}}$.

Afterwards, the new joint angles configuration, which effected by the torques τ is calculated with the elasiticity coefficients K_i by

$$\tilde{q}_i(m) = q_i - \frac{\tau_i(m)}{K_i}, \tag{4.46}$$

in which $i \in (\text{arm0}, \cdots, \text{arm6})$, only the joints on arms are interested. As the joint angles configuration changed, so changed the transformation matrix ${}^{\text{Basis}}T_{\text{TCP}}(\tilde{q})$ as well.

Therefore the new translation vector $\tilde{\mathbf{v}}(m)$ causd by an effective mass m can be calculated as:

$$\tilde{\mathbf{v}}(m) = {}^{\text{Camera}}T_{\text{Basis}}(q) \cdot {}^{\text{Basis}}T_{\text{TCP}}(\tilde{\mathbf{q}}(m)) \cdot {}^{\text{TCP}}T_{\text{Body}} \cdot \mathbf{o}_{\text{Body}} - \mathbf{o}_{\text{Camera}} \tag{4.47}$$

By minimizing the norm in $\tilde{\mathbf{v}}(m)$ with parameter m,

$$\arg\min_{m} \| {}^{\text{Camera}}T_{\text{Basis}}(q) \cdot {}^{\text{Basis}}T_{\text{TCP}}(\tilde{\mathbf{q}}(m)) \cdot {}^{\text{TCP}}T_{\text{Body}} \cdot \mathbf{o}_{\text{Body}} - \mathbf{o}_{\text{Camera}} \|^2 \tag{4.48}$$

it was assumed that, we can obtain the required force, which can pull the joint of **TCP** towards the commanded position.

4.3 Online Calibration

As mentioned in the previous sections, we know that, the pose estimation method with one marker does not work so well as the pose estimation method with three markers does. Therefore, in the following procedures for online calibration, it was assumed, all three used markers could be found in stereo camera images. In another word, we only work with the poses, for which all three makrers could be found in camera images.

The procedure of the pose estimation can be coarsely illustrated as the figure 4.10.

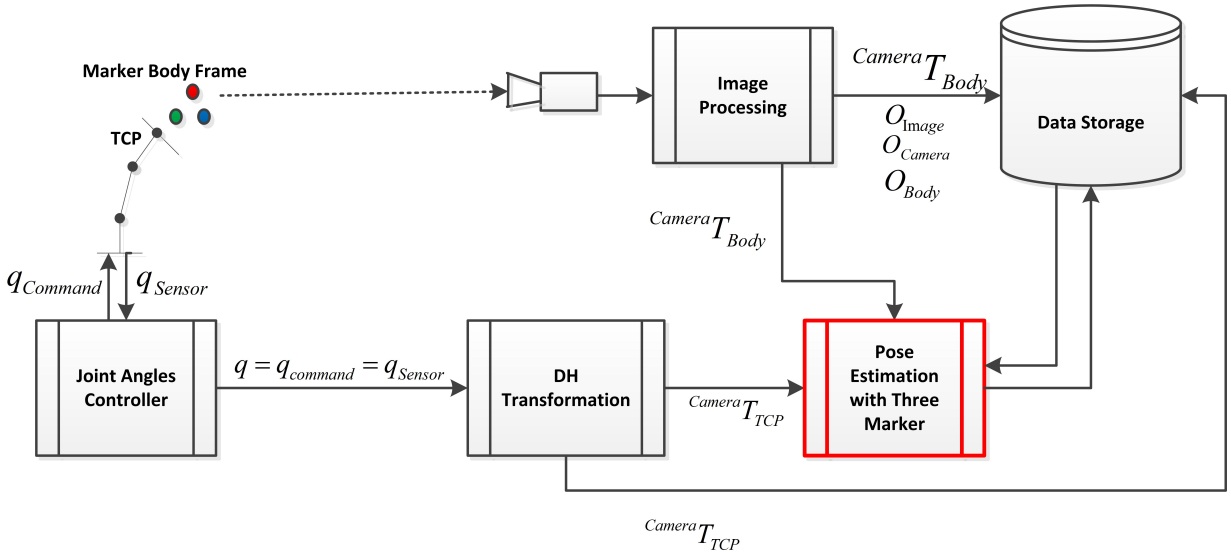

Figure 4.10: Whole Procedure Structure

The robot pose can be on one side commanded by a joint angle controller with $\mathbf{q}_{\text{command}}$. And on the other side, the robot can also move spontaneously or be pulled manuelly, and be observed by joint angles sensor of each joint to get the current joint angles configuration $\mathbf{q}_{\text{sensor}}$. The output of the joint angle controller, the joint angles configuration $\mathbf{q}$ should be the same as either $\mathbf{q}_{\text{command}}$ or $\mathbf{q}_{\text{sensor}}$.

Subsequently, the transformation matrix ${}^{\text{Camera}}T_{\text{TCP}}$ can be calculated with forward kinematics and the DH-transformation from subsection 2.2.3. As mentioned before, this transformation matrix is considered not accurate enough due the nature of the Light-Weight-Robot-Arm. In contrast, by extracting the marker points in camera images, the transformation matrix ${}^{\text{Camera}}T_{\text{Body}}$ can be obtained with more accurancy compared to ${}^{\text{Camera}}T_{\text{TCP}}$.

According to the section 4.1, we know that, the following data:

- the marker points in **Camera** coordinate system $\mathbf{o}_{\text{Camera}}$
- the marker points in **Image** coordinate system $\mathbf{o}_{\text{Image}}$
- the marker points in defined **Body** coordinate system $\mathbf{o}_{\text{Body}}$

- the transformation matrix ${}^{\text{Camera}}T_{\text{Body}}$
- the transformation matrix ${}^{\text{Camera}}T_{\text{TCP}}$

will be used for the estimation of the transformation matrix ${}^{\text{TCP}}T_{\text{Body}}$.

And it was known that, the more feasible observations have been recorded and learned in the process, the better the estimation method for ${}^{\text{TCP}}T_{\text{Body}}$ works. For online calibration method, all the data used for estimation of the transformation matrix ${}^{\text{TCP}}T_{\text{Body}}$ should be saved in a storage disk.

The question is, what is going to be found out after the whole procedure of online calibration method. As mentioned in Chapter 2, the method in thesis is different from either the classic **Open-Loop-Control** method or the classic **Closed-Loop-Control** method. The idea of this method is based on the ideas of both the **Open-Loop-Control** method and the **Closed-Loop-Control** method.

It trys to find out a way, which can not only estimate the position and orientation of the **TCPest**, but also minimize the deviations between the commanded position and orientation of **TCP**, and the estimated actual position and orientation of TCP. It will need a lot of observations to do the pose estimation method.

For the pose estimation method itself, it can be seperated into two different situations:

1. Pose Estimation for first Observation
2. Pose Estimation for following Observations

4.3.1 Pose Estimation for first Observation

For online calibration, we should run the programm as autonomously as possible. As we have glued more than three markers upon the robot hand, one problem of online calibration method is how to find the very same three marker in the very same order.

For first observation, it will be easier. The order of the three markers, which are needed for the processing, will be choosed manuelly in left camera image by running. The only problem is how to find out the corresponding three markers in right camera image.

As the transformation matrix ${}^{\text{Camera,L}}T_{\text{Camera,R}} = [\mathbf{R}\,\mathbf{t}]$ is known, the corresponding problem can be solved by the help of essential matrix $\mathbf{E}$ as follows:

Define the essential matrix $\mathbf{E}$,

$$\mathbf{E} = \mathbf{R}[\mathbf{t}]_x, \tag{4.49}$$

where $[\mathbf{t}]_x$ is the matrix representation of the cross product with $\mathbf{t}$.

Let the three marker points from left camera be homogeneous normalized in image coordinates as $\mathbf{o}_{1,\text{cameraLeft}}, \mathbf{o}_{2,\text{cameraLeft}}, \mathbf{o}_{3,\text{cameraLeft}}$. And all the extracted points from right camera be homogeneous normalized as $\mathbf{o}_{1,\text{cameraRight}}, \cdots, \mathbf{o}_{n,\text{cameraRight}}$. For each i-th point from right camera, it will be compared with each j-th point from left camera with the equation,

$$\left| \frac{\mathbf{o}'_{i,\text{cameraRight}} \mathbf{E} \mathbf{o}_{j,\text{cameraLeft}}}{\|\mathbf{o}_{i,\text{cameraRight}}\| \|\mathbf{o}_{j,\text{cameraLeft}}\|} \right| \leq \epsilon. \tag{4.50}$$

If the right side of the equation equals or smaller than a computing limit value **eps**, the i-th point from right camera is considered as the corresponding point to the j-th point from left camera. Do this process, until all three corresponding pairs could be found in both cameras.

In this case, the three markers could be stereo triangulated in 3D left camera coordinate system, and therewith the transformation matrix ${}^{\text{Camera}}T_{\text{Body}}$ could be obtained. If there couldn't be enough markers found, then discard current observation and try another pose, until there will be a feasible observation as the first observation.

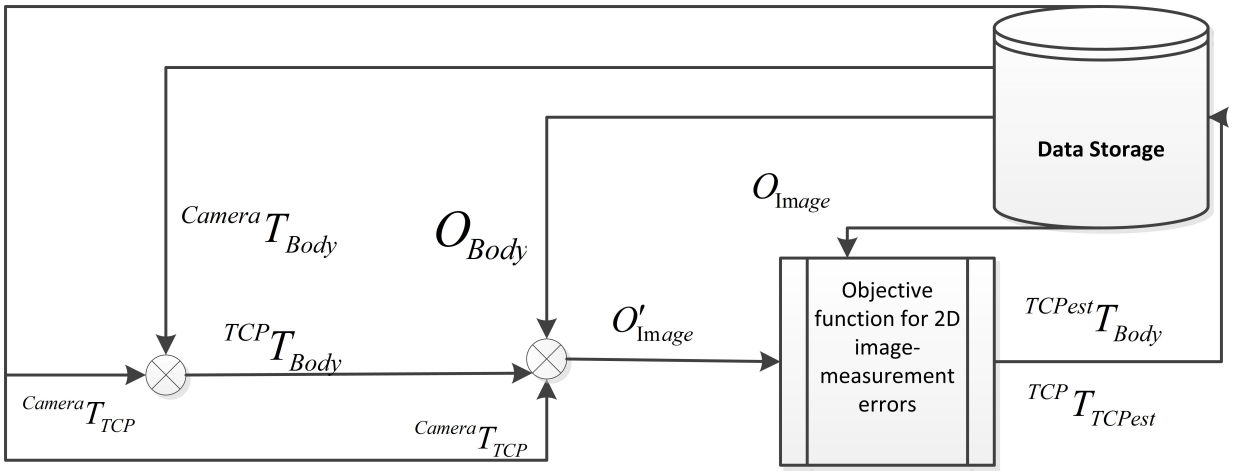

Figure 4.11: Pose Estimation for first Observation

As illustrated in 4.11, for first observation, the transformation matrix ${}^{\text{TCP}}T_{\text{Body}}$ can be calculated by equation 4.35 as an initial guess. With the Objective function for 2D marker-measurement errors from 4.1.7, we can get the estimated transformation matrix ${}^{\text{TCPest}}T_{\text{Body}}$ and transformation matrix ${}^{\text{TCP}}T_{\text{TCPest}}$, which should be calculated as:

$${}^{\text{TCP}}T_{\text{TCPest}} = ({}^{\text{Camera}}T_{\text{TCP}})^{-1} \cdot {}^{\text{Camera}}T_{\text{Body}} \cdot ({}^{\text{TCPest}}T_{\text{Body}})^{-1}. \tag{4.51}$$

But for the first observation, it should be the same as unit matrix $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$.

4.3.2 Pose Estimation for following Observations

As mentioned before, the order of the three markers only can be manuelly dertermined by first observation. For the following observations, neither the order of the markers nor the position of the markers are known before. By processing the images from both cameras, there could be lot more points found in images than the three we want. Without a manuelly labelling, it was uncertain, which three points from the camera images in which order are considered as the three markers we used. Therefore, it has to do a preprocess to deal with this problem:

For each extracted point from left camera image $\mathbf{o}_{m,\text{imageLeft}}$ $(m \in (1 \cdots M))$, it will be compared with each extracted point from right camera image $\mathbf{o}_{n,\text{imageRight}}$ $(n \in (1 \cdots N))$. Before the comparision, all of the points have to be normalized in homogenous corresponding camera coordinate system to get $\mathbf{o}_{m,\text{cameraLeft}}$ and $\mathbf{o}_{n,\text{cameraRight}}$.

The comparision is realized by the equation 4.50. If the result is equals or smaller than a limit value **eps**, the point $\mathbf{o}_{m,\text{cameraLeft}}$ and $\mathbf{o}_{n,\text{cameraRight}}$ are considered as the correspoinding pair and will be stereo triangulated and saved in a temporary storage disk as $\mathbf{o}_{m,\text{Camera}}$ for current observation.

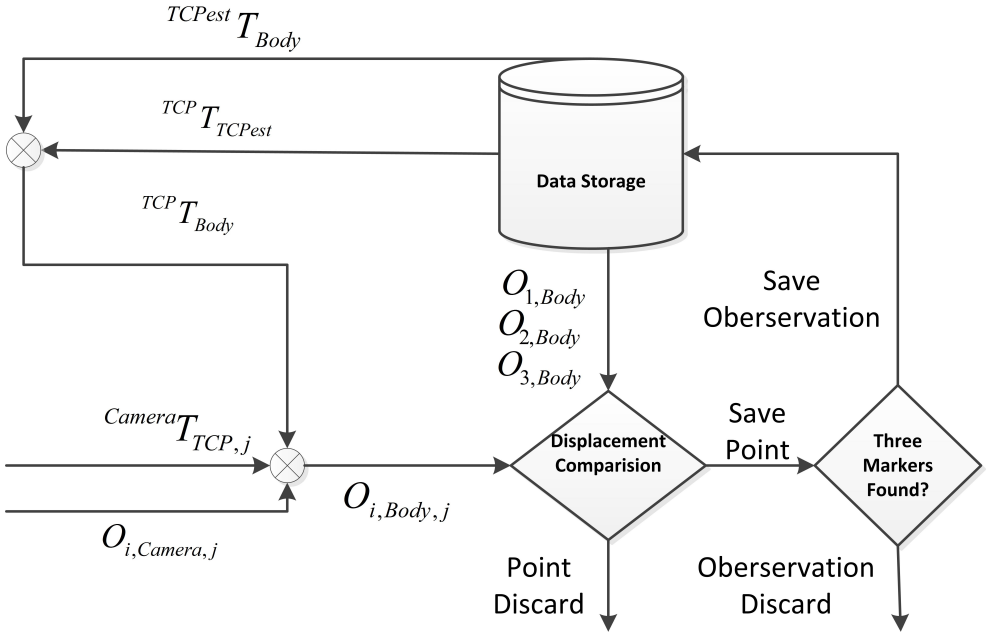

Figure 4.12: Preprocess for finding the three markers

As illustrated in figure 4.12, for observation j, for each i-th point $\mathbf{o}_{i,\text{Camera},j}$ from the array $\mathbf{o}_{\text{camera}}$ will be transformed from **Camera** coordinate system into **Body** coordinate system. It can be fulfilled by the equation:

$$\mathbf{o}_{i,\text{Body},j} = ({}^{\text{Camera}}T_{\text{TCP},j} \cdot {}^{\text{TCP}}T_{\text{TCPest}} \cdot {}^{\text{TCPest}}T_{\text{Body}})^{-1} \cdot \mathbf{o}_{i,\text{Camera},j}, \tag{4.52}$$

where the transformation matrix ${}^{\text{Camera}}T_{\text{TCP},j}$ is obtained with forward kinematics for j-th observation. The transformation matrix ${}^{\text{TCP}}T_{\text{TCPest}}$ and ${}^{\text{TCPest}}T_{\text{Body}}$ are obtained from data storage disk, which are estimated from (j-1)-th observation.

For each marker point $\mathbf{o}_{\text{Body}} = [\mathbf{o}_{1,\text{Body}}\ \mathbf{o}_{2,\text{Body}}\ \mathbf{o}_{3,\text{Body}}]$ defined from previous observations, it will be compared with the each extracted point in **Body** coordinate system $\mathbf{o}_{i,\text{Body},j}$:

$$\delta_{m-ij} = \|\mathbf{o}_{m,\text{Body}} - \mathbf{o}_{i,\text{Body},j}\|, \tag{4.53}$$

where $m \in [1 \cdots 3]$. By analyzing the displacement vector δ_{m-ij}, if the minumum value in the vector is found to be smaller than 0.005 (0.5 cm), it will be considered to be corresponding m-th marker point and to be saved in a temporary storage memory. Doing the processing for all the three markers until all of the them be found or all of the extracted corresponing pairs have been compared. If there are three marker points found from the array, the current j-th observation is considered as feasible and the data will be saved in storage disk for the following process. Otherwise, it will be discarded and a new pose will be commanded.

When we have more than one observation's data, the pose estimation method will be different from the method with only one observation.

As illustrated in figure 4.13, the main difference is that, the transformation matrix ${}^{\text{TCP}}T_{\text{Body}}$ is not calculated by the equation 4.35, but obtained from the storage disk, which is the output transformation matrix ${}^{\text{TCPest}}T_{\text{Body}}$ from last observations.

The same with the pose estimation method with one observation, here also using the objective function for 2D marker-measurement errors from 4.1.7. The output of the pose estimation procedure ${}^{\text{TCPest}}T_{\text{Body}}$ and the transformation matrix ${}^{\text{TCP}}T_{\text{TCPest}}$ by equation 4.51 will be stored in the memory data at the end of the process.

According to the knowledges before, we know that, the deviation transformation matrix ${}^{\text{TCP}}T_{\text{TCPest}}$ won't be an unit matrix anymore as for the first observation. For each new observation, the defined Body coordinate system could be kind of different because of the noises in extracted three marker points. And the transformation matrix ${}^{\text{Camera}}T_{\text{TCP}}$ could also derivates from ${}^{\text{Camera}}T_{\text{TCPest}}$ in different ranges.

The target of the whole online calibration method of hand-eye coordination is to find out, if the parameters of the transformation matrix ${}^{\text{TCPest}}T_{\text{Body}}$ could be converged into stable values. By the method of online calibration, as every process could run automatically

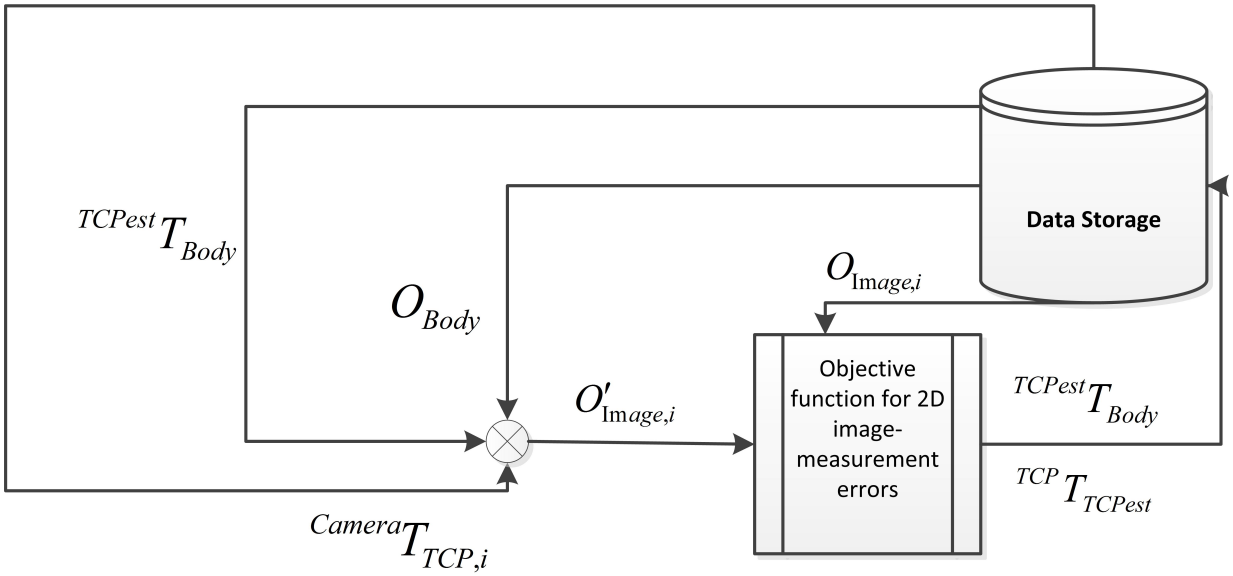

Figure 4.13: Pose Estimation with more than one Observation

except for the first observation, therewith the efficiency could help recording more and more data.

As known that, the more poses are observed, the more accurate of the data from pose estimation could be obtained. By analyzing the six translation and rotation parameters of the transformation matrix ${}^{\mathrm{TCPest}}T_{\mathrm{Body}}$ and ${}^{\mathrm{TCP}}T_{\mathrm{TCPest}}$, we may find out, how behave the deviations during the process of online calibration. The details will be declaired in the following chapter 5.2.3.

Chapter 5

Experiments

The experiments of the methods for hand-eye coordination, which was developed in this thesis for the robot „*Rollin'Justin*“, are seperated into two phases:

1. Experiments in Simulation
2. Experiments with „*Rollin'Justin*“

which represent seperately the first half and second half of my work on my diploma thesis.

5.1 Experiments in Simulation

In the first half of my work on this diploma thesis, the methods for hand-eye coordination was developed, and be programmed in Matlab Software environments [Mat]. Before testing the methods with real data, the movements and Poses of the robot are simulated and observed in a virtual viewer. It was to help by understanding, how to command the robot movement with the joint angle configurations.

5.1.1 Virtual Viewer

The Virtual Viewer is based on the software from [Vir]. The structure of the virtual robot „*Rollin'Justin*“with upper body was modeled with the VRML language by the colleges from DLR.

By sending the transformation matrix from robot basis to each link with a defined port number, the Poses of the robot can be stretched in the available areas. It can help to observe, which joint angle configurations are feasible to make the required movements.

With the transformation matrices, which are calculated with the foward kinematics and the default DH-Parameter of each link from A.1, the initial Pose of the robot can be seen in figure 5.1.

Figure 5.1: Initial Pose

With different joint angle offsets for the right arm, the Pose of the right arm can be changed in a different position (see figure 5.2). Extended from the basic structure of the robot, three small balls with different colors are modeled and attached to the last joint of right arm (TCP). It can be seen as the marker bodies, which are to be used in the hand-eye coordination methods. They have the fixed relative position and orientation from TCP to themselves.

By sending the transformation matrix from robot basis to the marker bodies frame with another defined port number, the marker bodies can also move around with the corresponding movement of the robot arm. It helps to find out the feasible joint angle configurations, which make the marker bodies can be seen from the robot cameras in the head.

5.1.2 Programm

After testing plenty of different joint angle configurations, six feasible joint angle configurations are recorded for the following experiment. As known, the transformation matrix ${}^{\mathrm{TCP}}T_{\mathrm{Body}}$ and the marker points in Body coordinate system are already defined for the

Figure 5.2: Commanded Pose

modeling of the marker bodies in virtual viewer. The procedure of method in simulation environment can be seen as follows:

1. Calculate the transformation matrix ${}^{\text{Camera}}T_{\text{TCP}}$ with the recorded joint angle configurations
2. Calculate the transformation matrix ${}^{\text{Camera}}T_{\text{Body}}$ with

$$ {}^{\text{Camera}}T_{\text{Body}} = {}^{\text{Camera}}\ T_{\text{TCP}} \cdot {}^{\text{TCP}}\ T_{\text{Body}} \tag{5.1} $$

3. assume the marker points are -1cm in Z-axis deviated from the commanded position in camera frame, which means,

$$ \mathbf{o}_{\text{Camera}} = {}^{\text{Camera}}\ T_{\text{Body}} \cdot \mathbf{o}_{\text{Body}} - \begin{pmatrix} 0 \\ 0 \\ 0.01 \end{pmatrix} \tag{5.2} $$

4. project the marker points in image coordinate system to get the $\mathbf{o}_{\text{Image}}$ with the equation 4.26
5. Minimize the deviations with different methods in different situations
   - With 3 markers, estimate the transformation matrix ${}^{\text{TCPest}}T_{\text{Body}}$ seperately with the three objective functions from 4.1.7

- With 1 marker, estimate the required mass value m so as to minimize the translation deviations in Camera frame with the method from 4.2.

5.1.3 Results

With 3 Markers

As for each observations, the deviations stay the same with -1cm in Z-axis, the objective functions from 4.1.7 can function well with very small residuals. The details of the results are not so insterested and will not be introduced here.

With 1 Marker

The results of the optimization method with one marker from 4.2 are declaired as follows:

v_array before the estimation method:

$$v\_array\_before = \begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -0.01 & -0.01 & -0.01 & -0.01 & -0.01 & -0.01 \end{matrix}$$

v_array after the estimation method:

$$v\_array\_after = \begin{matrix} 0.0010 & 0.0001 & -0.0000 & 0.0011 & 0.0002 & -0.0004 \\ 0.0043 & 0.0043 & 0.0048 & 0.0046 & 0.0034 & 0.0033 \\ -0.0073 & -0.0076 & -0.0063 & -0.0065 & -0.0087 & -0.0087 \end{matrix}$$

in which the v_array represents the translation vectors of the deviations in Camera frame for each observation (here six observations are recorded). The corresponding effective mass value vector $\vec{m}$ to minimize the deviations for each observation after the estimation method are seen as:

$$\vec{m} = -118.4823 \quad -155.9772 \quad -236.2245 \quad -392.6863 \quad -98.9942 \quad -171.1147$$

Based on these results, it can be seen that, the v_array before the estimation method are same as the assumed deviations for 1cm. Calculate the norm value from v_array for each observation, it was shown that:

$$\|v\_array\| = 0.0085 \quad 0.0087 \quad 0.0080 \quad 0.0081 \quad 0.0093 \quad 0.0093$$

It can be seen that, the v_array after the estimation method doesn't make big difference from the v_array before the estimation method in norm. The differences in norm of each

vector in v_array for each observation is about 0.14cm. There are still about 0.86cm deviations exist for each observation. The small improvement could be caused from the elastic joint model.

While the marker points in camera frame $\mathbf{o}_{\text{Camera}}$ are calculated without considering the elastic joint model, the resulting v_array of the deviations in Camera frame is calculated according to the equation 4.45 with the transformation matrix ${}^{\text{Camera}}T_{\text{TCP}}$ based on the new joint angle configurations from elastic joint model. This causes already some deviations.

Another reason may according to the equation 4.46, the new joint angle configurations won't make big difference from the original ones as the elastic coefficients are much bigger. As the assumed deviations are 1cm in camera frame for each observation, the method from 4.2 may not work well as the movement for each step in the optimization method is too small to reach the right direction.

5.2 Experiments with „*Rollin'Justin*“

5.2.1 General

After testing the developed methods in the simulation environments, the next step is to test them with real robot „*Rollin'Justin*“. A few markers were glued upon the surface of the robot hand. The whole experiments are also seperated with two phases:

1. Offline estimation with real data
2. Online Calibration

Offline estimation with real data

In the first phase, the robot was commanded with a few joint angle configurations to make different poses. With the cameras in the head of the robot, the images were taken and saved for the following procedures. Only the poses, which several markers can be seen in the cameras, are taken into consideration and the corresponding data of both the joint angle configurations and the torques for each joint were saved.

Different from the experiments in simulation environments, the marker points and its frame were not predefined. The procedure before the estimation method was run as follows:

1. extract the corner points in both camera images
2. label the three markers in left camera image $\mathbf{o}_{\text{Image}}$ to be used for the following procedure
3. find the correspoinding three points in right camera

4. make the stereo triangulation to determine the three marker points in camera fram $\mathbf{o}_{\text{Camera}}$
5. define the marker body frame and so as to define the $\mathbf{o}_{\text{Body}}$ and calculate the transformation matrix from Camera frame to Body frame ${}^{\text{Camera}}T_{\text{Body}}$
6. do the principal component analysis (PCA) to define a new body frame PCA and so well the $\mathbf{o}_{\text{PCA}}$ and ${}^{\text{Camera}}T_{\text{PCA}}$ (in the following PCA will be substitued with Body to make the meaning of the frame easier to understand)

The following procedures are same as the experiments in simulation environment. With three markers, the transformation matrix from estimated position and orientation of TCP (TCPest) to the Body ${}^{\text{TCPest}}T_{\text{Body}}$ were estimated with three different objective functions. With one marker, the effective mass values m were estimated for each observation to minimize the deviations between the commanded position of the marker and actual extracted position of the marker in camera frame.

Online Calibration

Based on the experiment of offline estimation with real data, it was followed by the real online experiments with the robot. The programm was running in real-time when the robot arm moved. The whole procedures were introduced from the section 4.3. The robot arms were moved by the manuelly guidance.

For each new pose, the joint angle configurations and the torques for each joint were received from the server. The images could also be grabbed from sensor server. In these experiments, the programm were processing with three markers.

In each new loop, the data were processed to check if the same three markers could be found for the following objective functions. When not enough markers were found in images, the corresponding observation will be abandoned and a new pose will be tried.

Moreover, only the robot arm moved more than 2 degrees, this pose can be considered as a new observation. In total, there were 100 observations were processed and recorded for the whole online calibration method.

5.2.2 Software Packets

The whole programm for the experiments with the real robot „$Rollin'Justin$“of online calibration was written in programming language C++ in the developing environment Eclipse [Ecl]. It could increase the efficiency by running compared to the programm in Matlab. Besides, there were also several software packages and libraries(see table 5.1) used by programming. They are not only from internet but also from the institute of Robotics and Mechatronics from German Aerospace Center (DLR).

Softwares and Libaries	Function Area
Eclipse	Developing Environment for online calibration
Matlab	Developing Environment for offline estimation
OpenCV 1.0.0	Image Processing
sensor-model 2.0.2	Image grabbing
ard	shared memory exchange
xml2	data reading and saving
GSL	scientific library

Table 5.1: Software and Library Packages

5.2.3 Results

Results under offline estimation

Although more than six observations were recorded for the experiments of offline estimation, there were only six observations taken into account.

When the programm was processed with three markers, the resulting transformation matrix from estimated position and orientation of TCP to Body frame are as follows:

By objective function for angle-axis errors:

$$ {}^{\mathrm{TCPest}}T_{\mathrm{Body}} = \begin{bmatrix} -0.9270 & -0.0924 & 0.3635 & -0.0704 \\ 0.0323 & 0.9459 & 0.3227 & 0.0100 \\ -0.3737 & 0.3109 & -0.8739 & 0.0571 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix}, $$

with the remained errors in angles and translation vectors:

$$ rms_{\mathrm{angle}} = 0.1953\, rad $$
$$ rms_{\mathrm{t}} = 0.0123\, m $$

By objective function for 3D marker-measurement errors:

$$ {}^{\mathrm{TCPest}}T_{\mathrm{Body}} = \begin{bmatrix} -0.9165 & -0.1835 & 0.3555 & -0.0747 \\ -0.0683 & 0.9473 & 0.3128 & 0.0126 \\ -0.3942 & 0.2624 & -0.8808 & 0.0565 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix}, $$

with the remained errors in camera frame:

$$ rms_{\mathrm{Camera}} = 0.0109\, m $$

By objective function for 2D image-measurement errors:

$$^{\text{TCPest}}T_{\text{Body}} = \begin{bmatrix} -0.9124 & -0.1675 & 0.3734 & -0.0663 \\ -0.0459 & 0.9485 & 0.3135 & 0.0082 \\ -0.4067 & 0.2689 & -0.8731 & 0.0590 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix},$$

with the remained errors in image coordinate system:

$$rms_{\text{Image}} = 9.1235\, pixel$$

It was shown that, with six observations, the remained errors after the estimation method are not small enough.

When the programm was processed with only one marker, the results for each observation are to be seen as follows:

The deviations of translation vector from the commanded TCP position to the actual TCP position in Camera frame v_array_before before the estimation method:

$$v\_array\_before = \begin{pmatrix} -0.0002 & -0.0047 & -0.0047 & 0.0080 & 0.0066 & 0.0042 \\ 0.0089 & 0.0003 & 0.0081 & 0.0003 & -0.0013 & -0.0007 \\ 0.0218 & -0.0048 & -0.0114 & 0.0011 & 0.0025 & -0.0091 \end{pmatrix},$$

with the norm values

$$\|v\_array\_before\| = \begin{pmatrix} 0.0236 & 0.0067 & 0.0147 & 0.0081 & 0.0072 & 0.0100 \end{pmatrix}$$

The deviations of translation vector from the commanded TCP position to the actual TCP position in Camera frame v_array_after:

$$v\_array\_after = \begin{pmatrix} 0.0019 & -0.0014 & -0.0066 & 0.0068 & 0.0057 & 0.0025 \\ 0.0037 & 0.0007 & 0.0050 & 0.0016 & -0.0007 & 0.0046 \\ 0.0226 & -0.0058 & -0.0126 & 0.0009 & 0.0024 & -0.0041 \end{pmatrix},$$

with the norm values

$$\|v\_array\_before\| = \begin{pmatrix} 0.0230 & 0.0060 & 0.0151 & 0.0070 & 0.0062 & 0.0066 \end{pmatrix},$$

and the effective mass value vector $\vec{m}$:

$$\vec{m} = \begin{pmatrix} -538.5688 & -634.4712 & 140.5460 & 102.5616 & 18.5176 & -422.7492 \end{pmatrix},$$

It was shown that, the same with the experiments in simulation environment, the results haven't made big changes in the remained deviations of translation vector $\vec{v}$ from v_array for each observation.

Results under online calibration

By the second phase of the experiments with robot „*Rollin'Justin*“, the programm is running real time with the movement of the robot. Plenty of poses are observed, and totally 100 observations are recorded.

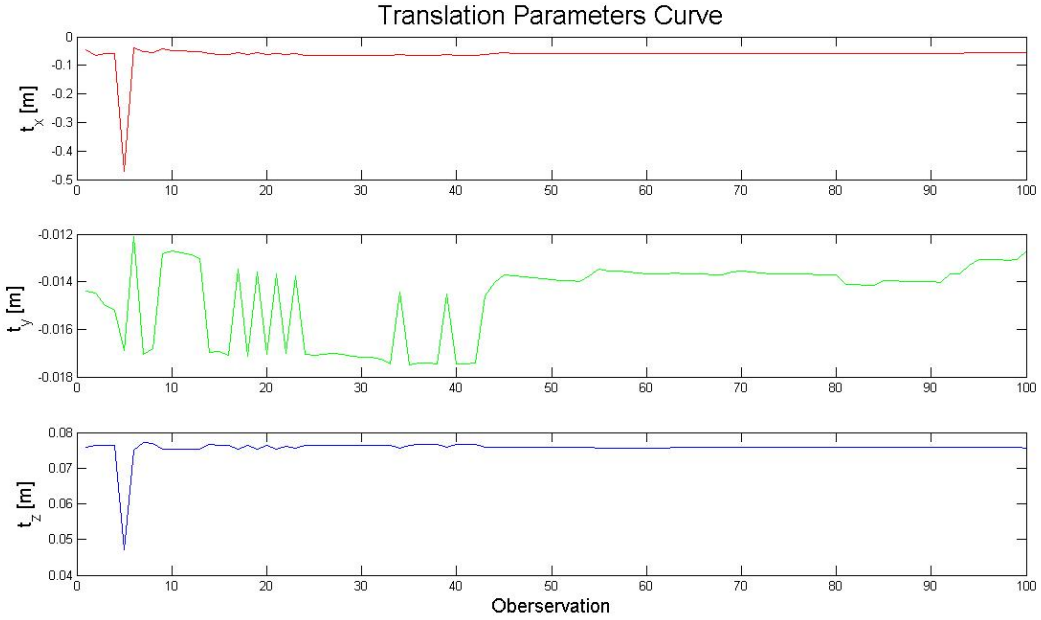

Figure 5.3: Translation Parameters Curve of ${}^{\mathrm{TCPest}}T_{\mathrm{Body}}$

The figures 5.3 and 5.4 illustrate the changes of the three translation parameters and three rotation parameters of the transformation matrix ${}^{\mathrm{TCPest}}T_{\mathrm{Body}}$ for these 100 observations.

On one hand, from the figure 5.3, it was shown that, the translation parameters of t_x and t_z are stable from beginning except for one observation at about 5th poses. The translation parameter of t_y is not stable until after about 45 observations are recorded.

On the other hand, from the figure 5.4, the situations are opposite from the situations by the three translation parameters. The rotation parameter α_y is stable after about 5 observations. But in the meanwhile, the rotation parameters of α_x and α_z are not stable until about 45 observations are recorded.

From the very beginning, it was known that, the transformation matrix ${}^{\mathrm{TCPest}}T_{\mathrm{Body}}$ was calculated with equation 4.35. It was assumed that, the transformation matrix ${}^{\mathrm{TCPest}}T_{\mathrm{Body}}$ was the same with the transformation matrix ${}^{\mathrm{TCP}}T_{\mathrm{Body}}$ for the first observation.

As known, the accutal position and orientation of TCPest is deviated from the command position and orientation of TCP. Therefore, the transformation matrix of ${}^{\mathrm{TCPest}}T_{\mathrm{Body}}$ by the first observation is not accurate. After about 45 observations, when there are enough

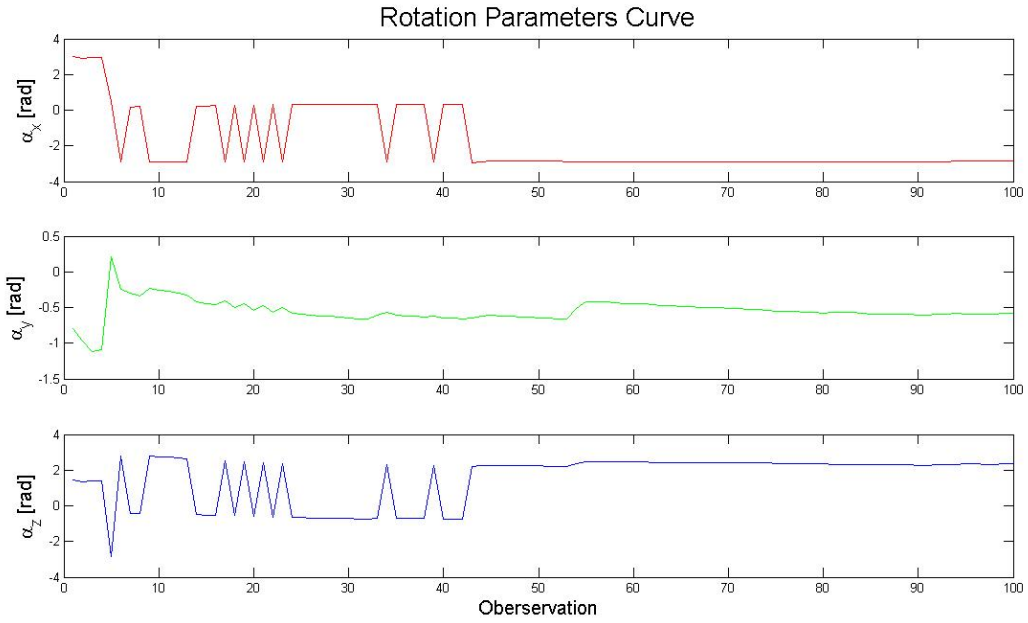

Figure 5.4: Rotation Parameters Curve of ${}^{\mathrm{TCPest}}T_{\mathrm{Body}}$

observations are recorded, all the six parameters of the ${}^{\mathrm{TCPest}}T_{\mathrm{Body}}$ are converged to stable values.

Based on the values from the estimated transformation matrix ${}^{\mathrm{TCPest}}T_{\mathrm{Body}}$, the transformation matrix ${}^{\mathrm{TCP}}T_{\mathrm{TCPest}}$ can be calculated with the equation 4.51.

The figures 5.5 and 5.6 illustrate the changes of the three translation parameters and three rotation parameters of the transformation matrix ${}^{\mathrm{TCP}}T_{\mathrm{TCPest}}$ for these 100 observations.

From the figure 5.5, it was shown that, after about 10 observations, the deviation of the translation in X-axis from commanded position of TCP to estimated position of TCP is tendential to 0. In the meanwhile, the deviation of the translation in Y-axis and Z-axis from commanded position of TCP to estimated position of TCP is tendential to 0 after about 45 observations.

On the other hand, the three rotation parameters seem to be less stable than the three translation paramters. From the figure 5.6, it was shown that, all the three rotation parameters of the transformation matrix ${}^{\mathrm{TCP}}T_{\mathrm{TCPest}}$ are tendential to 0 after about also 45 observations. But by 58th, 75th, and 98th observation, the three parameters are strongly deviated from 0.

As the transformation matrix ${}^{\mathrm{TCP}}T_{\mathrm{TCPest}}$ is calculated with the equation ${}^{\mathrm{TCP}}T_{\mathrm{TCPest}}$, it depends strongly on the accurancy of the transformation matrix ${}^{\mathrm{TCPest}}T_{\mathrm{Body}}$, ${}^{\mathrm{Camera}}T_{\mathrm{Body}}$ and the current transformation matrix of ${}^{\mathrm{Camera}}T_{\mathrm{TCP}}$.

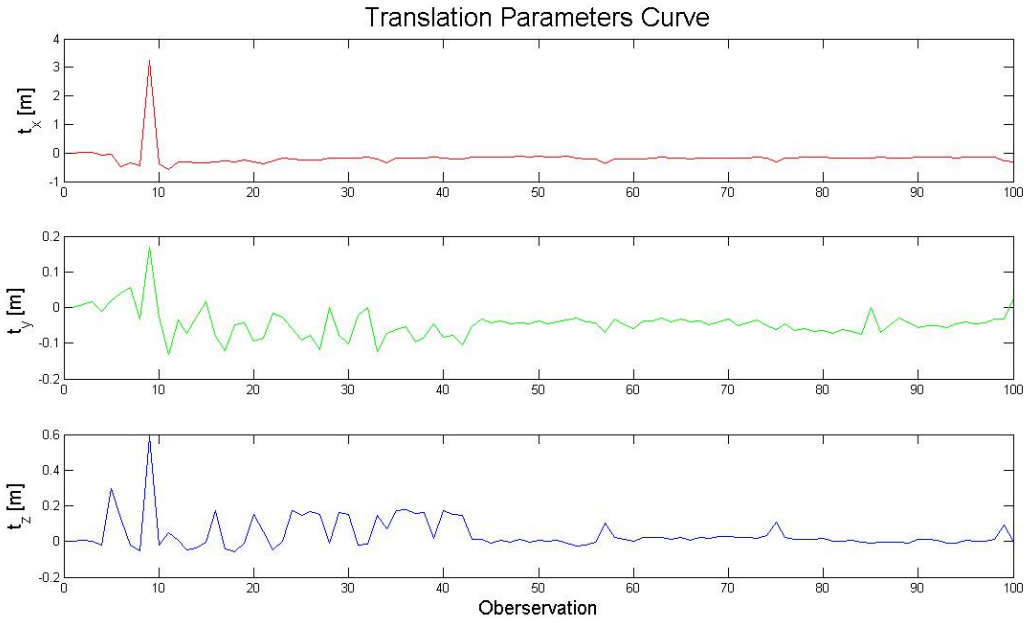

Figure 5.5: Translation Parameters Curve of ${}^{\mathrm{TCP}}T_{\mathrm{TCPest}}$

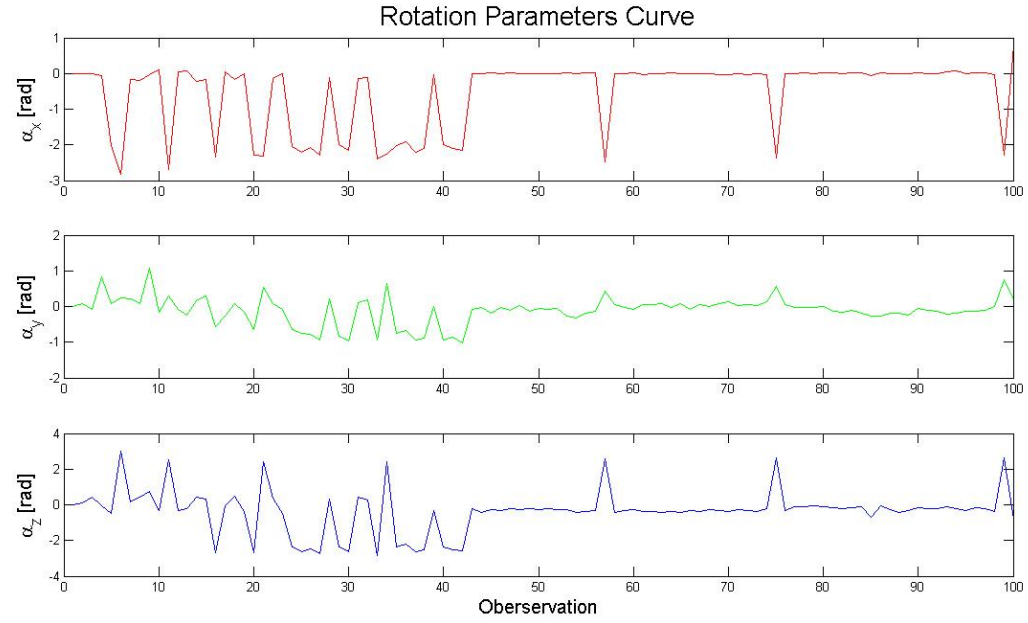

Figure 5.6: Rotation Parameters Curve of ${}^{\mathrm{TCP}}T_{\mathrm{TCPest}}$

As known that, the six parameters of the transformation matrix ${}^{\mathrm{TCPest}}T_{\mathrm{Body}}$ are converged to stable values after about 45 observations are recorded. The unstable parameters for

the observation 58, 75 and 98 could be caused by the unaccurancy of the transformation matrix ${}^{\text{Camera}}T_{\text{TCP}}$ or ${}^{\text{Camera}}T_{\text{Body}}$.

In the following figures, several images are showing the differences between the commanded position and orientation of TCP and the actual estimated position and orientation of TCPest in camera.

The three lines with color red, green and blue (RGB) represent the XYZ axis of the commanded position and orientation of TCP in camera. The three lines with color cyan, yellow and magenta (CYM) represent the XYZ axis of the estimated position and orientation of TCPest in camera.

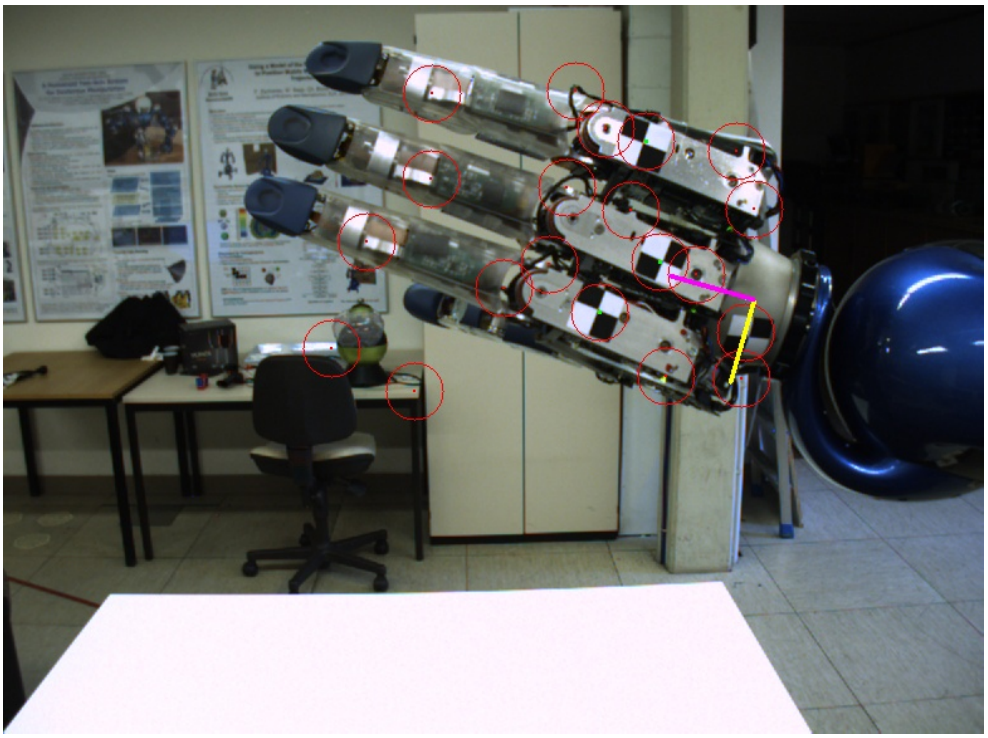

Figure 5.7: Deviation with one oberservation

In the figure 5.7, the image shows the deviation for first observation. As mentioned before, for the first observation, the estimated position and orientation of TCPest is assumed to be the same with the commanded position and orientation of TCP.

It was shown in the figure 5.7, the axes of the commanded position and orientation of TCP with color RGB coincide with the axes of the estimated position and orientation of TCPest with color CYM.

In the figure 5.8, the deviation between TCP and TCPest for the second observation was illustrated. Different from the situation by the first observation, it was shown that, the actual estimated position and orientation of TCPest deviates from the commanded position

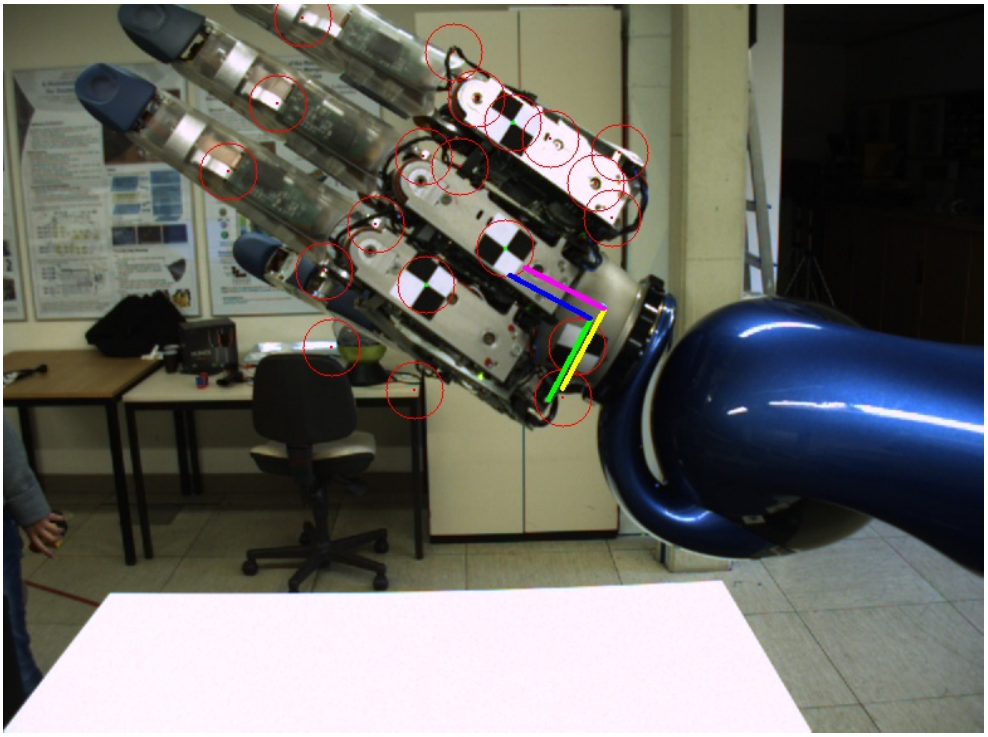

Figure 5.8: Deviation with 2 oberservations

and orientation of TCP.

But with only 2 observations, it can't be determined , if the estimated position and orientation of TCPest is accurate. More and more poses have to be observed to check stability of the values.

In the figure 5.9, it was shown that, the deviations between the estimated position and orientation of TCPest (CYM) and the commanded position and orientation of TCP (RGB) for observation 20 are much bigger than the previous ones and unreasonable.

From the figures 5.3, 5.4, 5.5 and 5.6, it was shown that, the values of the six parameters of the transformation matrix ${}^{\mathrm{TCPest}}T_{\mathrm{Body}}$ and ${}^{\mathrm{TCP}}T_{\mathrm{TCPest}}$ are unstable from about 5th observation to about 45th observation.

It causes the very unacceptable values of the estimated position and orientation of TCPest, which was shown in the figure 5.9. The reason of this phenomenon may due to the unconverged transformation matrix ${}^{\mathrm{TCPest}}T_{\mathrm{Body}}$ from the Nonlinear Least-Squares function of GSL [GSL].

In the figure 5.10, it was shown that, the deviations between the commanded position and orientation of TCP and the actual estimated position and orientation of TCPest are reasonable again.

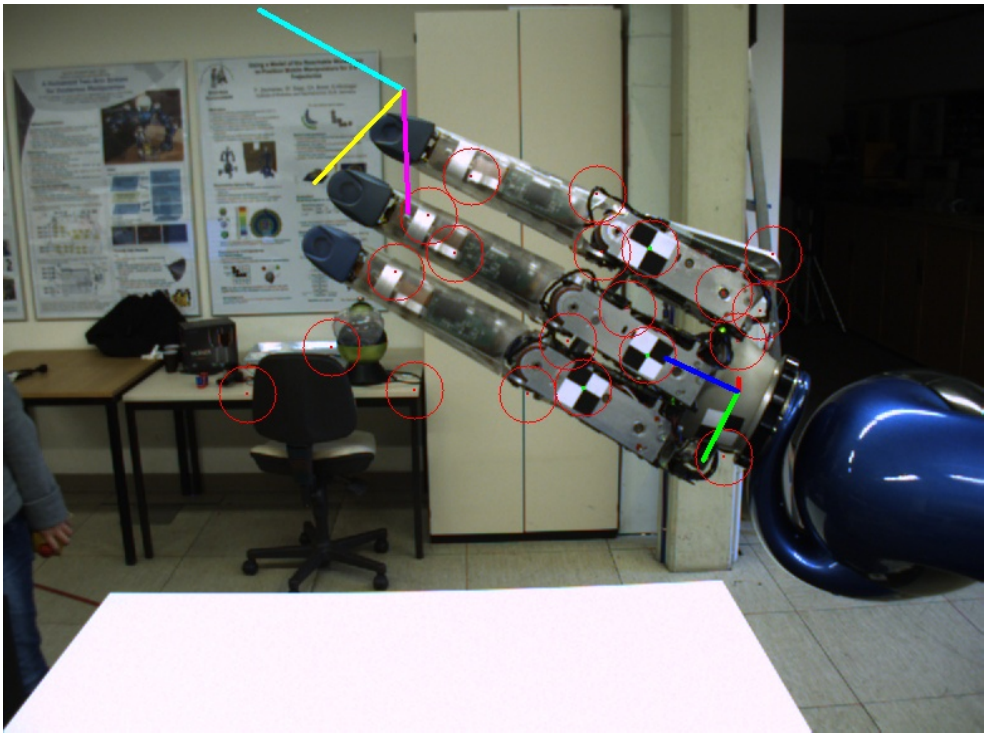

Figure 5.9: Deviation with 20 oberservations

As known from Appendix A.1, the actual position and orientation of TCP is already close to the estimated position and orientation of TCPest in the figure 5.10 with color system CYM.

As known from figures 5.3, 5.4, the values of the six parameters of the transformation matrix ${}^{\text{TCPest}}T_{\text{Body}}$ remain stable after about 45 observations. It could be said that, the estimated position and orientation of TCPest is accurate from the 45th observation on.

But as shown in figure 5.11, the estimated position and orientation of TCPest becomes large deviated again for the 75th observation. As shown in figures 5.3, 5.4, the six parameters of the transformation matrix ${}^{\text{TCPest}}T_{\text{Body}}$ is still stable by 75th observation. But in the meanwhile, as shown in figures 5.5 and 5.6, some parameters of the transformation matrix ${}^{\text{TCP}}T_{\text{TCPest}}$ deviated from the previous ones for the 75th observation.

According to the equation 4.51, ss the values of the transformation matrix ${}^{\text{TCPest}}T_{\text{Body}}$ are stable, the deviation may be caused from the unconverged transformation matrix of ${}^{\text{Camera}}T_{\text{Body}}$ from the Nonlinear Least-Squares function of GSL [GSL].

In the figure 5.12, it was shown that, the estimated position and orientation of TCPest for 95th observation is even more close to the actual position and orientation of TCP according to the Appendix A.1.

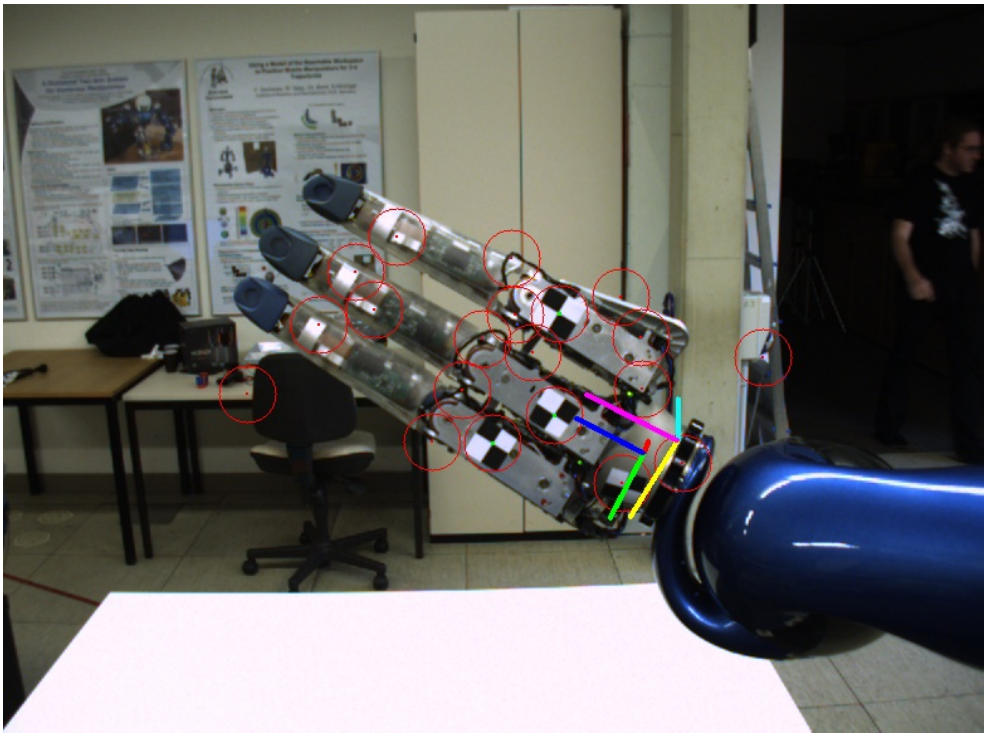

Figure 5.10: Deviation with 45 oberservations

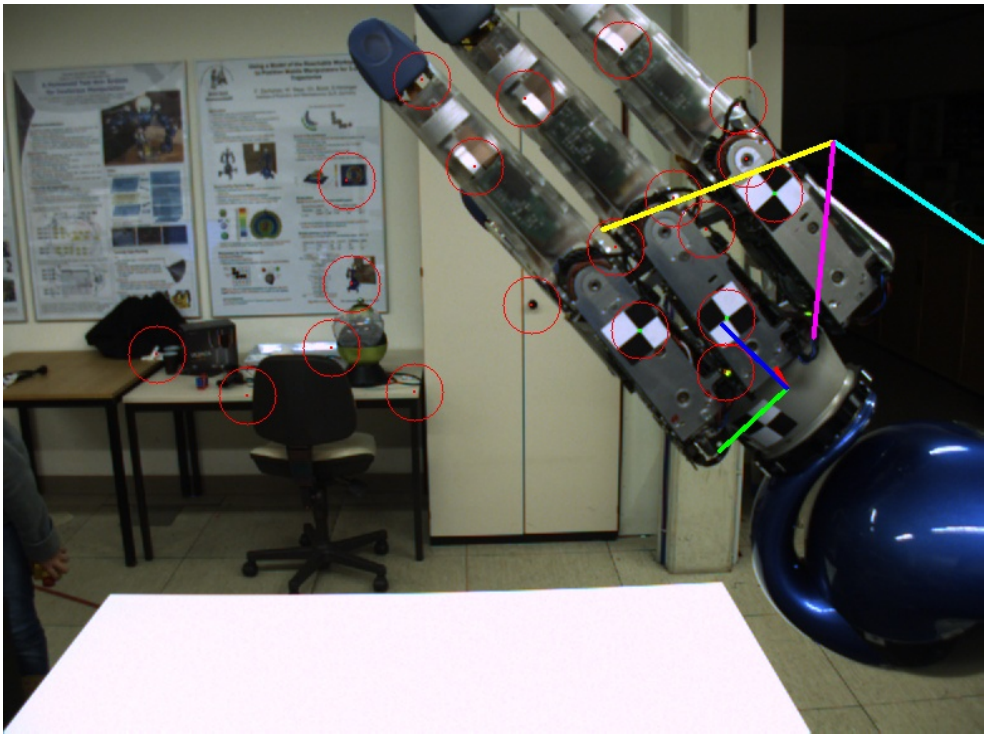

Figure 5.11: Deviation with 75 oberservations

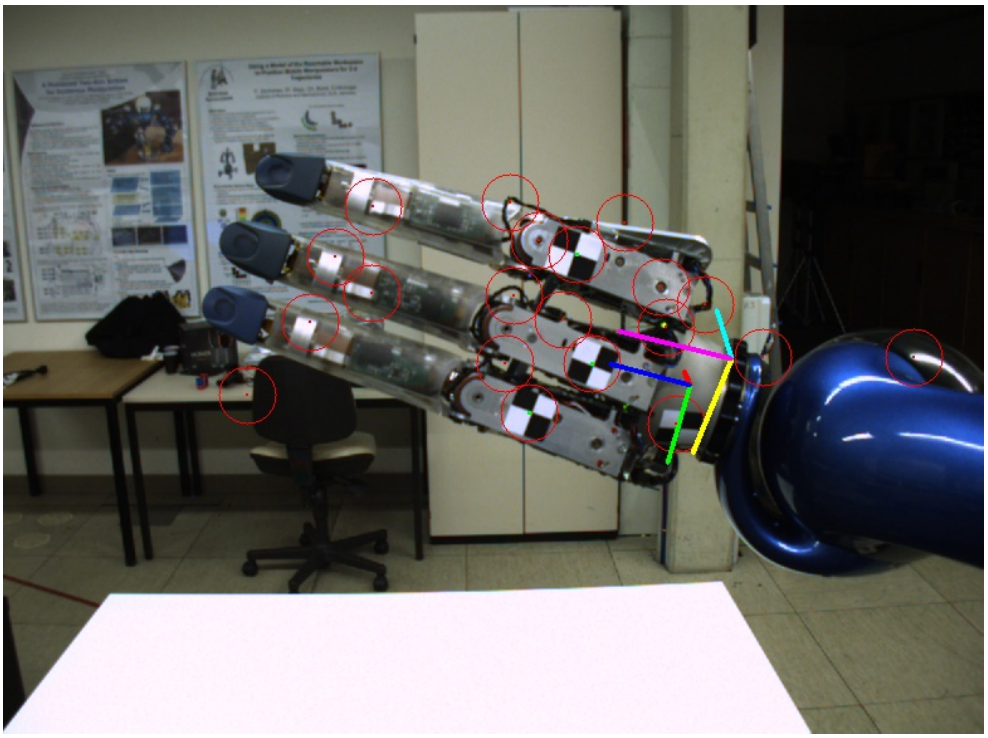

Figure 5.12: Deviation with 95 oberservations

Chapter 6

Conclusions and Outlook

For the tasks of object manipulation, the precise knowledge about the position and orientation of robot endeffector relative to object is necessary. As introduced in the previous chapters, the target of this thesis is to develop a image processing based method to tracking the position of the robot end-effector for the tasks of object manipulation.

As mentioned before, the robot poses are commanded with joint angle configurations. But the actual achieved position and orientation of the robot endeffector (TCP) deviates from the commanded position and orientation of TCP. It makes the novel open-loop controll system hard and unappropriate to controll the robot end-effector for grapping the objects.

In the meanwhile, a novel so-called closed-loop controll system has the advantage compared to the novel open-loop controll system, that it can servo the current achieved position and orientation of robot end-effector in camera views. But it is difficult in our case to track the real position and orientation of TCP as it is occluded by the robot hands.

The concept of the method for this thesis is to develop a marker-based tracking system. That means, several markers with chess board pattern are glued upon the surface of the robot hand, which has the fixed relative position to the TCP. By tracking the marker points, the position and orientation of TCP can be determined.

But there is still a problem, that the relative position and orientation between the TCP and marker coordinate system is also uncertain from the beginning. The idea of the pose estimation method of this thesis is to track the marker points in real time and estimate the transformation matrix from the TCP frame to marker Body frame with learning process.

As the transformation matrix composed of three rotation parameters and three translation parameters (6 DOF) and there are much more values of constraints in objective functions to be used. It can be realised by solving the least squares function.

Thus, it can also estimate the actual position and orientation of the TCPest relative to Camera according to the results from the converged rigid body transformation matrix from

TCPest frame to marker Body frame.

The method can be in principal seperated in two different technologies. First, it is assumed that three markers can be tracked for each new poses. With three markers, the frame for the marker body self can be defined. And three different objetive functions are presented to minimizing the errors from least squares function and therewith finding the most appropriate relationship between the TCP frame to the marker Body coordinate system.

Second, it is assumed that only marker can be found in the camera views for each observation. Under this condition, only the translation between the camera to the marker can be obtained. And therefore the orientation from the TCP frame to the marker Body frame is also underdetermined.

For this consideration, another method by using of the property of the elastic deformation was developed. It assumes that, a force can be exerted on the robot end-effector to pull or drag the current position of TCP towards the real commanded position. It also deals with the least squares function to finding the required mass payload and therewith minimize the deviations in between.

But based on the results under simulation environment, the method with one marker doesn't seem to work well as the remained errors from the objective function are not big improved. Therefore, we only focus on the method of tracking three markers.

As the more values of the constraints in objective function, the better works the least squares function. Based on the method of pose estimation with three markers, the tracking system was extended with a real-time functionalities to track the current position and orientation of TCPest autonomously and efficiently.

The idea of this real time tracking system includes the concept of online calibration. By learning process with large plenty of observed poses, the accuracy of the estimated real position and orientation of TCP increases.

As shown in Chapter 5, several results yield the performance of the method of the online calibration in this thesis. It was shown that, after about 50 observations are learned, the estimation of the transformation matrix ${}^{\mathrm{TCPest}}T_{\mathrm{Body}}$ is considered as accurate enough, and therewith the estimated position and orientation of TCPest are also considered as accurate position except for some special cases.

Although the method introduced in this thesis could estimate the accurate position and orientation of robot end-effector with plenty of data of poses, it still has some limitations. For example, by extracting the marker points in camera images, the accuracy of the results rely strongly on the orientation from the marker points to camera. On the other side, the least-squares function also has the problem with robustness by converging the results.

Moreover, as known, our method is based on the assumption that all three markers could be found in the camera images. But the fact is, by rotation or movement of the robot

end-effector, it couldn't be guaranteed that all the same three markers could be extracted for most of the observations.

For these considerations, the method of this thesis could be extended for further development with some other ideas. For example, instead of marker-based tracking method, the idea of shape-based tracking system may help by increasing the accuracy of the results. But in contrast, it may also increase the calculation time and gain less efficiency by identifying the extracted shape with the model.

Last but not least, the development of this method has realised the idea of determing the accurate position of robot end-effector and minimizing the deviations between the actual position and commanded position of robot end-effector by the help of image-processing based method. Although there are still limitations or weakness in it, the idea the method is the beginning for further research in the area of hand-eye coordination of a humanoid robot.

Appendix A

Appendix

A.1 Structure of „*Rollin'Justin*“ Upper Body

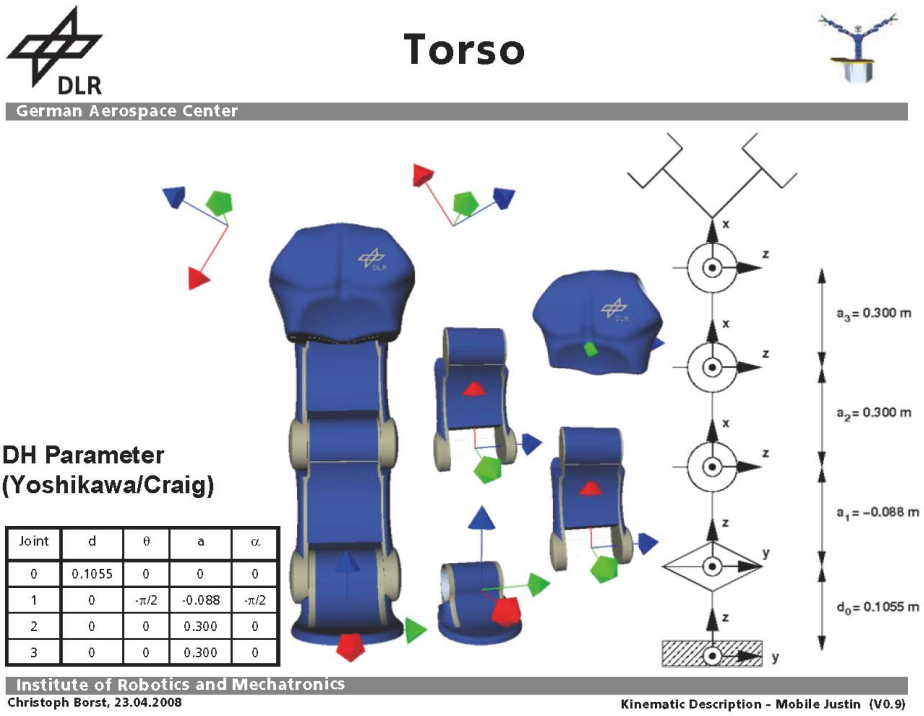

Figure A.1: Justin - Torso

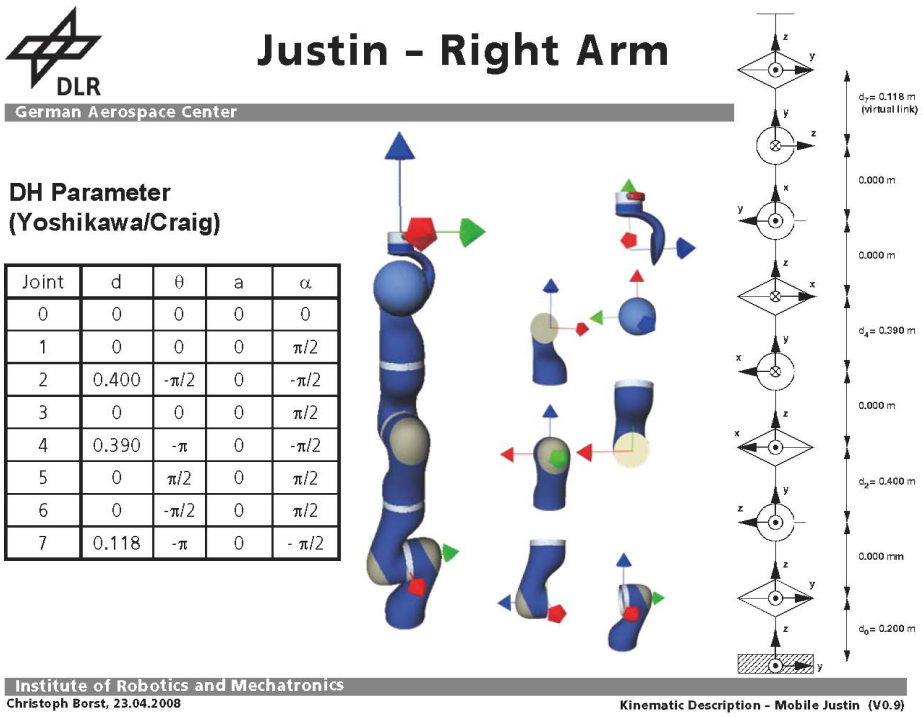

Figure A.2: Justin - Right Arm

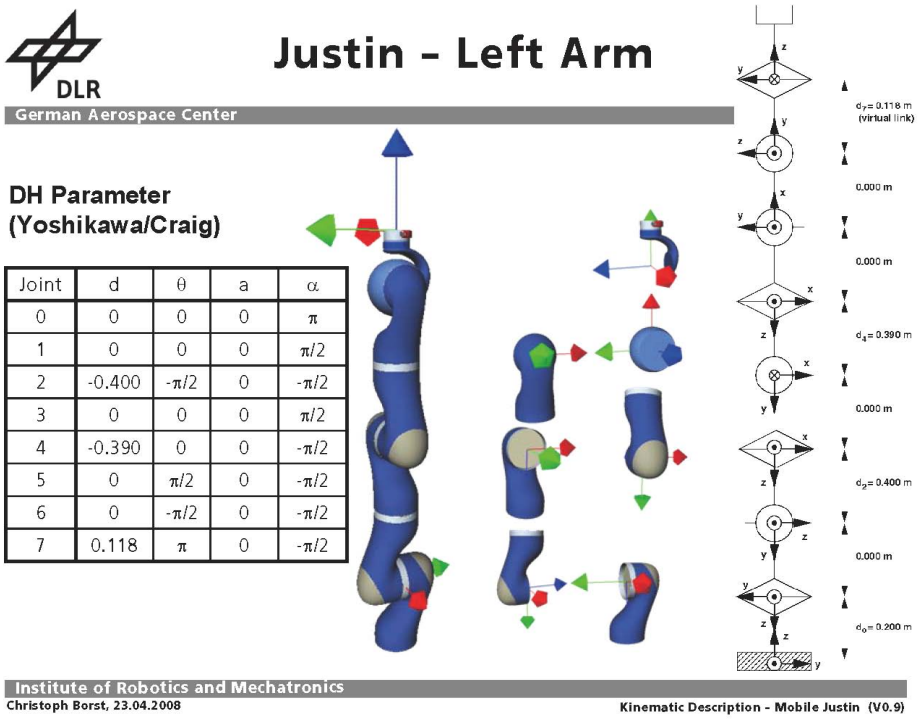

Figure A.3: Justin - Left Arm

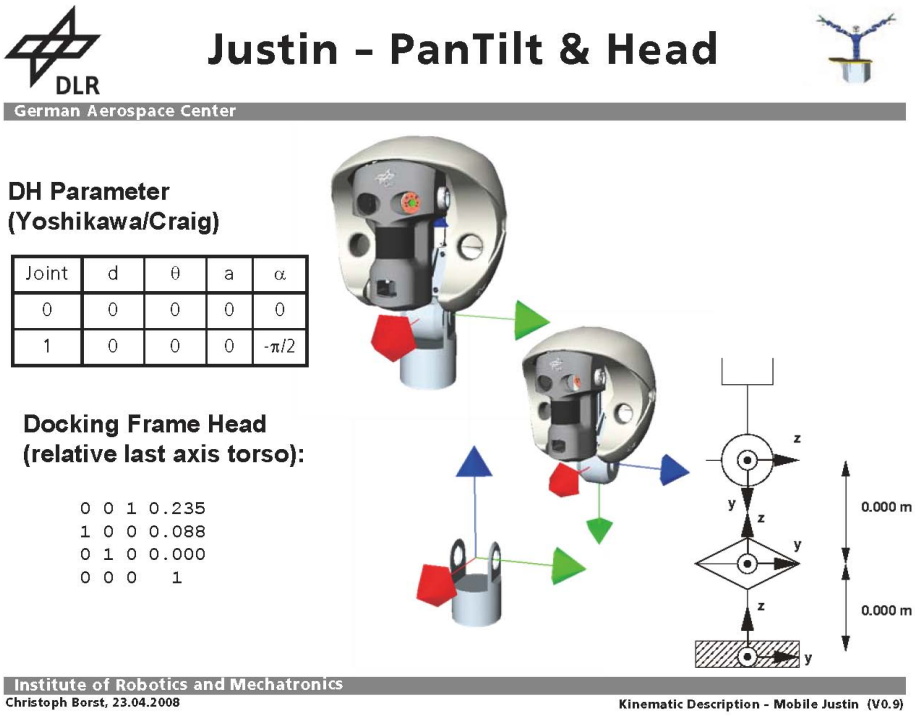

Figure A.4: Justin - Head

List of Figures

List of Tables

Bibliography

[BB09] Jan Bandouch and Michael Beetz. Tracking Humans Interacting with the Environment Using Efficient Hierarchical Sampling and Layered Observation Models. 2009.

[BGLH01] J. Butterfass, M. Grebenstein, H. Liu, and G. Hirzinger. DLR-Hand II: Next generation of a dextrous robot hand. pages 109–114, 2001.

[Bou98] Jean-Yves Bougunet. Stereo Triangulation in Matlab. 1998.

[Bra08] Gary Bradski. *Learning OpenCV*. 2008.

[BWS+09] C. Borst, T. Wimböck, F. Schmidt, M. Fuchs, B. Brunner, F. Zacharias, P. R. Giordano, R. Konietschke, W. Sepp, S. Fuchs, C. Rink, A. Albu-Schäffer, and G. Hirzinger. Rollin' Justin - Mobile Platform with Variable Base. pages 2572–2573, 2009.

[Cra88] John J. Craig. Issues in the design of off-line programming systems. pages 379–389, 1988.

[Cra05] John J. Craig. *Introduction to Robotics: Mechanics and Control*. Prentice Hall, 2005.

[Dru02] Tom Drummond. Real-Time Visual Tracking of Complex Structures. 2002.

[Ecl] `http://www.eclipse.org/`.

[GSL] `http://www.gnu.org/software/gsl/manual/html_node/Nonlinear-Least_002dSquares-Fitting.html`.

[HSAS+02] G. Hirzinger, N. Sporer, A. Albu-Schäffer, M. Hähnle, R. Krenn, A. Pascucci, and M. Schedl. DLR's torque-controlled light weight robot III - are we reaching the technological limits now? pages 1710–1716, 2002.

[HW88] B. Horn and E. Weldon. Direct Methods for Recovering Motion. pages 51–76, 1988.

[KB02] Wisama Khalil and Sebastien Besnard. Geometric Calibration of Robots with Flexible Joints and Links. pages 357–379, 2002.

[KFH+08] C. C. Kemp, P. M. Fitzpatrick, H. Hirukawa, K. Yokoi, K. Harada, and Y. Matsumoto. Humanoids. In *Springer Handbook of Robotics*, pages 1307–1333. 2008.

[Lig] DLR Light Weight Robot III.

[Mat] `http://www.mathworks.com/`.

[MHS06] Takayuki Moritani, Shinsaku Hiura, and Kosuke Sato. Real-Time Object Tracking without Feature Extraction. pages 747–750, 2006.

[OEF+06] Ch. Ott, O. Eiberger, W. Friedl, B. Bäuml, U. Hillenbrand, Ch. Borst, A. Albu-Schäffer, B. Brunner, H. Hirschmüller, S. Kielhöfer, R. Konietschke, M. Suppa, T. Wimböck, F. Zacharias, and G. Hirzinger. A humanoid two-arm system for dexterous manipulation. pages 276–283, 2006.

[Vir] `http://www.instantreality.org/`.

[WS10] German Aerospace Center (DLR) Wolfgang Sepp. personal communication, 2010.