Vision Guiding System for 6-DoF Robotic Manipulator

Lei Shi

03.01.2017

TALLINN UNIVERSITY OF TECHNOLOGY
SCHOOL OF ENGINEERING

Department of Electrical Power Engineering and Mechatronics

Vision Guiding System for 6-DoF Robotic Manipulator

Masinnägemisel Põhinev Juhtssteem 6-e Vabadusastmega Roboti Manipulaatorile

MASTER THESIS

Student: Lei Shi

Student code: 144751

Supervisor: Prof. Mart Tamre

Tallinn, 2017

AUTHORS'S DECLARATION

Hereby I declare, that I have written this thesis independently.

No academic degree has been applied for based on this material.

All works, major viewpoints and data of the other authors used in this thesis have been referenced.

".....” ..... 201....

Author: .....
(signature)

Thesis is in accordance with terms and requirements.

".....” ..... 201....

Supervisor: .....
(signature)

Accepted for defence

".....” ..... 201....

Chairman of theses defence commission: .....
(name and signature)

Department of Electrical Power Engineering and Mechatronics
Mechatronics

MASTERS THESIS SHEET OF TASKS

Year: 2016/2017

Student:	Lei Shi, ID: 144 751
Curricular:	Mechatronics
Speciality:	Mechatronics
Supervisor:	Mart Tamre
Advisor:	N/A

MASTERS THESIS TOPIC:

(in english): Vision Guiding System for 6-DoF Robotic Manipulator
(in estonian): Masinnägemisel Põhinev Juhtssteem 6-e Vabadusastmega Roboti Manipulaatorile
Thesis tasks to be completed and the timetable:

Nr.	Description of tasks	Timetable
1.	Setup UR5 and ROS	03.10.2016
2.	Test and Configure Cameras	03.10.2016
3.	Color Descriptor Algorithm	03.10.2016
4.	Pose Estimation Algorithm	19.10.2016
5.	Controller Design	15.11.2016
6.	Conduct Experiments	15.12.2016
7.	Results Summary	22.12.2016
8.	Compile Thesis	03.01.2017

Solved engineering and economic problems:
1. Design color-based descriptor and online correction algorithm
2. Build stereo vision and estimate pose
3. Controller for vision control

Additional comments and requirements: No
Language: English

Application is filed not later than: 09.10.2017
Deadline for submitting the theses: 16.01.2017

Student: _____
Supervisor: _____

Confidentiality requirements and other conditions of the company are formulated as a company official signed letter.

List of symbols

B	Baseline
C	Controller
C_1	Controller 1
C_2	Controller 2
$\mathbf{C}_L$	Left Camera Center
$\mathbf{C}_o$	Object Color Intensity
$\mathbf{C}_r$	Reference Color Intensity
$\mathbf{C}_R$	Right Camera Center
D	Disparity
d	Disturbance
$\mathbf{d}_l$	Left Camera Distortion Vector
$\mathbf{d}_r$	Right Camera Distortion Vector
$\mathbf{D}(x, y)$	Disparity Map
$\mathbf{D}_{tot}(x, y)$	Combined Disparity Map
$\mathbf{e}_l$	Left Epipole
$\mathbf{e}_r$	Right Epipole
$\mathbf{E}$	Camera Extrinsic Matrix
$\mathbf{E}_l$	Left Camera Extrinsic Matrix
$\mathbf{E}_r$	Right Camera Extrinsic Matrix
$\mathbf{E}_{stereo}$	Stereo Camera Essential Matrix
f	Focal Length
$\mathbf{F}_{stereo}$	Stereo Camera Fundamental Matrix
G	Plant with Delay
G_0	Plant without Delay
G_{0m}	Model of Plant without Delay
G_{dr}	Load Disturbance Response
G_m	Model of Plant with Delay
G_t	Desired Dynamic
G_{yr}	Setpoint Response
I	Identity Matrix
I_c	Color Intensity Classifier
$\mathbf{I}_l(x, y)$	Left Image
$\mathbf{I}_l(x, y)$	Right Image
$\mathbf{K}$	Camera Intrinsic Matrix
K	Gain of Controller
K_{c1}	Gain of Controller 1
K_{c2}	Gain of Controller 2
$\mathbf{K}_l$	Left Camera Intrinsic Matrix

$\mathbf{K}_r$	Right Camera Intrinsic Matrix
K_{uc2}	Ultimate Gain of Controller 2
$\mathbf{l}_1$	Left Epipolar Line
$\mathbf{l}_2$	Right Epipolar Line
$\mathbf{O}_c$	Camera Coordinate System
$\mathbf{O}_{TCP}$	TCP Coordinate System
$\mathbf{p}$	Principal Point
$\mathbf{p}_d$	Desired Plane Points
$\mathbf{p}_{dcen}$	Centroid of Desired Plane
$\mathbf{p}_o$	Object Plane Points
$\mathbf{p}_{ocen}$	Centroid of Object Plane
$\mathbf{P}$	Camera Model
$\mathbf{P}_o$	Object Pose
$\mathbf{P}_r$	Desired Pose
$\mathbf{P}_{ref}$	Reference Pose in TCP Coordinate System
$\mathbf{P}_{TCP}$	Pose in TCP Coordinate System
$\mathbf{q}$	Joint Position of Manipulator
$\dot{\mathbf{q}}$	Joint Velocity of Manipulator
r	Controller Reference Input
$\mathbf{R}_c$	Rotation Matrix between Deisred Pose and Object Pose
$\mathbf{R}_{stereo}$	Stereo Camera Rotation Matrix
$\mathbf{R}_{TCP}^c$	Rotation Matrix of Coordinate Transformation
$\mathbf{t}_c$	Integral Time of Controller
t_d	Translation Vector between Object Pose and Desired Pose
t_m	Time Delay of Plant
$\mathbf{t}_{stereo}$	Time Delay Model of Plant
$\mathbf{t}_{TCP}^c$	Stereo Camera Translation Vector
T_c	Translation Vector of Coordinate Transformation
$\mathbf{T_c}$	Transformation Matrix between Object Pose and Desired Pose
$\mathbf{T_{color}}$	Color Intensity Threshold
$\mathbf{T_D}$	Color Distribution Threshold
$\mathbf{T_{TCP}}$	Transformation Matrix in TCP Coordinate System
$\mathbf{T_{TCP}^c}$	Transformation Matrix of Coordinate Transformation
$\mathbf{x_L}$	Point on Left Image
$\mathbf{x_R}$	Point on Right Image
$\bar{\mathbf{x}}$	Mean Pixel Value in RGB Plane
$\mathbf{X}$	Point in World
y	Plant Output
δ_{TCP}	Pose Controller
$\mathbf{\Delta}Z_m$	Displacement of Camera Measured Depth
ΔZ_{shift}	Displacement of Shifted Depth
σ_o	Object Color Distribution
σ_{ref}	Reference Color Distribution
λ	Time Constant of Desired Dynamic
μ	Pixel Size
τ	Total Time Delay

List of abbreviations

DoF	Degree of Freedom
EKF	Extended Kalman Filter
FOPDT	First Order Plus Dead Time
FoV	Field of View
fps	frame per second
IBVS	Image-Based Visual Servoing
ICP	Iterative Closest Point
IFOPDT	Integrating First Order Plus Dead Time
IMU	Inertial Measurement Unit
IPDT	Integrating Plus Dead Time
OpenCV	Open Source Computer Vision
PCL	Point Cloud Library
PBVS	Position-Based Visual Servoing
RANSAC	Random Sample Consensus
RGB	Red Green Blue
RGB-D	Red Green Blue-Depth
ROS	Robot Operation System
ROS-I	Robot Operation System Industrial
SAD	Sum of Absolute Differences
SLAM	Simultaneous Localization And Mapping
SVM	Support Vector Machine
TCP	Tool Center Point

Foreword

In English:

The inspiration of this thesis work comes from the time when the author was participating a project in IAT Institution from University of Bremen. The thesis is conducted in the Department of Electrical Power Engineering and Mechatronics, Tallinn University of Technology. This thesis work is dedicating in the robotic vision and vision control and trying to bring more intelligence

Author would like to thank supervisor Prof. Mart Tamre for all the supports to the thesis work. And all the helps offered by the colleagues from department are very much appreciated. The author also wants to thank Dhanushka Liyanage for designing the camera frame, Robin Ehrminger for creating the latex thesis template and sharing it. Ott Ruus, who kindly translated the foreword and summary for me.

In Estonian:

Antud lõputöö teema pärineb ajast, kui töö autor viibis Bremeni Ülikooli Automaatika Instituudis. Lõputöö on kirjutatud Tallinna Tehnikaülikooli Elektroenergeetika ja mehhatroonika instituudis. Töö temaatika on masinnägemine ja sellel põhinev juhtimissüteem ning selle arukas disain.

Töö autor avaldab tänu Mart Tamrele lõputöö juhendamise eest ja kolleegidele teaduskonnas abi eest. Eraldi soovib töö autor tänada Dhanushka Liyanaget, kes disainis kaamera raami, Robin Ehrmingeri, kes lõi Latexi suvandid ja Ott Ruusi, kes oli abiks Eesti keelsete tõlgetega.

Contents

Chapter 1

Introduction

Robotic manipulators can perform complicated grasp task fast and precisely with known environment. But there exists limitation. No feedback information for grasp task is provided. The trajectory and path are pre-planned, if there is any change in the environment, for example, the object is missing, the robot system will not know. With visual perception, it brings vision feedback to the robot system to make it a close loop to control robot. This is known as Visual Servoing.
Generally, there are 3 types of visual servoing, image based (2D) visual servoing, position based (3D) visual servoing and 2 1/2D visual servoing. 2D visual servoing uses features from image for control and 3D visual servoing estimates the pose first and uses the pose information for control. 2 1/2 D visual servoing is a hybrid method which combines the 2D and 3D visual servoing. Figure 1.1 and Figure 1.2 below are the 2D and 3D visual servoing.

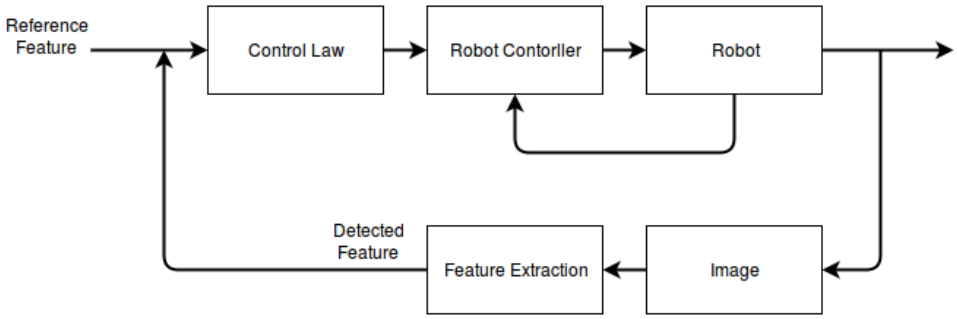

Figure 1.1: Block Diagram of Image Based Visual Servoing, derived from Hutchinson et al. (1996)

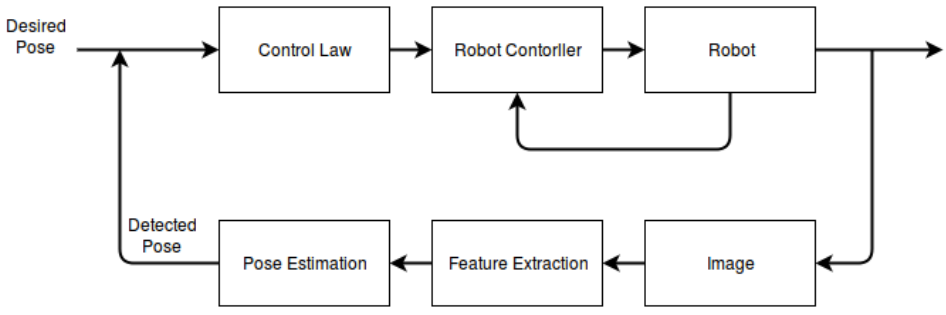

Figure 1.2: Block Diagram of Position Based Visual Servoing, derived from Hutchinson et al. (1996)

For image based visual servoing (IBVS), pose estimation is not needed, but the controller will be complicated since control law uses the discrepancy of desired image features and detected image features as input. For position based visual servoing (PBVS), on contrary, the controller is rather simpler but requires pose estimation which is complex. And a-prior knowledge like detailed 3D object model is required usually. 2 1/2D visual servoing is a hybrid method, it decomposes the pose into position and orientation first and then control them separately. It is a hybrid visual servoing. Another hybrid approach

is partitioned IBVS developed by Corke and Hutchinson (2001), it derives z axis rotation and translation apart from pose and controls it separately.

Depth information is very important for PBVS since it is required for calculating the pose. Popular approaches for getting depth map include using RGB-D camera such as Kinect and stereo vision technique. Another solution is monocular SLAM, which uses one camera to get images from different pose to obtain depth knowledge. Kinect (from Microsoft) can directly output depth map which is quite convenient, but the accuracy of Kinect is not suitable for robotic grasp. Wasenmüller and Stricker (2016) evaluated the depth accuracy among other characteristics. Monocular SLAM requires less hardware than stereo vision, but the algorithm is more complicated. In addition it's not applicable in visual servoing since camera has to move to different place in order to get depth information. Stereo vision can provide better depth accuracy than Kinect but lots of additional work is required to get depth map.

Cameras can be set up in different configurations as in Figure 1.3. With eye-in-hand camera is fixed on the manipulator and with eye-to-hand camera is fixed in a point in the scene. Redundant configuration combines eye-in-hand and eye-to-hand together.

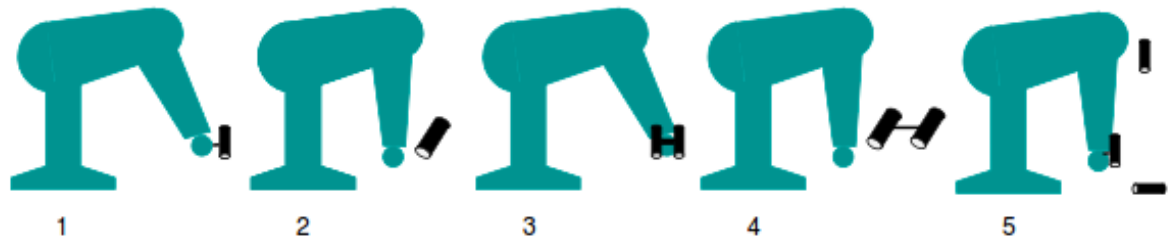

Figure 1.3: Different Camera Configurations for Visual Servoing, derived from Kragic et al. (2002). Configuration 1: Monocular eye-in-hand. Configuration 2: Monocular eye-to-hand. Configuration 3: Binocular eye-in-hand. Configuration 4: Binocular eye-to-hand. Configuration 5: Redundant.

1.1 Motivation

The primary motivation in this thesis work is to bring more intelligence to the robot. One of the approach, as mentioned earlier, is bringing the visual perception to the robot so that the robot can "see" the environment. Another motivation is reducing the a-prior knowledge about the object and increasing the robustness in vision perception part.

Another motivation is working towards practical visual servoing. Although the idea of visual servoing has been brought up a long time ago and the topic has been discussed a lot, still there hasn't a lot of successful practical integrations in the field of above. Robustness, accuracy and convenience are important factors that need to be considered. Fomena et al. (2013) also discuss the problems toward practical visual servoing. This thesis work also tries to make effort to contribute in the factors which is possible to make visual servoing more practical.

The visual servoing could be applied extensively in the field of robotics. In industry, for example, it can be used in assisting human for assembling or smart pick and place. It is also applicable in robot manipulator interacting with each other. Another application field is assistive robotics in domestic environment. For instance, helping disable person grabbing cup, or taking a book from shelf, etc. Furthermore the idea of visual sensor can also be applied not just to robots. It can be implemented in various scenes regarding to control. For example, Wang et al. (2008) and Hirata and Mizuno (2008), camera is used as feedback to control the inverted pendulum.

1.2 Task definition

In this thesis work, PBVS system will be designed for a 6-DoF robotic manipulator. To accomplish PBVS, a few tasks need to be finished. First part is object recognition. It is obvious that the target object has to be detect and recognized. Secondly, the position and orientation of the object needs to be estimated. What is important is the depth information. To get the depth data, 2 cameras are used as stereo vision system to obtain depth. The camera configuration is eye in hand which means the cameras are mounted on the end effector of the manipulators. The third part is the controller design. Controller uses visual feedback information to control the manipulator.

The design can be divided into 3 parts according to the tasks mentioned above. In object detection and recognition part, an algorithm is proposed. This algorithm uses an edge-based segmentation method and a descriptor based on color information. A simple teaching stage is required in order to get the information of the object. The algorithm is designed to have a descriptor invariant to the scale, orientation and light condition change within a certain range while the computational cost will be relatively low.

In pose estimation part, depth information is acquired by stereo vision first. Pose estimation is based on Kabsch algorithm developed by Kabsch (1976). A few feature points of object will be used to represent the plane of object. And this plane is compared to desired plane and then transformation matrix is obtained thus the estimated pose is also obtained.

In controller design part, the main objective is focused on dealing with the delay problem, since the image capture, image transportation, processing and algorithm takes a lot of time. It makes the system difficult to achieve real time performance. Another problem is the disturbance from environment and hardware. The control law design aims at solving these 2 problems.

1.3 Literature Review

Visual servoing researches have been carried out by many researchers. Chaumette and Hutchinson (2006) and Hutchinson et al. (1996) categorized into visual servoing into image based visual servoing and position based visual servoing. A hybrid method, 2 1/2D visual servoing which combines 2D and 3D method is proposed by Malis et al. (1999). Regarding to the PBVS, Lippiello et al. (2007) developed a PBVS system with a eye-in-hand and eye-to-hand hybrid camera configuration. Thuilot et al. (2002) focused on keeping the object in the center of camera while manipulator is moving.

Successfully detect and recognize the object during the entire visual servoing process is essential. The quality of the feedback information will affect the total performance directly. During the movement of manipulator the scale and the orientation of the detected contour is changing. Also from different angle, the lighting is different even if the environment lighting is not changing. Thus, ideally, the descriptor should have following characteristics for visual servoing. It should be invariant to scale, orientation and light condition, it should be reliable and meanwhile have relatively low computational cost.

There exist various methods for object detection and recognition. A very basic one is to put marker on the object to be detected. The marker is usually made of patterns which can be easily detected. Another popular solution is cross correlation which is based on template matching. Also lots of color information based object recognition researches have been conducted. Chang and Krumm (1999) and Ancuti and Bekaert (2007) investigated how color co-occurrence can be used for object recognition. An extensive evaluation on different color descriptors have been presented by Van De Sande et al. (2010). For scale and orientation invariance, SIFT descriptor developed by Lowe (2004) and SURF descriptors developed by Bay et al. (2008) are widely used for pattern match. More recently, a descriptor called BRISK, which is even faster than SURF is proposed by Leutenegger et al. (2011).

The proposed algorithm doesnt require large amount of template or lots of priory knowledge about the object. It is invariant to scale, orientation and lighting condition within a certain range while the algorithm itself still be relatively simple thus it doesnt require high computational cost.

Pose estimation is one important stage in PBVS. The main difference between IBVS and PBVS is the pose estimation. In PBVS pose information is fed as controller input while in IBVS, the image features are the inputs for controller. Different methods have been implemented for estimating the pose. Lippiello et al. (2007) used Extended Kalman Filter (EKF) to estimate pose. Kragic and Christensen (2002) used a model based method for estimation.
Corke et al. (1996) pointed out the importance of controller design for visual servoing performance and presented controller design. Compare to IBVS, the control law of PBVS is relatively simple. Various controllers based on classic control theory and modern control theory have been used for visual servoing. Siciliano et al. (2010) designed PD controller with gravity compensation. Hashimoto et al. (1996) developed a LQ controller for visual servoing. Sliding mode controller is investigated by Becerra and Sagues (2008).

1.4 Outline

This thesis is organized in the following way. In Chapter 2, the proposed descriptor and online correction algorithm is explained in detail. In Chapter 3, how stereo vision system is constructed and its theoretical background is given. After that, pose estimation approach and coordinate transformation calculation is given and explained. In Chapter 4 possible controller structures are discussed to solve the time delay and disturbance rejection problem. Visual servoing system is modelled and controller calculation is performed. How the entire design is implemented from hardware and software point of view is described in Chapter 5. The results of the experiments regarding to the contents from Chapter 2 to Chapter 5 are summarized in Chapter 6. An overall discussion is given in Chapter 7.

Chapter 2

Object Recognition

There exists methods to perform object recognition easily and successfully, for example, put a sticker with designed dot pattern. But it is less intelligent and inconvenient towards real practice. Taking one example in assistive robotics, each items to be grasped needs one sticker and add or change items require new sticker or change sticker. In addition sticker is easily being broken. It is convenient for research purpose but not suitable for practical application. Some machined learning based methods are more intelligent and don't require a lot of maintenance as the previous one. But it requires a lot of training samples.

The proposed algorithm uses edge-based segmentation for object detection and a color-based descriptor for classification. This algorithm intends to reducing a-prior knowledge about the object and increase the robustness. To be more specific in robustness, it should be robust in shape, orientation and scale as well as lighting condition. During the motion of robotic manipulator, the scale and orientation of object will be changing. The lighting condition might also change due to environment change. If camera loses the object during servoing, it can cause problems. Thus increasing the robustness of object detection can decrease the probability in losing the object.

Figure 2.1 below shows the common step in object recognition. Acquired image is first processed to decrease noise and enhanced (increase contrast, for example). Processed image is then segmented. Commonly either by edge-based or region-based segmentation techniques. Then features of the segmented object is extracted for classification.

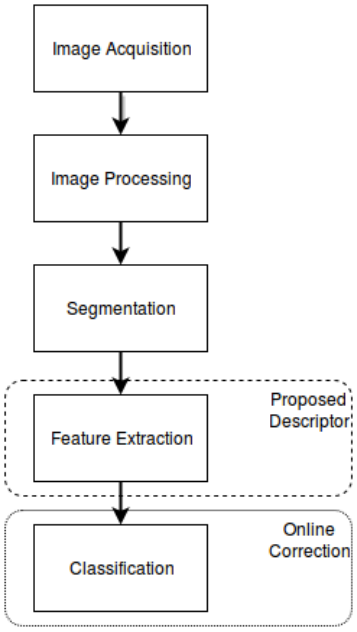

Figure 2.1: Steps for Object Detection and Recognition

The descriptor in this thesis work is used in phase feature extraction. A classifier is used to classify either the segmented object is target object or not. A online correction algorithm modifies the classifier if necessary. Figure 2.2 is the overview of the object detection and recognition algorithm. A simple teaching stage is required before using. In the teaching stage, feature vector is recorded first and saved as file. Before visual servoing started, the feature vector is read from the saved file as reference values. The proposed descriptor is used to recognize the object. If the object is not recognized, then an online correction algorithm is called to modify the classifier.

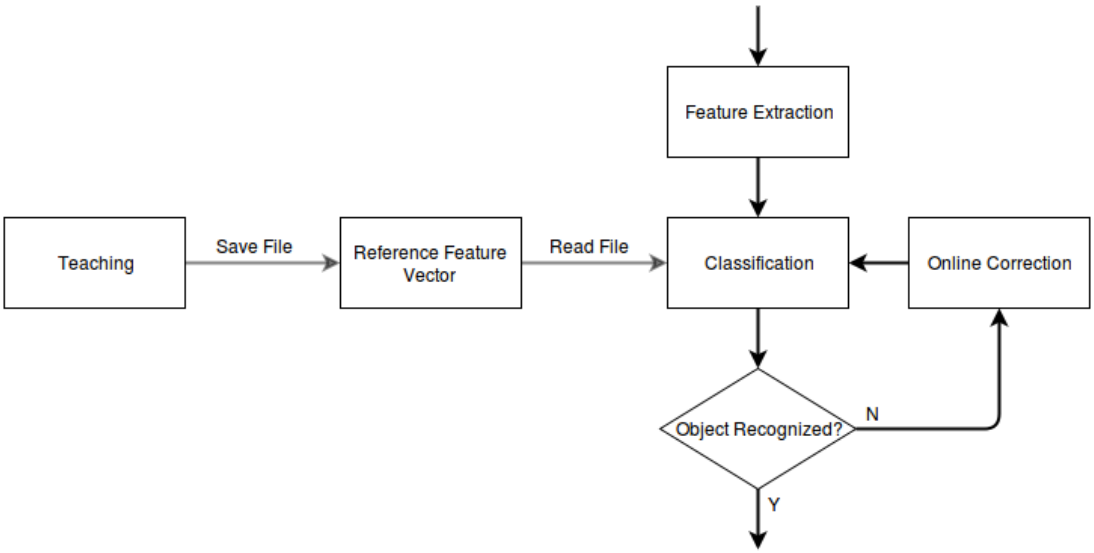

Figure 2.2: Overview of Object Detection and Recognition

2.1 Descriptor and Classifier

The descriptor is the color intensity of the RGB image plane. Equation (2.1) below indicates the definition,

$$C_r = \frac{1}{n}\sum_{i=1}^{n} R_i + G_i + B_i \ . \tag{2.1}$$

Where C_r is the reference value of color intensity, n is the size of segmented object, R_i, G_i and B_i are the pixel value in red, green and blue plane respectively. Usually in classification, different classifiers like Support Vector Machine (SVM), Minimum Distance, Maximum Likelihood and so on. Depends on different applications, different classifiers with different complexity are used. In addition, combined classifier can also be used. Gangeh et al. (2007) evaluated a 2 stage combined classifier. The classifier used in this work is a threshold as in Equation (2.2)

$$I_c = \begin{cases} \text{True}, & \text{if } \frac{|C_r - C_o|}{C_r} < T_{color} \\ \text{False}, & \text{else} \end{cases} \ . \tag{2.2}$$

Where C_o is the color intensity of segmented object. T_{color} is the threshold value. However, there exists a problem. The descriptor is vulnerable to the light condition change. As it is linear to the light intensity. C_o will be higher when the environment is brighter and lower when it is darker. When environment light changes, the classifier may not be effective and this means it is variant to light condition change. To overcome this problem, the online correction is introduced.

2.2 Online Correction

Figure 2.3 below is the overview of the online correction algorithm. If the target object can't be recognized, then color distribution, which is nothing more than the pixel value standard deviation in RGB image plane, will be used to determine whether it is target object. The reference color distribution is acquired as one of the feature during the teaching stage. When object is recognized by using the color distribution, the reference color intensity will be modified. Current value will be the new value.

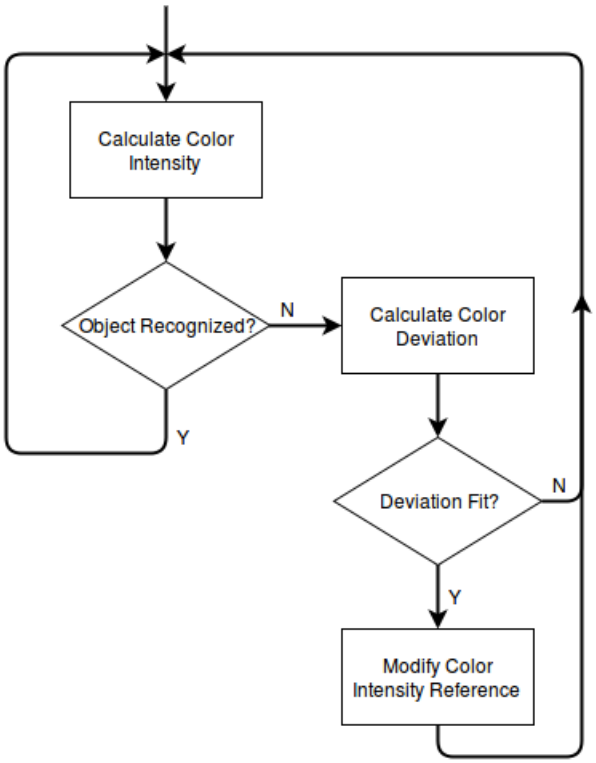

Figure 2.3: Overview of Online Correction Algorithm

Equation (2.3) below is the color distribution,

$$\sigma_{\mathbf{ref}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\mathbf{x_i} - \bar{\mathbf{x}})} \ . \tag{2.3}$$

Where $\sigma_{\mathbf{ref}} = [\sigma_r, \sigma_g, \sigma_b]$, σ_r, σ_g and σ_b are pixel value distributions in RGB image plane respectively. $\mathbf{x_i}$ equals to $[x_{ir}, x_{ig}, x_{ib}]$ which represents ith pixel value in RGB plane. $\bar{\mathbf{x}}$ is the average pixel value in RGB plane which equals to $[\bar{x_{ir}}, \bar{x_{ig}}, \bar{x_{ib}}]$. Equation (2.4) below decides whether a object is target object or not.

$$D = \begin{cases} \text{True}, & \text{if } \frac{|\sigma_{\mathbf{ref}} - \sigma_{\mathbf{o}}|}{\sigma_{\mathbf{ref}}} < \mathbf{T_D} \\ \text{False}, & \text{else} \end{cases} \ . \tag{2.4}$$

Where $\sigma_{\mathbf{o}}$ is the color distribution of segmented object. $\mathbf{T_D}$ is threshold values. If D is true, then C_o is assigned to C_r. The program will then use the updated C_r to recognize object again.

2.3 Discussion

The overall algorithm is designed for the purpose of being more practical for visual servoing application. The reliability is important to the performance of visual servoing. The color based descriptor is invariant to scale and orientation, with the help of online correction invariance to light condition is also achieved.

The reliability is increased by increasing the robustness. There are other descriptors which is invariant to scale and orientation, such as image moment. The reason of author's interest in color based descriptor is the possibility to research what's the relationship between color pattern and perspective projection and the possibility to use it for pose estimation.

Chapter 3

Stereo Vision and Pose Estimation

There are several commercially available stereo camera on market. ZED camera (from STEREOLABS) provides a position accuracy of 1 mm and it has extensive third party support includes Robot Operating System (ROS), Open Source Computer Vision (OpenCV) and Point Cloud Library (PCL). From compatibility and accuracy point of view, it suits the application. But the minimum depth range is 0,7 m which is not suitable. In this thesis work, instead of using commercially available stereo cameras. The main advantage is the freedom to select baseline according to the application. 2 cameras will be used to form a stereo rig. The cameras are parallel to each other and camera centres locate on the x axis. A point in the world is projected onto image planes of both cameras. The projected points will have different image coordinates. With different depth of the point in world, the difference in image coordinates is also different. Depth image is calculated on this basis. Figure 3.1 is a overview of stereo algorithm. First, cameras need to be calibrated, then the distortion of images from left and right camera are removed and images are rectified. Depth is calculated after correspondence matching is matched. Details of each step will be explained in this chapter.

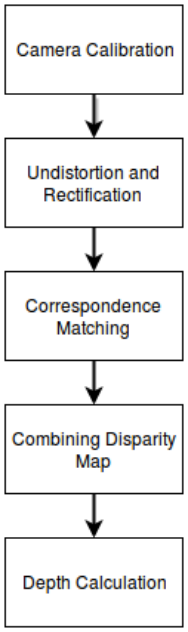

Figure 3.1: Overview of Stereo Algorithm

After depth is calculated, pose of object will be estimated. The algorithm is based on Kabsch (1976), Kabsch (1978), Besl and McKay (1992) and Arun et al. (1987)'s method. The idea is to create a plane representing the object's pose. The desired pose and current pose is thus projected as desired plane and current plane. By aligning the 2 planes, current pose is aligned with desired pose. A transformation

matrix between 2 planes is obtained. The relevant pose can be calculated from the transformation matrix. Using this approach, the exact 3D model of object is not required. It is consistent with the motivation of reducing "a-prior" knowledge. Figure 3.2 is an overview of the pose estimation calculation.

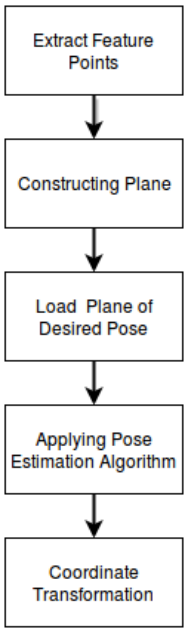

Figure 3.2: Overview of Pose Estimation Algorithm

3.1 Camera Model

Pinhole camera model (3.3) can be used as camera model. Equation 3.1 to Equation 3.8 is derived from OpenCV.

$$k \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{KE} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} . \tag{3.1}$$

Where k is scaling factor, $\mathbf{K}$ is the intrinsic matrix (3.2) and $\mathbf{E}$ is extrinsic matrix (3.3). $[X, Y, Z, 1]^T$ is the homogeneous world coordinate and $[u, v, 1]^T$ is the homogeneous image coordinate.

$$\mathbf{K} = \begin{bmatrix} f_x & sk & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} , \tag{3.2}$$

$$\mathbf{E} = [\mathbf{R} \mid \mathbf{t}] . \tag{3.3}$$

Where f_x and f_y are focal length along x and y axis. (c_x, c_y) is principal point. sk is skew parameter. $\mathbf{R}$ represents the rotational matrix and $\mathbf{t}$ is translational vector. Figure 3.3 illustrates the pinhole camera model.

$\mathbf{p}$ is the principle point. $\mathbf{X}$ is the point in world. $\mathbf{x}$ is the point projected on image plane. Focal length f is the distance from camera center to principal point. The projection is done via the camera model $\mathbf{P} = \mathbf{KE}$. Skew parameter sk equals to 0 is x and y axis of pixel is perpendicular to each other. In most cases the pixel shape is rectangle or square.

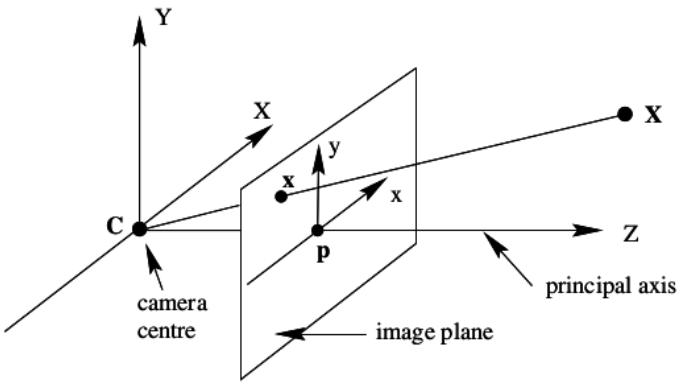

Figure 3.3: Pinhole Camera Model, derived from Hartley and Zisserman (2003)

3.2 Calibration

Figure 3.4 is the overview of calibration for stereo vision. Firstly, each camera is calibrated separately, the single calibration parameter file is saved and the results are used in stereo calibration. The stereo calibration parameter is also saved as file. Both single calibration file and stereo calibration file are read for further stereo vision algorithm.

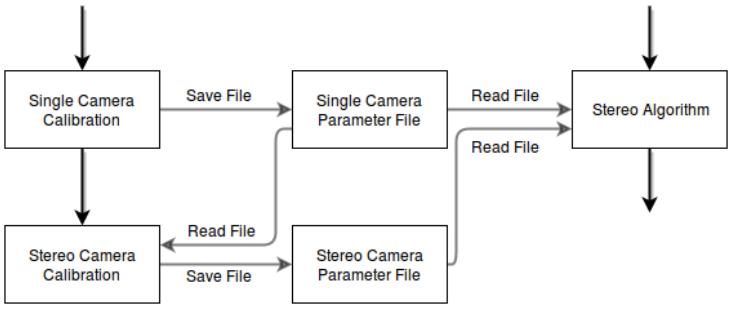

Figure 3.4: Overview of Camera Calibration

The principle of camera calibration is, given a calibration pattern with known size and dimension, camera model is estimated and distortion coefficient is obtained to get undistorted image. Figure 3.5 illustrates the distortion of image.

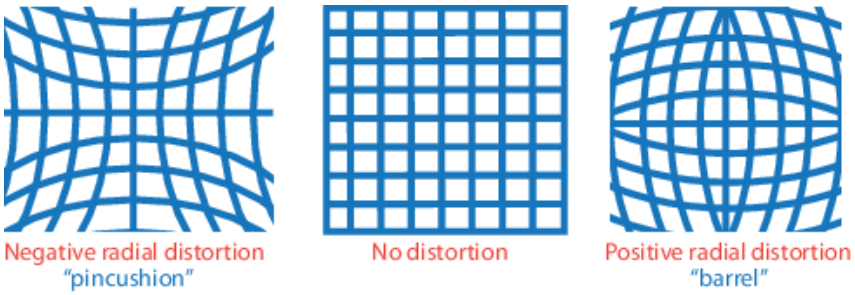

Figure 3.5: Camera Distortion (from Matlab)

The calibration pattern is usually a black and white chessboard pattern or black circles. For single camera calibration in the thesis, a chessboard pattern is used. Figure 3.6 is the algorithm of single camera calibration. The corner of black and white squares are detected first, then the sub pixel of detected corners are interpolated to increase the performance of calibration. The parameters of the single camera calibration are $\mathbf{K}_l$, $\mathbf{K}_r$, $\mathbf{E}_l$, $\mathbf{E}_r$, $\mathbf{d}_l$ and $\mathbf{d}_r$, which represents intrinsic and extrinsic matrix and distortion vector for left camera and right camera. The stereo calibration calculates the transformation

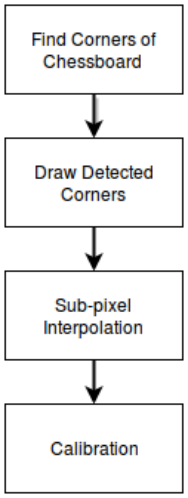

Figure 3.6: Single Camera Calibration

between left and right cameras. The results parameters of stereo calibration are rotational matrix $\mathbf{R}_{stereo}$, translational vector $\mathbf{t}_{stereo}$, essential matrix $\mathbf{E}_{stereo}$ (3.4) and fundamental matrix $\mathbf{F}_{stereo}$ (3.5).

$$\mathbf{E}_{stereo} = \begin{bmatrix} 0 & -t_2 & t_1 \\ t_2 & 0 & -t_0 \\ -t_1 & t_0 & 0 \end{bmatrix} \mathbf{R}_{stereo} \ , \tag{3.4}$$

$$\mathbf{F}_{stereo} = {\mathbf{K}_r}^{-T} \mathbf{E}_{stereo} {\mathbf{K}_l}^{-1} \ . \tag{3.5}$$

Where t_0, t_1 and t_2 are elements of $\mathbf{t}_{stereo}$ (3.6).

$$\mathbf{t}_{stereo} = [t_0, t_1, t_2] \ . \tag{3.6}$$

3.3 Epipolar Geometry

Figure 3.7 is the epipolar geometry of stereo system. $\mathbf{C}_L$ and $\mathbf{C}_R$ are the camera centers of left camera and right camera. $\mathbf{X}$ is a point in world, $\mathbf{x}_L$ and $\mathbf{x}_R$ are the point projected in left camera plane and right camera plane. $\mathbf{e}_R$ is the epipole on right image plane. It is the $\mathbf{C}_L$ on the right image plane. Same for left epipole $\mathbf{e}_L$. $\mathbf{l}_1$ and $\mathbf{l}_2$ are the left epipolar lines. The correspondence point of point $\mathbf{x}_L$ will appear on the $\mathbf{l}_2$ and vice versa. Finding the correspondence points of left image in right image is referred as correspondence problem.
(3.7) is the mathematical representation of epipolar geometry.

$$\mathbf{x}_R \mathbf{F}_{stereo} \mathbf{x}_L = 0 \ . \tag{3.7}$$

Epipolar lines are characterized in (3.8).

$$\mathbf{l}_2 = \mathbf{F}_{stereo} \mathbf{x}_L, \quad \mathbf{l}_1 = {\mathbf{F}_{stereo}}^T \mathbf{x}_R \ . \tag{3.8}$$

Rectification is performed before finding the correspondence point. The purpose of rectification is narrowing down the search range of epipolar lines. As in Figure 3.7, in order to find correspondence point on epipolar line, it is necessary to search in both x and y direction in image coordinates. After rectification, the image planes are parallel and epipoles are moved to infinity,, the epipolar lines are parallel as well so that the search is only required along one direction. It is much less complicated than search in 2 directions. Figure 3.8 shows the rectified image epipolar lines.

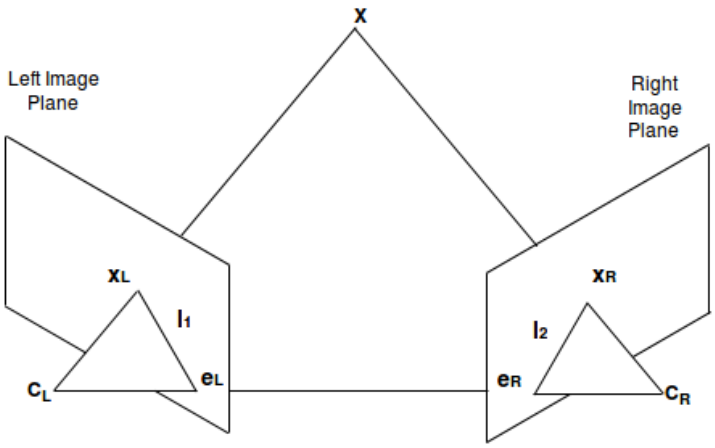

Figure 3.7: Epipolar Plane

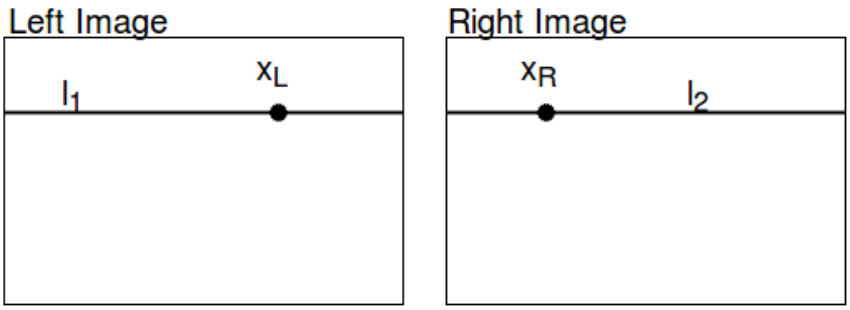

Figure 3.8: Rectified Image

3.4 Disparity and Depth

Depth can be calculated as in (3.9)

$$Depth = \frac{Bf}{D\mu} \ . \tag{3.9}$$

Where B is baseline value, f is focal length, μ is pixel size, and D is the disparity value. Depth is inversely proportional to disparity, disparity map should be obtained before depth map can be calculated. Depth resolution (3.10) is another important metric.

$$\Delta Depth = \frac{(Depth)^2}{Bf}\Delta D \ . \tag{3.10}$$

Where $\Delta Depth$ and ΔD are the minimal change of depth and disparity value respectively. Equation 3.9 and Equation 3.10 is derived from NI.

Correspondence matching will give disparity value. Different algorithms can be employed to solve correspondence matching problem. Blocking matching (from OpenCV) is used in this thesis work. It is one of the dense matching algorithm. Equation 3.11 and Equation 3.12 explain the algorithm. An window in size of n by n is convoluted through image to search the corresponding point. Search range is defined by disparity range. Sum of Absolute Differences (SAD)(3.11) is calculated.

$$SAD(x, y; d) = \sum_{i,j} \mid I_l\left(x_i, y_j\right) - I_r\left(x_i + d, y_j\right) \mid \ . \tag{3.11}$$

Where n is window size, d is disparity search range. $I_l\left(x, y\right)$ and $I_r\left(x, y\right)$ are left image and right image pixel value. The size of window will affect the disparity map directly, smaller size will preserve more details of image but meanwhile more noise is remained. Larger size will remove more noise but details are lost. The size of window depends on application.

Disparity map $D(x, y)$ is calculated as in (3.12)

$$D(x, y) = argmin(SAD(x, y; d)) \ . \tag{3.12}$$

Before calculating the depth map, 2 disparity maps with different searching disparity is combined together (3.13). The reason of this to reduce the occlusion.

$$D_{tot}(x, y) = \begin{cases} D_1(x, y), & \text{if } D_1(x, y) \neq 0 \\ D_2(x, y), & \text{else} \end{cases} \ . \tag{3.13}$$

Where $D_{tot}(x, y)$ is combined disparity map, $D_1(x, y)$ and $D_2(x, y)$ are disparity maps with different disparity search range. Block matching is fast, but it also has limitations. It can't deal with texture-less surfaces. It has fattening effect at borders.

3.5 Pose Estimation and Coordinate Transformation

Pose estimation is essential for PBVS. The error between desired pose $\mathbf{P}_r$ and object pose $\mathbf{P}_o$ is fed to visual controller. Both $\mathbf{P}_r$ and $\mathbf{P}_o$ are in camera coordinate frame.
One solution is project the current object scene and reference object scene into point cloud and then use Iterative Closest Point (ICP) algorithm (Besl and McKay (1992)) to get the $\mathbf{T}_c$ which consists of rotation matrix $\mathbf{R}_c$ and translational vector $\mathbf{t}_c$. $\mathbf{P}_o$ can be aligned with $\mathbf{P}_r$ by using $\mathbf{T}_c$. Given a initial guess of the difference between 2 sets of point cloud, ICP iteratively calculates transformation matrix and move the current object point cloud towards desired object point cloud until they align with each other. But ICP is rather computationally expensive which doesn't fit the purpose of real time. For the same reason iteration based algorithm Random Sample Consensus (RANSAC) proposed by Fischler and Bolles (1981) is also not a good option.
The solution used in this thesis uses plane to represent the object. Then problem is transferred to align the plane of current object in scene with the desired one. The plane is represented by 3 points which is not in a line. 3 feature points are selected representing the desired plane and object plane are selected based on a same rule. They are denoted as desired plane points $\mathbf{p}_d$ and current object plane points $\mathbf{p}_o$. By aligning these 2 sets of points, 2 planes are also aligned. The criterion is to minimize the accumulated error of transformation (3.14).

$$\sum e = \sum_{i=1}^{3} \| \mathbf{R}_c \mathbf{p}_{o_i} + \mathbf{t}_c - \mathbf{p}_{d_i} \| \ . \tag{3.14}$$

Where $\sum e$ is accumulated error. Minimizing $\sum e$ involves both $\mathbf{R}_c$ and $\mathbf{t}_c$. As explained by Besl and McKay (1992) and Arun et al. (1987), Equation (3.14) can be decomposed so that only $\mathbf{R}_c$ is involved. To solving decomposed accumulated error, Singular Value Decomposition (SVD) can be applied and this non-iterative method is tested and proven faster than ICP mehtods. Equation (3.15) to (3.19) explained the calculation of $\mathbf{T}_c$. First the centroids of $\mathbf{p}_d$ and $\mathbf{p}_o$ are calculated (3.15).

$$\mathbf{P}_{dcen,ocen} = \frac{1}{3} \sum_{i=1}^{3} \mathbf{p}_{d_i,o_i} \ . \tag{3.15}$$

Where $\mathbf{P}_{dcen}$ and $\mathbf{P}_{ocen}$ are centroids for $\mathbf{p}_d$ and $\mathbf{p}_o$ respectively. Covariance matrix $\mathbf{M}$ is calculated using $\mathbf{p}_d$, $\mathbf{p}_o$ and their centroids.

$$\mathbf{M} = \sum_{i=1}^{3} (\mathbf{p}_{o_i} - \mathbf{p}_{ocen})(\mathbf{p}_{d_i} - \mathbf{p}_{dcen})^T \ . \tag{3.16}$$

Then SVD is performed on $\mathbf{M}$ (3.17) . $\mathbf{R}_c$ is then calculated by unitary matrix $\mathbf{U}$ and $\mathbf{V}$ (3.18). $\mathbf{t}_c$ is

calculated in (3.19).

$$[\mathbf{U}, \mathbf{S}, \mathbf{V}] = SVD(\mathbf{M}) \ . \tag{3.17}$$

Where $\mathbf{S}$ is diagonal matrix.

$$\mathbf{R}_c = \mathbf{V}\mathbf{U}^T \ . \tag{3.18}$$

$$\mathbf{t}_c = -\mathbf{R}_c\mathbf{p}_{ocen} + \mathbf{p}_{dcen} \ . \tag{3.19}$$

The pose estimated is in the camera coordinate $\mathbf{O_c}$. It needs to be transformed into TCP coordinate $\mathbf{O_{TCP}}$. Equation 3.20 is the transformation.

$$\mathbf{T}_{TCP} = \mathbf{T}^c_{TCP}\mathbf{T}_c \ . \tag{3.20}$$

Where $\mathbf{T}^c_{TCP}$ is the transformation matrix which transform the pose from $\mathbf{O_c}$ to $\mathbf{O_{TCP}}$. $\mathbf{T}^c_{TCP}$ is relevant to how stereo rig is mounted on the end effector of the manipulator and its mechanical dimension. If the stereo rig is mounted in a way that the orientation of axes of both coordinate system is same, then the difference is only in the translation (3.21).

$$\mathbf{T}^c_{TCP} = [\mathbf{R}^c_{TCP} \mid \mathbf{t}^c_{TCP}] = [\mathbf{I} \mid \mathbf{t}^c_{TCP}] \ . \tag{3.21}$$

Where $\mathbf{I}$ is identity matrix.

3.6 Discussion

The depth accuracy is important, it affects the accuracy of the pose estimation. Depth accuracy depends on several factors. From hardware point of view, pixel size, camera sensor quality and lens will affect the depth calculation. But once the hardware is selected, there is not much one can do with hardware to affect depth accuracy. Other factors affect depth accuracy is during setting up the stereo system. First is calibration. Normally the result of calibration is measured by the reprojection error. If the value of reprojection error is around 1 pixel, it is usually accepted. In order to calibrate the camera properly, all of the calibration pattern should be visible in both cameras. Images shouldn't be blurry. Usually more images for calibration results in better calibration result, but there is no such an exact minimal number of images that should be taken. Also when taking images, the calibration pattern should be viewed from different perspective as much as possible. Another factor is the baseline and depth range meaning the distance from cameras. Figure 3.9 indicates the effects on depth resolution. Depth resolution is defined in Equation 3.10.

As indicated in Figure 3.9, for a certain baseline, the further the distance is from the camera, the less accurate in depth the stereo rig can detect. With the distance increases, the error increases exponentially. If an object is fixed at a certain distance from the cameras, larger baseline will lead in a better resolution.Additionally, different baselines and depth ranges will also lead in different maximum disparity value. Figure 3.10 is the plot. As indicated in the plot, the further the object is away from the cameras, the smaller the disparity value will be. For a certain distance, a smaller baseline means smaller maximum disparity value.

The underlying idea of the pose estimation is aligning 2 planes. The plane is extracted from 3 feature points of object. The plane then represents the pose of object. The absolute pose is not important since the algorithm calculates the relative transformation between the desired pose and current object pose. This pose deviation can be used to control the manipulator. Another solution is projecting the recognized object into 3D space as point cloud and then aligning the object point cloud with the desired point cloud.

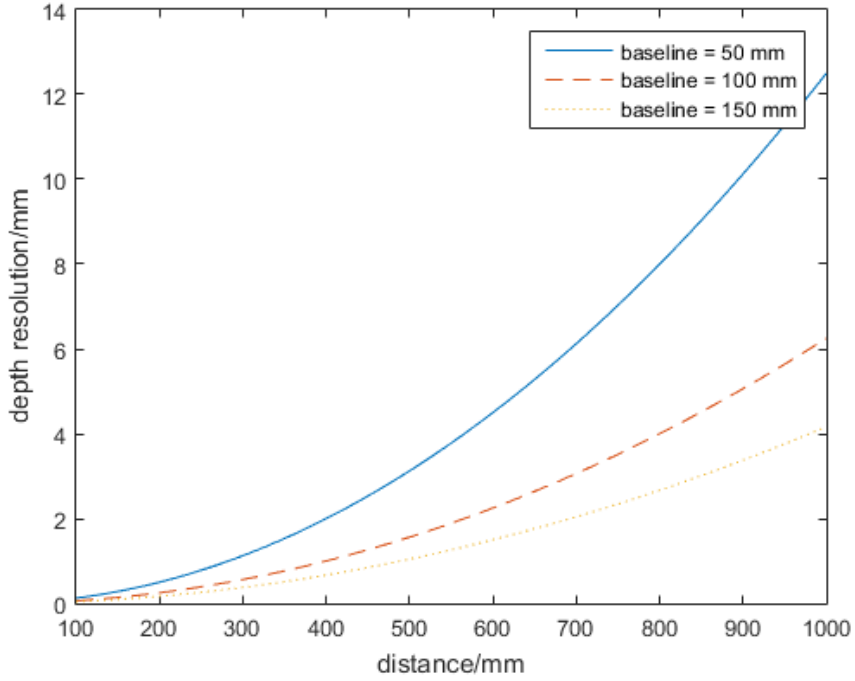

Figure 3.9: Plot of Depth Resolution in Different Distance Range. Focal length is 6 mm. Pixel size is 3,75 μm.

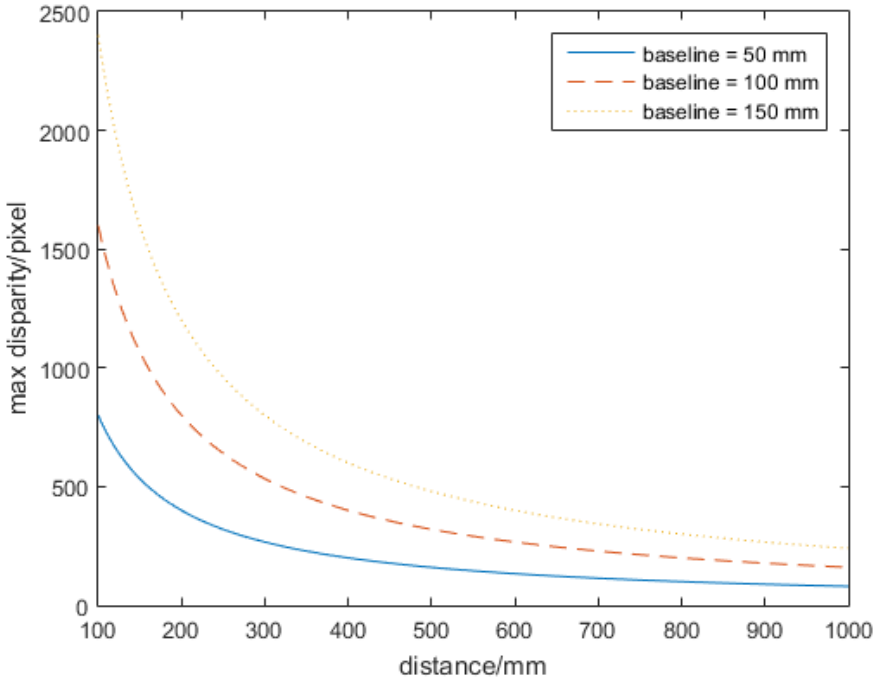

Figure 3.10: Plot of Maximum Disparity in Different Distance Range. Focal length is 6 mm. Pixel size is 3,75 μm.

But the latter method requires the the model of object thus increasing the a-priori knowledge. In addition, the increased amount of point in the point cloud set will also increase the noise level when reproject from 2D space to 3D space. Also the time it takes to align the point sets is increased with the higher amount of point number. The method used in this thesis reduces the calculation and a-priori knowledge. One thing needs to be taken care of. The feature extraction and feature point selection should be as steady as possible. If the feature points are not steady due to high noise or other reasons, then the planes extracted

from feature points are not accurately representing the pose of object which may increase the error of the total pose estimation.

The pose estimation uses SVD method to calculate Transformation matrix, which is a non-iterative based algorithm. Obviously it should be faster than the ICP algorithm, but here the set of point only consists of 3 points. One might guess that the difference between 2 algorithms is not so significant. But in the result of Arun et al. (1987)'s experiment, it showed that even for the set of point cloud consists of 3 points, the time ICP took is more than 2 times compare to the time taken by SVD method. In the same experiment, another quaternion based method is also compare and it shows that this method is faster than SVD based method for a set of 3 points. But in Lorusso et al. (1995)'s work, it shows that SVD based method has better accuracy and stability than quaternion based methods and it has best accuracy and stability performance among all 4 algorithms. Although it has slightly speed loss, for the visual servoing system, stability is more important than the speed. Thus SVD method is selected over quaternion based method.

Chapter 4

Controller Design

The controller design part of visual servoing is aiming at solving the time delay and disturbance problems. Before starting to design the controller, the source of the delay and disturbance needs to be addressed. And the plant model should be identified. Delay comes from several parts. Image acquisition, image processing and algorithm, communication to robot controller and robot controller processing, they all add delay to the total system. Total delay is denoted as τ. τ are not constant, it is variant since the components of τ is not constant each iteration. The time of image processing and algorithm and controller processing is different depends on different circumstance. Disturbance on the other hand mainly comes from the vision part, image is sensitive to the noise, which will affect depth calculation and pose estimation.

One approach is model predictive control. As the name suggests, this approach is based on prediction. At time t, the control output of future until $[t, t+k]$ is predicted by using the plant model and minimizing a cost function. The first prediction is implemented and time $t+1$, the predication is made again and it is iteratively predicted as time moves on. Thus, this is a online prediction method.

Smith predictor propsed by Smith (1959) can also deal with a system with large delay. Figure 4.1 is the block diagram of smith predictor. C is the controller, G is the plant with dead time, G_{0m} is the model of plant without dead time and G_m is the model of plant with deadtime. d is disturbance. The dead time compensation is achieved by eliminating the controller effect on dead time.

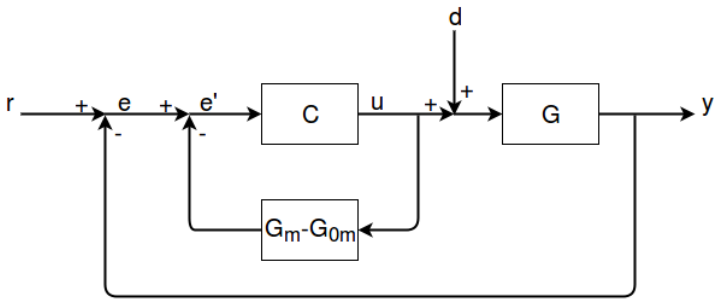

Figure 4.1: Block Diagram of Smith Predictor

An equivalent block diagram is shown in Figure 4.2, G_0 is the plant without delay, t_d is the time delay of plant, t_m is the time delay of the model plant. As shown in the block diagram, the effect of controller output on dead time is removed from the feedback. The transfer function from reference r to plant output y, known as setpoint response, is in (4.1), the transfer function from d to y, known as load disturbance response, is in (4.2).

$$\frac{y(s)}{r(s)} = \frac{CG_0e^{-t_ds}}{1 + C(G_{0m} + G_0e^{-t_ds} - G_{0m}e^{-t_ms})} \ . \tag{4.1}$$

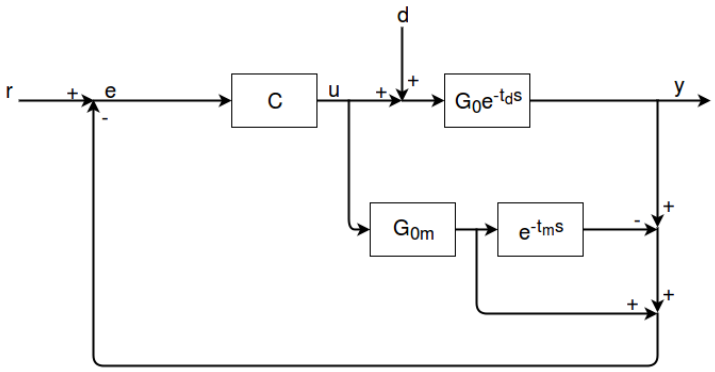

Figure 4.2: Block Diagram of Equivalent Smith Predictor

$$\frac{y(s)}{d(s)} = \frac{(1 + C(G_{0m} - G_{0m}e^{-t_m s}))e^{-t_d s}}{1 + C(G_{0m} + G_0 e^{-t_d s} - G_{0m}e^{-t_m s})} \ . \tag{4.2}$$

4.1 Modelling Plant and Controller

A 6-DoF manipulator can be controlled in either cartesian space or joint space. For cartesian space control, pose of TCP $\mathbf{P}_{TCP}$ is controlled if it is in position control mode and velocity of $\mathbf{P}_{TCP}$ if it is in velocity control mode. Same for joint space control, joint position $\mathbf{q}$ and joint velocity $\dot{\mathbf{q}}$ are controlled. The controller in the thesis is designed in cartesian space. Consider the pose of TCP $\mathbf{P}_{TCP}$ in (4.3).

$$\mathbf{P}_{TCP} = \begin{bmatrix} x \\ y \\ z \\ \alpha \\ \beta \\ \gamma \end{bmatrix} . \tag{4.3}$$

Where x, y and z is the coordinate of position. α, β and γ is the Euler angle of pose. For each element in $\mathbf{P}_{TCP}$, a smith predictor based controller is assigned (4.4).

$$\delta_{TCP} = \begin{bmatrix} \delta(x) \\ \delta(y) \\ \delta(z) \\ \delta(\alpha) \\ \delta(\beta) \\ \delta(\gamma) \end{bmatrix} . \tag{4.4}$$

Where δ_{TCP} is pose controller. From the result of pose estimation, a transformation matrix is obtained and it is further transformed from the the camera coordinate system to the manipulator TCP coordinate system. The reference pose given to the controller is defined as in (4.5)

$$\mathbf{P}_{ref} = \begin{bmatrix} x_{ref} \\ y_{ref} \\ z_{ref} \\ \alpha_{ref} \\ \beta_{ref} \\ \gamma_{ref} \end{bmatrix} . \tag{4.5}$$

$\mathbf{P}_{ref}$ is in TCP coordinate system $\mathbf{O}_{TCP}$. Equation 3.21 gives the calculation of transformed relative

transformation $\mathbf{T}_{TCP}$. It has to be transformed in to pose Equation 4.6.

$$\mathbf{\Delta P}_{ref} = \xi(\mathbf{T_{TCP}}) \ . \tag{4.6}$$

Then the reference pose can be calculated in (4.7).

$$\mathbf{P}_{ref} = \mathbf{P}_{TCP} + \mathbf{\Delta P}_{ref} \ . \tag{4.7}$$

The reference pose given to controller is the current pose of TCP plus the relative pose transformed from camera coordinate system $\mathbf{O_c}$.
Robot controller and robot in combination can be view as the plant to be controlled. Corke et al. (1996) defined that for a position control, the plant can be treated as an unit delay. For velocity control, the plant can be treated as integrator with first order delay. Figure 4.3 is the closed loop scheme of visual servoing in position control mode. G can be modelled as Integrating Plus Dead Time (IPDT) process model (4.8).

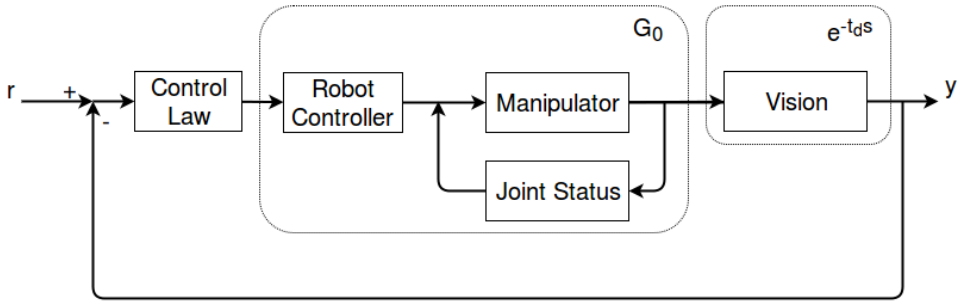

Figure 4.3: Block Diagram of Visual Servoing

$$G = G_0 e^{-t_d s} = \frac{1}{s} e^{-t_d s} \ . \tag{4.8}$$

The model of plant without delay G_{0m} is also set to $\frac{1}{s}$. The dead time is modeled by measuring the total time of image acquisition and all algorithms to get the estimate of pose. Since the delay time is not exactly same every time. The model of delay time t_m is set to 2 seconds, which is less than the measured delay time. If the delay time is overestimated, the output of plant will have oscillating behaviour which is dangerous. Figure 4.4 is an example of overestimated time delay. As indicated, for a step input signal, the smith predictor will have strange oscillating behaviour compare to an damped behaviour. Figure 4.5 shows an example of underestimated time delay, the step response of smith predictor is still damped, but will have a trend leading to overshoot.

4.2 Controller Analysis and Calculation

Smith predictor can compensate the dead time in system. However, there is one problem, as pointed out by Watanabe and Ito (1981), Smith predictor will fail with unstable plant due to cancellation of unstable poles. Either the manipulator is in position control mode or velocity control mode, the plant contains the unstable term $\frac{1}{s}$, thus the smith predictor in Figure 4.1 is not applicable. In order to solve the problem, several modified smith predictors are developed.
Astrom et al. (1994) proposed a modification on the original smith predictor. Figure 4.6 is the block diagram of this modified smith predictor. G_0 equals to $\frac{1}{s}$. In this modified smith controller, the set point response and load disturbance response is decoupled. M has 2 purposes, one is to stabilise the unstable plant, another is to reject the disturbance. It contains 3 parameters to tune in order to achieve the purposes.

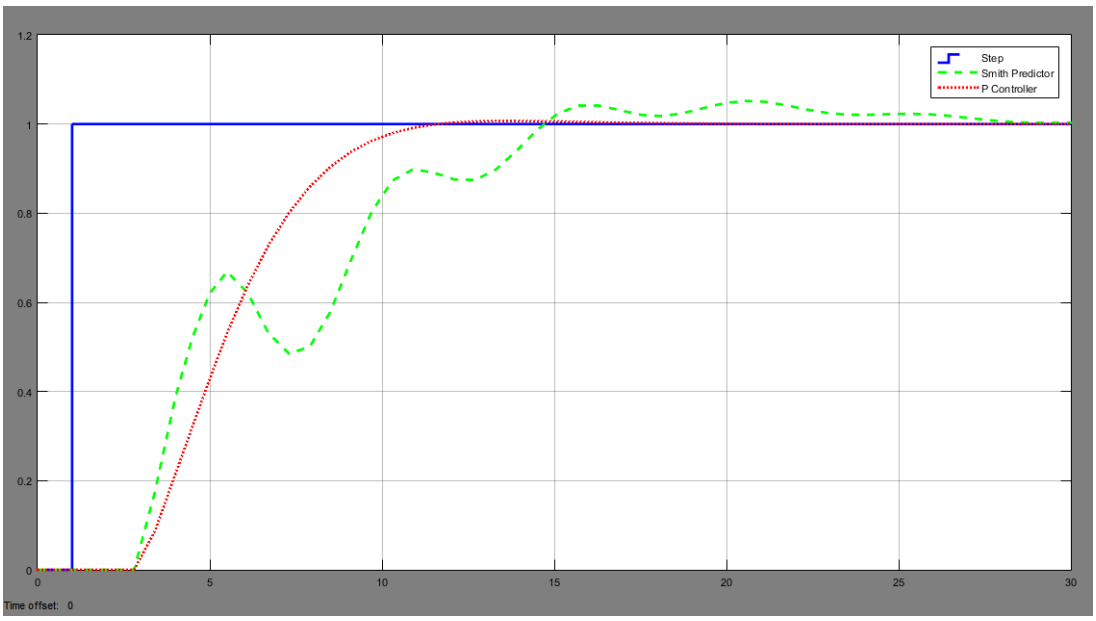

Figure 4.4: Example of Overestimated Time Delay

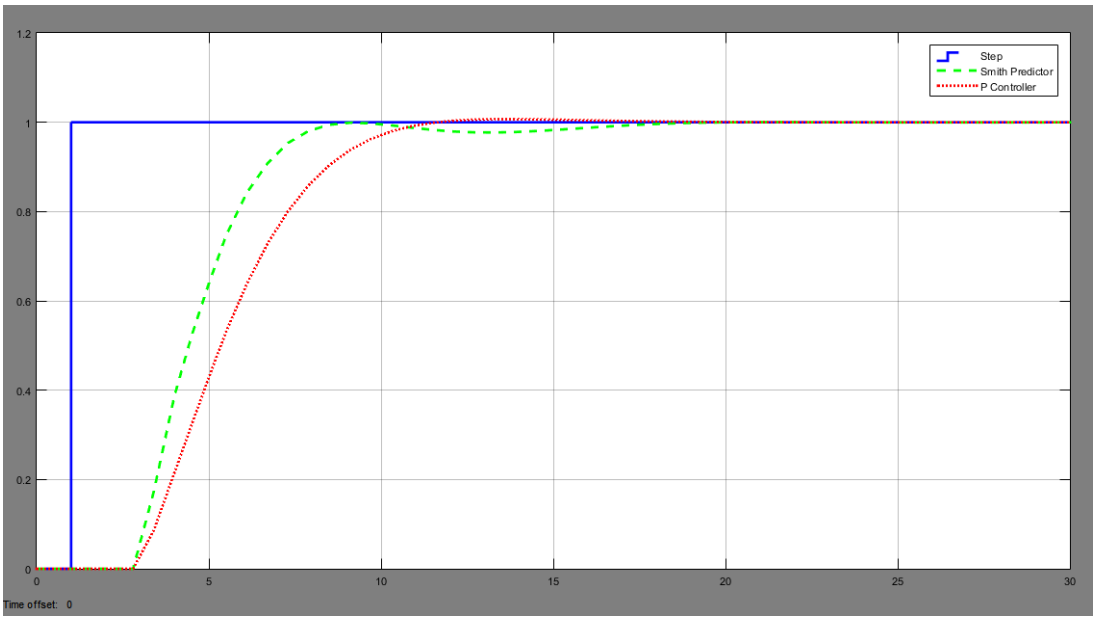

Figure 4.5: Example of Underestimated Time Delay

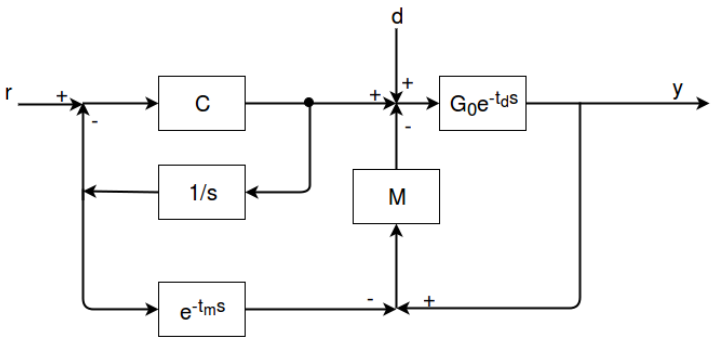

Figure 4.6: Block Diagram of Smith Predictor by Astrom et al. (1994)'s Method

The modified smith predictor used in this thesis is developed by Hongdong et al. (Equation (4.9), (4.10), (4.11), (4.12), (4.19),(4.20), (4.21), (4.24), (4.25)). It improves modified smith predictor mentioned above

by decoupling the functions of M so that an easy tuning of parameter is achieved. The decoupling is done via a cascade structure. Figure 4.7 is the block diagram. C, C_1 and C_2 are the controllers. C_1 and C_2 are used to stabilising the unstable plant. To reject the disturbance, input of C_2 is also fed back to controller C, which can be designed to reject the disturbance. Compare to Figure 4.6, the structure of inner loop of the cascade smith predictor is same, disturbance rejection is decoupled to the outer loop of cascade smith predictor.

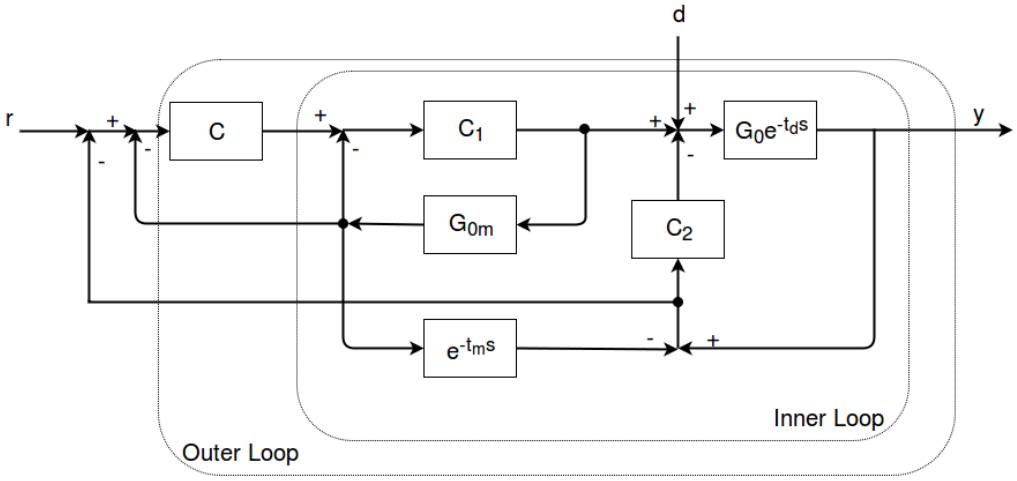

Figure 4.7: Block Diagram of Cascade Smith Predictor by Hongdong et al.'s Method

The setpoint response of the cascade smith predictor G_{yr} is indicated in (4.9). The load disturbance response is indicated in (4.10).

$$G_{yr} = \frac{CC_1G_{0m}}{1 + G_{0m}C_1(1 + C)}e^{-t_d s} \ . \tag{4.9}$$

$$G_{yd} = \frac{G_{0m}e^{-t_d s}}{1 + G_{0m}C_1(1 + C)} \frac{1 + C_1G_{0m}(1 + C - Ce^{-t_m s})}{1 + C_2G_{0m}e^{-t_d s}} \ . \tag{4.10}$$

To design the controller C, C_1 and C_2, the system must be stable. The poles of G_{yr} and G_{yd} should be on the left part of complex plane. The denominator of (4.9) is identical with the denominator of first term in (4.10). The problem narrows down to keep the poles of G_{yd} in the left part of complex plane. Since the denominator of second term in G_{yd} only contains C_2, it can be designed separately. Calculate the roots of Equation 4.11 will give the poles of second term of G_{yd}. Similarly, calculate the roots of Equation 4.12 will give the poles of first term of G_{yd}.

$$1 + C_2G_{0m}e^{-t_d s} = 0 \ . \tag{4.11}$$

$$1 + G_{0m}C_1(1 + C) = 0 \ . \tag{4.12}$$

The structure of C_2 could be either P or PD controller for stabilizing purpose according to Matausek and Micic (1996) (Equation (4.14), (4.15), (4.16)) and Matausek and Micic (1999). P controller form will be used in this thesis, the gain of C_2 is K_{c2}. Consider the manipulator is in position control mode, then G_{0m} is $\frac{1}{s}$. Equation 4.11 is transform to (4.13),

$$1 + \frac{K_{c2}}{s}e^{-t_d s} = 0 \ . \tag{4.13}$$

Note $\frac{K_{c2}}{s}e^{-t_d s}$ as $A(s)$. K_{c2} is calculated in (4.14),

$$K_{c2} = \frac{\frac{\pi}{2} - \phi_A}{t_d} \ . \tag{4.14}$$

Where ϕ_A (4.15) is the phase margin of characteristic equation. The amplitude of $A(s)$ is in (4.16).

$$\phi_A = \pi + arg(A(j\omega)) \ . \tag{4.15}$$

$$\mid A(j\omega) \mid = 1 \ . \tag{4.16}$$

To ensure Equation 4.11 is stable, K_{c2} should always less than ultimate gain K_{uc2} which is calculated in (4.17) with $\phi_A = 0$. The detailed calculation is given in section A.1.

$$K_{uc2} = \frac{\pi}{2t_d} \ . \tag{4.17}$$

Matausek and Micic (1996) gave an optimal ϕ_A to 61.3065°, which means

$$K_{c2} = \frac{1}{2t_d} \ . \tag{4.18}$$

After controller C_2 has been designed, C and C_1 should be designed together. C is selected as PI controller (4.19), C_1 is selected as P controller (4.20).

$$C = K_c(1 + \frac{1}{T_c s}) \ . \tag{4.19}$$

$$C_1 = K_{c1} \ . \tag{4.20}$$

Tuning rule of C and C_1 will be tune the setpoint response to the First Order Plus Dead Time (FOPDT) (4.21).

$$G_t = \frac{1}{\lambda s + 1} e^{-\tau s} \ . \tag{4.21}$$

After tuning setpoint response G_{yr} to G_t, parameters of C and C_1 is given in Equation 4.22 and Equation 4.23. The detailed calculation is given in section A.2.

$$K_c = \frac{T_c}{\lambda} \ . \tag{4.22}$$

$$K_{c1} = \frac{1}{\lambda K_c} \ . \tag{4.23}$$

Normally Integral time T_c is set to the desired integral time λ, then controller parameter of C (4.24) and C_1 (4.25) becomes

$$T_c = \lambda \ , K_c = 1 \ . \tag{4.24}$$

$$K_{c1} = \frac{1}{\lambda} \ . \tag{4.25}$$

Insert C and C_1, setpoint response G_{yr} is transformed into Equation 4.26. Detailed calculation is referred to section A.3.

$$G_{yr} = \frac{K_c T_c s + K_c}{\frac{T_c}{K_{c1}} s^2 + T_c s + K_c T_c s + K_c} \ . \tag{4.26}$$

For a step reference,

$$\lim_{s \to 0} sy(s) = \lim_{s \to 0} sG_{yr} \frac{1}{s} = 1 \ . \tag{4.27}$$

Which yields that the setpoint is tracked, and steady state error is eliminated. The load disturbance response is transformed into the following (4.28),

$$G_{yd} = \frac{T_c s e^{-t_d s}}{T_c s^2 + T_c K_{c1}(1+K_c)s + K_c K_{c1}} \cdot \frac{T_c s^2 + T_c K_{c1} s + T_c K_c K_{c1} s - T_c K_c K_{c1} s e^{-t_m s} + K_c K_{c1} - K_c K_{c1} e^{-t_m s}}{T_c s^2 + T_c K_{c2} e^{-t_d s}} . \tag{4.28}$$

Detailed calculation see section A.3. The limit from disturbance to output is in (4.29).

$$\lim_{s \to 0} G_{yd} = 0 . \tag{4.29}$$

Which yields that the disturbance is rejected.

4.3 Discussion

Smith predictor is selected as primary controller for the visual servoing system. It can compensate the time delay in the system effectively. But for the original smith predictor, it is only valid for the stable plant, for the unstable plant such as IPDT it will not reject the disturbance. Since the robot manipulator contains unstable term, the original smith predictor is not applicable. A lot of different kinds of modification have been made in order to make it is applicable on the unstable plant. Among which, a cascaded smith controller structure proposed by Hongdong et al. is used. This controller has a huge advantage which is the parameter tuning is decoupled. The controller C, C_1 and C_2 is able to be designed separately.

Two transfer functions, setpoint response G_{yr} and load disturbance response G_{yd}, are considered to track the setpoint and reject the disturbance. The controller selection is different with the original controller. C_2 is selected as P controller instead of PD controller. Controller is decoupled in such way that C_2 is designed separately while C and C_1 are designed together. To ensure the stability, the poles of both transfer functions must be on the left part of the complex plane. If characteristic equations Equation 4.12 and Equation 4.11 are fulfilled, stability is then ensured. Direct substitution method is applied for calculating Equation 4.11. The limit of the stable poles is on the imaginary axis. This method calculates the gain of the poles who locates on the imaginary axis. This gain is called ultimate gain, as long as the controller gain is less than ultimate gain, poles are located on the left part of complex plane. For Equation 4.12, direct substitution is not applied. The reason is, after calculation, ω depends on the controller parameter. Instead, the tuning rule applied it to tune the transfer function to the FOPDT dynamic $\frac{1}{\lambda s+1} e^{-} \tau s$. For $\lambda > 1$, the pole is on the left part of complex plane. Thus the stability can be assured as long as parameters of controllers are selected according the results of the 2 stability analysis methods.

After the calculation of system stability is done, controller parameters still needs to be optimized. For controller C_2, Matausek and Micic (1996) gives an optimal parameter selection. For controller C and C_1, K_{c1} depends on K_c, the controller parameter selection is narrowed downed to considering parameters of C_1. As a PI controller, the integral time T_c is usually selected so that $T_c = \lambda$. Other parameters are obtained too. The limit of setpoint response equals to 1, which means the setpoitn is followed. The limit of load disturbance is 0, which means the disturbance is rejected.

The model of the dead time should be considered carefully. If the dead time is overestimated, the plant output will have undesired oscillating behaviour, this could create problems especially for 6 DoF manipulators. When the dead time is underestimated, the output of plant is kept damped, which is better than oscillation. The model rule will be using underestimated dead time.

Chapter 5

Implementation

The implementation work is divided into hardware part and software part. In hardware implementation part, what kind of hardware is used and how it is set up is explained. In the software implementation part, it is explained how the algorithms and controller are implemented and how the communication and driver are working.

5.1 Hardware

The 6 DoF manipulator used is UR5, a collaborative robot developed by Universal-Robots. Table 5.1 is the data of UR5.

Payload	5 kg
Reach	800 mm
Joint Range	$\pm 360°$
Speed	Joint: 180 $°/s$ Tool: 1000 mm/s
Repeatability	$\pm 0{,}1$ mm
Communication	TCP/IP 100 Mbit IEEE 802.3u 100BASE-TX Ethernet Socket Modbus TCP

Table 5.1: UR5 Data from Universal-Robots

Cameras are 2 USB 3,0 Leopard RGB camera module, type is LI-USB30-M021C (leopard image). Table 5.2 is the data of cameras.

FoV (D/H/V)	0° / 48° /34,5°
Focal Length	6,0 mm
Pixel Size	3,75 μm
Interface	USB 3,0
Resolution	1280*960 @ 30 fps 1280*720 @ 60 fps 800*460 @ 90 fps 600*460 @ 30 fps

Table 5.2: Camera Data from leopard image

Camera is mounted on a frame which is 3D printed. Camera is connected to a PC where all software is implemented. 2 cameras are connected to different USB ports of PC. The control of manipulator is also from PC, details of how it is achieved will be discussed in software implementation part. But the

communication between robot controller and PC must be established. Communication is via Ethenet cable. Figure 5.1 indicates the hardware architecture.

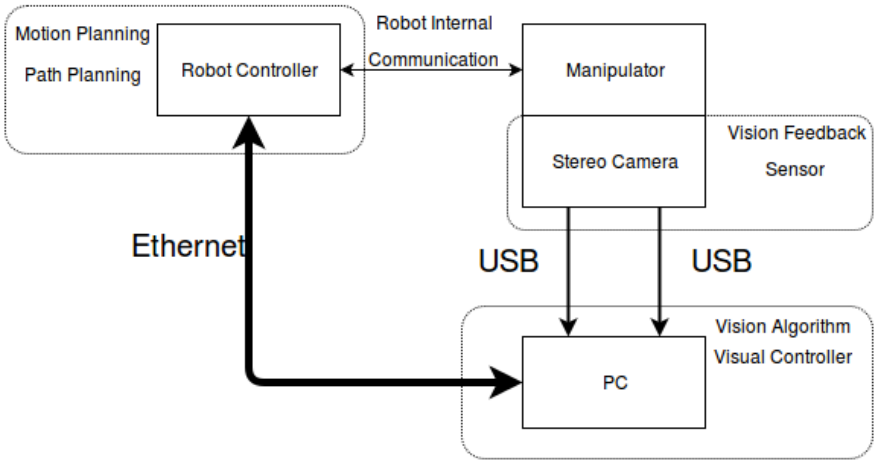

Figure 5.1: Hardware Architecture of System

Figure 5.2 indicates the actual hardware configuration.

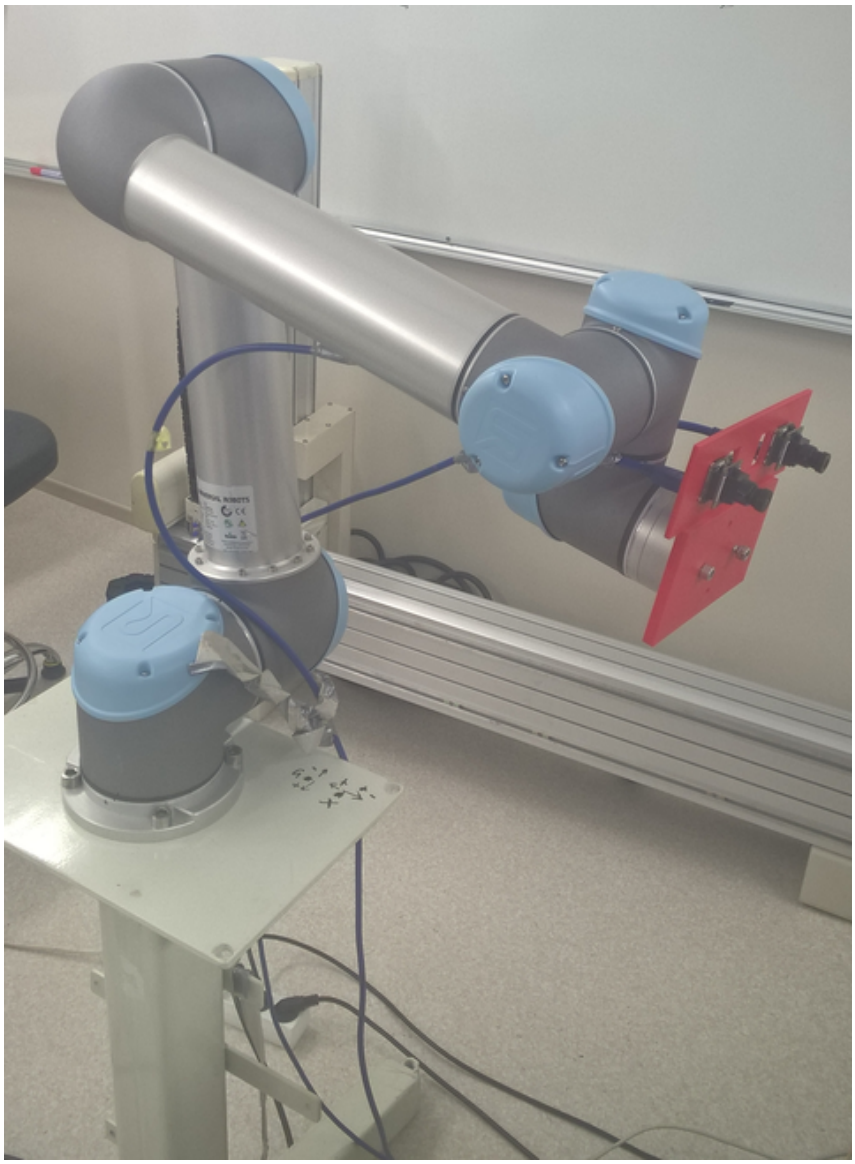

Figure 5.2: Actual Hardware of System

5.2 Software

The entire software is based on ROS. The major reason for choosing ROS is that it derives the dependency to the hardware. Once the software is developed, it can be implemented on other manipulators without any changed. Only the ROS driver for the new manipulator needs to be installed. This gives more flexibility on hardware and reduce the requirement for re-development. Another reason is the architecture is suitable for teamwork and further development. The concept of ROS is that one master called "core" which handles slaves called "node". The communication between nodes are topics or services. A node can both publish messages to and subscribe messages from topics. Messages are data types which contain the information. Topics are periodically updated. Service is however based on request and reply mechanism. Figure 5.3 illustrates the ROS node communication mechanism.

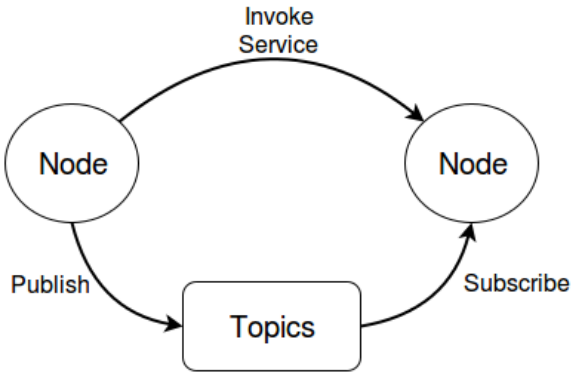

Figure 5.3: ROS Topic and Service

For example, an image could be published to an topic in the form of image message, another node subscribing this topic will get the message periodically. The image is sent from one node to other node by topic mechanism.

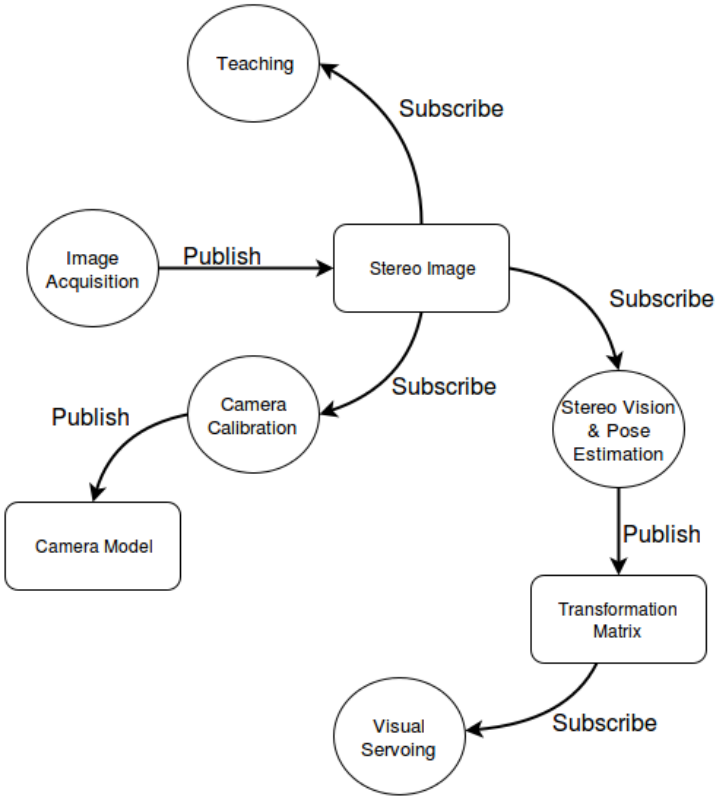

Figure 5.4: Overview of ROS Topics and Nodes

Figure 5.4 is the overview of the nodes and topics used in this thesis. The rounded boxes are the nodes and rectangular boxes are the topics. Image acquisition node gets the images from cameras and publishes image topics. Camera calibration node and teaching node is responsible for calibration and teaching described in Chapter 2. In stereo vision and pose estimation node, object recognition, stereo vision and pose estimation algorithms are implemented. Controller is implemented in node visual servoing, where it also sends commands to control robot and get status from robot.

In image acquisition node, some additional processing steps are required. The output of camera is raw image data, in order to apply image processing and vision algorithms, it has to be converted into RGB form. Figure 5.5 shows the algorithm for image conversion.

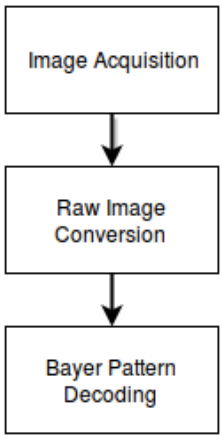

Figure 5.5: Image Acquisition and Conversion

Raw image has a different way to pack pixel value compare the RGB image. The sensor data form camera is minimal processed. After conversion, a bayer filter needs to be applied for decoding, so that color arrangement will be correct. After both camera can acquire RGB images, Sharpness should be tuned for both camera. Cameras can be manually adjusted so that images are properly focused. Image blurriness of both images are calculated and focus is adjusted to make both blurriness of images falls into a certain range. Then the 2 cameras are considered both properly focused. After the images are pre-processed, it will be applied with further image processing techniques and vision algorithms. OpenCV library is used for this part. It is a well developed open source library which covers most of the important topics in image processing and computer vision. It also has quite extensive implementation of machine learning algorithms.

For the purpose of controlling the manipulator, ROS-Industrial (ROS-I) package provides the possibility to control industrial robots. Except for the UR5, a few more other industrial robots are also supported by ROS-I such as ABB, FANUC and Motoman. In addition, Robotiq gripper is also supported. Figure 5.6 is the overview of architecture.

ROS-I provides the interface between the robot controller from manufacturer and ROS. ROS layer handles other nodes described in Figure 5.4. UR5 model is stored in configuration files. An improved driver, UR_Modern_Driver, for Universal Robots developed by Andersen (2015) is used. The new driver has several improvements compare to the original one. First of all, teach pendant can be used while driver is running. Secondly, velocity control in joint space of UR5 is added which suits for the purpose of visual servoing. Velocity of joints is set via writing to a topic. And install the new driver won't affect the program you developed before.

MoveIt package (from MoveIt) provides motion planning, trajectory planning, cartesian space control and joint space control over a wide range of robots. Supported robots include Universal Robots, ABB, Kuka, Motorman, Baxter and many other. Figure 5.7 indicates the overview of the MoveIt package. A node called move_group handles all the functionalities. User interface supports C++ and Rviz. It provides flexible integration opportunities. The interface in Rviz is a GUI interface which is suitable for manual

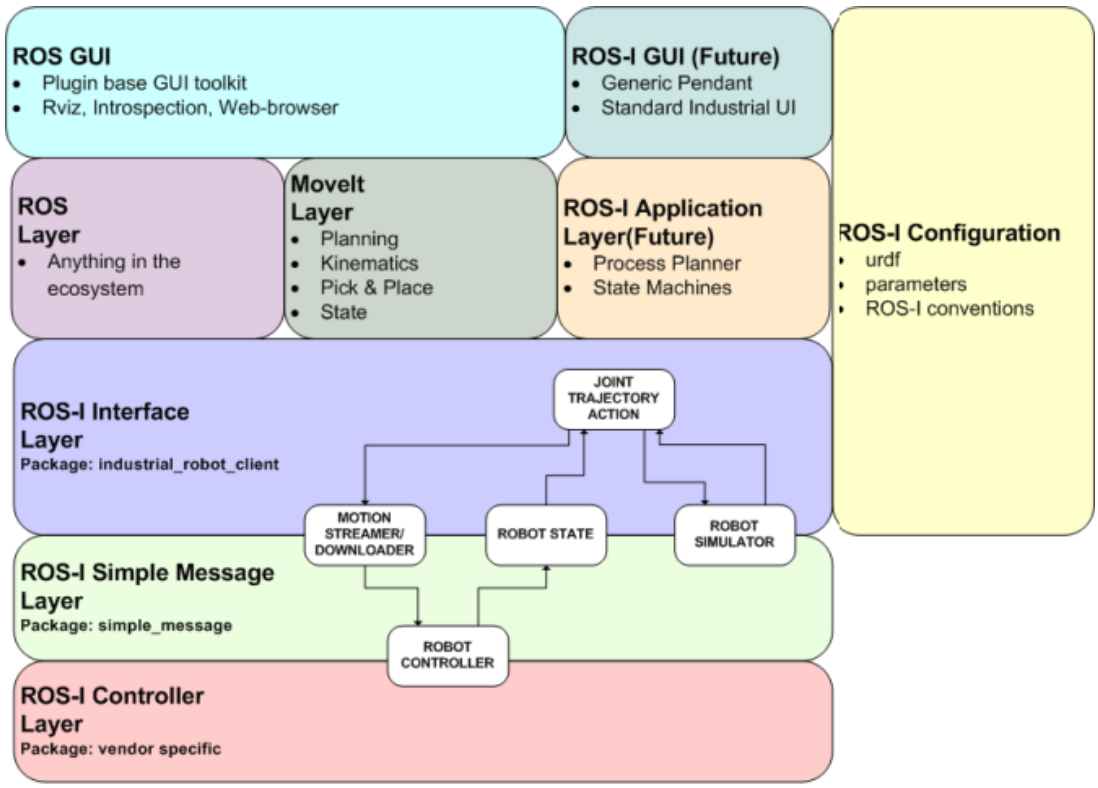

Figure 5.6: ROS Industrial Architecture (from ROS-Industrial)

test. With C++ interface, it is easily being able to integrated into the total visual servoing system.

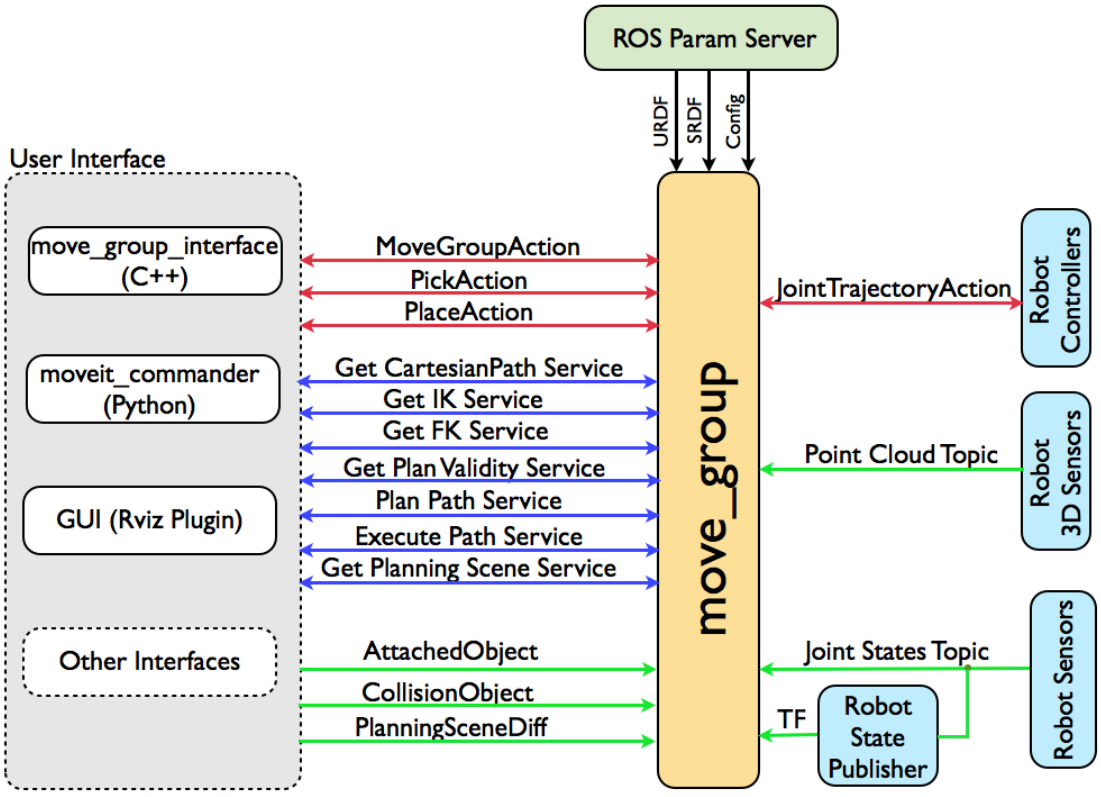

Figure 5.7: Overview of MoveIt Architecture (from MoveIt)

The communication between robot controller and ROS-I is TCP/IP connection. But user doesn't need

to consider the protocol and low-level communication. ROS Topic/Service mechanism is the interface. Node UR_Driver handles the communication. It publishes and subscribes different topics and this is how communication to robot controller is done by user. The topics published and subscribed by UR_Driver is in the Table 5.3.

Table 5.3: Topics Published and Subscribed by UR Driver

Topic Name	Description	Published/Subscribed
io_states	Status of DI, DO and AI, AO of robot controller	Published
joint_states	Joint status of manipulator	Published
wrench	Force/torque at TCP	Published
follow_joint_trajectory	Integration with MoveIt	Published
URScript	Send movel/movej commands to robot controller	Subscribed
joint_speed	Send joint speed and acceleration to robot controller	Subscribed

Except for the packages mentioned above, a few other libraries are used to complete the implementation work. Table 5.4 below is the complete libraries and packages used in the implementation.

Library and Package	Field of Application
ROS	Overall Architecture
OpenCV	Vision
M021_V4L2	Image Acquisition
MoveIt	Robot Planning
UR_Modern_Driver	Robot Driver

Table 5.4: Libraries and Packages Used

In summary, all the selections of packages and libraries are based on the flexibility and possibility of future development and system expansion. All the software implementation is based on ROS. Using ROS can separate the hardware from software. This decoupling makes it possible to seamlessly transfer the developed code with another hardware. Another advantage of ROS is that it is suitable for team work. Tasks can be easily divided and it is easy for management. Furthermore, it has integrated a lot of third party library and package. For instance, OpenCV and PCL for 2D and 3D computer vision process. Gazebo for robot simulation. ROS-I for industrial robot control and MoveIt for manipulation. And of course it has C++ interface which is the programming language in the software implementation work. The reason for selecting it as programming language is the requirement of real time processing.
ROS-I supports the UR5 among other industrial manipulator hardware. It provides the possibility to control the manipulator within ROS frame. This package can be viewed as an driver for industrial robot, so that different programming environments from different manufacturer is abandoned and a unified development environment is assured. An improved package, UR_Modern_Driver is used instead of the one from ROS-I. The improved one offers more freedom in control.
To control the robot manipulator, MoveIt package is considered. It supports a wide range of robots and it provides all necessary functionalities for robot manipulation. Although not all of them are used in this thesis, it is good that the package has the possibility to integrate other functionality in case this design will be further developed in the future.
For image processing and vision algorithm, OpenCV is used. But a pre-processing step is performed

first. Raw images is transformed to RGB plane images and then images from 2 cameras are tuned at same focus. After this step, OpenCV can be applied properly. PCL could also be used for visualization purpose, but 3D vision processing is not considered in this thesis.

Chapter 6

Results

The results are presented by the different tasks in this thesis. Firstly object recognition algorithm is tested. Then the stereo vision result is analysed and at last the results of controller design is interpreted.

6.1 Object Recognition

In order to test the object recognition design, 2 experiments are carried out. The first one is to test the effectiveness of the algorithm regarding the invariance to scale, orientation and lighting condition. The second one is to test the orientation range that it could work.
For the first object detection and recognition experiment, the proposed descriptor and online correction algorithm is tested with 3 different objects. The first one is a tea box with a mark of black circle. The second one is a tea box with a mark of randomly selected colored image. The third one is a colored image on a cup, the color image is also randomly selected. Each object is tested with 4 different poses in dark and bright lighting conditions. With this setup, whether the proposed descriptor is invariant to the orientation and scale in different lighting condition can be tested. Figure 6.1 and Figure 6.2 is tested with black circle in different lighting conditions. Figure 6.3 and Figure 6.4 is tested with a color pattern in different lighting conditions. Figure 6.5 and Figure 6.6 is tested with a color pattern on a cup in different lighting conditions.

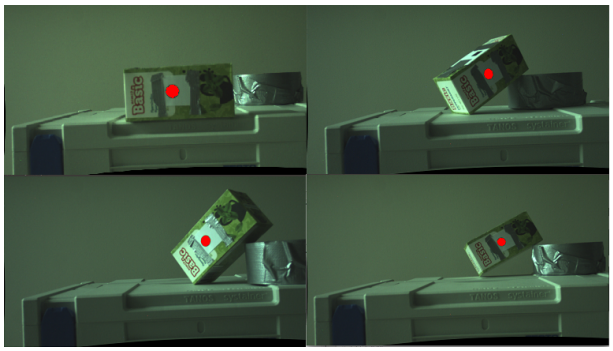

Figure 6.1: Object Recognition with Black Circle Mark from Different Position and Orientation in a Dark Light Condition

As indicated in the figures above, the recognized contour is drawn in red color. All objects are recognized from different poses in both dark and bright lighting conditions. The 4 different poses tested covers different position and orientation, which proves the invariance to scale and orientation. Using black mark and color pattern in the experiment, the proposed descriptor is shown effective on both textured or texture-less pattern. The different shapes, circle and rectangle, are also compared, which indicates that the descriptor is also invariant to shape. Putting the color pattern on a cup evaluates the validation of a non-flat surface. All the experiment setups are tested with 2 different lighting conditions, one is dark and

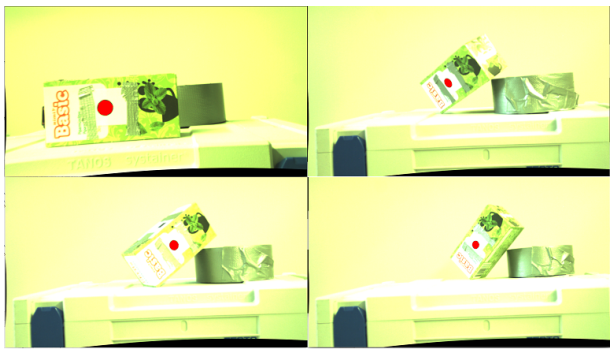

Figure 6.2: Object Recognition with Black Circle Mark from Different Position and Orientation in a Bright Light Condition

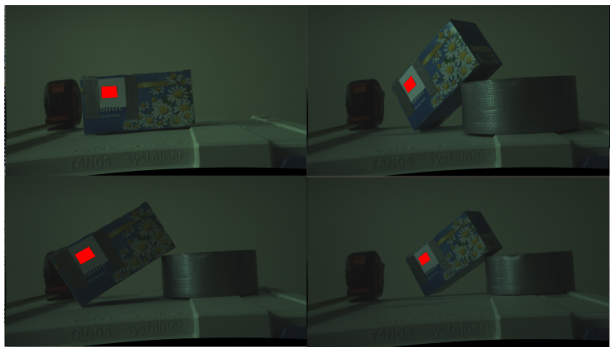

Figure 6.3: Object Recognition with Color Pattern from Different Position and Orientation in a Dark Light Condition

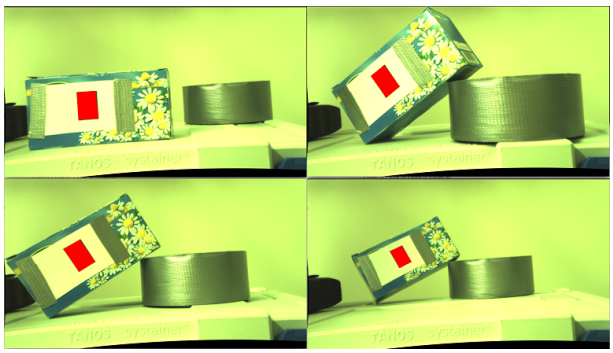

Figure 6.4: Object Recognition with Color Pattern from Different Position and Orientation in a Bright Light Condition

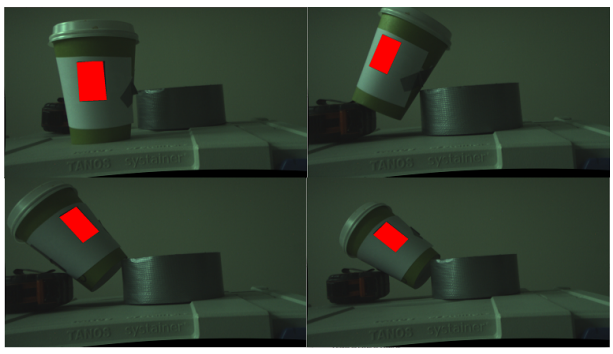

Figure 6.5: Object Recognition with Color Pattern on a Cup from Different Position and Orientation in a Dark Light Condition

the other one is bright. In the dark lighting condition, the lights in the lab are off. The bright lighting

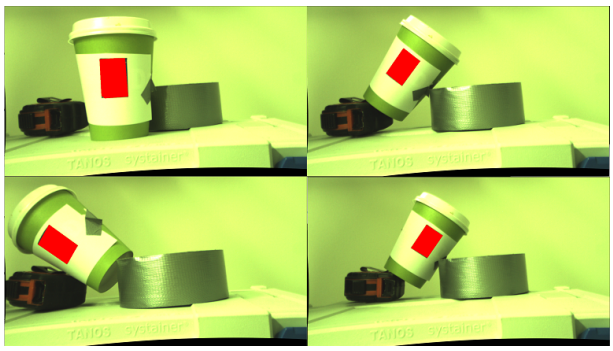

Figure 6.6: Object Recognition with Color Pattern on a Cup from Different Position and Orientation in a Bright Light Condition

condition is with lights in the lab on. Proposed online correction algorithm is tested effective in both lighting conditions for all setups.

In the second experiment, the orientation range that within which proposed descriptor is effective is tested. Taking camera coordinate as reference coordinate. The object are rotated around x axis and y axis both in clockwise direction and counter-clockwise direction. Clockwise is noted as direction "+" and counter-clockwise is noted as direction "−". Rotation around z axis is not considered since it will not have perspective projection. The object is place in front of camera, the distance from center of object to the left camera lens is 400 mm. In test the black circle pattern is used and the in bright light condition.

Table 6.1: Results of Effective Orientation Range of Proposed Descriptor

	Angle	Direction
X Axis	77°	+
	72°	−
Y Axis	68°	+
	58°	−

6.2 Stereo Vision

Cameras need to be calibrated before constructing the stereo rig. The number of images taken for calibration is 14. Table 6.2 shows the result of calibration. In single calibration, both the errors of left and right camera are less than 0,5 pixel. For stereo calibration, the error just exceeds 1 pixel.

Table 6.2: Results of Single Camera Calibration and Stereo Camera Calibration

	Error (pixel)
Left Camera Single Calibration	0,456
Right Camera Single Calibration	0,458
Stereo Camera Calibration	1,106

To verify the stereo rig, the following experiment is carried out. A small window is drawn on the image, the mean depth value within this small window is treated as camera measured depth value. In the experiment, 20 measures are conducted. The small window helps image center is on the same part of object in all the measures. In each measure, the object is placed 5 mm away from the cameras. The depth information will be measured and analysed. However it is difficult to measure the absolute depth from object to camera center. The structure and dimension within camera is unknown from the data

sheet provided by manufacturer. The depth taken into account is shifted depth which is the distance from object to the lens of left camera. The unknown distance from camera lens to sensor plane can be compensated. Figure 6.7 below is the camera measured depth versus shifted depth.

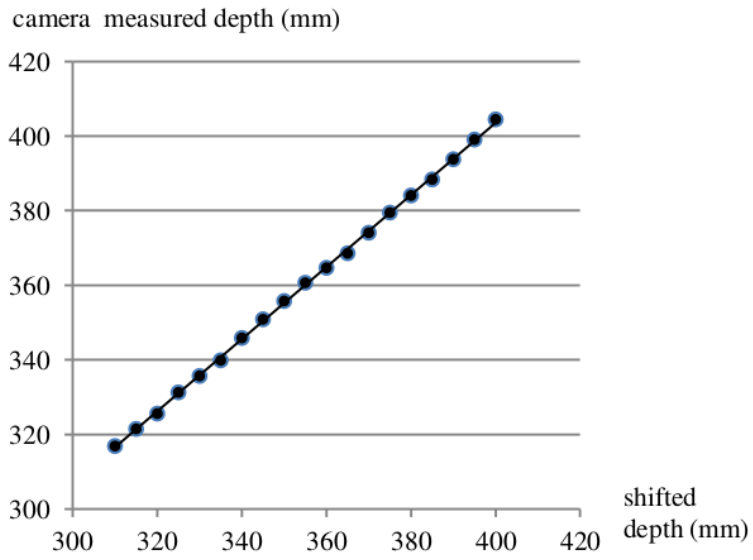

Figure 6.7: Measured Depth vs. Shifted Depth

The measured depth is the depth calculated by Equation 3.9. The shifted depth and camera measured depth fits in a linear regression. The error is interpreted in the following way.

$$\textbf{Error} = \mid \Delta Z_{shift} - \mathbf{\Delta Z_m} \mid \ . \tag{6.1}$$

Where ΔZ_{shift} is the displacement of shifted depth between 2 measures and $\mathbf{\Delta Z_m}$ is the displacement of camera measured depth between 2 measures. Using this error could eliminate the necessity of compensating the distance between lens and sensor plane which is one source of uncertainty. For all the measurements in the experiment, ΔZ_{shift} is 5 mm. Figure 6.8 is the displacement of camera measured depth $\mathbf{\Delta Z_m}$.

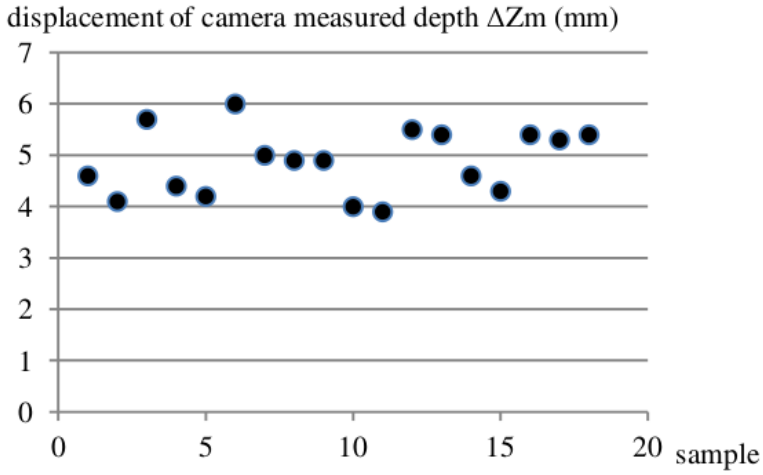

Figure 6.8: Displacement of Camera Measured Depth $\mathbf{\Delta Z_m}$

Almost all of the measures falls in the range 1 mm around the ΔZ_{shift}. This implies that for almost every time the object is moved 5 mm away from camera, the camera measured depth displacement is within 5 mm$\pm$1 mm. Figure 6.9 displays the **Error**.

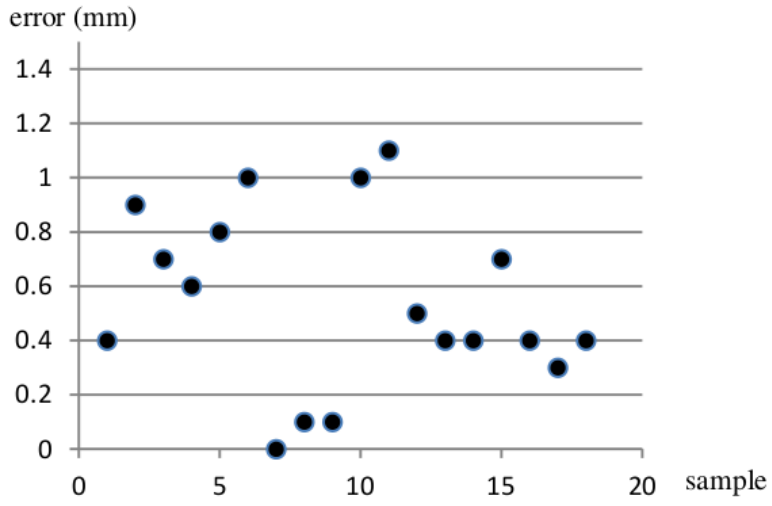

Figure 6.9: Error of Stereo Vision

Most of the error is within 1 mm. The maximum error is 1,1 mm. Table 6.3 indicates the mean value and standard deviation of $\mathbf{\Delta Z_m}$ and **Error**.

Table 6.3: Mean Error and Standard Deviation of $\mathbf{\Delta Z_m}$

	Mean (mm)	Standard Deviation
Displacement of Camera Measured Depth	4,867	0,63
Error	0,544	0,33

6.3 Controller

The controller is simulated in Matlab before it could be implemented so that it is sure that no unwanted behaviour will occur which may cause damage. Simulink will be used for simulation. Figure 6.10 is the block diagram of the cascade controller structure in Chapter 4. Figure 6.11 is the block diagram of a P controller with feedback control. The actual simulink block diagram used for simulation can be found in Appendix B. The P controller with closed loop feedback can stabilize the unstable plant, but it can't compensate the dead time. The purpose of constructing a P controller is to compare performance with the cascade smith predictor.

The controllers parameters need to be determined first. Parameter selection of cascade smith predictor involves selecting λ in Equation 4.23. Parameters of C and C_1 depends on λ. Different λ means location of pole $(-\frac{1}{\lambda})$ is different. Table 6.4 shows the parameters of different values of λ. Parameter of C_2 relates to delay time. t_d is set to 2,5 seconds. K_{c2} equals to 0,2. For the Feedback P controller, gain is set to 0,18.

Table 6.4: Parameters of Controller C and C_1 for Different λ

	$\lambda = 0,2$	$\lambda = 1$	$\lambda = 2$
K_c	1	1	1
T_c	0,2	1	2
K_{c1}	5	1	0,5

There are several factors which will affect the system and worth to investigate in. The simulation for controller design should be analysed thoroughly. Since the cascade smith predictor structure decouples the

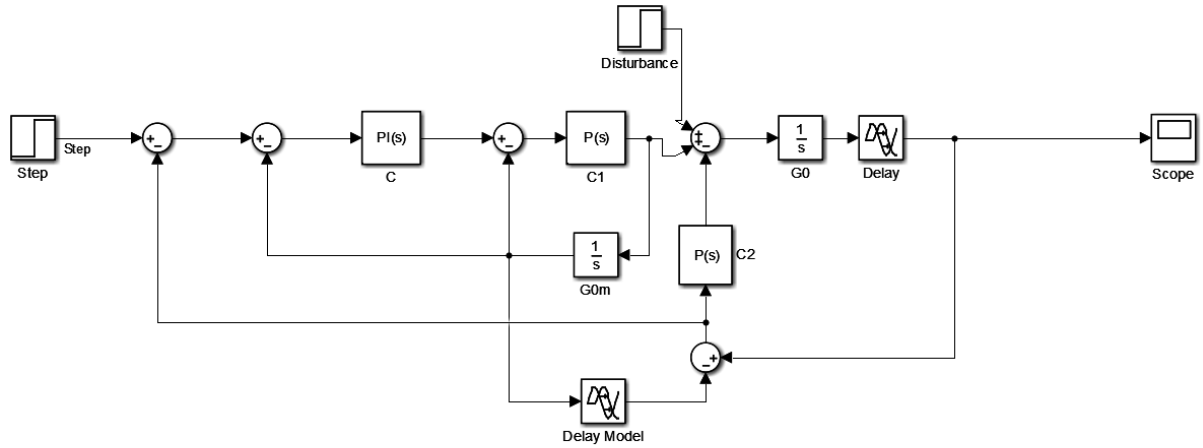

Figure 6.10: Block Diagram of Cascade Smith Predictor in Matlab

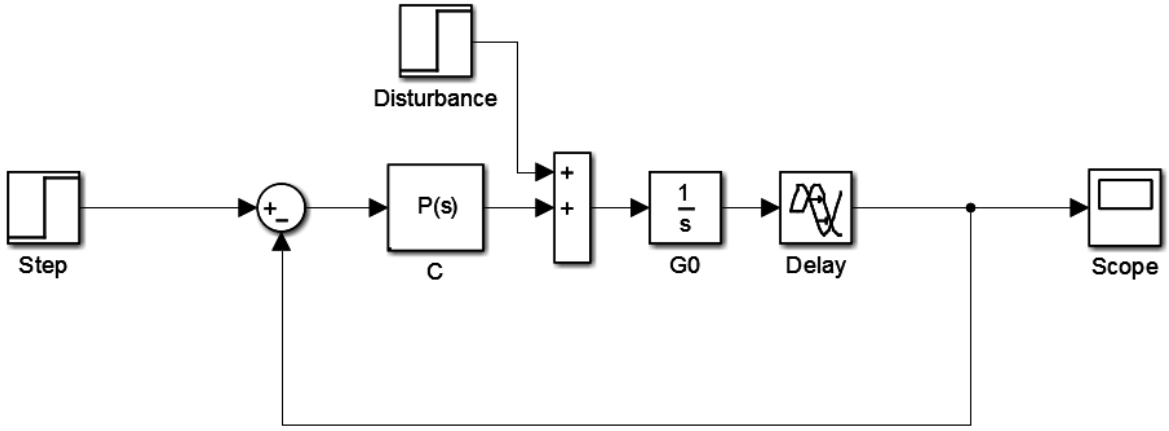

Figure 6.11: Block Diagram of P Controller in Matlab

controller for following setpoint and rejecting disturbance. The parameters of controllers are manipulated to analyse different performance. Firstly, whether and how the cascade smith predictor can compensate the deadtime is simulated. The step response of it is compared to a P controller to see the difference. Secondly, analysis of the performance of step response is analysed. By doing this, different λ is given to compare different dynamics. After that, the load disturbance response is simulated. Cascade smith controller and P controller are given with 2 different disturbance. One is triggered after the system has reached equilibrium and the other one is triggered during the rising time. Next is to test the effects of different controllers on the disturbance dynamics. C_2 is decoupled with C and C_1. Thus how K_{c2} and λ is affecting disturbance rejection is analysed separately. Finally, The modelling error of deadtime is investigated. The possible situation, overestimating the deadtime and underestimating it, is simulated to evaluate the different impact on the system behaviour by modelling different deadtime.

As mentioned above, λ will change the place of pole. Selection of model of time delay t_m will also affects the performance as discussed in Chapter 4. Disturbance is certainly another important factor. All the factors will be manipulated in order to find out how one single factor affects the total system. In all simulations, a step reference signal is trigged at time 1 second. Firstly disturbance is turned off, model of time delay t_m is set same as t_d. The effect of different λ will be analysed. Figure 6.12 shows the step

response of cascade smith predictor and P controller. Figure 6.13 shows the responses with differnet λ values.

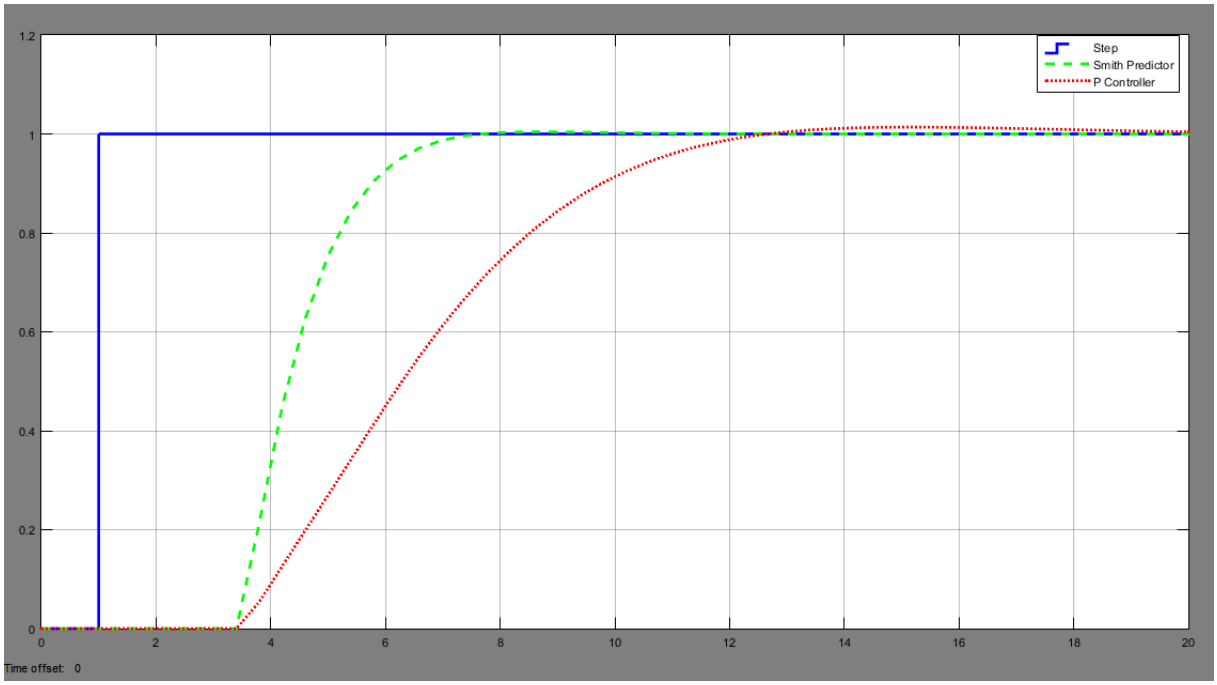

Figure 6.12: Step Response from Cascade Smith Predictor and P controller without Disturbance

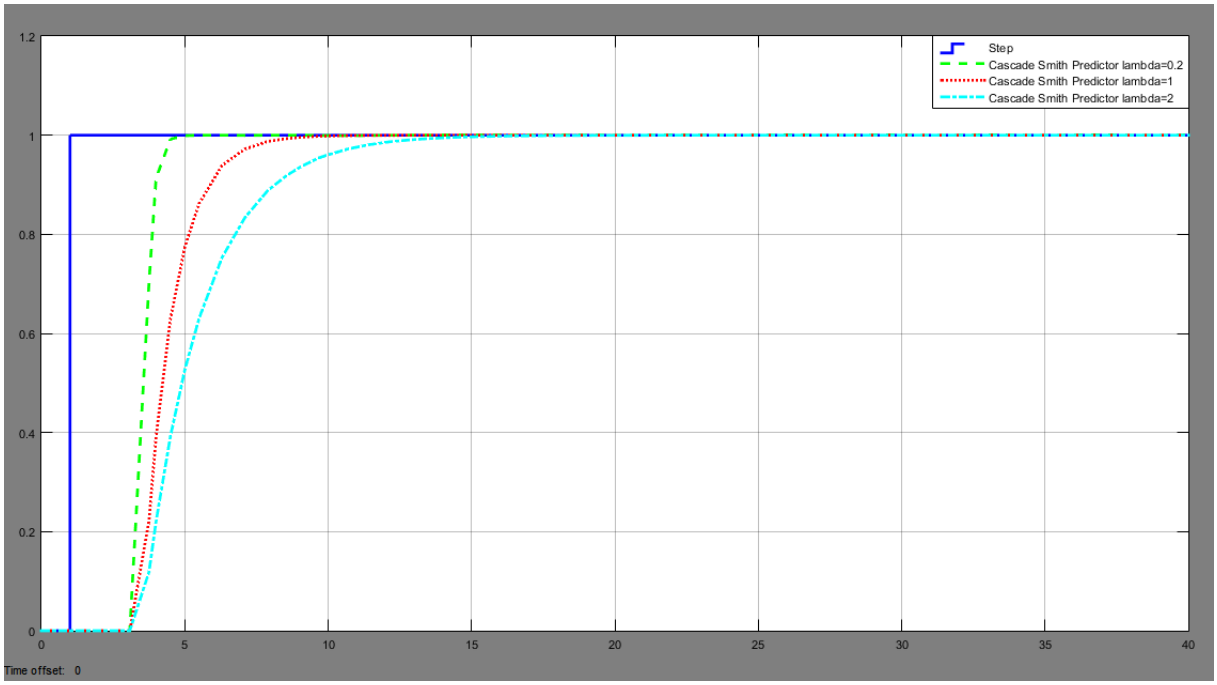

Figure 6.13: Step Responses from Cascade Smith Predictor with Different λ without Disturbance

As indicated in the figures, the cascade smith predictor can compensate the dead time as expected while the P controller has much slower response to the dead time. There is no overshoot in cascade smith predictor. With smaller value of λ, the pole is moved further away towards left from the origin of complex plane, which means the system faster response. The reason is the more left the pole is from origin, the higher gain it is. This can be observed from Figure 6.13. Also both smith predictor can P controller converge to the setpoint which indicates the stability. Now with the same parameters, but disturbance is added. The disturbance come in at 15th second and its amplitude is 0,1. Figure 6.14 shows the response.

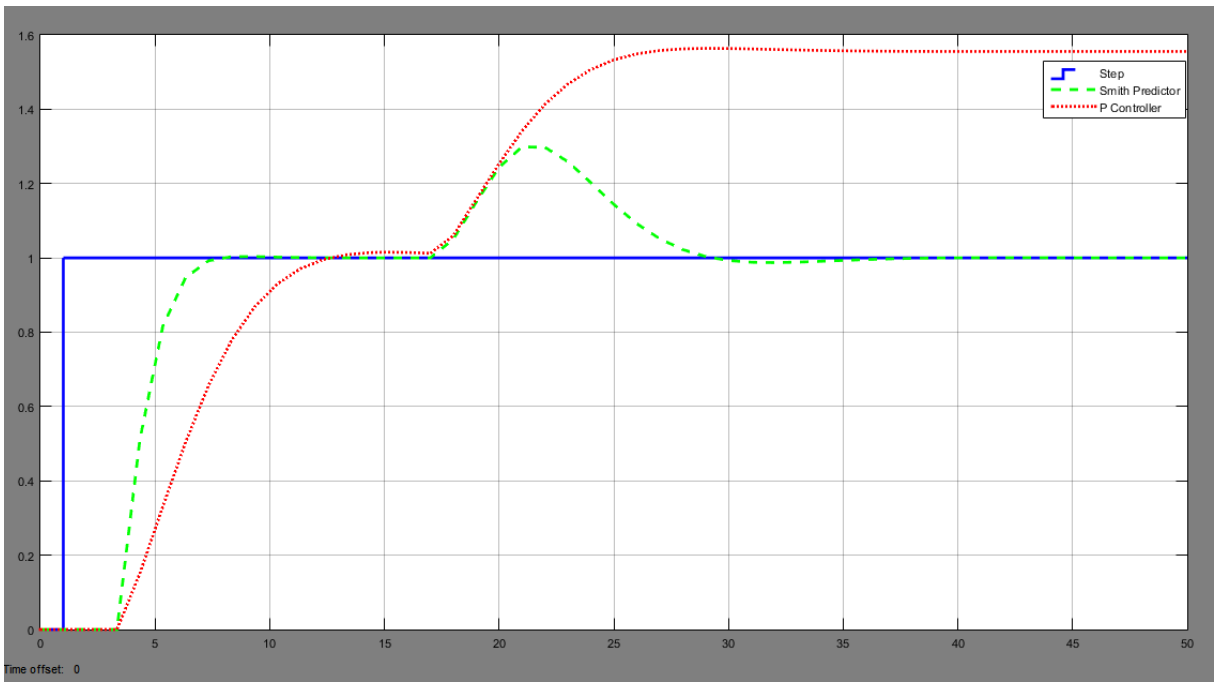

Figure 6.14: Step Responses from Cascade Smith Predictor and P Controller with Disturbance at 10th Second

The disturbance comes after both cascade smith predictor and P controller have reached steady state. Cascade smith predictor can reject the disturbance. Although P controller can stabilize system after disturbance is injected, it can't reject it. The disturbance may also come when controllers haven't reached equibrilum yet. Figure 6.15 is the responses of both controllers when the disturbance comes at 3rd second.

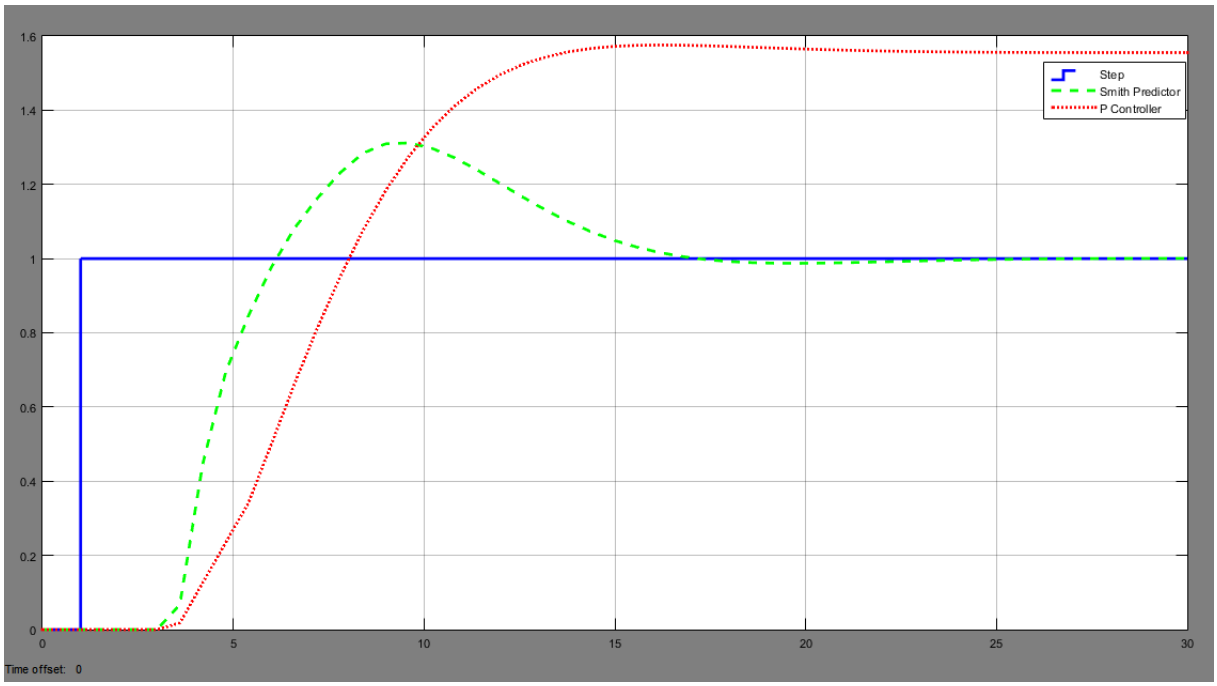

Figure 6.15: Step Responses from Cascade Smith Predictor and P Controller with Disturbance at 3rd Second

The disturbance is again rejected by cascade smith predictor. From the load disturbance response G_{yd}, parameters of C, C_1 and C_2 will all affect the response. But it can be analysed separately as proved in Chapter 4. Set the disturbance trigging time back to 10th second. The rest parameters remain same.

Load disturbance responses with different λ values is compared in Figure 6.16.

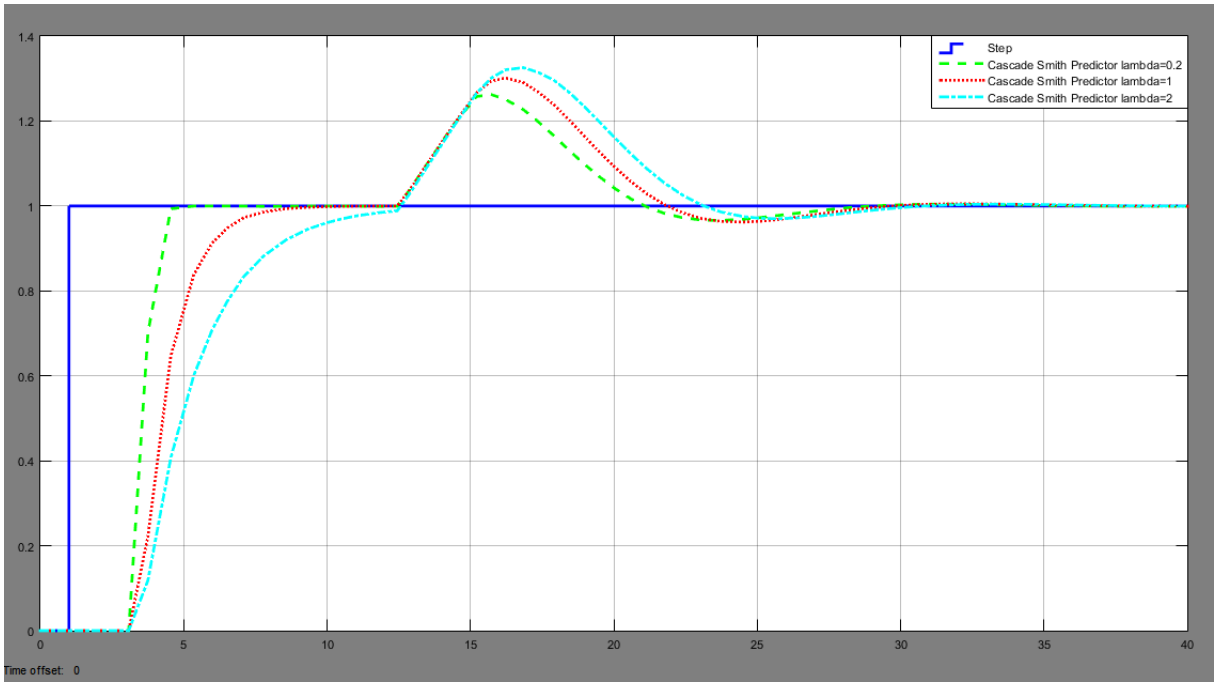

Figure 6.16: Responses to Disturbance with Different λ When Disturbance Comes at 10th Second

As it can be observed that the smaller the λ is, the smaller the amplitude of response to disturbance. Furthermore, the response patterns are quite similar, the effects of different λ is mainly on the amplitude of disturbance response. To evaluate the effect of K_{c2} on load disturbance, λ is set to 1 and 3 different K_{c2} are tested. Figure 6.17 shows the result.

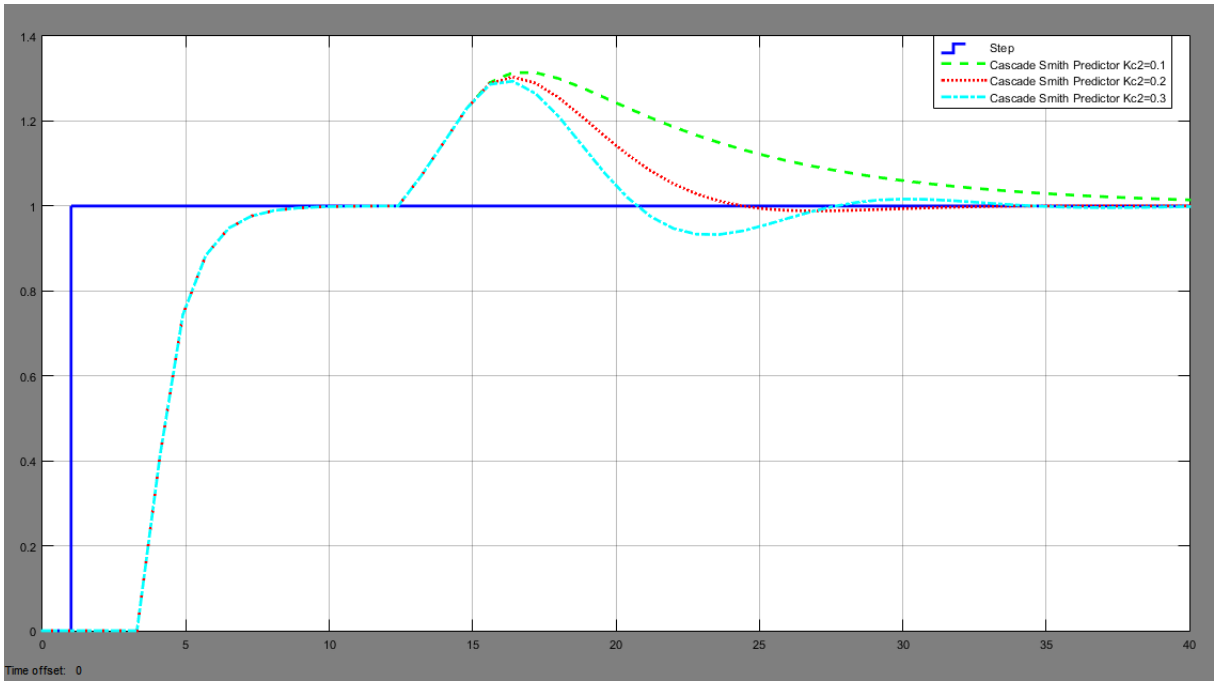

Figure 6.17: Responses to Disturbance with Different K_{c2} When Disturbance Comes at 10th Second

$K_{c2} = 0,2$ is calculated according to the optimized phase margin suggests by Matausek and Micic (1996), the response is optimized as well. Increasing K_{c2} will have oscillating behaviour and decreasing K_{c2} will increase the time the system reaches equilibrium.

The model of time delay will also have effect on the system behaviour. In next simulation it will be

analysed. Both Overestimated and underestimated t_m will be investigated. Figure 6.18 shows the effects of overestimated t_m. Figure 6.19 shows the effects of underestimated t_m.

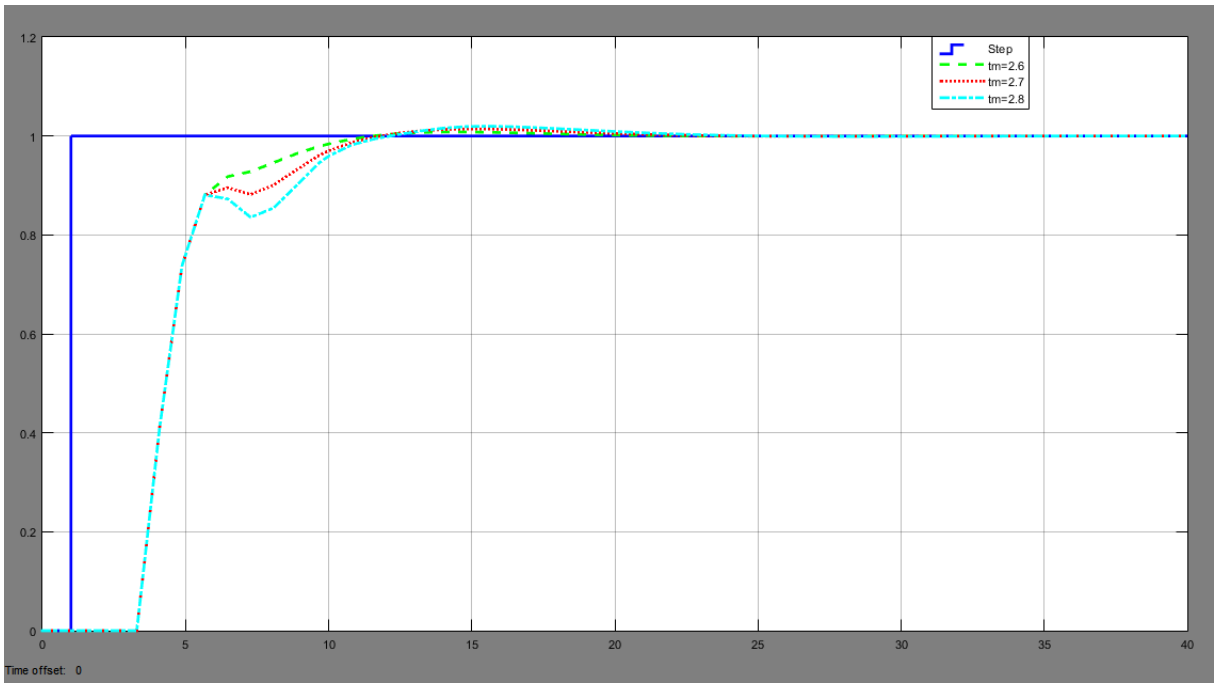

Figure 6.18: Step Responses with Different Underestimated t_m without Disturbance

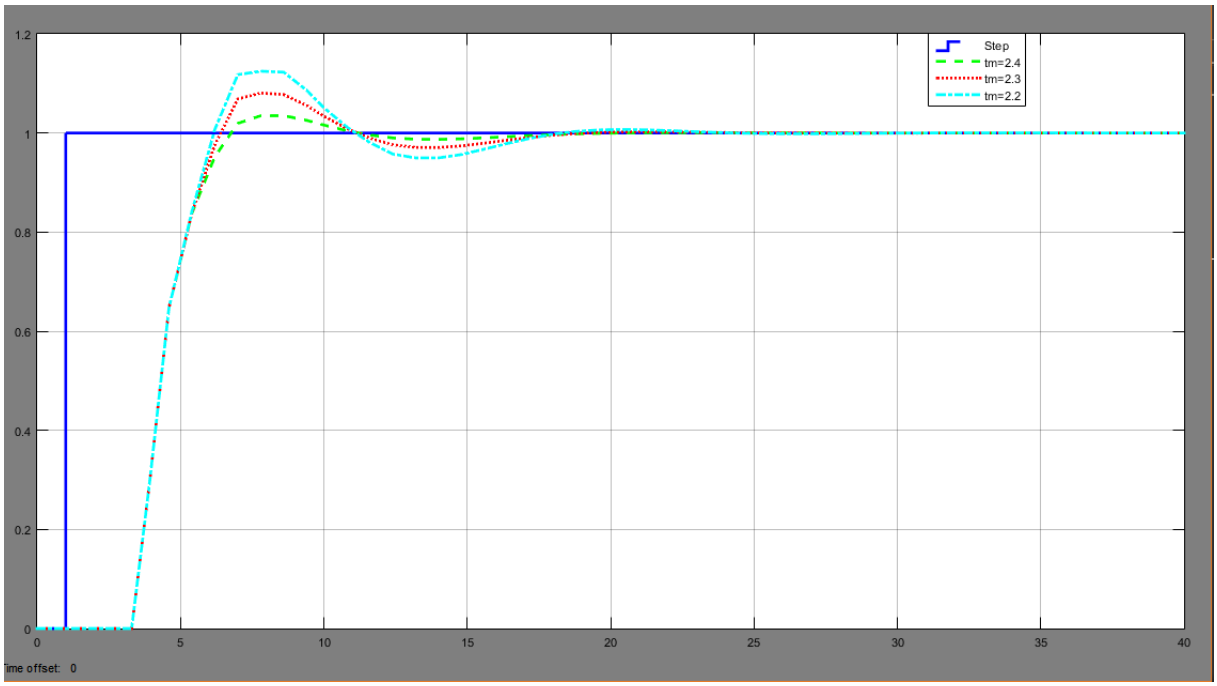

Figure 6.19: Step Responses with Different Overestimated t_m without Disturbance

For overestimated t_m, it is set to $2,6s$, $2,7s$ and $2,8s$. For underestimated t_m, it is set to $2,4s$, $2,3s$ and $2,2s$. In the case of overestimated time delay, with the difference of modelled delay and actual delay increases, the setpoint step response is more distorted. In the case of underestimation, setpoint response will have overshoot. With the difference increases, the overshoot is also increasing. Figure 6.20 shows the effects of larger modelling error of t_m. For underestimated parameter, $t_m = 1,5s$ and for overestimated parameter, $t_m = 3,5s$. The plot indicates that with larger error in modelling the delay time, the effects of overestimation and underestimation is magnified respectively. Thus when selecting the deadtime model,

it should be as close as to the actual deadtime as possible.

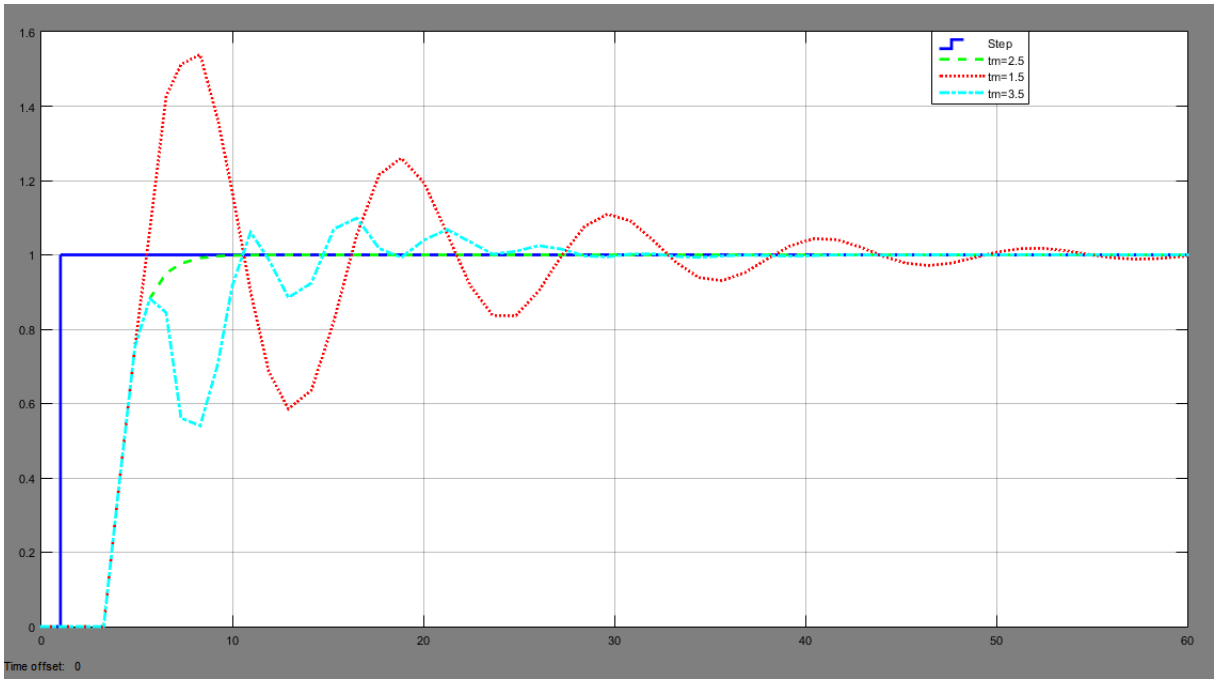

Figure 6.20: Step Responses with Underestimated and Overestimated t_m without Disturbance

Chapter 7

Discussion

The proposed descriptor and correcting algorithm is proven working effectively in different lighting conditions. Object is detected from different orientation and distance. Both monochromatic color pattern and randomly selected RGB color pattern which represents untextured and textured surface are tested. Although in this paper the untextured surface is not applicable for block matching, it still demonstrates that it will work on untextured surface. From the third object in experiment, it demonstrates that the descriptor can also work on a non-flat surface. The working orientation range is also tested. Result shows that it can work with a quite severe perspective projection. The proposed online correction algorithm implies that the color distribution is working more robustly than the color intensity. But using color intensity requires less time than the color distribution requires.

The stereo vision cameras can reach an adequate accuracy to perform a grasping task. However, a working range in the cartesian workspace should be considered. With different depth range the accuracy is different. Baseline selection should also consider the depth range as indicated in the discussion part of Chapter 3. The images acquired from should be synchronized. Since the block matching is applied to solve corresponding problem, if the the images differs too much in time when it is captured, it might create additional problem. The approach in this thesis to synchronize the image is combining the left and right image first in image acquisition node and then publishes it as one image. The other nodes which subscribe this topic the separate it into left and right image again. This method is better than process the left and right image separately and then publish them into different topics. In order to successfully perform block matching, a few factors should be considered carefully. Window size of block matching is definitely important. It depends on how many details you want to preserve and the noise level you could accept. A lot of researches is dedicated to solve this issue. One idea is adaptive window instead of fix size window. Several different sized windows are applied on same images and disparity map is obtained by processing the results from different sized windows.

The control law of the PBVS is designed in a way that a cascade smith predictor is used to control every element of pose. The reference input pose is calculated by the addition of current pose in TCP coordinate frame and the transformation which brings the current object pose to the desired pose. Generally smith predictor is considered as compensator for deadtime, but it will not work for unstable plant. Thus a cascade smith predictor control scheme is selected to overcome this problem. There are a lot of different modifications based on smith predictor have been conducted by different researchers in the world to solve the problem. The cascade smith predictor controller structure has the advantage of decoupled setpoint response and load disturbance response as well as the easy parameter tuning. The original controller C_2 presented by Hongdong et al. is a PD controller. The author mentioned that P controller could also be the controller form for C_2, but it is not presented in paper. In the thesis, it is designed as a P controller. All controller parameters are calculated again and system stability and performance are analysed to ensure that changed the controller C_2 could be used. The Matlab simulation results show that the setpoint could be successfully tracked, the time delay is compensated and the disturbance is rejected effectively whether

it is coming after system is in a steady state or it is in the rising phase. Tuning the setpoint response is simplified to tuning λ which is the time constant of desired system dynamic. To change the dynamic behaviour of the disturbance rejection, K_{c2} could be designed and changed separately. Modelling the deadtime should be taken care of. Either underestimated deadtime or overestimated deadtime will affect the desired response. With higher λ value, less deadtime modelling error will be tolerated. Thus the more accurate of the deadtime model, the better performance the system can reach. However in the PBVS case, the actual deadtime t_d itself is not constant, it brings more uncertainty for modelling the deadtime. In this case, λ should be smaller so that robustness is assured by sacrificing some response performance as the stability of the visual servoing is more important.

One motivation for selecting PBVS as the visual servoing scheme is, the possibility that it could be applied with different feedback sensors. Figure 7.1 illustrates the idea. In PBVS, camera acts as the feedback sensor, then pose is derived as the value to control. Instead of camera, other sensor technology such as IMU, ultrasound and so on could be also used for estimate the pose. The sensor could be used alone or together with a sensory data fusion stage to obtain pose information. In addition, different vision sensors can also be used together to get different sensory data. For instance, Kinect and stereo rig can be used together and images from them are fed to sensor fusion step. Another example could be place cameras with different configuration (eye-in-hand, eye-to-hand) for sensory data fusion. With ant case, the controller designed in PBVS still could be used since the controller input is pose information. Compare to IBVS, the control law part is less coupled with sensory data processing. It means more flexibility on the selection of sensor technology.

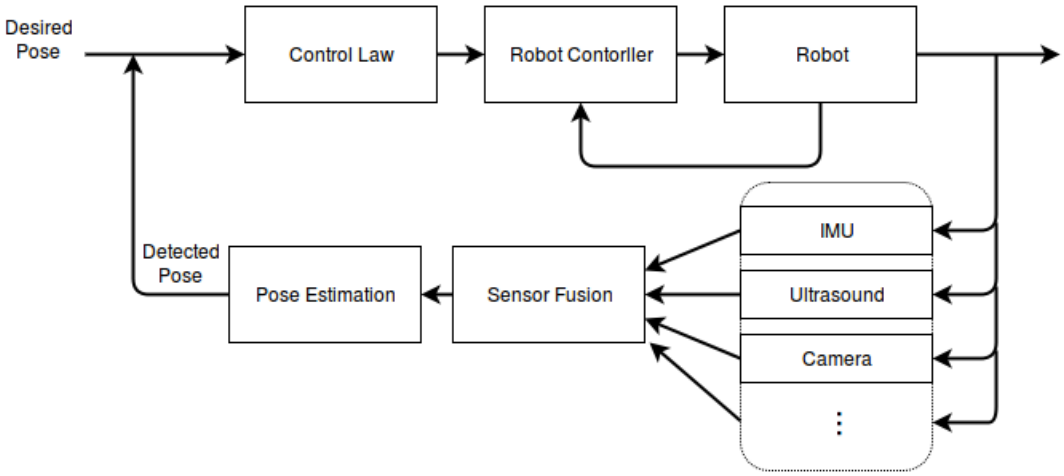

Figure 7.1: Other Feedback Fusion with PBVS Structure

As mentioned in the motivation, this thesis work is trying to bring the visual servoing towards more practical. The factor accuracy is contributed by constructing stereo vision to acquire depth map instead of using Kinect. Robustness is related to reliability. More robust means that the system can work with more different conditions which also implies more reliability. It is difficult to be defined precisely. But through out the 3 main tasks, at least some work contributes the increase the robustness. The descriptor is invariant to scale and orientation and correction algorithm make it also invariant to lighting condition. The disparity map is generated by combining 2 disparity maps with 2 different disparity ranges. This method can decrease the possibility of occurrence of occlusion. The selected cascade controller has good ability to reject the disturbance whether the disturbance comes at steady state or rising phase. The proposed descriptor requires little a-priori knowledge and teaching stage is simple, which brings more convenience. The controller structure brings an easy parameter tuning, only 2 parameters needs to be considered which simplifies the controller tuning to get desired performance. In addition this meaning of parameter is easy to understand, even for the people who is not expert in control engineering and not familiar with complicated theory.

Chapter 8

Summary

8.1 Summary (English)

In this thesis, a position based visual servoing is designed for a 6 DoF robot manipulator. The name of visual servoing means that the robot is controlled with vision feedback, normally cameras, to form a closed loop control for robot. A color-based descriptor and online correction algorithm is developed for object recognition and it provides invariance to scale, orientation as well as lighting condition. A stereo vision camera is constructed by 2 cameras and it is used to obtain depth information, which is further used to obtain the object pose. Finally a controller structure is selected to overcome the deadtime caused by image processing and vision algorithm as well as reject the possible disturbance.
Several improvements could be further developed in order to make the system more intelligent in the future. For the correction algorithm, now the threshold value is manually selected by observing the color intensity and color distribution value. More intelligent method can be applied to accomplish selecting parameter automatically. Another improvement could be dynamically change the controller parameter. Since the deadtime of system is variant, if the parameter is assigned dynamically, performance could be improved too. The criterion of parameter could be model based method or machine learning. If the machine learning method is applied, the training data can be collected during the running. Thus it will not against the idea of convenience by avoiding manually collecting a large quantity of data.

8.2 Summary (Estonian)

See lõputöö käsitleb masinnägemisel põhineva juhtimissüsteemi disainimist kuue vabadusastmega robot manipulaatorile. Masinnägemisel põhinev juhtimissüsteem kujutab endast tagasisidestatud juhtimissüsteemi, mille tagasiside põhineb visuaalsel informatsioonil, mis saadakse kaamerast. Objekti tuvastuseks arendati välja värvidel põhinev tuvastussüsteem ja korrigeeriv algoritm. Seetõttu ei sõltu objektide tuvastamine nende kaugusest ega asendist kaamera suhtes ega ka valgustingimustest. Töös on kasutatud kahest kaamerast koosnevat stereosüsteemi. Saadud kasutatakse, et saada infot objekti kauguse kohta. Seda infot kasutatakse objekti paiknemise kindlaks-tegemisel. Töös on pakutud välja ka täiendava kontrolleri lahendus, sest pilditöötluse ja masinnägemis algoritmi tõttu võivad tekkida viited süsteemi reageerimisajas. Kontroller aitab likvideerida ka võimalike signaali häirete mõju.
Töös kirjeldatud lahendusele on võimalik teha mitmeid edasiarendusi. Korrigeeriva algoritmi piirväärtused määratakse manuaalselt, hinnates värvide jaotust ja intensiivsust. Antud piirväärtuste määramiseks on võimalik luua automaatne lahendus, mis määrab piirväärtused eelpool mainitud parameetrite alusel iseseisvalt. Täiendavaks edasiarenduseks oleks ka kontrolleri parameetrite dünaamiline muutmine. Kuna viited süsteemi reageerimisajas ei ole konstantsed, siis kontrolleri parameetrite dünaamiline muutmine parandaks kogu süsteemi sooritust. Kontrolleri parameetrite dünaamiliseks muutmiseks oleks vaja luua vastav mudel või kasutada masinõpet. Masinõppe lahenduse puhul oleks süsteemi õpetamiseks vajalikud andmed võimlaik koguda süsteemi töö ajal. Seeläbi ei ole tarvis suure hulga andmete kogumiseks teha täiendavaid pingutusi.

Bibliography

C. Ancuti and P. Bekaert. Sift-cch: Increasing the sift distinctness by color co-occurrence histograms. In *Image and Signal Processing and Analysis, 2007. ISPA 2007. 5th International Symposium on*, pages 130–135. IEEE, 2007.

T. Andersen. *Optimizing the Universal Robots ROS driver.* Technical University of Denmark, Department of Electrical Engineering, 2015.

K. S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3-d point sets. *IEEE Transactions on pattern analysis and machine intelligence*, (5):698–700, 1987.

K. J. Astrom, C. C. Hang, and B. Lim. A new smith predictor for controlling a process with an integrator and long dead-time. *IEEE transactions on Automatic Control*, 39(2):343–345, 1994.

H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.

H. M. Becerra and C. Sagues. A sliding mode control law for epipolar visual servoing of differential-drive robots. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3058–3063. IEEE, 2008.

P. J. Besl and N. D. McKay. Method for registration of 3-d shapes. In *Robotics-DL tentative*, pages 586–606. International Society for Optics and Photonics, 1992.

P. Chang and J. Krumm. Object recognition with color cooccurrence histograms. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2. IEEE, 1999.

F. Chaumette and S. Hutchinson. Visual servo control. i. basic approaches. *IEEE Robotics & Automation Magazine*, 13(4):82–90, 2006.

P. I. Corke and S. A. Hutchinson. A new partitioned approach to image-based visual servo control. *IEEE Transactions on Robotics and Automation*, 17(4):507–515, 2001.

P. I. Corke et al. *Visual Control of Robots: high-performance visual servoing.* Research Studies Press Baldock, 1996.

M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

R. T. Fomena, C. P. Quintero, M. Gridseth, and M. Jagersand. Towards practical visual servoing in robotics. In *Computer and Robot Vision (CRV), 2013 International Conference on*, pages 303–310. IEEE, 2013.

M. J. Gangeh, B. M. ter Haar Romeny, and C. Eswaran. Scale-space texture classification using combined classifiers. In *Scandinavian Conference on Image Analysis*, pages 324–333. Springer, 2007.

Gazebo. Gazebo. http://gazebosim.org/. Accessed: 10.01.2017.

R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

K. Hashimoto, T. Ebine, and H. Kimura. Visual servoing with hand-eye manipulator-optimal control approach. *IEEE Transactions on Robotics and Automation*, 12(5):766–774, 1996.

K. Hirata and T. Mizuno. Robust visual feedback control of inverted pendulum system against camera misalignment. In *2008 10th IEEE International Workshop on Advanced Motion Control*, pages 153–158. IEEE, 2008.

Z. Hongdong, L. Ruixia, and S. Huihe. Control for integrating processes based on new modified smith predictor.

S. Hutchinson, G. D. Hager, and P. I. Corke. A tutorial on visual servo control. *IEEE transactions on robotics and automation*, 12(5):651–670, 1996.

W. Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 32(5):922–923, 1976.

W. Kabsch. A discussion of the solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 34(5):827–828, 1978.

D. Kragic and H. I. Christensen. Vision techniques for robotic manipulation and grasping. In *Proceedings of the 33rd ISR (International Symposium on Robotics) October*, volume 7, page 11, 2002.

D. Kragic, H. I. Christensen, et al. Survey on visual servoing for manipulation. *Computational Vision and Active Perception Laboratory, Fiskartorpsv*, 15, 2002.

leopard image. leopard image. https://www.leopardimaging.com/LI-USB30-M021.html. Accessed: 10.01.2017.

S. Leutenegger, M. Chli, and R. Y. Siegwart. Brisk: Binary robust invariant scalable keypoints. In *2011 International conference on computer vision*, pages 2548–2555. IEEE, 2011.

V. Lippiello, B. Siciliano, and L. Villani. Position-based visual servoing in industrial multirobot cells using a hybrid camera configuration. *IEEE Transactions on Robotics*, 23(1):73–86, 2007.

A. Lorusso, D. W. Eggert, and R. B. Fisher. *A comparison of four algorithms for estimating 3-D rigid transformations*. University of Edinburgh, Department of Artificial Intelligence, 1995.

D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

E. Malis, F. Chaumette, and S. Boudet. 21/2d visual servoing. *IEEE Transactions on Robotics and Automation*, 15(2):238–250, 1999.

M. Matausek and A. Micic. A modified smith predictor for controlling a process with an integrator and long dead-time. *IEEE Transactions on Automatic Control*, 41(8):1199–1203, 1996.

M. R. Matausek and A. Micic. On the modified smith predictor for controlling a process with an integrator and long dead-time. *IEEE Transactions on Automatic Control*, 44(8):1603–1606, 1999.

Matlab. Camera distortion. https://de.mathworks.com/help/vision/ug/camera-calibration.html. Accessed: 29.12.2016.

Microsoft. Kinect sensor. https://msdn.microsoft.com/en-us/library/hh438998.aspx. Accessed: 10.01.2017.

MoveIt. Moveit system architecture. http://moveit.ros.org/documentation/concepts/. Accessed: 29.12.2016.

NI. Ni. http://zone.ni.com/reference/en-XX/help/372916P-01/nivisionconceptsdita/guid-10d358bd-3dcd-4ccd-a73c-672e48aed39a/. Accessed: 10.01.2017.

OpenCV. Opencv. http://docs.opencv.org/2.4/. Accessed: 10.01.2017.

PCL. Pcl. http://pointclouds.org/. Accessed: 10.01.2017.

ROS. Ros. http://www.ros.org/. Accessed: 10.01.2017.

ROS-Industrial. Ros-industrial overview. http://http://wiki.ros.org/Industrial. Accessed: 29.12.2016.

B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo. *Robotics: modelling, planning and control*. Springer Science & Business Media, 2010.

O. J. Smith. A controller to overcome dead time. *iSA journal*, 6(2):28–33, 1959.

STEREOLABS. Zed stereo camera. https://www.stereolabs.com/. Accessed: 10.01.2017.

B. Thuilot, P. Martinet, L. Cordesses, and J. Gallice. Position based visual servoing: keeping the object in the field of vision. In *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, volume 2, pages 1624–1629. IEEE, 2002.

Universal-Robots. Universal robots. https://www.universal-robots.com/download/. Accessed: 10.01.2017.

K. Van De Sande, T. Gevers, and C. Snoek. Evaluating color descriptors for object and scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1582–1596, 2010.

H. Wang, A. Chamroo, C. Vasseur, and V. Koncar. Stabilization of a 2-dof inverted pendulum by a low cost visual feedback. In *2008 American Control Conference*, pages 3851–3856. IEEE, 2008.

O. Wasenmüller and D. Stricker. Comparison of kinect v1 and v2 depth images in terms of accuracy and precision. In *Asian Conference on Computer Vision Workshop (ACCV workshop), Springer*, 2016.

K. Watanabe and M. Ito. A process-model control for linear systems with delay. *IEEE Transactions on Automatic control*, 26(6):1261–1269, 1981.

List of Figures

List of Tables

Appendix A

Appendix A

A.1 Direct Substitution

Direct Substitution is used to find out the limit of stable poles on the complex plane. Calculate the ultimate gain of C_2 is given as follow. Characteristic equation is $W(s)$ (A.1).

$$W(s) = 1 + A(s) = 1 + \frac{K_{c2}}{s} e^{-t_d s} \ . \tag{A.1}$$

$$\begin{aligned} W(j\omega) &= 1 + \frac{K_{c2}}{j\omega} e^{-t_d j\omega} \\ &= 1 + \frac{K_{c2}}{j\omega}(cos(t_d\omega) - jsin(t_d\omega)) \\ &= 1 + \frac{K_{c2}\omega}{-\omega^2} j(cos(t_d\omega) - jsin(t_d\omega)) \\ &= 1 - \frac{cos(t_d\omega)K_{c2}}{\omega} j - \frac{K_{c2}sin(t_d\omega)}{\omega} \\ &= \frac{\omega - K_c sin(t_d\omega)}{\omega} - \frac{cos(t_d\omega)K_{c2}}{\omega} j \ . \end{aligned} \tag{A.2}$$

The limit of stable poles yields

$$\mathbf{Re}\{W(j\omega)\} \stackrel{!}{=} 0$$

and

$$\mathbf{Im}\{W(j\omega)\} \stackrel{!}{=} 0,$$

Which leads to the following calculation.

$$\begin{aligned} cos(t_d\omega) &= 0 \ , \\ \omega = \frac{\frac{\pi}{2} + 2n\pi}{2t_d} &= \frac{(4n+1)\pi}{2t_d} \ , \quad n \in N \ . \end{aligned} \tag{A.3}$$

$$\begin{aligned} \omega - K_{c2}sin(t_d\omega) &= 0 \ , \\ \omega - K_{c2}sin(t_d\frac{(4n+1)\pi}{2t_d}) &= 0 \ , \\ \omega - K_{c2} &= 0 \ , \\ K_{uc2} = K_{c2} = \omega &= \frac{(4n+1)\pi}{2t_d} \ . \end{aligned} \tag{A.4}$$

Setting $n = 0$, then

$$K_{uc2} = \frac{\pi}{2t_d} \ .$$

A.2 Controller Parameter Tuning

C and C_1 are tuned to FOPDT form

$$\frac{1}{\lambda s + 1} e^{-\tau s} \ .$$

setpoint response G_{yr} is

$$\begin{aligned}
G_{yr} &= \frac{CC_1G_{0m}}{1 + G_{0m}C_1(1+C)} e^{-t_m s} \ , \\
&= \frac{C}{\frac{1}{C_1G_{0m}} + 1 + C} e^{-t_m s} \ , \\
&= \frac{K_c(1 + \frac{1}{T_c s})}{\frac{s}{K_{c1}} + 1 + K_c + \frac{K_c}{T_c s}} e^{-t_m s} \ , \\
&= \frac{1}{\frac{s}{K_{c1}K_c(1+\frac{1}{T_c s})} + \frac{1}{K_c(1+\frac{1}{T_c s})} + 1} e^{-t_m s} \ , \\
&= \frac{1}{\frac{s+K_{c1}}{K_cK_{c1}(1+\frac{1}{T_c s})} + 1} e^{-t_m s} \ , \\
&\ .
\end{aligned} \tag{A.5}$$

Apply the tuning rule yields

$$\frac{s + K_{c1}}{K_cK_{c1}(1 + \frac{1}{T_c s})} \stackrel{!}{=} \lambda s \ .$$

$$\begin{aligned}
\frac{s + K_{c1}}{K_cK_{c1}(1 + \frac{1}{T_c s})} &= \lambda s \ , \\
s + K_{c1} &= \lambda K_cK_{c1}s + \frac{\lambda K_cK_{c1}}{T_c} \ , \\
s &= \lambda K_cK_{c1}s + \frac{\lambda K_cK_{c1}}{T_c} - K_{c1} \ , \\
s &= \lambda K_cK_{c1}s + \frac{\lambda K_cK_{c1} - K_{c1}T_c}{T_c} \ .
\end{aligned} \tag{A.6}$$

This leads to

$$\lambda K_cK_{c1} \stackrel{!}{=} 1$$

and

$$\lambda K_cK_{c1} - K_{c1}T_c \stackrel{!}{=} 0 \ .$$

$$\begin{aligned}
\lambda K_cK_{c1} - K_{c1}T_c &= 0 \\
\lambda K_c - T_c &= 0 \\
K_c &= \frac{T_c}{\lambda} \ .
\end{aligned} \tag{A.7}$$

$$\begin{aligned}
\lambda K_cK_{c1} &= 1 \\
K_{c1} &= \frac{1}{\lambda K_c} \ .
\end{aligned} \tag{A.8}$$

A.3 Transfer Function Calculation

Setpoint response G_{yr} and load disturbance response G_{yd} with controller C, C_1 and C_2 are calculated as follows.
For the controllers in the form of

$$\begin{aligned} C &= K_c(1+\frac{1}{T_c s}), \\ C_1 &= K_{c1}, \\ C_2 &= K_{c2}\ . \end{aligned} \tag{A.9}$$

$$\begin{aligned} G_{yr} &= \frac{CC_1G_{0m}}{1+G_{0m}C_1(1+C)} = \frac{K_c(1+\frac{1}{T_c s})K_{c1}\frac{1}{s}}{1+\frac{1}{s}K_{c1}(1+K_c(1+\frac{1}{T_c s}))} \\ &= \frac{\frac{K_{c1}}{s}\frac{K_cT_cs+K_c}{T_cs}}{1+\frac{K_{c1}}{s}(1+\frac{K_cT_cs+K_c}{T_cs})} = \frac{\frac{K_cT_cs+K_c}{T_cs}}{\frac{K_{c1}}{s}+1+\frac{K_c}{T_cs}+K_c} = \frac{K_cT_cs+K_c}{\frac{T_c}{K_{c1}}s^2+T_cs+K_cT_cs+K_c}\ . \end{aligned} \tag{A.10}$$

$$\begin{aligned} G_{yd} &= \frac{G_{0m}e^{-t_ds}}{1+G_{0m}C_1(1+C)}\frac{1+C_1G_{0m}(1+C-Ce^{-t_ms})}{1+C_2G_{0m}e^{-t_ds}} \\ &= \frac{\frac{1}{s}e^{-t_ds}}{1+\frac{1}{s}K_{c1}(1+K_c(1+\frac{1}{T_cs}))}\frac{1+K_{c1}\frac{1}{s}(1+K_c(1+\frac{1}{T_cs})-K_c(1+\frac{1}{T_cs})e^{-t_ms})}{1+K_{c2}\frac{1}{S}e^{-t_Ds}} \\ &= \frac{e^{-t_ds}}{s+K_{c1}+K_cK_{c1}(1+\frac{1}{T_cs})}\frac{s+K_{c1}(1+K_c(1+\frac{1}{T_cs})-K_c(1+\frac{1}{T_cs})e^{-t_ms})}{s+K_{c2}e^{-t_ds}} \\ &= \frac{T_cse^{-t_ds}}{T_cs^2+T_cK_{c1}s+T_cK_cK_{c1}s+K_cK_{c1}}\frac{T_cs^2+T_cK_{c1}s(1+K_c(1+\frac{1}{T_cs})-K_c(1+\frac{1}{T_cs})e^{-t_ms})}{T_cs^2+T_cK_{c2}e^{-t_ds}} \\ &= \frac{T_cse^{-t_ds}}{T_cs^2+T_cK_{c1}(1+K_c)s+K_cK_{c1}}\cdot \\ &\quad \frac{T_cs^2+T_cK_{c1}s+T_cK_cK_{c1}s-T_cK_cK_{c1}se^{-t_ms}+K_cK_{c1}-K_cK_{c1}e^{-t_ms}}{T_cs^2+T_cK_{c2}e^{-t_ds}}\ . \end{aligned} \tag{A.11}$$

Appendix B

Appendix B

In Figure B.1, it has cascade smith predictor with P controller for comparison. In Figure B.2, 3 cascade smith predictors with different λ are compared. Figure B.3 shows the details within the subsystem of Figure B.2.

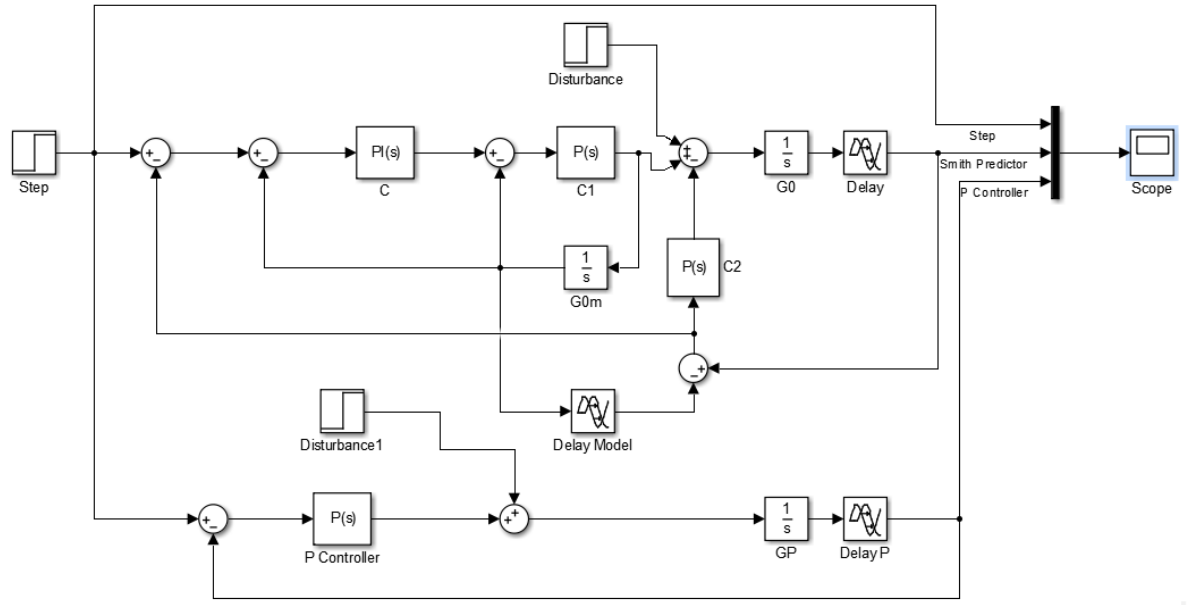

Figure B.1: Block Diagram of Cascade Smith Predictor and P Controller in Matlab

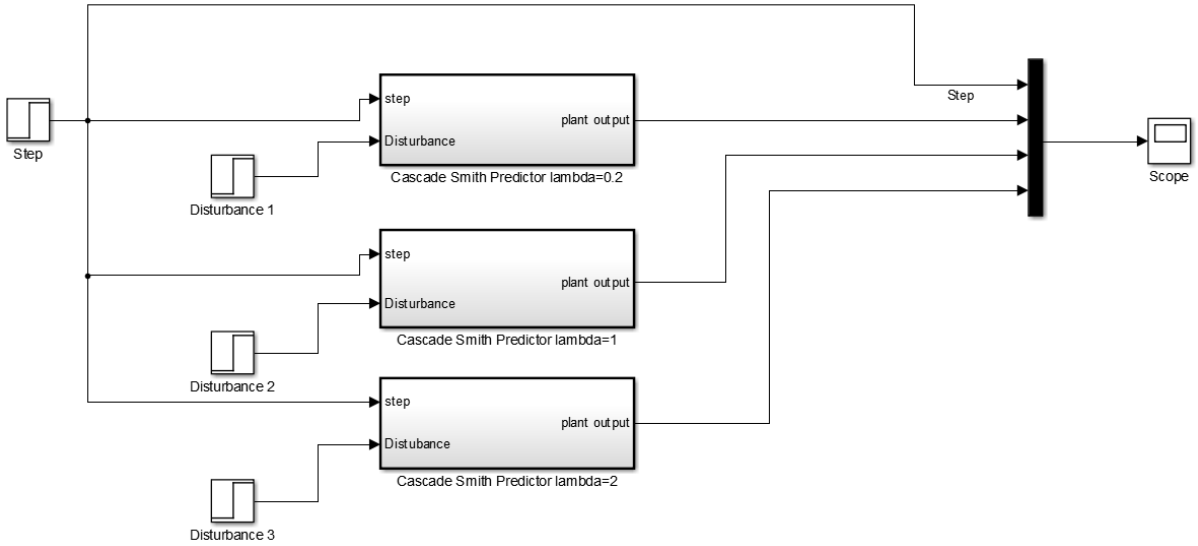

Figure B.2: An Overview of Block Diagram of Cascade Smith Predictor with Different λ in Matlab

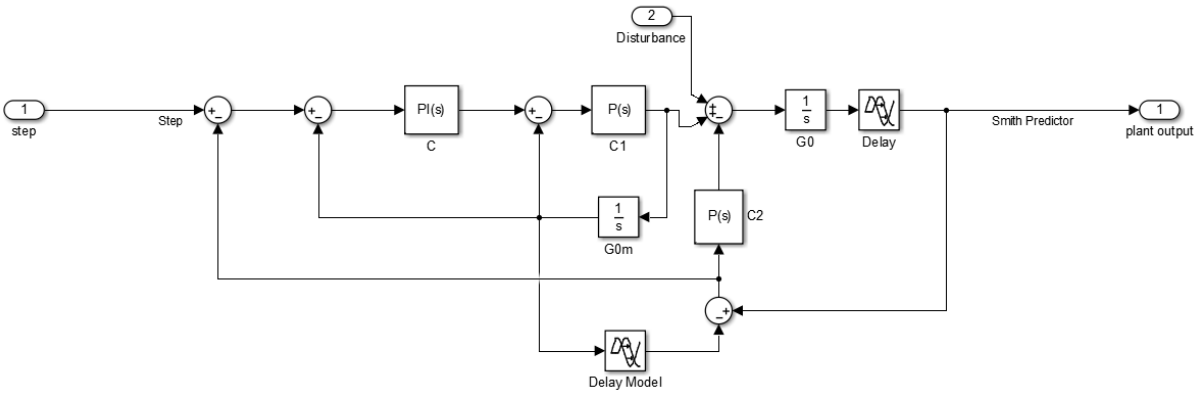

Figure B.3: Block Diagram of Cascade Smith Predictor Subsystem in Matlab.