Image Based Visual Servoing Using Trajectory Planning and Augmented Visual Servoing Controller

Mohammad Keshmiri

A Thesis

in

The Department

of

Mechanical and Industrial Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of Doctor of Philosophy at

Concordia University

Montréal, Québec, Canada

October 2014

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis proposal prepared

By: **Mohammad Keshmiri**

Entitled: **Image Based Visual Servoing Using Trajectory Planning and Augmented Visual Servoing Controller**

and submitted in partial fulfilment of the requirements for the degree of

Doctor of Philosophy

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

_____ Dr. Hashem Akbari, Chair

_____ Dr. Fengfeng Xi , External Examiner

_____ Dr. Ben Hamza, External to Program

_____ Dr. Youmin Zhang, Examiner

_____ Dr. Henry Hong, Examiner

_____ Dr. Wen-Fang Xie, Thesis Supervisor

Approved by _____
Chair of the Department of Mechanical and Industrial Engineering

_____

Dean of the Faculty of Engineering and Computer Science

ABSTRACT

Image Based Visual Servoing Using Trajectory Planning and Augmented Visual Servoing Controller

Mohammad Keshmiri,

Concordia University, 2014

Robots and automation manufacturing machineries have become an inseparable part of industry, nowadays. However, robotic systems are generally limited to operate in highly structured environments. Although, sensors such as laser tracker, indoor GPS, 3D metrology and tracking systems are used for positioning and tracking in manufacturing and assembly tasks, these devices are highly limited to the working environment and the speed of operation and they are generally very expensive. Thus, integration of vision sensors with robotic systems and generally visual servoing system allows the robots to work in unstructured spaces, by producing non-contact measurements of the working area. However, projecting a 3D space into a 2D space, which happens in the camera, causes the loss of one dimension data. This initiates the challenges in vision based control. Moreover, the nonlinearities and complex structure of a manipulator robot make the problem more challenging. This project aims to develop new reliable visual servoing methods that allow its use in real robotic tasks.

The main contributions of this project are in two parts; the visual servoing controller and trajectory planning algorithm. In the first part of the project, a new image based visual servoing controller called Augmented Image Based Visual Servoing (AIBVS) is presented. A proportional derivative (PD) controller is developed to generate acceleration as the controlling command of the robot. The stability analysis of the controller is conducted using Lyapanov theory. The developed controller has been tested on a 6 DOF Denso robot. The experimental results on point features and image moment features demonstrate the performance of the proposed

AIBVS. Experimental results show that a damped response could be achieved using a PD controller with acceleration output. Moreover, smoother feature and robot trajectories are observed compared to those in conventional IBVS controllers. Later on, this controller is used on a moving object catching process.

Visual servoing controllers have shown difficulty in stabilizing the system in global space. Hence, in the second part of the project, a trajectory planning algorithm is developed. The trajectory planning algorithm can take the robot to its desired position froma any initial position using visual data. The trajectory planning is carried out by parameterizing the camera's velocity screw. The camera's velocity screw is parameterized using time-based profiles. The parameters of the velocity profile are then determined such that the velocity profile guides the robot to its desired position. This is done by minimizing the error between the initial and desired features. This method provides a reliable path for the robot considering all robotic constraints. The developed algorithm is tested on a Denso robot. The results show that the trajectory planning algorithm is able to perform visual servoing tasks which are unstable when performed using visual servoing controllers.

To my beloved wife,

Mohaddeseh Assari

and my lovely parents,

Mehdi Keshmiri

and

Zahra Mirza Jaffar

ACKNOWLEDGEMENTS

I would like to take the opportunity to greatly thank my supervisor, Dr. Wen-Fang Xie for her nonstop supports, suggestions, and direction. I will always count on her to provide direction and support whenever I feel lost or stressed. I would like to thank her for all her contributions, guidance and encouragements.

Hearty thanks also go to my colleagues and friends who accompanied me along these years: Mr. Abolfazl Mohebbi, Mr. Ali Fellah Jahromi, Mr. Amir Hajiloo, Mr. Yimin Zhao. I am also grateful to all those people who have given me support and I didn't have the chance to thank them.

I ultimately would like to thank my lovely parents Zahra Mirza Jafar and Dr. Mehdi Keshmiri for their support, kindness and being always there, regardless of my choices, and for having allowed me, with their efforts, to get to where I am now. I also would like to thank my lovely parents in law, Fatemeh Eshraghi and Dr. Abass Assari for all their kindness and support.

My deepest heartfelt gratitude goes out to my wife, Mohaddeseh. She supported and encouraged me from the very beginning step of my PhD study to the end. I am really grateful for having her in my life.

I would like to acknowledge the financial support from the Natural Sciences and Engineering Research Council of Canada (NSERC) throughout the course of this study and also Quanser Company for providing us with the experimental setup which brought a great value to this research.

TABLE OF CONTENTS

List of Figures

List of Tables

Nomenclature

$[\ddot{X}, \ddot{Y}, \ddot{Z}]$ 3D point acceleration in space with respect to camera frame

$[{}^cX, {}^cY, {}^cZ]$ Coordinates of a 3D point represented in camera frame

$[R_c, \theta_c, \alpha_c]$ Camera coordinates in Polar system with respect to robot base

$[u, v]$ Feature point coordinates in pixel

$[u_0, v_0]$ Pixel coordinates of the image plane principal points

$[X_c, Y_c, Z_c]$ Camera coordinates in Cartesian space with respect to robot base

α_u Scaling factor along x coordinate of the image plane

α_v Scaling factor along y coordinate of the image plane

$\boldsymbol{\alpha}_c = [\alpha_{cx}, \alpha_{cy}, \alpha_{cz}]^{\mathrm{T}}$ Camera angular acceleration with respect to base frame represented in camera frame

$\boldsymbol{\omega}_c = [\omega_{cx}, \omega_{cy}, \omega_{cz}]^{\mathrm{T}}$ Camera angular velocity with respect to base frame represented in camera frame

$\boldsymbol{\xi}$ Point feature vector

$\boldsymbol{\xi}_d$ Desired point feature vector

$\boldsymbol{\tau}$ Robot joint torques vector

$\boldsymbol{\theta}_I$ Interceptor's velocity angle

$\boldsymbol{\theta}_T$ Target's velocity angle

$\boldsymbol{\theta}_{LOS}$ Angle of the line of sight

$\ddot{\mathbf{p}} = [\ddot{x}, \ddot{y}]$ Features acceleration in image plane

$\dot{\mathbf{p}} = [\dot{x}, \dot{y}]$ Features velocity in image plane

κ_p Visual servoing controller's proportional gain

κ_v Visual servoing controller's derivative gain

κ_{m_p} Visual servoing controller's proportional gain for image moments

κ_{m_v} Visual servoing controller's derivative gain for image moments

$\boldsymbol{\epsilon}$ Point feature error vector

$\boldsymbol{\epsilon}_n$ Error vector of the nominal system

$\epsilon_{\mathbf{m}}$ Moment feature error vector

$\xi_{\mathbf{n}}$ New features defined for the trajectory planning algorithm

ξ_m Moment feature vector

ξ_{m_d} Desired moment feature vector

$\mathbf{A}$ Camera acceleration screw with respect to base frame represented in camera frame

$\mathbf{A}_c$ Generated acceleration command form the visual servoing block

$\mathbf{a}_c = [a_{cx}, a_{cy}, a_{cz}]^{\mathrm{T}}$ Camera linear acceleration with respect to base frame represented in camera frame

$\mathbf{A}_I$ Interceptor's acceleration vector

$\mathbf{B}$ Camera intrinsic matrix

$\mathbf{e}_{q_i}$ Robot joint ith error vector

$\mathbf{G}(\mathbf{q})$ Robot gravitational matrix

$\mathbf{I}(\mathbf{P_c})$ Robot inverse kinematics equations

$\mathbf{J}$ Robot Jacobian matrix

$\mathbf{K}_d$ Computed torque control derivative gain

$\mathbf{K}_p$ Computed torque control proportional gain

$\mathbf{L}_a$ Visual servoing interaction matrix for point features

$\mathbf{L}_v$ Visual servoing velocity interaction matrix for point features

$\mathbf{L}_{a4}$ Interaction matrix for 4 feature points

$\mathbf{L}_{ma_6}$ Image moment interaction matrix for six moment features

$\mathbf{L}_{ma_{ij}}$ Interaction matrix for the m_{ij} moment

$\mathbf{L}_{mv_6}$ Image moment velocity interaction for six moment features

$\mathbf{L}_s$ Interaction matrix for a 4 DOFs robot

$\mathbf{M}(\mathbf{q})$ Robot inertial matrix

$\mathbf{p} = [^c x, {}^c y]$ Coordinates of projection of a 3D point on the camera frame, represented in camera frame

$\mathbf{q}$ Robot joint angle vector

$\mathbf{q}_d$ Robot desired joint angle vector

$\mathbf{q}_{max}$ Robot maximum joint limit

$\mathbf{q}_{min}$ Robot minimum joint limit

$\mathbf{V}$ Camera velocity screw with respect to base frame represented in camera frame

$\mathbf{V}_c$ Generated velocity command form the visual servoing block

$\mathbf{v}_c = [v_{cx},\, v_{cy},\, v_{cz}]^{\mathrm{T}}$ Camera linear velocity with respect to base frame represented in camera frame

$\mathbf{V}_I$ Interceptor's velocity vector

$\mathbf{V}_{c_4}$ Generated velocity command form the visual servoing block for the 4 DOFs robot

$\mathbf{V}_{tp}(t)$ Predefined velocity profiles

μ_{ij} Centered moment of order $i + j$

ν Lyapunov function candidate for the nominal system

σ_d Robot joint controller's derivative gain

σ_i Robot joint controller's integral gain

σ_p Robot joint controller's proportional gain

θ_z Camera rotation angle about z axis of the camera

A Constant used to define the depth equation

a Image area

A_{In} Interceptor's normal acceleration

A_{It} Interceptor's tangential acceleration

B Constant used to define the depth equation

b Distance between two camera in a stereo vision system

C Constant used to define the depth equation

$C(\dot{q}, q)$ Robot centripetal and coriolis matrix

CE Camera calibration error

$F_b(x_b,\ y_b,\ z_b)$ Robot's base frame

$F_c(x_c,\ y_c,\ z_c)$ Camera frame

$F_e(x_e,\ y_e,\ z_e)$ Robot's end-effector frame

$L_{mv_{ij}}$ Velocity interaction for the m_{ij} moment

m_{ij} Image moment of order $i + j$

n_{ij} The normalized moments

OF Optimization objective function

R Length of the line of sight

V_{m_i} Robot ith joint required voltage

x_g x component of the image center

x_l feature position on the left camera of a stereo vision system

x_r feature position on the right camera of a stereo vision system

y_g y component of the image center

Z_c Object depth with respect to camera

AIBVS Augmented image based visual servoing

AIPNG Augmented ideal proportional navigation guidance

APNG Augmented proportional navigation guidance

DOF Degree of freedom

IBVS Image based visual servoing

IPNG Ideal proportional navigation guidance

LOS Line of sight

P Proportional

PBVS Position based visual servoing

PD Proportional and derivative

PID Proportional, integral and derivative

PNG Proportional navigation guidance

UDP User Datagram Protocol

Chapter 1

Introduction

Robots and automation machineries are an inseparable part of the industry, nowadays. However, robotic systems are generally limited to operate in highly structured environments. Conventionally, robotic systems use an open loop algorithms to calculate the position of the end effector, with respect to a work piece. The work piece must be also placed in a known position with respect to the robot base coordinate frame. Any uncertainty in the robotic system, would cause the task to fail. Visual servoing solves this problem by providing non contact and real-time measurements of the environment and the work piece with respect to the robot and hence does not rely on open loop kinematic calculations.

"Visual Servoing" is an approach for controlling the motion of a robot using visual feedback signals [8, 9]. In other words, in visual servoing, the robot uses the vision captured by a camera to acquire the accurate position of the target objects and/or the end-effector and uses it as a feedback for controlling the system. Visual servoing is the fusion of many active research areas including high speed image processing, kinematics, dynamics, control theory and real-time computation [10].

Vision based control or visual servoing has been used as a solution in robotic industry to increase the dexterity and intelligence of robotic systems [11]. Visual

servoing becomes more interesting in situations where robot needs to work in an unstructured environment. In such situations, ordinary sensors cannot gather enough information for robots operation. On the other hand, vision systems provide a wide range of information of its environment such as structure, color, motion and more. Furthermore, the use of vision system allows to develop human intelligence in robotic system by imitating human vision system and the great talent of fast and accurate pattern recognition. Visual servoing is used in a wide variety of applications such as lane tracking for cars, navigation for mobile platforms, teleoperation, missile tracking, fruit picking and specially manipulation of objects for grasping, assembly or welding applications.

Implementing visual servoing in real robotic system is very challenging. Vision systems projects 3D space into 2D space, causing one dimension loss of data from the environment. The lost dimension is the depth of the objects. This property of vision system and also the nonlinearity involved in the projection, introduce the difficulties in integrating machine and vision. Visual servoing was presented as a solution to such problems [8, 11, 12].

1.1 Visual Servoing Applications

Visual servoing is mainly used in industrial robotics. In such applications the goal is to control the end-effector pose (position and orientation) with respect to the pose of objects or obstacles which could be fixed or moving in the workspace of the robot. Positioning or moving objects, assembling and disassembling mechanical parts, paintings, welding, etc. are some examples of the tasks which could be performed using robotic systems (Figure 1.1) [13–17].

Besides industrial applications, visual servoing tends to be used in medical and surgical applications to position instruments or perform the medical operations.

(a) (b)

Figure 1.1: Robots using visual feedbacks to perform various tasks [1]

Figure 1.2: Vision based robotic used in medical applications [1]

For instance in laparoscopic surgery, which needs only several small incisions in the abdominal wall to introduce instruments such as scalpels, scissors, etc., and a laparoscopic camera, such that the surgeon can operate by just looking at the camera images. To avoid the need for another assistant and to free the surgeon from the control task, an independent system that automatically guides the laparoscope is highly desirable. Several researchers have tried to use visual servoing techniques to guide the instrument during the operation (Figure 1.2) [18–20].

Control and guidance of unmanned vehicle systems such as unmanned ground vehicles (UGV) and unmanned aerial vehicles (UAV) is other examples of using

(a) (b)

Figure 1.3: Visual servoing systems used in UGVs and UAVs [1]

Figure 1.4: Google driver-less car [2]

visual servoing technique for the exploration or reconnaissance operations [21, **?**] (Figure 1.3). A good example of UGV application of visual servoing is the Google driver-less car [2] (Figure 1.4).

In another application of visual servoing systems, space robots are used to perform autonomous on-orbit servicing which includes approaching and docking to a target satellite and grasping some complex parts for the purpose of refueling and servicing [3] (Figure 1.5).

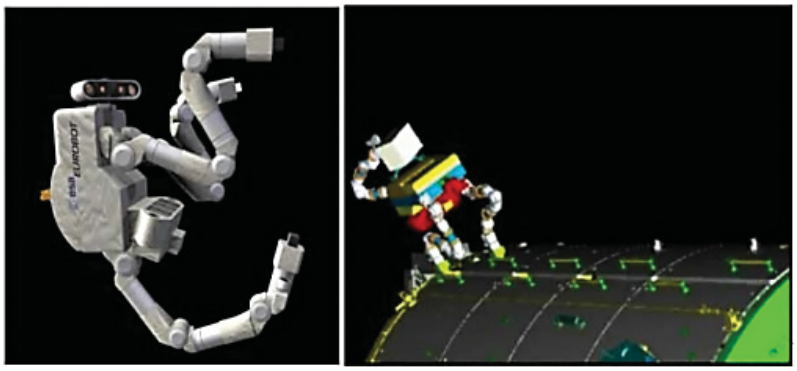


Figure 1.5: A space robotic system in a mission of on-orbit servicing [3]

1.2 Literature Review

Researchers have been working on the topic of visual servoing for more than 30 years [22]. However, recent developments in high speed processors and cameras has made it possible for real-time implementation and industrial applications. Visual servoing techniques have been investigated in literature since 1970 [8]. The phrase "Visual Servoing" was first used by Hill and Park in 1979 [23]. Further on, researchers have made considerable progress in this field. To better review the visual servoing in literature, different categories and classes of visual servoing are presented and discussed separately. Due to various types of implementation and configuration, visual servoing could be mainly classified as follows.

1. Visual Servoing Strategy
   (a) Position Based Visual Servoing (PBVS)
   (b) Image Based Visual Servoing (IBVS)
   (c) Hybrid Visual Servoing
2. Visual Servoing Controller
   (a) Proportional Controller
   (b) Adaptive Controller
   (c) Model Predictive Controller

3. Image Feature
    (a) Point Features
    (b) Line Features
    (c) Image Moment Features
4. Camera Configuration
    (a) Eye-to-Hand
    (b) Eye-in-Hand
5. Number of Cameras
    (a) Mono Vision
    (b) Stereo Vision
    (c) Multiple Cameras
6. Target Situation
    (a) Static Object
    (b) Moving Object
7. Trajectory Planning in Visual Servoing

The following literature review is prepared based on the categories introduced above.

1.2.1 Visual Servoing Strategy

Based on the ways in which the visual feedback is used to control the robot, visual servoing is classified into three different classes, Image Based Visual Servoing (IBVS), Position Based Visual Servoing (PBVS) and Hybrid Visual Servoing (HVS) [5, 24–26]. Each of these strategies is discussed separately in the following sections.

Position Based Visual Servoing

In a position based visual servoing (PBVS) controller, position and orientation of the object are extracted from the image captured by the camera. Comparing the object position and orientation with the desired ones, the errors of position and orientation are computed as the input to the controller. The PBVS controller generates a controlling signal to reduce the relative position and orientation errors. Finally, a joint level servo controller tracks the controlling command produced by the Cartesian controller [27]. The block diagram of a PBVS controller is shown in Figure (1.6).

Obviously, the robot control problem is prevalent and well established. Thus, the main challenge is the robustness, accuracy and speed of computing the object position and orientation [8]. This method is also known as 3D visual servoing [5]. In order to reconstruct 3D information from 2D image information, accurate camera calibration is required. Besides, in order to be able to calculate the object pose, a complete geometric model of the object is needed. Furthermore, using PBVS, no control is available on the image trajectory and the object may leave the camera's field of view (FOV). This normally causes the visual servoing task to fail [28].

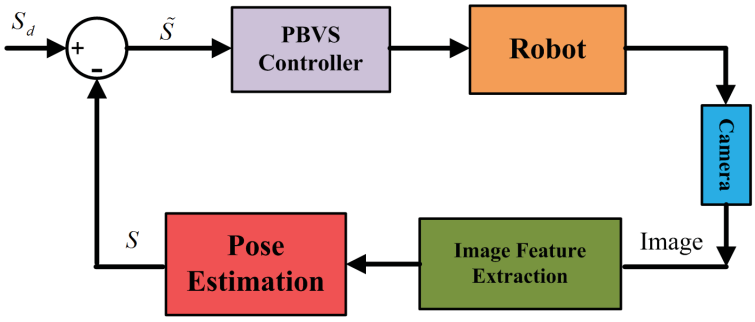


Figure 1.6: Position based visual servoing block diagram [4]

Image Based Visual Servoing

In image based visual servoing (IBVS), the control commands are computed based on the image features data directly. The difference between the current features and the desired ones creates the visual servoing error and the controller moves the robot to reduce this error until it becomes zero. Figure 1.7, shows the structure of an IBVS controller. Image based visual servoing are also called feature based visual servoing. Image features usually used in this approach could be position of points, size of a region, center of a region, length of a line, segment and rotation angle of line and etc. Since there are no image interpolation and 3D reconstruction in IBVS, computation cost is noticeably less than that in PBVS method. In addition, IBVS is more robust to camera and robot calibration errors. The IBVS controller eliminates the effect of calibration errors. Furthermore, as opposed to PBVS, object model is not required in this method [8, 11, 29–31].

Although IBVS controls the image feature trajectory in an almost straight line, the uncontrolled Cartesian trajectory may violate the robot's joint limit, especially when large rotational and translational displacements are required to reach the target. Furthermore, potential image singularities and occurrence of local minima are the other drawbacks of this method [28].

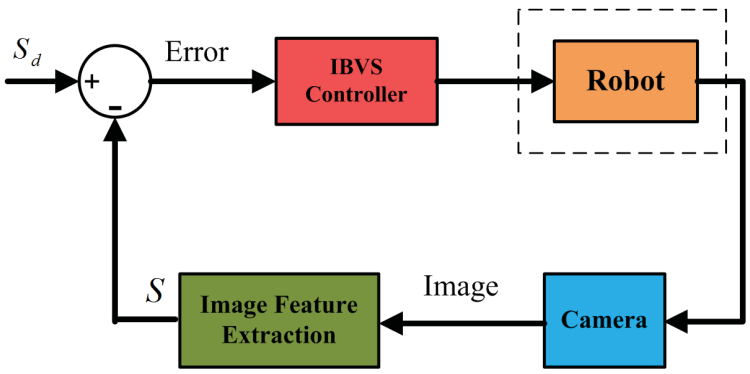


Figure 1.7: Image based visual servoing block diagram[4]

Hybrid Visual Servoing

Considering the drawbacks mentioned for PBVS and IBVS, Hybrid Visual Servoing (HVS) method is introduced combining the previous two strategies. The most well-known hybrid method is $2\frac{1}{2}$D method [9]. This method controls the robot by decoupling the end-effector rotational motion from the translational motion control. The 2-1/2D visual servoing system block diagram is shown in Figure 1.8.

The advantages of this method are that the trajectories of end effector in both Cartesian and image spaces are simultaneously straight lines. In addition, this method is known as a target model free method.

Nevertheless, there are some shortcomings for hybrid system. First, it is necessary to find at least 4 and 8 different feature points for a planar and non-coplanar target object respectively to be able to extract the orientation of the object. Second, it also requires partial pose estimation.

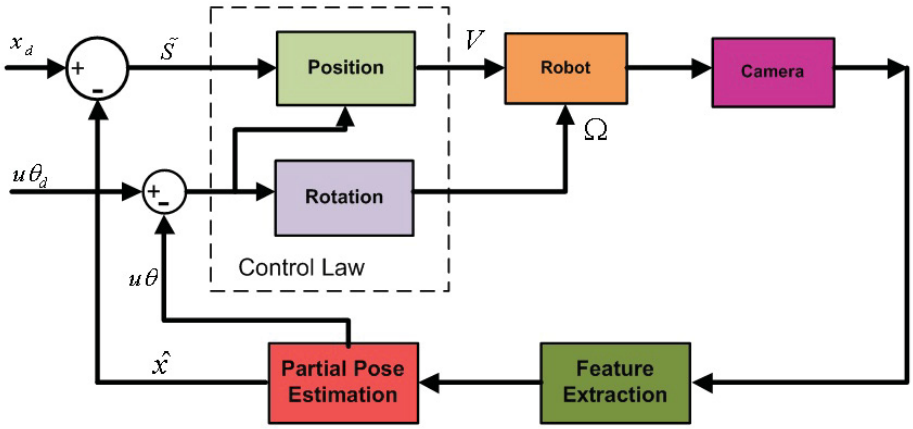


Figure 1.8: 2 1/2 D visual servoing block diagram [5]

A switching method was proposed to partially solve the uncontrolled Cartesian trajectory of IBVS and the uncontrolled image trajectories of PBVS. The controller is switched from IBVS control to PBVS control to avoid robot joint limits when the distance from the end-effector to the target object is too large, and it is switched back to the image based control if the image trajectories come close to the image

boundary [32]. However, this method requires even greater amount of computation, compared to IBVS and PBVS, which is not desirable.

Reviewing the main visual servoing strategies, one can conclude that the overall benefits of IBVS strategy is greater than those of the other two methods. This is due to the less computation that it requires and higher robustness that it provides. In this thesis, all of the research work and developments are based on the IBVS strategy.

1.2.2 Visual Servoing Controller

In addition to the various types of feedback used to control a visual servoing system, various controlling algorithms have been used to guide the features and the robot to their desired position. The most basic controller used in all three strategies is a proportional controller [11, 12]. This controller aims to reduce the features errors exponentially. Later on, more advanced controllers were used to overcome the shortcomings of this controllers. Adaptive control was introduced to estimate the unknown or uncertain parameters of the system such as camera calibration parameters or the object depth [33]. Robust visual servoing was proposed in order to perform more reliable visual servoing tasks in presence of big camera or robot calibration errors which provides more stability to the system [34, 35]. Model predictive controllers have also been developed for visual servoing systems in order to consider the system constraints such as image boundaries and robot's joint limitation during the motion of the robot [36]. Furthermore, other non-linear controllers have been practiced on visual servoing systems such as sliding mode control to increase the robustness of the task in feature trajectory tracking [37].

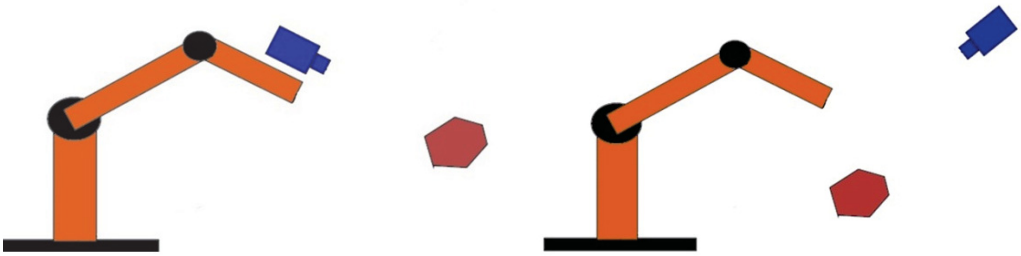
1.2.3 Image Feature

Image feature is a piece of information defined in the image. Image features are used in visual servoing to define the system error by comparing it to the desired values. The desired features information is gathered from an image taken at the place where the camera is in its desired position relative to the target object. Initially, geometric features such as points, segments or straight lines are usually used as image features in visual servoing [9]. Although these features are very easy to be detected for various kinds of objects, they are also prone to getting lost by being occluded by another object or a human hand, during operation. This leads to a visual servoing failure. Recently, in order to apply the visual servoing technology to track the complicated objects and enhance the robustness, several novel features were adopted such as laser points [38] and image moments[6]. In [38], laser point is also used as an instrument to estimate the object's depth. Image moments have widely been used in computer vision for a very long time in applications such as pattern recognition[6]. Recently, utilizing image moments in visual servoing have been widely taken into consideration. Image moment is discussed in details in chapter 3.

1.2.4 Camera Configuration

Based on the camera setup configuration, visual servoing is categorized into eye-in-hand and eye-to-hand visual servoing. In situations where the camera is mounted on the robot's end-effector, it is called an eye-in-hand configuration (Figure 1.9a). If the camera is statically installed looking toward the robot and its workspace, the configuration is called eye-to-hand (Figure 1.9b). Another configuration has been considered where the camera is installed on another robot or on a pan/tilt head in order to be able to reposition the camera to observe the robot from the best position and angle.

The majority of the visual servoing operations are carried out using eye-in-hand



(a) (b)

Figure 1.9: Eye-in-Hand(a) and Eye-to-Hand(b) Configurations

configuration. The eye-in-hand configuration could be focused on the object, while the eye-to-hand configuration provides a picture including irrelevant information. Using an eye-to-hand configuration, the image of target object also may be occluded during the motion by the manipulator itself or other obstacles. However, an eye-to-hand configuration or stationary camera gives a wider field of view and it is mostly used in applications which includes moving objects.

Regardless of camera configuration, camera calibration must be performed in order to obtain the camera intrinsic parameters such as focal length, resolution and the principle point [11]. Some camera calibration methods can be found in [39] and [40].

Chesi and Hashimoto have investigated the stability problem of each configuration in [41]. Due to indispensable benefits of each configuration, visual servoing can take advantage of both configurations, simultaneously. A cooperation of eye-in-hand/eye-to-hand configuration is introduced in [42] and [43].

Calibrating the camera and the robot is both time and money consuming. In order to overcome such problem, some uncalibrated visual servoing methods have been introduced. Malis proposed a visual servoing method invariant to camera parameters [44]. He proved the stability of the controller in presence of large calibration errors. Other uncalibrated and automatic calibration visual servoing techniques are introduced in [45, 46]. It is shown that position based visual servoing does not have

good precision using uncalibrated camera and manipulators. Since this project aims at investigating visual servoing of robotic manufacturing systems, the eye-in-hand configuration is adopted in this thesis, which allows the camera to focus on the workpiece in the work space.

1.2.5 Number of Cameras

Visual servoing can be performed using different number of cameras; single camera (Monocular), two cameras (Stereo Vision or Binocular) and multiple cameras. In each of these categories, the camera(s) could be installed in an eye-in-hand or eye-to-hand configuration. Kragic and Christensen [47] reported a comprehensive survey on each category. Among all categories, single camera needs less processing time to extract the visual information. Due to the fact that, every point in the 2D image plane corresponds to a line[11] in 3D space, a single camera can not provide a good estimation of the distance between the camera and the object. The solution to obtain more precise position and depth of the object is to use a stereo camera vision system. Depth computation is done by comparing the small differences between multiple views of the same scene. The existing constraint between images that allows the computation of the depth is called Epipolar Geometry. It is shown that the depth of a 3D point with respect to the camera is inversely proportional to the position difference of their projection on the left and right image plane. Depth estimation is discussed in detail in chapter 5. Stereo vision could be easily implemented, but the stereo matching problem or finding the correct match is one of the most active research areas in computer vision[48]. Some related works are presented in [49, 50].

Stereo vision system is rarely used in an eye-in-hand configuration. Stereo vision systems usually have smaller field of view than mono vision. This is because stereo vision systems only work with the shared part of the field of view of each

camera. This also limits the camera baseline distance, which affects the accuracy of depth estimation. In contrast, stand-alone stereo vision configuration has fewer limitations and hence is very common in visual servoing systems. Kragic addressed some eye-in-hand and stand-alone stereo vision configuration applications in [47].

The third and rarest case is the multiple camera system. Although, such systems provide more information and may seem worthy for accurately detecting the objects and its depth, the image processing and features matching is more time consuming compared to the other types of configurations. However, in cases where the target object is very large, using multiple camera is inevitable. Recently, Zhao et al.[51] proposed a configuration of four camera for a visual servoing system to deal with large work pieces, where one or two cameras is not enough to observe the whole work piece. Some related works is presented in [47]. A single camera system is used in this thesis in order to be able to process the system fast.

1.2.6 Stationary and Moving Object

One of the fundamental capabilities of vision system is the ability to track and catch moving objects. This can be very useful in production lines, where pausing the process for catching the objects may be inefficient. Thus the visual servoing methods for moving object can be utilized to increase the production efficiency.

Catching a moving object with a robot becomes a challenging problem in visual servoing which needs a well-designed trajectory planning. Various methods and strategies have been developed for this matter. All these strategies can also be classified under two main classes of visual servoing, Position Based and Imaged Based, which are listed as follows:

1. Position Based Methods

    (a) Trajectory regeneration methods[52–55]

(b) Potential field methods[56]

(c) Navigation Guidance methods[57]

2. Imaged Based Methods

(a) Potential Field methods [58]

In a catching task, the visual servoing is responsible for keeping the object in the field of view and reducing the error between the actual position and the desired position [59]. In such applications, motion prediction algorithms can help in tracking and catching the objects precisely [53]. An adaptive visual servoing method was also introduced in [60], in cases of unknown object movements. In [61], an example of catching moving object is presented, where the robot uses an on-line trajectory planning algorithm to follow and grasp the object. The on-line trajectory planning is based on the adaptive prediction, planning and execution. This algorithm generates and modifies the trajectory repeatedly throughout the procedure.

1.2.7 Trajectory Planning in Visual Servoing

Visual servoing controllers were initially developed to perform a complete visual servoing task. However, the developed controller cannot deal with all the challenges. During the evolution of visual servoing, researchers have made a great effort on solving image based visual servoing problems. Most probable problems of an IBVS are presented in [28] such as interaction matrix singularity, local minima and etc. Furthermore, IBVS suffers from a number of deficiencies which can be classified as follow.

1. Instability of system in long distance tasks
2. Interaction matrix singularity
3. Local minima

4. Instability of the system in tasks with rotation of 180^o about camera's center
5. Having no control on the speed of the robot during the visual servoing task
6. Unknown path of the robot prior to the tasks
7. Features leaving the field of view

All these drawbacks causes IBVS not to guarantee a successful and complete visual servoing task. Combining visual servoing with trajectory planning techniques is a possible solution to overcome the above mentioned problems. In addition, the robot trajectory could be planned to keep system within its constraints limits during visual servoing execution. Both image and physical constraints can be considered. Some of the image constraints are the field of view boundaries, local minima spots and singularities in interaction matrix. Some of the physical constraints are the joint limits, robots Jacobian singularities, actuator limitations, and collision with obstacle and self-collisions. Although image constraints are specific to visual servoing tasks, physical constraints of the robot is a known challenge in robotic researches area and a great deal of investigation has been devoted to it [62, 63].

Generally speaking, in approaches where trajectory planning is combined with visual servoing algorithms, a trajectory is planned using the information from the image features and the desired ones, to take the robot to or close to its desired position. This path is produced by the trajectory planner block considering the constraints of the robot. After the trajectory was executed on the robot, a visual servoing controller eliminates any remaining error in the system. The overall process increases the stability of the system and creates a more reliable visual servoing method.

Knowing that more than just one trajectory could fulfill the task, optimal trajectory planning arises aiming to find the path which minimizes a cost function such as distance from the image boundary, distance to a straight line from initial to desired features position in image space, length of the path traversed by the robot,

energy consumption, etc. [64–66].

The challenging problem in trajectory planning is that the planned image trajectory may not match to any feasible trajectory of the robot in Cartesian space and joint space. The most basic method developed for solving this problem is using stereo vision and the epipolar geometry constraint between two camera images [67]. Utilizing the privilege of epipolar geometry, an image trajectory is generated on both images in a way that corresponds to a feasible or even straight line trajectory in Cartesian space [68, 69].

Moreover, other methodologies are presented to partially determine the euclidean displacement of the camera using projective homography matrix. Projective homography introduces a relationship between the images taken of one scene from different views. Knowing the camera calibration, the euclidean homography matrix can be computed up to a scale. The euclidean homography matrix could be decomposed to the rotation matrix relating the initial and desired camera configuration and their translation vector.

Mezouar et al. [58] used this method to generate an image based trajectory considering physical and image constraints. The trajectory is based on a potential field techniques. Utilizing the homography decomposition, the Cartesian space and joint space trajectory could be predicted and forced to obey the constraints. Later on, in [64] they parametrized the trajectory in order to plan an optimized trajectory. Similarly an optimal and constrained path planning for visual servoing is presented in [66].

By parameterizing the trajectory from the initial to the desired configuration, Chesi [65] presented a new framework for path planning based on the use of homography decomposition. He introduced the constraints in the linear matrix inequalities format to ensure the global minima of the optimization problem.

Homography decomposition techniques considerably address the image based trajectory planning corresponding to feasible 3D paths. However, one should note its shortcomings listed as follows. The homography matrix can only be found up to a scale which could affect the translation vector decomposition. Furthermore, its high dependency on camera calibration is not desirable as image based technique.

Apart from all other path planning challenges, generating the path requires a parameterizing technique which defines the trajectory between the initial and the desired points. Different parameterizing techniques could be named such as polynomial functions [61, 66], summation of weighed monomial [65], etc. The number of parameters should be at least the number of initial and final conditions. Defining more parameters could make more degrees of freedom in planning specially in planning an optimal path.

A whole different method of trajectory planning is potential field based planning. The potential field produces an attraction force to guide the camera to the desired location. Meanwhile, a repulsive force is produced to keep the features from nearing the image boundary in order to lower the risk of features leaving the field of view. In addition, other repulsive forces can be produced in order to make the robot respect other physical and image constraints. A possible problem of this method is that it could get stuck in a local minima which is created by equal attractive and repulsive forces [58].

1.3 Image Based Visual Servoing Basics

The basic idea of a visual servoing system is illustrated in a block diagram format as shown in Figure 1.10. As it can be seen, the whole system consists of two main parts; the visual servoing block and the robotic block.

The camera is attached to the robot's end-effector and captures the image of

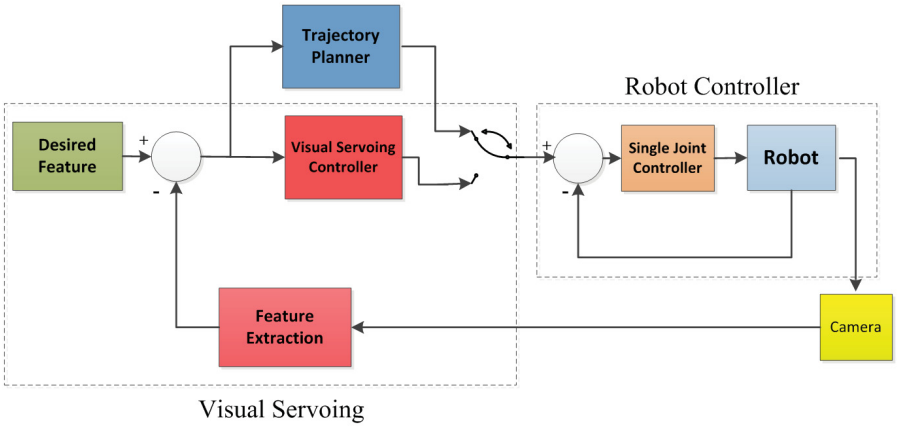
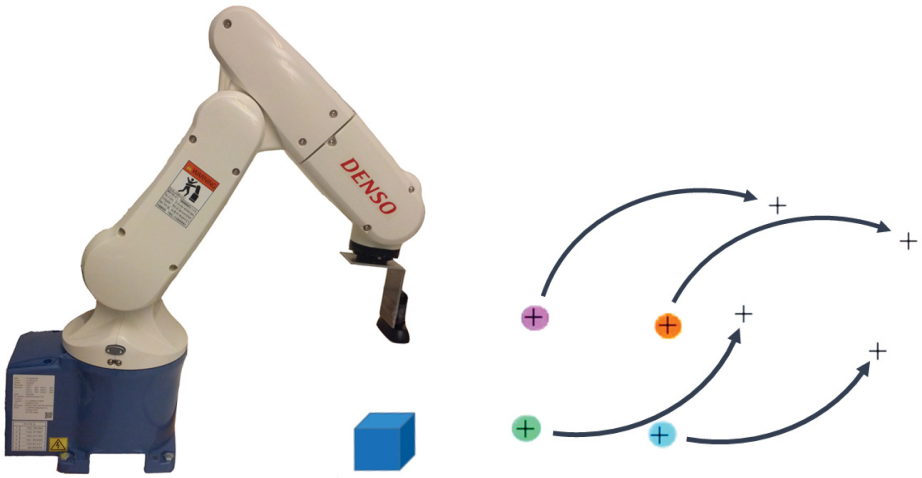


Figure 1.10: General structure of a IBVS block diagram

the object. The features are extracted in the feature extraction block and sent to the visual servoing controller or trajectory planner. The visual servoing block generates a velocity or acceleration command for the robot according to system error. The system error is defined as the difference between the features in the image and the desired image. The command could be generated by a visual servoing controller or a visual servoing trajectory planner. The robot is required to follow the generated command in order to reduce the system error. This is done by the controller in the robotic block. The robotic controller matches the robots end-effectors velocity or acceleration with the command received from the visual servoing block. Either a single joint controller or a computer torque controller could be used to control the robot. In this thesis, a single joint controller is used. More explanation of the robot controller is presented in chapter 2. The focus of this dissertation is on designing the visual servoing block. The use of visual servoing controller and visual servoing trajectory planning is considered in this thesis.

In an image based visual servoing, the extracted features are compared with the desired ones which defines the visual servoing error vector. The desired features are the features, extracted from the image where the robot end-effector or the camera (which are attached to each other,) is in a known position and orientation with



(a) Robot positioned in a known relative situation with respect to the object

(b) Image features and desired features

Figure 1.11: Features and robot in desired positions

respect to target object. Having the desired image features, if the camera is taken to a position and orientation which the image features match the desired features, the camera (end-effector) is in the known position and orientation with respect to the object. At this position and orientation, the end-effector could catch the object with a previously known motion such as moving in vertical direction. The desired features are usually captured when the end-effector is parallel to the object with a vertical distance to it, shown in Figure 1.11a. Based on this explanation, a visual servoing task is complete when the features match the desired ones. Figure 1.11b, shows how this could happen. In this picture the circles with plus sign in the middle are the image features and the plus signs are the desired ones.

1.4 Research Objectives and Scope

As mentioned in section 1.2.7, a visual servoing task performed using conventional methods tends to fail due to their drawbacks and limitation. Conventional visual servoing only assure local stability. Thus, industries cannot rely on such

systems. The main goal of this research is to develop a complete visual servoing algorithm in order to fulfill a complete visual servoing task. In other words, the proposed theory should guarantee the stability of the system in all tasks. This is fulfilled by developing a new visual servoing controller and trajectory planning algorithm. The research work in this Ph.D. thesis is carried out in different stages.

First, a controller is designed to resolve some of the existing problems with IBVS controller. This controller is designed following all the image based visual servoing assumptions and it is called Augmented Image Based Visual Servoing (AIBVS). The main idea of this controller is that it produces acceleration as the controlling command. In contrast to IBVS, a proportional derivative (PD) controller is developed to provide the robot with the controlling command. This controller can achieve smoother and more linear feature trajectory in the image space and decrease the risk that the features leave the field of view. The developed control method also enhances the camera trajectory in 3D space. The stability of the proposed method is fully investigated by using Lyapunov and perturbed systems theory. Experimental tests are performed on a 6 DOFs robotic system to validate the effectiveness of the controller. The performance of the controller is compared with that of a conventional Image Based Visual Servoing (IBVS) [70, 71].

This controller is developed for point features. In some cases where it is hard to extract point features from the image, image moment is used as the visual servoing features. In the next step, the controller is adapted to be used with image moment features. A general formulation for the visual servoing interaction matrix is derived. Three different sets of image moments features from the literature are used in the AIBVS and the corresponding controllers have been fully tested. Experimental tests show the validity of this controllers [72].

As discussed in the literature review, visual servoing controllers could stabilize the visual servoing task only in a local region around the desired position. This

means that if the initial locations of the features are far from the desired features, the visual servoing most probably fails. Moreover, visual servoing controller is not aware of the robot's configuration during the task. In the situation where reaching the desired features requires a complicated motion in the joint space of the robot, the visual servoing could cause the robot to collide itself or reach it's joint limits. In addition, there are some special tasks that visual servoing controllers could not handle, such as tasks which include 180 degree rotation of the camera about its center. Due to these reasons, researchers have started to use trajectory planning algorithms in visual servoing, recently. Various approaches have been developed for this matter. However, their shortcoming such as their sensitivity to camera calibration errors or being dependent on 3D reconstruction algorithms, trajectory planning in visual servoing requires more study.

In the last stage of this project, a new approach is presented in planning a trajectory for an image based visual servoing system. In this method, the camera's velocity screw is parameterized using time-based profiles. The parameters of the velocity profile are then determined such that the velocity profile takes the robot to its desired position. This is done by minimizing the error between the initial and desired features. A depth estimation technique is proposed to provide the trajectory planning algorithm with an accurate initial depth. This algorithm is tested and validated via experiment on a 6 DOFs Denso robot in an eye-in-hand configuration. This method extends the stability range of the system compared to traditional IBVS controllers. The merit of the proposed method is that it respects the system constraints such as robotic and image constraints. Experimental results demonstrate that the proposed method provides a reliable visual servoing algorithm by overcoming the IBVS drawbacks such as surpassing the system limits and causing instability of the system in fulfilling the tasks which requires a 180^o rotation of the camera about its center.

1.5 Thesis Contribution

The presented research work is published (or submitted for publication) in a number of journals and conference proceedings ([70–80]). Following is the list of author's contributions followed by the related publications;

1. Augmented image based visual servoing controller

   (a) M. Keshmiri, W. F. Xie, and A. Mohebbi, "Augmented Image Based Visual Servoing of a Manipulator Using Acceleration Command," Industrial Electronics, IEEE Transactions on, vol. 61, no. 10, pp. 5444-5452, Oct. 2014.

   (b) M. Keshmiri and W. F. Xie, "Augmented Imaged Based Visual Servoing Controller for a 6 DOF Manipulator Using Acceleration Command," in Decision and Control (CDC), 2012 IEEE 51st Annual Conference on, pp. 556-561, Maui, HI, Dec. 2012.

2. AIBVS controller for visual servoing with image moment features

   (a) M. Keshmiri and W. F. Xie, "Augmented Image Based Visual Servoing Using Image Moment Features," Accepted at the ASME 2014 International Mechanical Engineering Congress & Exposition, Montreal, Canada, Nov. 2014.

3. Visual servoing using trajectory planning algorithm

   (a) M. Keshmiri and W. F. Xie, "Visual Servoing of a Robotic Manipulator Using an Optimized Trajectory Planning Technique," in Electrical and Computer Engineering (CCECE), 2014, 27th Annual IEEE Canadian Conference on, Toronto, Canada, May 2014.

(b) M. Keshmiri and W. F. Xie, "Visual Servoing Using an Optimized Trajectory Planning Technique for a 4 DOFs Robotic Manipulator," Submitted to Journal of Intelligent & Robotic Systems, 2014.

(c) M. Keshmiri and W. F. Xie, "Visual Servoing Using a Trajectory Optimization Technique," Submitted to Mechatronics, IEEE/ASME Transactions on, 2014.

4. Real time model predictive controllers for visual servoing

   (a) A. Hajiloo, M. Keshmiri, and W. F. Xie, "Real-Time Model Predictive Visual Servoing Controller," To be submitted to Industrial Electronics, IEEE Transactions on, 2014.

5. Catching moving object using the AIBVS controller and navigation guidance technique

   (a) M. Keshmiri and W. F. Xie, "Catching Moving Objects Using a Navigation Guidance Technique in an Robotic Visual Servoing System," American Control Conference (ACC), pp. 6302-6307, Washington, DC, Jun. 2013.

   (b) A. Mohebbi, M. Keshmiri, and W. F. Xie, "Eye-in-Hand Image Based Stereo Visual Servoing for Tracking and Grasping Moving Objects," Processing of 33rd Chinese Control Conference (CCC), Nanjing, China, Jul. 2014.

   (c) A. Mohebbi, M. Keshmiri, and W. F. Xie, "An Acceleration Command Approach to Stereo Image Based Robot Visual Servoing," in Proceedings of the 19th IFAC World Congress, Cape Town, South Africa, Aug. 2014.

Besides the above publications, the results of a course work project was published in a journal publication and a conference proceeding [81, 82], which are listed

as follows;

1. M. Keshmiri, A. F. Jahromi, A. Mohebbi, M. H. Amoozgar, and W.-F. Xie, "Modeling and Control of Ball and Beam System Using Model Based and Non-Model Based Control Approaches," International Journal of Smart Sensing and Intelligent systems, vol. 5, no. 1, pp. 14 - 35, Mar. 2012.

2. X.-M. Ma, W.-F. Xie, M. Keshmiri, and A. Mohebbi, "Wavelet-based linearization for single-degree-of-freedom nonlinear systems," in Intelligent Robotics and Applications, Lecture Notes in Computer Science. Springer Berlin Heidelberg, vol. 7506, pp. 99 - 110, Oct. 2012.

1.6 Thesis Outline

An introduction along with a literature review on visual servoing, the objectives and research scope of this thesis are presented in Chapter 1. In Chapter 2, the new AIBVS controller is presented. In Chapter 3, the image moment is introduced and used with the AIBVS controller. In Chapter 4, the AIBVS controller is used in an algorithm to catch a moving object using image based visual servoing techniques. In Chapter 5, the trajectory planning algorithm is introduced. The conclusion of this thesis and the proposed future works are presented in Chapter 6. The references cited in this thesis are sorted in the bibliography section.

In this thesis, bold capital letters such as $\mathbf{B}$ are used for matrix definitions with the following exception, where $\mathbf{A}$, $\mathbf{A}_c$, $\mathbf{V}$ and $\mathbf{V}_c$ are used for acceleration, acceleration command, velocity and velocity command screw vectors, receptively. Bold lower-case letters such as $\mathbf{x}$ are used for vectors and unbold letters such as ν are used for scalar and constant values.

Chapter 2

Image Based Visual Servoing Controller

2.1 Introduction

In this chapter, a new visual servoing controller is developed. Various IBVS controllers have been developed such as: proportional controller[12], adaptive visual servoing[83, 84], sliding mode control[85], model predictive control[36], etc. The common idea behind these strategies is that the visual servoing controller generates a velocity screw as the controlling command for the robotic systems, i.e., visual servoing produces a velocity profile which guides the end-effector towards its target. However, the velocity command may not be satisfactory as the command signal of the robot due to the following reasons.

First, visual servoing which generates a velocity command can only allow for a proportional controller to be designed to ensure the error convergence. Although, some literature have reported using PD controller in visual servoing controller design[33], the controller still produces a velocity command. Hence, the control performance deteriorates in terms of its overshoot and its smoothness of trajectory.

Second, image noises from the vision system and the motion vibration of the robot may cause sudden changes to the velocity command. Since the robot controller is tuned to follow these velocity commands exactly, small shakiness and sudden changes in the direction of the robot can occur as the robot moves towards the target.

On the other hand, generating acceleration as the controlling command for the robot controller could overcome these drawbacks. First, in order to generate an acceleration profile, the visual servoing could be designed using a PD or PID controller that could be properly tuned to achieve smoother response and thus better performance.

Second, acceleration generated controllers could solve the implementation problem and eliminate the shakiness and sudden changes in the direction of the robot. This is due to the fact that the states of the robot are related to the integral of the controlling command, which filters out the noise of the system.

In [86], an acceleration controller is presented for a position based visual servoing system. Thus, the modeling and controller design are carried out in Cartesian space. It is well known that PBVS is more sensitive to calibration errors compared with IBVS. Hence, an acceleration controller designed in image space is needed to augment the visual servoing performance.

In addition, the availability of the acceleration profile opens up the possibility to use a single joint control or a computed torque control[87, 88] on the robot controller. Knowing the robot model sufficiently enough, the computed torque method could deliver a high speed and accurate trajectory tracking. Moreover, this type of controller can drive the robot to follow the trajectories by producing the required robots torques. This will be useful for designing industrial robots that deal with heavy payloads where it is necessary to keep the joint loads in actuator torque range.

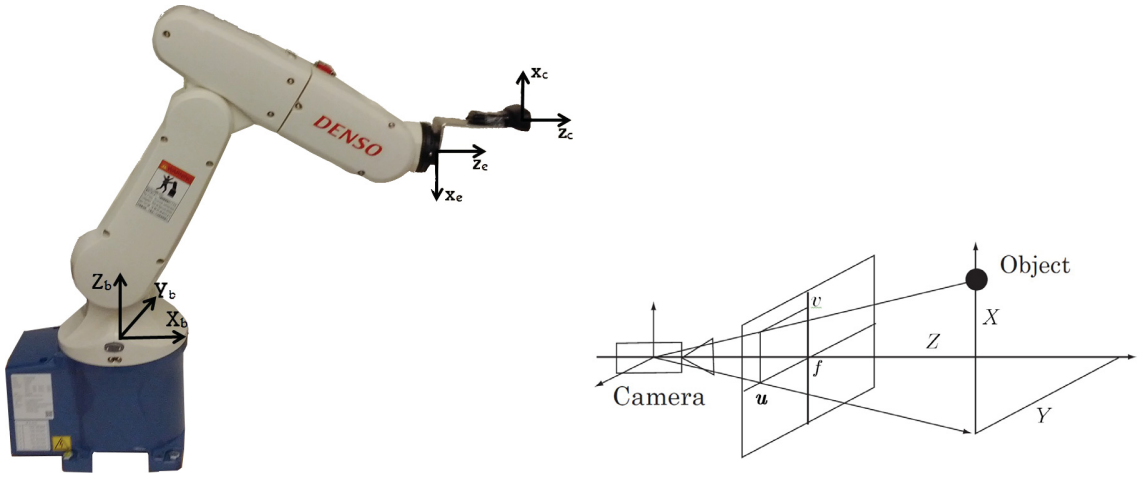
In this chapter, a new visual servoing controller is developed that produces an acceleration screw instead of a velocity screw for the robot controller, in order to achieve smoother and more reliable feature tracking responses [71]. Due to the improvements that the controller makes to a visual servoing task this controller is named Augmented Image Based Visual Servoing (AIBVS). The kinematic equations of the new visual servoing model is derived and a PD controller is used to achieve the exponential convergence of the system, instead of the common proportional (P) controller in conventional IBVS methods. The proper gains of PD controller have been chosen to get an over damped response. For the robot controller, a single joint controller is designed to follow the acceleration profile. In the proposed visual servoing controller where the acceleration is generated, highly nonlinear terms will appear in the visual servoing model. Considering these nonlinearities, the stability of the visual servoing system is proven in this chapter. To validate the proposed controllers, simulation and experimental tests are performed.

2.2 Augmented Image Based Visual Servoing

The robotic system consists of a 6 DOFs manipulator with a camera installed on its end-effector. The target object is assumed to be stationary with respect to robot's reference frame. In this section, point features are used as image features. However, the developed controller in this chapter is also tested with image moment features in the next chapter. The number of point features required in a visual servoing task depends on number of degrees of freedom (DOFs) of the robot. To complete a visual servoing task with a 6 DOFs robot, at least four feature points are required [89].

Let F_b be the robot base frame, F_e be the end-effector frame and F_c be the

camera frame (Figure 2.1a). The object is stationary in the workspace and is characterized by 4 feature points on its four corners. A distinguished merit of IBVS is that it does not require the object frame. A pinhole CCD camera is mounted on the robots end-effector. The image of the 3D points of the object is projected on the image plane of the camera, shown in Figure 2.1b, where ${}^{c}x$ and ${}^{c}y$ are the point coordinates in image plane in meter represented in camera frame. The ${}^{c}x$ and ${}^{c}y$ coordinates can be calculated from equation (2.1) [78].



(a) Robot and Camera Frames (b) Pinhole camera model

Figure 2.1: Robot frames and camera model

$$\begin{bmatrix} {}^{c}x \\ {}^{c}y \end{bmatrix} = \frac{1}{{}^{c}Z} \begin{bmatrix} {}^{c}X \\ {}^{c}Y \end{bmatrix}, \tag{2.1}$$

where $\mathbf{P} = ({}^{c}X, {}^{c}Y, {}^{c}Z)$ is the object coordinate in 3D space presented in camera frame, $\mathbf{p} = ({}^{c}x, {}^{c}y)$ is the coordinate of $\mathbf{P}$ in image space presented in F_c. Throughout this thesis, x and y are used instead of ${}^{c}x$ and ${}^{c}y$. The camera usually provides the pixel information of the image. Considering $\mathbf{m} = (u, v)$ as the pixel coordinate of $\mathbf{p}$, the $\mathbf{p} = ({}^{c}x, {}^{c}y)$ coordinate can be calculated using

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{B} \begin{bmatrix} {}^{c}x \\ {}^{c}y \\ 1 \end{bmatrix}, \tag{2.2}$$

where

$$\mathbf{B} = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.3}$$

is the intrinsic parameters matrix of the camera, u_0 and v_0 are the pixel coordinates of the image plane principal points and α_u and α_v are the scaling factor along x and y coordinates of the image plane in pixel/meter.

Suppose that the features and the desired features are defined in the image plane. In order to find the motion that takes the camera to its desired position, where the features match its desired ones, it is required to find the relationship between the motion of the camera and the motion of the features in the image plane. In other words, it is required to know how the features move in the image as the camera makes a specific motion. This relationship could be found by taking the time derivative of the equation (2.1), given as

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} \frac{\dot{X}}{Z} - \frac{X\dot{Z}}{Z^2} \\ \frac{\dot{Y}}{Z} - \frac{Y\dot{Z}}{Z^2} \end{bmatrix}, \tag{2.4}$$

where $\dot{\mathbf{P}} = (\dot{X}, \dot{Y}, \dot{Z})$ is the velocity of the 3D point in space with respect to camera frame, and $\dot{\mathbf{p}} = (\dot{x}, \dot{y})$ is the velocity of the image of $\mathbf{P}$ in the image space. Furthermore, the relationship between the acceleration of the camera and the acceleration of image feature in the image is required. Taking the second time derivative of the features from equation (2.1), the features acceleration could be written as

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} = \begin{bmatrix} \frac{\ddot{X}}{Z} - \frac{\ddot{Z}X}{Z^2} - 2\frac{\dot{Z}\dot{X}}{Z^2} + 2\frac{\dot{Z}^2X}{Z^3} \\ \frac{\ddot{Y}}{Z} - \frac{\ddot{Z}Y}{Z^2} - 2\frac{\dot{Z}\dot{Y}}{Z^2} + 2\frac{\dot{Z}^2Y}{Z^3} \end{bmatrix}, \tag{2.5}$$

where $\ddot{\mathbf{P}} = (\ddot{X}, \ddot{Y}, \ddot{Z})$ is the acceleration of the 3D point in space with respect to camera frame, and $\ddot{\mathbf{p}} = (\ddot{x}, \ddot{y})$ is the acceleration of the image of $\mathbf{P}$ in the image space. The kinematic relationship between the camera motion and the 3D point in space is given by ([90])

$$\dot{\mathbf{P}} = -\mathbf{v}_c - \boldsymbol{\omega}_c \times \mathbf{P}, \tag{2.6}$$

$$\ddot{\mathbf{P}} = -\mathbf{a}_c - \boldsymbol{\alpha}_c \times \mathbf{P} + 2\boldsymbol{\omega}_c \times \mathbf{v}_c + \boldsymbol{\omega}_c \times (\boldsymbol{\omega}_c \times \mathbf{P}), \tag{2.7}$$

where $\mathbf{v}_c$ and $\mathbf{a}_c$ are the camera's linear velocity and acceleration vectors, which are written as $\mathbf{v}_c = [v_{cx}, v_{cy}, v_{cz}]^{\mathrm{T}}$ and $\mathbf{a}_c = [a_{cx}, a_{cy}, a_{cz}]^{\mathrm{T}}$, respectively. ω_c and α_c are the camera's angular velocity and acceleration vectors, which are written as $\boldsymbol{\omega}_c = [\omega_{cx}, \omega_{cy}, \omega_{cz}]^{\mathrm{T}}$ and $\boldsymbol{\alpha}_c = [\alpha_{cx}, \alpha_{cy}, \alpha_{cz}]^{\mathrm{T}}$, respectively. Substituting equations (2.6) and (2.7) in (2.5) gives

$$\ddot{\mathbf{p}} = \mathbf{L}_a \mathbf{A} + \mathbf{L}_v, \tag{2.8}$$

where, $\mathbf{A}$ is the camera acceleration screw, which is written as

$$\mathbf{A} = [a_{cx} \;\; a_{cy} \;\; a_{cz} \;\; \alpha_{cx} \;\; \alpha_{cy} \;\; \alpha_{cz}]^{\mathrm{T}}, \tag{2.9}$$

and $\mathbf{L}_a$ is the interaction matrix written as

$$\mathbf{L}_a = \begin{bmatrix} -\frac{1}{Z} & 0 & \frac{x}{Z} & xy & -(1+x^2) & y \\ 0 & -\frac{1}{Z} & \frac{y}{Z} & 1+y^2 & -(xy) & x \end{bmatrix}. \tag{2.10}$$

$\mathbf{L}_v$ is obtained from substituting the two last terms of equation (2.7) in equation (2.5) and can be written as

$$\mathbf{L}_v = \begin{bmatrix} \mathbf{V}^{\mathrm{T}} \boldsymbol{\Omega}_x \mathbf{V} \\ \\ \mathbf{V}^{\mathrm{T}} \boldsymbol{\Omega}_y \mathbf{V} \end{bmatrix}, \tag{2.11}$$

where, $\mathbf{V}$ is the camera acceleration screw, which is written as

$$\mathbf{V} = [v_{cx} \;\; v_{cy} \;\; v_{cz} \;\; \omega_{cx} \;\; \omega_{cy} \;\; \omega_{cz}]^{\mathrm{T}}, \tag{2.12}$$

ans $\boldsymbol{\Omega}_x$ and $\boldsymbol{\Omega}_y$ can be calculated from

$$\boldsymbol{\Omega}_x = \begin{bmatrix} 0 & 0 & \frac{1}{Z^2} & -\frac{y}{Z} & \frac{3x}{2Z} & 0 \\ 0 & 0 & 0 & -\frac{x}{2Z} & 0 & -\frac{1}{2Z} \\ \frac{1}{Z^2} & 0 & \frac{2x}{Z^2} & \frac{2xy}{Z} & -\frac{1}{2Z} - \frac{2x^2}{Z} & \frac{y}{Z} \\ -\frac{y}{Z} & -\frac{x}{2Z} & \frac{2xy}{Z} & x + 2xy^2 & -\frac{y}{2} - 2x^2y & \frac{1}{2} - \frac{x^2}{2} + y^2 \\ \frac{3x}{2Z} & 0 & -\frac{1}{2Z} - \frac{2x^2}{Z} & -\frac{y}{2} - 2x^2y & 2x + 2x^3 & -\frac{3xy}{2} \\ 0 & -\frac{1}{2Z} & \frac{y}{Z} & \frac{1}{2} - \frac{x^2}{2} + y^2 & -\frac{3xy}{2} & -x \end{bmatrix} \tag{2.13}$$

and

$$\boldsymbol{\Omega}_y = \begin{bmatrix} 0 & 0 & 0 & 0 & \frac{2y}{Z} & \frac{1}{2Z} \\ 0 & 0 & \frac{1}{Z^2} & -\frac{3y}{2Z} & \frac{x}{Z} & 0 \\ 0 & \frac{1}{Z^2} & \frac{2y}{Z^2} & \frac{1}{2Z} + \frac{y^2}{Z} & -\frac{xy}{Z} & -\frac{x}{Z} \\ 0 & -\frac{3y}{2Z} & -\frac{1}{2Z} + \frac{y^2}{Z} & 2y + 2y^3 & -\frac{x}{2} - 2xy^2 & -\frac{3xy}{2} \\ \frac{1}{2Z} & \frac{x}{y} & -\frac{xy}{Z} & -\frac{x}{2} - 2xy^2 & y + 2x^2y & \frac{1}{2} + x^2 - \frac{y^2}{2} \\ \frac{1}{2Z} & 0 & -\frac{x}{Z} & \frac{3xy}{2} & \frac{1}{2} + x^2 - \frac{y^2}{2} & -y \end{bmatrix}. \tag{2.14}$$

As mentioned before, four feature points are used to complete the visual servoing task. The coordinate position of the features vector are stacked to form a feature vector of $\boldsymbol{\xi}$ given as

$$\boldsymbol{\xi} = \begin{bmatrix} x_1 & y_1 & \cdots & x_4 & y_4 \end{bmatrix}^{\mathrm{T}}, \tag{2.15}$$

where $x_1, y_1, \cdots, x_4, y_4$ are the coordinates of the feature points. Thus, the equation (2.8) for four feature point is written as

$$\ddot{\boldsymbol{\xi}} = \mathbf{L}_{a4}\mathbf{A} + \mathbf{L}_{v4}, \tag{2.16}$$

where

$$\mathbf{L}_{a4} = \begin{bmatrix} \mathbf{L}_a|_{\mathbf{p}=\mathbf{p}_1} \\ \vdots \\ \mathbf{L}_a|_{\mathbf{p}=\mathbf{p}_4} \end{bmatrix}, \tag{2.17}$$

$$\mathbf{L}_{v4} = \begin{bmatrix} \mathbf{L}_v|_{\mathbf{p}=\mathbf{p}_1} \\ \vdots \\ \mathbf{L}_v|_{\mathbf{p}=\mathbf{p}_4} \end{bmatrix}. \tag{2.18}$$

The term $\mathbf{L}_{v4}$, is a nonlinear term that improves the visual servoing task significantly as considered in the controller design. On the other hand, as it is shown in the following sections, this term also forces some limitations on the controlling gains, i.e., the gains should not be less than a specific number to keep the system stable.

2.3 Controller Design

In this section, the visual servoing controller and the controller used to control the robot are presented.

2.3.1 Visual Servoing Controller

The augmented controller is designed based on the visual servoing model (2.16). The visual servoing system error is defined as

$$\boldsymbol{\epsilon} = \boldsymbol{\xi} - \boldsymbol{\xi}_d \tag{2.19}$$

where $\boldsymbol{\xi}_d$ is the desired feature vector. As indicated above, a PD controller is designed in such a way that the system error decreases subject to the following second order system;

$$\ddot{\boldsymbol{\epsilon}} + \kappa_v \dot{\boldsymbol{\epsilon}} + \kappa_p \boldsymbol{\epsilon} = 0, \tag{2.20}$$

where κ_v and κ_p are positive scalars. The following control law could be designed by letting $\ddot{\boldsymbol{\epsilon}} = \ddot{\boldsymbol{\xi}}$ and $\mathbf{A}_c = \mathbf{A}$. Thus,

$$\mathbf{A}_c = \mathbf{L}_{a4}^{+}(-\kappa_v \dot{\boldsymbol{\epsilon}} - \kappa_p \boldsymbol{\epsilon} - \mathbf{L}_{v4}), \tag{2.21}$$

where $\mathbf{A}_c$ is the control signal representing the camera acceleration command, $\mathbf{L}_{a4}^{+}$ is the pseudo inverse of the interaction matrix. Due to some inaccurate estimation

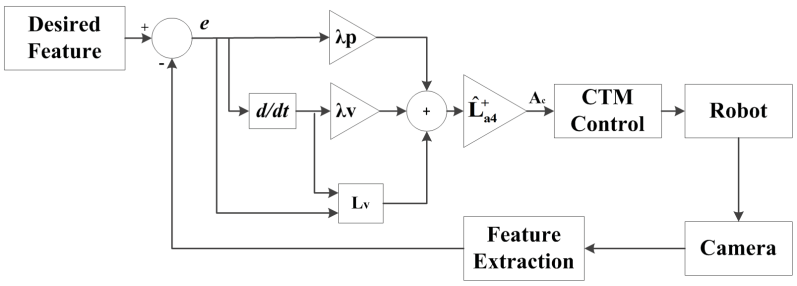


Figure 2.2: AIBVS controller block diagram

of depth (Z) and the presence of camera calibration errors, equation (2.21) should be written as

$$\mathbf{A_c} = \hat{\mathbf{L}}^{+}_{a4}(-\kappa_v \dot{\boldsymbol{\epsilon}} - \kappa_p \boldsymbol{\epsilon} - \mathbf{L}_{v4}), \tag{2.22}$$

where $\hat{\mathbf{L}}^{+}_{a4}$ is the interaction matrix with the uncertainty presented above. Considering (2.22) and (2.16) and knowing that $\ddot{\boldsymbol{\epsilon}} = \ddot{\boldsymbol{\xi}}$, the error equation could be written as

$$\ddot{\boldsymbol{\epsilon}} = \mathbf{L}_{a4}\hat{\mathbf{L}}^{+}_{a4}(-\kappa_v \dot{\boldsymbol{\epsilon}} - \kappa_p \boldsymbol{\epsilon} - \mathbf{L}_{v4}) + \mathbf{L}_{v4}. \tag{2.23}$$

The block diagram of the visual servoing system using the proposed controller is shown in Figure 2.2. It is necessary to mention that getting the features velocity requires taking the time derivative of the feature position which may produce big noise in the velocity signal. To solve this problem, the kinematic relation between the features velocity and the camera velocity is used given as

$$\dot{\boldsymbol{\xi}} = \mathbf{L}_a \mathbf{V} = \mathbf{L}_a(\mathbf{J}^{-1}\dot{\mathbf{q}}). \tag{2.24}$$

to calculate the features velocity. However, the time derivatives of the robot's joints ($\dot{\mathbf{q}}$) are required, which can be acquired directly using tachometers without introducing additional noise. Furthermore, since the target object is stationary and $\dot{\boldsymbol{\xi}}_d = 0$, equation (2.24) can be used to calculate $\dot{\boldsymbol{\epsilon}}$.

2.3.2 Robot Controller

To control the robot either a computed torque controller or a single joint control could be used [87, 91]. A single joint controller is referred to a controller in which each joint of the robot has an independent PID controller. This PID controller provides the required voltage for the joint motor to move the joint to its desired position. Usually, the dynamics of the robot is not taken into consideration in such controllers. The PID control law for ith joint of the robot is given by

$$V_{m_i} = \sigma_p e_{q_i} + \sigma_d \dot{e}_{q_i} + \sigma_i \int e_{q_i} dt, \tag{2.25}$$

where V_{m_i} is ith joint required voltage, σ_p, σ_d and σ_i are the proportional, derivative and integral gain of the PID controller respectively and e_{q_i} is the ith joint angle error given as

$$e_{q_i} = q_{d_i} - q_i, \tag{2.26}$$

and q_{d_i} is ith joint desired angle.

On the other hand, a computed torque control uses the dynamic equation of the robot to calculate the required torque for each joint to take the robot to its desired location. The robot dynamics equations can be written as follows

$$\boldsymbol{\tau} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\dot{\mathbf{q}}, \mathbf{q})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}), \tag{2.27}$$

where $\mathbf{q}, \dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ are the robot joints position, velocity and acceleration vector, respectively, $\mathbf{M}(\mathbf{q})$ is inertia matrix, $\mathbf{C}(\dot{\mathbf{q}}, \mathbf{q})$ is the centripetal and coriolis matrix, and $\mathbf{G}(\mathbf{q})$ is the gravitational term. The control law is calculated as

$$\begin{aligned} \boldsymbol{\tau} = \; & \mathbf{M}(\mathbf{q})(\ddot{\mathbf{q}}_d + \mathbf{K}_d(\dot{\mathbf{q}}_d - \dot{\mathbf{q}}) + \mathbf{K}_p(\mathbf{q}_d - \mathbf{q})) \\ & + \mathbf{C}(\dot{\mathbf{q}}, \mathbf{q})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}), \end{aligned} \tag{2.28}$$

where $\mathbf{q}_d, \dot{\mathbf{q}}_d$ and $\ddot{\mathbf{q}}_d$ are the robot's desired joints position, velocity and acceleration vector, respectively. $\mathbf{K}_p$ and $\mathbf{K}_d$ are the proportional and derivative gain matrices,

respectively. The visual servoing produces the end-effector acceleration. The joint acceleration could be calculated using the robot's Jacobian matrix and its calculation is shown as follows

$$\ddot{\mathbf{q}}_d = \mathbf{J}^{-1}(\mathbf{A}_c - \dot{\mathbf{J}}\dot{\mathbf{q}}), \tag{2.29}$$

where $\mathbf{J}$ is the robot's Jacobian matrix and $\mathbf{J}^{-1}$ and $\dot{\mathbf{J}}$ are its inverse and its derivative, respectively. $\mathbf{A}_c$ is the acceleration command which is produced by the visual servoing block.

2.4 Stability Analysis

Stability analysis of a visual servoing system is highly challenging but essential, especially in the presence of system uncertainties such as camera calibration errors and lack of object depth in an IBVS using monocular camera system. The stability of different visual servoing approaches has been investigated in the literature [5, 12, 28]. In [31], the stability of the IBVS is analyzed in the presence of camera calibration errors. The IBVS interaction matrix has a complex structure because of a large number of variables and their nonlinear formation in this matrix. Thus, no analytical solution is yet available for the pseudo inverse of the interaction matrix. This causes the stability analysis problem to be more challenging. Researchers use the properties of the pseudo inverse matrix to prove the stability of the IBVS and PBVS [29].

In this section, the stability of the proposed controller is analyzed to ensure the effectiveness of the controller. Furthermore, the tuning rules for the PD gains are given to achieve the best response. Some preliminary assumptions are taken into consideration. Since $\mathbf{L}_{a4}\hat{\mathbf{L}}_{a4}^{+}$ has the maximum rank of 6, $\mathbf{L}_{a4}\hat{\mathbf{L}}_{a4}^{+}$ has two null vectors that satisfy $\left\{\mathbf{L}_{a4}\hat{\mathbf{L}}_{a4}^{+}\dot{\boldsymbol{\epsilon}} = 0, \dot{\boldsymbol{\epsilon}} \in \mathbb{R}^8, \dot{\boldsymbol{\epsilon}} \neq 0\right\}$. Assuming that $\mathbf{x}$ does not fall in the null space of $\mathbf{L}_{a4}\hat{\mathbf{L}}_{a4}^{+}$ [12],

$$\mathbf{L}_{a4}\hat{\mathbf{L}}_{a4}^{+} > 0. \tag{2.30}$$

To deal with this problem, first the system equation (2.23) is rewritten in the following format.

$$\ddot{\boldsymbol{\epsilon}} = \underbrace{\mathbf{L}_{a4}\hat{\mathbf{L}}_{a4}^{+}(-\kappa_v\dot{\boldsymbol{\epsilon}})}_{\mathbf{f}(\mathbf{x},t)} + \underbrace{\mathbf{L}_{a4}\hat{\mathbf{L}}_{a4}^{+}(-\kappa_p\boldsymbol{\epsilon}) + (\mathbf{I} - \mathbf{L}_{a4}\hat{\mathbf{L}}_{a4}^{+})\mathbf{L}_{v4}}_{\mathbf{g}(\mathbf{x},t)}, \tag{2.31}$$

where $\mathbf{x}=\begin{bmatrix}\boldsymbol{\epsilon}\\ \dot{\boldsymbol{\epsilon}}\end{bmatrix}$. We consider $\mathbf{f}(\mathbf{x},t)$ as the nominal system and name it

$$\ddot{\boldsymbol{\epsilon}}_{\mathbf{n}} = \mathbf{f}(\mathbf{x},t). \tag{2.32}$$

Considering the function $\mathbf{g}(\mathbf{x},\mathbf{t})$ as the perturbation function, equation (2.31) can be considered as a perturbed system. Choosing

$$\nu = \frac{1}{2}\dot{\boldsymbol{\epsilon}}_n^{\mathrm{T}}\dot{\boldsymbol{\epsilon}}_n, \tag{2.33}$$

as the Lyapunov function candidate, the time derivative of the Lyapunov function is

$$\dot{\nu} = \ddot{\boldsymbol{\epsilon}}_n^{\mathrm{T}}\dot{\boldsymbol{\epsilon}}_n = -\kappa_v\mathbf{L}_{a4}\hat{\mathbf{L}}_{a4}^{+}\dot{\boldsymbol{\epsilon}}_n^{\mathrm{T}}\dot{\boldsymbol{\epsilon}}_n. \tag{2.34}$$

It can be concluded that, as long as the calibration errors and the depth assumption are not too coarse and

$$\mathbf{L}_{a4}\hat{\mathbf{L}}_{a4}^{+} > 0, \tag{2.35}$$

the system given in equation (2.32) is exponentially stable.

Knowing that the system in equation (2.32) is exponentially stable for all $\mathbf{L}_{a4}\hat{\mathbf{L}}_{a4}^{+} > 0$, and also $\mathbf{g}(\mathbf{0},\mathbf{t}) = \mathbf{0}$, using the stability lemma for perturbed systems with vanishing perturbation[92], the stability of system (2.31) can be proven. Assume that the nominal system (2.32) has an equilibrium point of $\boldsymbol{\epsilon} = \mathbf{0}$ with exponential stability and $\nu(t,\mathbf{x})$ is a Lyapunov function candidate for the nominal system that satisfies the following three conditions,

$$c_1\|\mathbf{x}\|^2 \leq \nu(t,\mathbf{x}) \leq c_2\|\mathbf{x}\|^2, \tag{2.36}$$

$$\frac{\partial \nu}{\partial t} + \frac{\partial \nu}{\partial \mathbf{x}} f(t, \mathbf{x}) \leq -c_3 \|\mathbf{x}\|^2, \tag{2.37}$$

$$\|\frac{\partial \nu}{\partial \mathbf{x}}\| \leq c_4 \|\mathbf{x}\|, \tag{2.38}$$

for $\left\{\forall (t, \mathbf{x}) \in [0, \infty) \times D, D \equiv \mathbb{R}^{2n}\right\}$, where n is the number of features, c_1, c_2, c_3 and c_4 in the above mentioned conditions are positive constants. If the perturbation function satisfies (2.39)

$$\|\mathbf{g}(t, \mathbf{x})\| \leq \gamma \|\mathbf{x}\|, \forall t \geq 0, \forall \mathbf{x} \in D, \tag{2.39}$$

where

$$\gamma < \frac{c_3}{c_4}, \tag{2.40}$$

then, the perturbed system is exponentially stable with an equilibrium point at origin, $\boldsymbol{\epsilon} = \mathbf{0}$.

Applying $\nu(t, \mathbf{x})$ and $\dot{\nu}(t, \mathbf{x})$ from equation (2.33) and (2.34) to conditions (2.36) to (2.38) gives,

$$0\|\mathbf{x}\|^2 \leq \frac{1}{2} \dot{\boldsymbol{\epsilon}}^{\mathrm{T}} \dot{\boldsymbol{\epsilon}} \leq \|\mathbf{x}\|^2, \tag{2.41}$$

$$-\kappa_v \dot{\boldsymbol{\epsilon}}^{\mathrm{T}} \mathbf{L}_{a4} \hat{\mathbf{L}}_{a4}^{+} \dot{\boldsymbol{\epsilon}} \leq -\kappa_v \|\mathbf{x}\|^2, \tag{2.42}$$

$$\|\dot{\boldsymbol{\epsilon}}\| \leq \|\mathbf{x}\|. \tag{2.43}$$

Thus, c_1, c_2, c_3 and c_4 can be found as $c_1 = 0, c_2 = 1, c_3 = \kappa_v$ and $c_4 = 1$. Furthermore, a value for γ in (2.39) is required to complete the stability proof. Calculating the norm of $\mathbf{g}(t, \mathbf{x})$ gives,

$$\begin{aligned} \|\mathbf{g}(t, \mathbf{x})\| &= \|\mathbf{L}_{a4} \hat{\mathbf{L}}_{a4}^{+} (-\kappa_p \boldsymbol{\epsilon}) + (\mathbf{I} - \mathbf{L}_{a4} \hat{\mathbf{L}}_{a4}^{+}) \mathbf{L}_{v4}\| \\ &\leq \|\mathbf{L}_{a4} \hat{\mathbf{L}}_{a4}^{+}\| (\kappa_p \|\boldsymbol{\epsilon}\| + \|\mathbf{L}_{a4}\|) + \|\mathbf{L}_{v4}\|. \end{aligned} \tag{2.44}$$

To be able to find a boundary for $\mathbf{g}(t, \mathbf{x})$ in the format of equation (2.39), it is required to find $\|\mathbf{L}_{a4}\|$ as a function of $\|\mathbf{x}\|$. From (2.24), $\mathbf{V}$ can be computed as

$$\mathbf{V} = \hat{\mathbf{L}}_{a4}^{+} \dot{\boldsymbol{\xi}}, \tag{2.45}$$

by substituting (2.45) in (2.11) gives,

$$\mathbf{L}_v = \begin{bmatrix} \mathbf{V}^{\mathrm{T}}\boldsymbol{\Omega}_x\mathbf{V} \\ \mathbf{V}^{\mathrm{T}}\boldsymbol{\Omega}_y\mathbf{V} \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{\xi}}^{\mathrm{T}}\boldsymbol{\Phi}_x\dot{\boldsymbol{\xi}} \\ \dot{\boldsymbol{\xi}}^{\mathrm{T}}\boldsymbol{\Phi}_y\dot{\boldsymbol{\xi}} \end{bmatrix}, \tag{2.46}$$

where $\boldsymbol{\Phi}_x$ and $\boldsymbol{\Phi}_y$ are

$$\begin{aligned} \boldsymbol{\Phi}_x &= \mathbf{L}_{a4}^{+\mathrm{T}}\boldsymbol{\Omega}_x\mathbf{L}_{a4}^{+}, \\ \boldsymbol{\Phi}_y &= \mathbf{L}_{a4}^{+\mathrm{T}}\boldsymbol{\Omega}_y\mathbf{L}_{a4}^{+}. \end{aligned} \tag{2.47}$$

Thus, for four feature points it becomes as,

$$\mathbf{L}_{v4} = \begin{bmatrix} \dot{\boldsymbol{\xi}}^{\mathrm{T}}\boldsymbol{\Phi}_{x_1}\dot{\boldsymbol{\xi}} & \dot{\boldsymbol{\xi}}^{\mathrm{T}}\boldsymbol{\Phi}_{y_1}\dot{\boldsymbol{\xi}} & \dots & \dot{\boldsymbol{\xi}}^{\mathrm{T}}\boldsymbol{\Phi}_{x_4}\dot{\boldsymbol{\xi}} & \dot{\boldsymbol{\xi}}^{\mathrm{T}}\boldsymbol{\Phi}_{y_4}\dot{\boldsymbol{\xi}} \end{bmatrix}^{\mathrm{T}}. \tag{2.48}$$

Each element of $\mathbf{L}_{v4}$ is in a quadratic form. Since $\boldsymbol{\Phi}_{x_i}$ and $\boldsymbol{\Phi}_{y_i}$ $i = 1,2,\dots,4$ are symmetric matrices, one can write

$$\begin{aligned} \dot{\boldsymbol{\xi}}^{\mathrm{T}}\boldsymbol{\Phi}_{x_1}\dot{\boldsymbol{\xi}} &\leq \max(eig(\boldsymbol{\Phi}_{x_1}))\|\dot{\boldsymbol{\xi}}\|^2, \\ &\vdots \\ \dot{\boldsymbol{\xi}}^{\mathrm{T}}\boldsymbol{\Phi}_{y_4}\dot{\boldsymbol{\xi}} &\leq \max(eig(\boldsymbol{\Phi}_{y_4}))\|\dot{\boldsymbol{\xi}}\|^2. \end{aligned} \tag{2.49}$$

Without the loss of generality, the problem is solved for second order norms, and the problem can similarly be solved for other norm orders. Thus one has

$$\|\mathbf{L}_{v4}\| \leq \sqrt{(\max(eig(\boldsymbol{\Phi}_{x_1})))^2 + \cdots + (\max(eig(\boldsymbol{\Phi}_{y_4})))^2}\|\dot{\boldsymbol{\xi}}\|^2, \tag{2.50}$$

where $\|\mathbf{L}_{v4}\|$ should be written as a first order function of the states. So the above inequality is written as

$$\|\mathbf{L}_{v4}\| \leq a\|\dot{\boldsymbol{\xi}}\|, \tag{2.51}$$

where a is a constant, and is defined as

$$a = max(\|\dot{\boldsymbol{\xi}}\|)\sqrt{\max(eig(\boldsymbol{\Phi}_{x_1}))^2 + \ldots + \max(eig(\boldsymbol{\Phi}_{y_4}))^2}. \tag{2.52}$$

Equation (2.50) is obtained only if the norm of feature variations satisfy the following inequality

$$\|\dot{\boldsymbol{\xi}}\| < \frac{a}{\sqrt{(\max(eig(\boldsymbol{\Phi}_{x_1})))^2 + \cdots + (\max(eig(\boldsymbol{\Phi}_{y_4})))^2}}. \tag{2.53}$$

This condition means that the norm of the feature variation vector must be less than a specific value which depends on the configuration of the features.

A numerical investigation has been made for a case of a Denso robot in section 2.6.1, to find the number of parameter a. Consequently, $\|\mathbf{g}(t,\mathbf{x})\|$ can be written as

$$\|\mathbf{g}(t,\mathbf{x})\| \leq \|\mathbf{L}_{a4}\hat{\mathbf{L}}_{a4}^{+}\|(\kappa_p\|\boldsymbol{\epsilon}\| + a\|\dot{\boldsymbol{\epsilon}}\|) + a\|\dot{\boldsymbol{\epsilon}}\|, \tag{2.54}$$

thus,

$$\|\mathbf{g}(t,\mathbf{x})\| \leq (\kappa_p + a)\|\mathbf{L}_{a4}\hat{\mathbf{L}}_{a4}^{+}\|\|\mathbf{x}\| + a\|\mathbf{x}\|. \tag{2.55}$$

Comparing (2.55) with (2.39), it can be concluded that

$$\gamma = (\kappa_p + a)\|\mathbf{L}_{a4}\hat{\mathbf{L}}_{a4}^{+}\| + a. \tag{2.56}$$

In order to satisfy the system stability relation, inequality (2.40) should be satisfied. By substituting (2.56) in (2.40), a relationship between the system gains is established as

$$(\kappa_p + a + \frac{a}{\|\mathbf{L}_{a4}\hat{\mathbf{L}}_{a4}^{+}\|}) < \kappa_v. \tag{2.57}$$

Thus, if $\mathbf{L}_{a4}\hat{\mathbf{L}}_{a4}^{+}>0$ and inequality (2.57) is satisfied, the system (2.31) will be locally exponentially stable.

Remark: The stability of the robot using either a computed torque controller or a single joint controller is fully proven[87]. Thus, combining the AIBVS with a computed torque control or a single joint controller in a cascade format, it could be concluded that the whole system is stable.

2.5 Experimental Setup

In order to validate the developed algorithms in this research the algorithms are tested on a experimental setup. In this section, the experimental setup is presented. Figure 2.3 shows the experiment setup components and connections. The

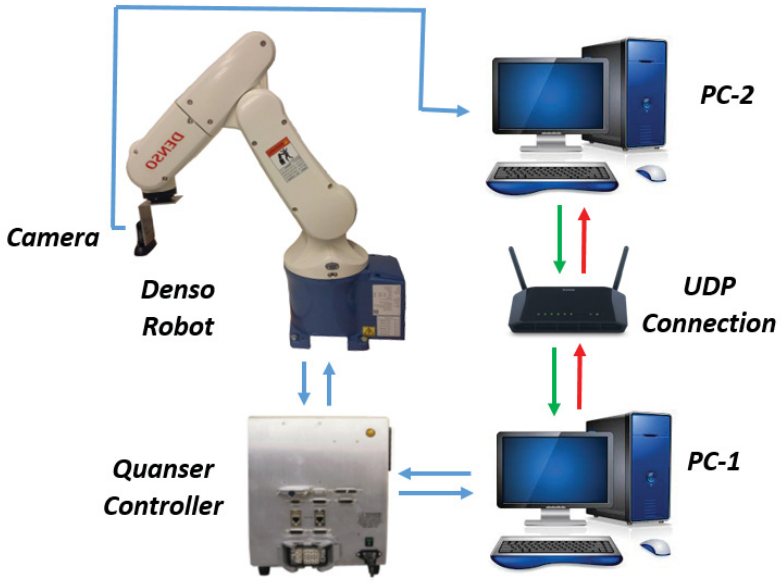


Figure 2.3: Experimental setup components

system shown in this figure consists of a 6-DOF DENSO VP-6242G robot, a Quanser open architecture control module, a Logitech C270 digital camera with 1280 by 720 pixels resolution and two PCs, one for controlling the robot and the other for image acquisition.

Denso VP-6242G is a high precision manipulator robot with 6 rotating joints which are powered by AC servo motors. The detailed specification and description of the robot is given in Appendix A. The position feedback is provided by absolute encoders mounted on each joints. The Denso robot end-effector has the ability to host various devices for different applications. The Denso robot, is supplied with Quanser open-architecture control module. The Quanser module includes six independent amplifiers and built-in feed-forward with PID controllers. The feed-forward lets user to apply current signals to the amplifier in addition to the velocity signal. The controller communicates with PC-1 at a rate of 1kHz. The vision system consists of a Logitech C270 digital cameras with 1280 by 720 pixels resolution, mounted on the end-effector using a bracket.

PC-1 reads the position of the robot from the Quanser controller and generates the controlling command and sends it to the Quanser controller. The image processing and feature extraction algorithms require a considerably high load of computation. This will cause the whole implementation process to slow down and affect the system's real-time performance. For this matter, the vision system is connected to another PC (PC-2) and the image processing and feature extraction algorithms are processed on this PC. The extracted features data is transmitted to PC-1 connected to the robot using a User Datagram Protocol (UDP) network connection protocol. The visual servoing algorithm running on PC-1 uses the transferred image processing data to guide the robot.

2.5.1 Camera Calibration

Camera calibration is the process of determining the camera's intrinsic parameters and the extrinsic parameters with respect to the world coordinate system. Calibrations techniques rely on sets of world points whose relative coordinates are known and whose corresponding image-plane coordinates are also known [93–95].

The Camera Calibration Toolbox for MATLAB implements the calibration method to find the camera's intrinsic and extrinsic parameters. The inputs of this toolbox are several images of a model chessboard plane containing the calibration points. The corners on the calibration plane are used as calibration points. Figure 2.4 illustrates this procedure. The camera intrinsic parameters extracted from this method are given in Table 2.1.

2.5.2 Image Processing

In order to simplify the image processing task, a white environment is created around the object. A rectangular object with four different colors on each corner

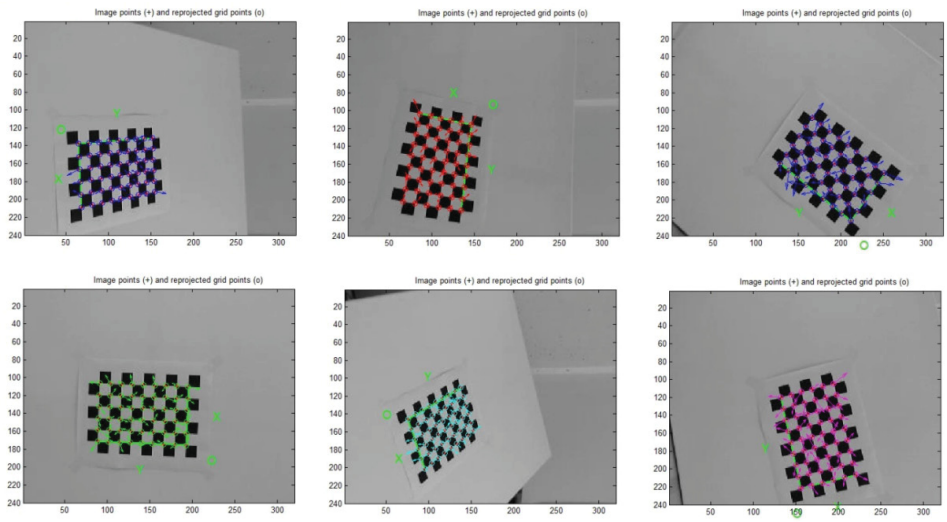


Figure 2.4: Calibrating the camera using a chessboard pattern

Table 2.1: Camera Parameters

Parameters	Values
Focal length	0.004 (m)
X axis scaling factor	110000 (pixel/m)
Y axis scaling factor	110000 (pixel/m)
Image plane offset of X axis	120 (pixel)
Image plane offset of Y axis	187 (pixel)

is placed in the workspace. The centers of the colored spots are used as the feature points. The Red Green Blue (RGB) image model acquired by the camera is transferred to the computer as a three-dimensional matrix. The colored spots are distinguished and located through a search algorithm. The centers of the spots are calculated by averaging each colored area's x and y pixel value. The main image processing algorithm can be briefly explained in the following three steps.

- Step 1. Convert the RGB model image to Hue Saturation Value (HSV) model image to reduce the effect of changes in the light and brightness of the environment;

- Step 2. Extract the HSV values of the colored spots and detect the spots based on their colors;

- Step 3. Calculate the center of each color spot using the following equations

$$X_{center} = \frac{1}{N}\sum_{n=0}^{N} x_n,$$
$$Y_{center} = \frac{1}{N}\sum_{n=0}^{N} y_n, \tag{2.58}$$

where N is the number of pixels belonging to the color spot, x_n and y_n are the x and y coordinates of the colored spot pixels, X_{center} and Y_{center} are the x and y coordinates of the colored spot center. The speed of the image acquisition is 30 frames per second. The image processing is carried out for each frame acquired from the camera. In order not to reduce the speed of the manipulator controller, the image processing is performed on the second computer(PC-2) and the results are sent via UDP to the manipulator controller.

2.5.3 Quarc Interface

QUARC is a multi-functional software suite that connects with Mathworks Simulink for rapid controls programming and hardware based experiments. QUARC provides Windows-based procedures to make Simulink designed controllers to be converted into real-time Microsoft Visual Studio based code that can run on many target processor and operating systems combinations. Figure 2.5 shows the implemented Simulink models for an experimental IBVS test.

2.6 Simulation and Experimental Results

In this section, the proposed controller is tested on a 6 DOFs Denso robot and the results are compared with those of conventional IBVS controller in [12]. Both simulation and experimental results are presented in this section. An explanation is given on finding the controlling gains for the Denso manipulator to ensure the stability of the visual servoing system.

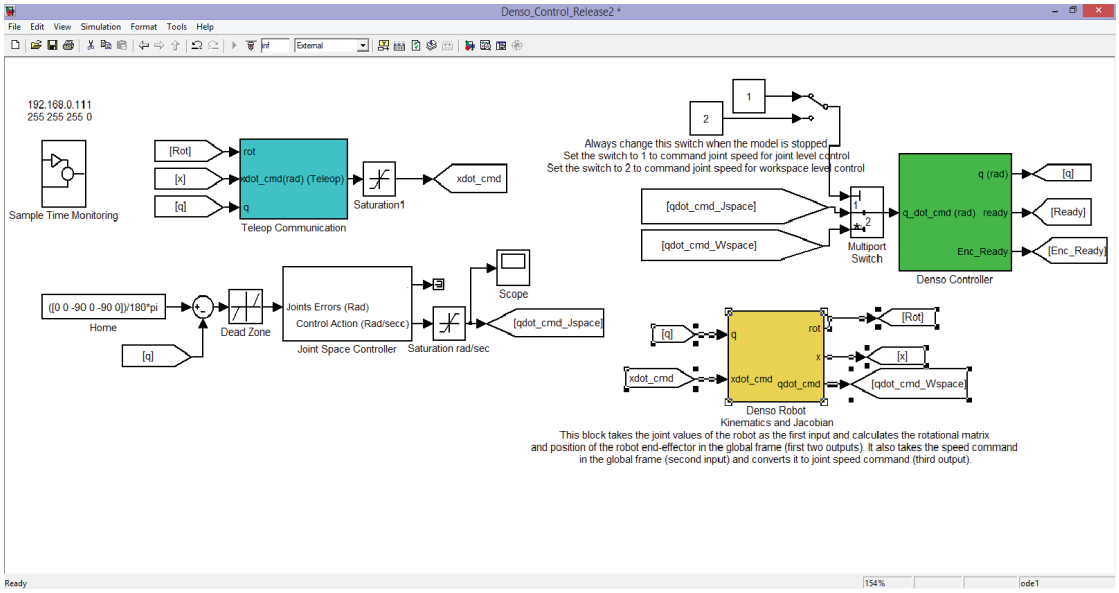


Figure 2.5: Simulink diagram for an experimental IBVS test

2.6.1 Case Study of a Denso VS-6556G

As discussed before, in order to have equation (2.51), condition (2.53) must be satisfied. However, an analytical calculation for $\mathbf{\Phi}_{x_i}$ and $\mathbf{\Phi}_{y_i}, i = 1,2,\ldots,4$ is not available. In this section, a numerical investigation is carried out to show that condition (2.53) could be met in a robotic system. Initially, the investigated system is defined as such; Denso VS-6556G is used as the manipulator [87, 96] and the intrinsic parameters of the camera are given in Table 2.1.

By manually adjusting the robot in a configuration such that the object could be captured by the camera, the distance between each two points will be at most 300 pixels.

Thus,

$$\frac{a}{\sqrt{(\max(eig(\mathbf{\Phi}_{x_1})))^2 + \cdots + (\max(eig(\mathbf{\Phi}_{y_4})))^2}} \approx 3.02 \times 10^{-6} a. \tag{2.59}$$

According to the limits set for the tests, the robot will move with the maximum linear velocity of 0.5 (m/s).

By calculating $\mathbf{L}_a$ for this situation $\|\dot{\boldsymbol{\xi}}\|$ will be approximately less than $8.5 \times$

10^{-3}(m/s) . Thus, a should be chosen as 2.808×10^3.

To find a numerical value for the stability criteria, a numerical value is required for $\|\mathbf{L}_{a4}\hat{\mathbf{L}}_{a4}^{+}\|$. Investigating the image screen, it can be seen that as the distance between the image features grows, the norm $\|\mathbf{L}_{a4}\hat{\mathbf{L}}_{a4}^{+}\|$ decreases and vice versa. Considering the above mentioned conditions, for the maximum distance between the feature points, the norm $\|\mathbf{L}_{a4}\hat{\mathbf{L}}_{a4}^{+}\|$ will be around 250. Thus, the final stability condition can be derived as

$$(\kappa_p + 2812) < \kappa_v. \tag{2.60}$$

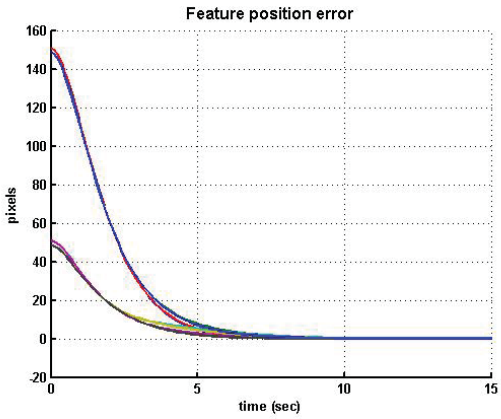
The validity of this condition has been proven in simulation and experiments.

2.6.2 Simulation Results

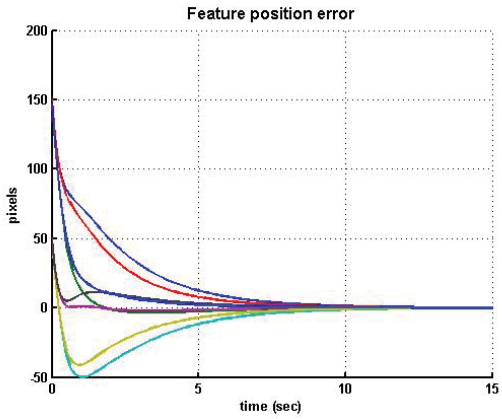
Simulation results are presented in this section. The experimental setup is modeled in the simulation as close as possible to achieve the best possible results. The robotic toolbox for Matlab [97] is used to model the robot and the computer vision toolbox for Matlab [98] is used to model the camera. The camera intrinsic parameters are given in Table 2.1. The specification of the Denso robot is presented in Appendix A. A comparison with the conventional IBVS has been made in terms of feature error and smoothness of the trajectory. The simulation results are illustrated in Figures 2.6a to 2.6b.

The results shown in Figures 2.6a to 2.6b, demonstrate that the feature error is reduced without any overshoot in the AIBVS method. This causes the features to move in a straighter line than the features using the IBVS method. This merit reduces the probability of features running out of the image plane.

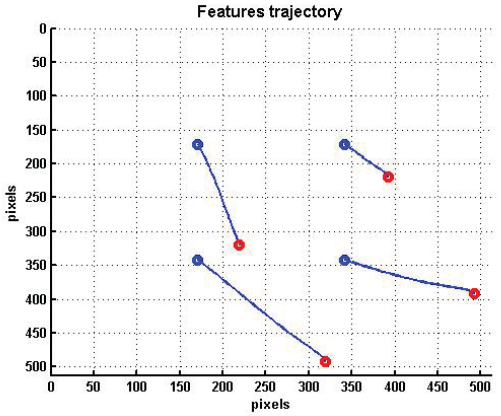
Another simulation test has been done to show how the proposed method reduces the risk of features leaving the field of view. Results show how AIBVS improves the feature trajectory and keep them from leaving the field of view. Figures 2.7a and Figure 2.7b show that for the same task, IBVS has left the field of view



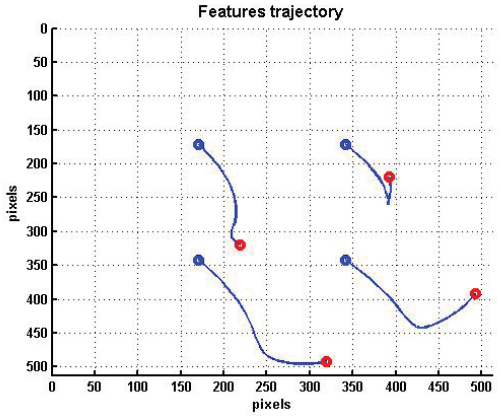
(a) Feature error for AIBVS



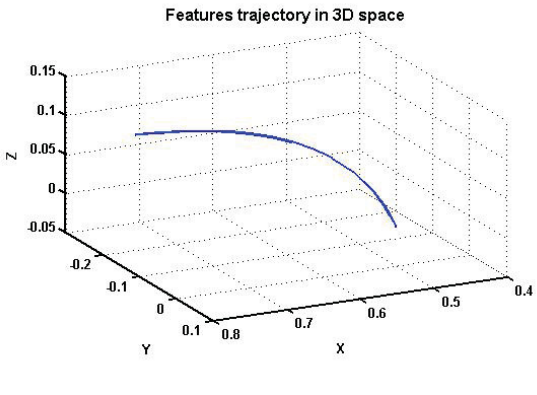
(b) Feature error for IBVS



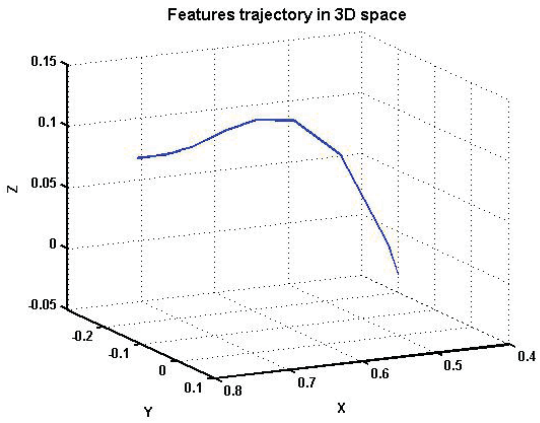
(c) Features Trajectory for AIBVS



(d) Features Trajectory for IBVS

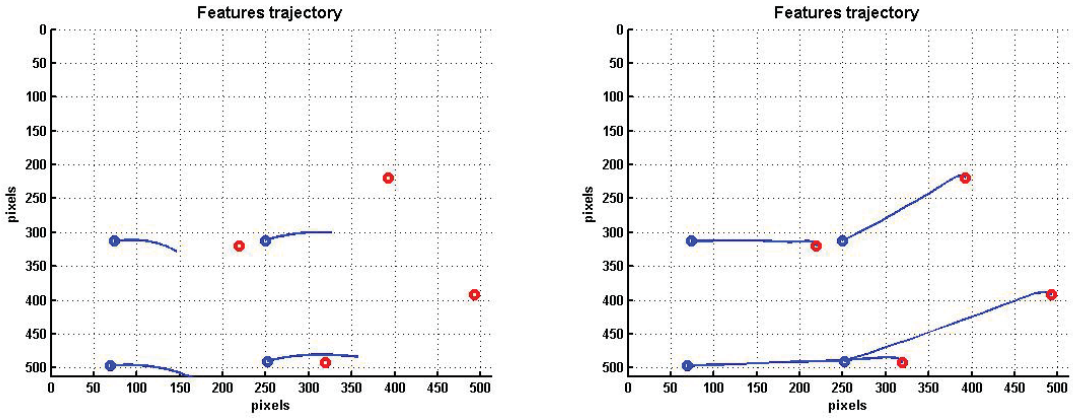


(e) Camera Trajectory for AIBVS



(f) Camera Trajectory for IBVS

Figure 2.6: AIBVS simulation results



(a) Features trajectory IBVS (b) Features trajectory for AIBVS

Figure 2.7: AIBVS simulation results for FOV test

while AIBVS kept the features inside the camera frame.

This advantage is attributed to the fact that the proposed controller creates acceleration and allows the use of a PD controller. Consequently, with the properly tuned PD gains to get an over damped response, the proposed controller generates the linear motions for the feature points without leaving field of view.

2.6.3 Experimental Results

Experimental results are presented in this section. The results of five different tests are presented here. The initial and desired configurations of the image features for each test are given in Table 2.2. As mentioned before, a constant depth value is chosen as the depth of the object with respect to the camera. In the experiments performed in this thesis, the average working distance of the robot in z_c direction is chosen as the depth of the object.

Test 1

In the first test, the convergence of each image feature point is examined when the desired location is far away from the initial one. Figure 2.8 shows the results of

Table 2.2: Initial(I) and desired(D) location of the feature points in pixel for AIBVS controller tests in pixels

		Point1		Point2		Point3		Point4	
		(x	y)	(x	y)	(x	y)	(x	y)
Test 1	I	113	82	115	107	89	109	88	83
	D	252	100	256	128	230	133	225	105
Test 2	I	200	79	196	153	126	148	129	79
	D	197	154	123	147	128	78	200	80
Test 3	I	118	167	26	163	32	91	123	93
	D	183	147	99	122	126	59	200	87
Test 4	I	107	210	16	206	26	133	114	137
	D	291	212	203	229	187	154	276	136
Test 5	I	123	149	123	189	83	188	83	148
	D	104	212	77	202	121	161	112	69

these tests.

Figure 2.8a shows the feature errors which converge to zero very smoothly without any overshoots. Figure 2.8b shows the trajectory of the features in the image space. In this figure, the image features start on the left and end on the right. It is noticed that the features are moving along an exact straight line. Figure 2.8c shows the trajectory of the camera in 3D Cartesian space.

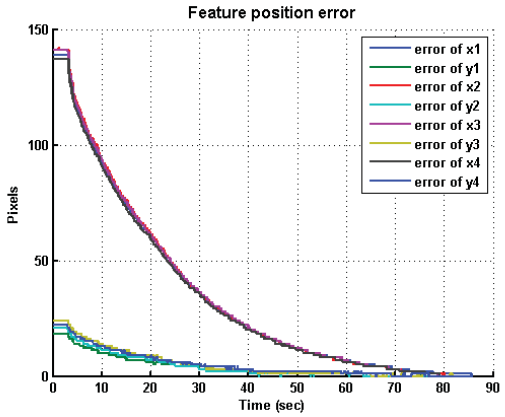
Test 2

For the second test, a pure rotation of features about the image's z_c axis by 90 degrees is performed. The initial and desired features are shown in Table 2.2. The test results are shown in Figure 2.9.

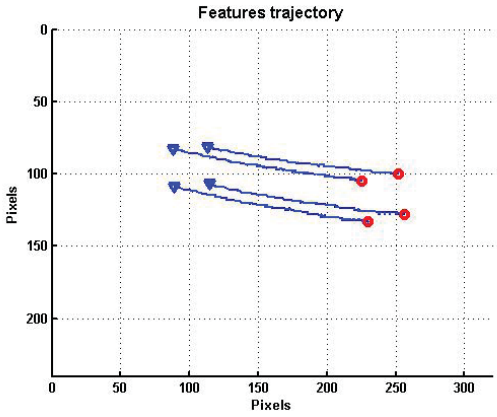
The test was successfully performed with the proposed AIBVS controller. The figure sequences are similar to those of Test 1.

Test 3

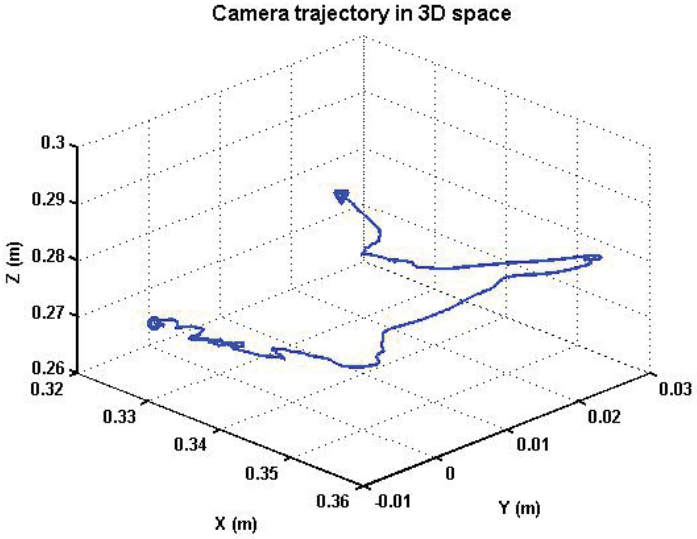
For the third test, a comparison between the proposed AIBVS and the conventional IBVS method is performed. The initial and desired locations of the feature



(a) Feature error variation in time

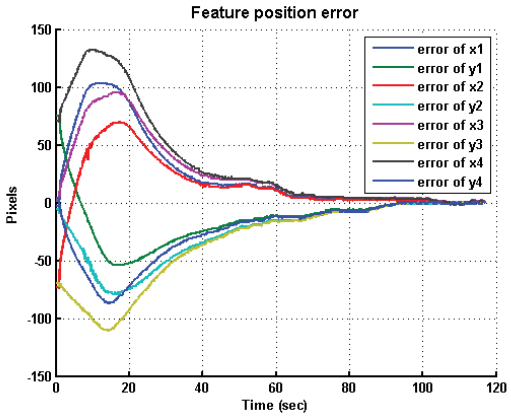


(b) Features trajectory from ▽ to ○

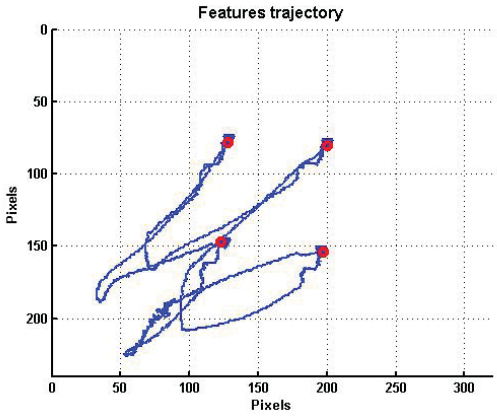


(c) 3D trajectory of the camera from ▽ to ○

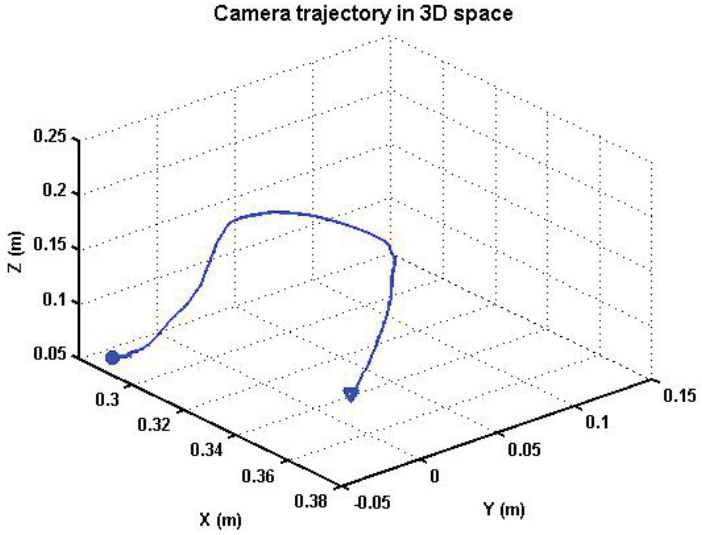
Figure 2.8: Test 1: AIBVS performance test



(a) Feature error variation in time



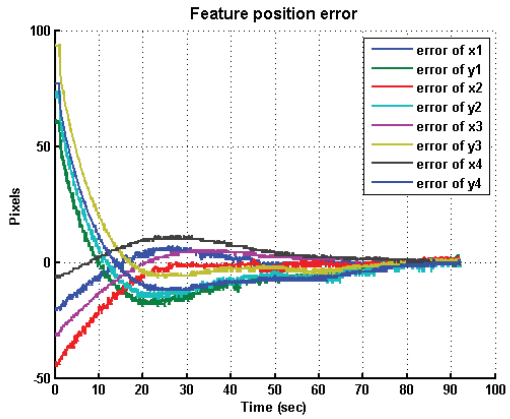
(b) Features trajectory from ▽ to ○



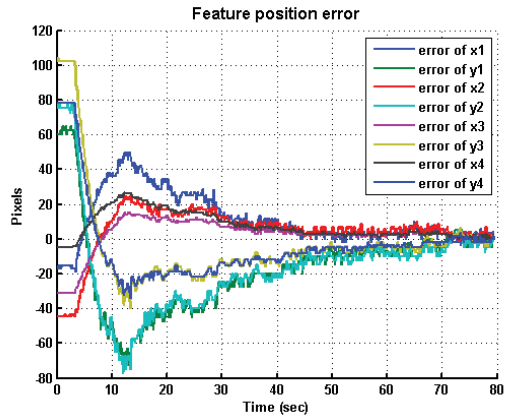
(c) 3D trajectory of the camera from ▽ to ○

Figure 2.9: Test 2: AIBVS performance test

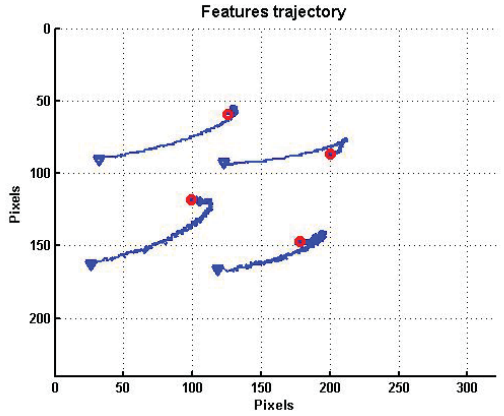
points are given in Table 2.2. All the IBVS test conditions are the same as AIBVS ones. The results of the mentioned test are given in Figure 2.10.



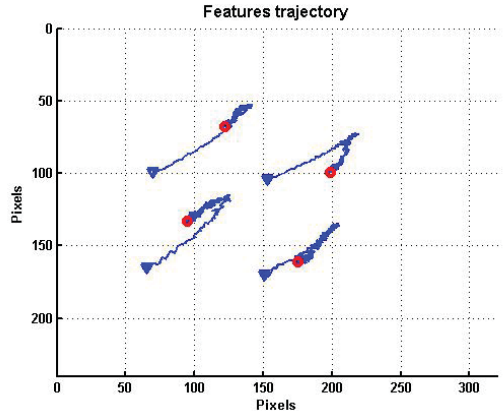
(a) Feature error variation in time for AIBVS



(b) Feature error variation in time for IBVS



(c) Features trajectory in image space for AIBVS from ∇ to $\circ$



(d) Features trajectory in image space for IBVS from ∇ to $\circ$

Figure 2.10: Test 3: Comparison of IBVS and AIBVS

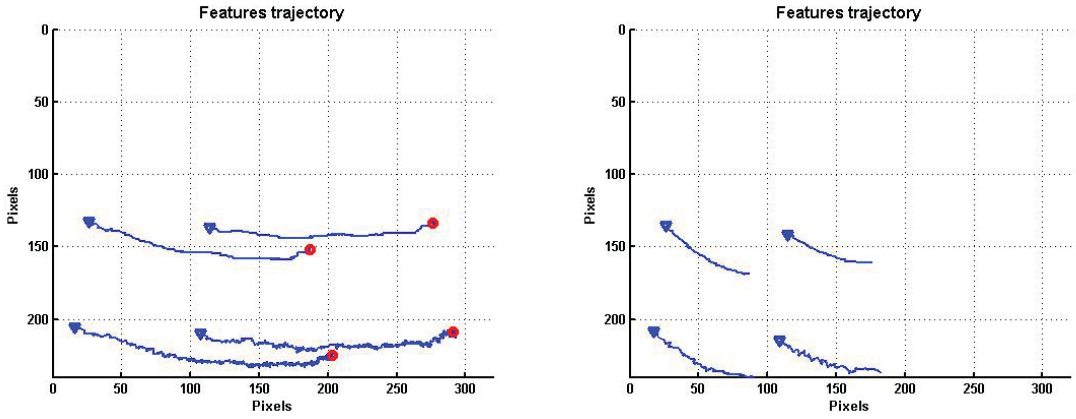
The results demonstrate that the feature error is reduced and the trajectories are smoother than those in the conventional method. The smoothness of the trajectories can be seen in the image plane trajectories. The IBVS results consist of shakes and disturbances due to the use of velocity as the controlling command.

Another benefit of using acceleration command is that two controlling parameters can be adjusted to give more flexibility for tuning the system performance. As shown in the results, the gains are adjusted to have the least overshoot when using

AIBVS. On the other hand, no specific adjustment can be made on an IBVS system and changing the gains only results in a slower or a faster response.

Test 4

This test shows that the proposed method keeps the image features trajectory on a straighter line and has a lower risk of leaving the FOV, compared with conventional IBVS. The initial and desired feature points are given in Table 2.2. Figure 2.11 illustrates the results of this test.



(a) Features for AIBVS stay inside the FOV (b) Features for IBVS leave the FOV

Figure 2.11: Test 4: Features leaving the FOV test. Trajectories start from ▽ symbol

The results validate the expected behavior of the AIBVS controller and show that, for a situation where the conventional IBVS fails to perform the visual servoing, AIBVS succeeds to complete the task.

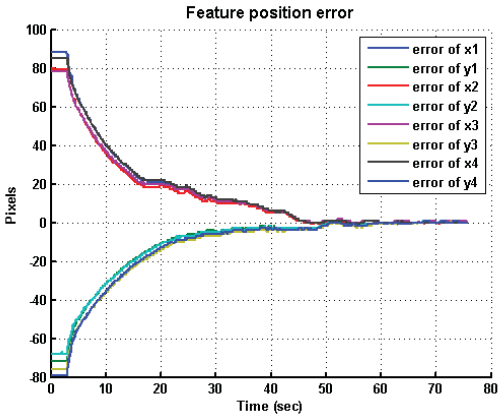
Test 5

For the last test, the robustness of the system with calibration errors is examined. The initial and desired positions of the image feature for this test are given in Table 2.2. The percentages of calibration error (CE%) affects the intrinsic

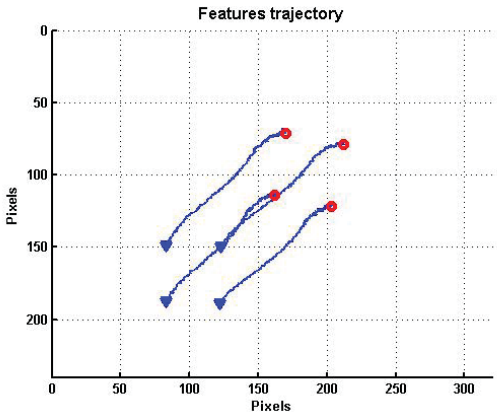
parameters matrix of the camera, $\mathbf{B}$, through the following equation

$$\mathbf{B} = \mathbf{B}_0(1 + CE\%), \tag{2.61}$$

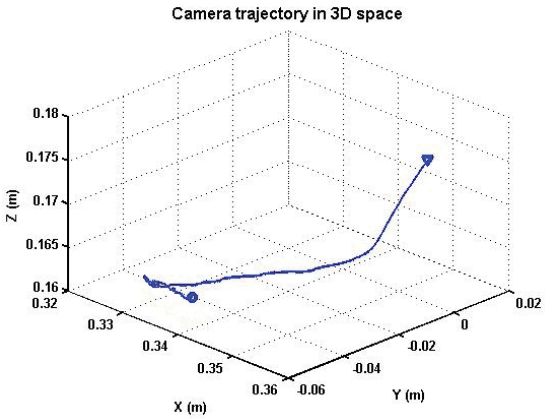
where $\mathbf{B}_0$ is the real intrinsic parameter matrix. This test was performed for different percentage of camera calibration errors. The results of the test with 50% of camera calibration error are illustrated in Figures 2.12 and 2.13 and are compared with the results of a task without any camera calibration errors.



(a) Features error



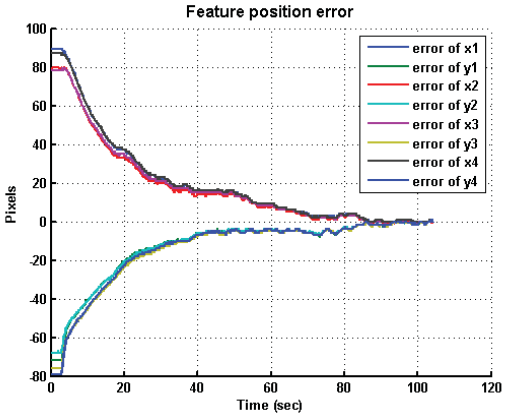
(b) Features trajectory from ▽ to ○



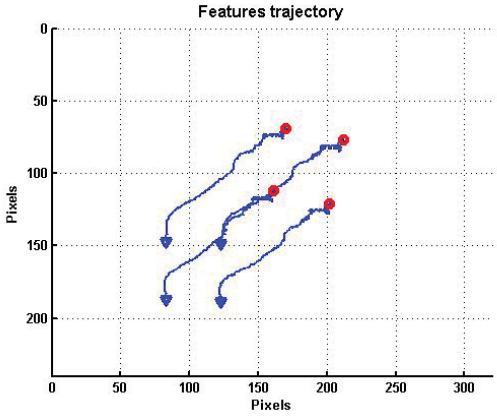
(c) Camera trajectory, from ▽ to ○

Figure 2.12: Test 5: Results for AIBVS controller test with 0% camera calibration error

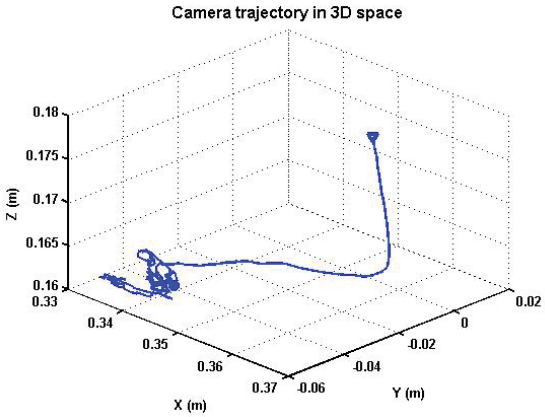
The results demonstrate the robustness of the controller against camera calibration errors up to 50%. It can also be seen that camera calibration errors have



(a) Features error



(b) Features trajectory from ▽ to ○



(c) Camera trajectory, from ▽ to ○

Figure 2.13: Test 5: Results for AIBVS controller test with 50% camera calibration error

affected the performance of the controller in terms of smoothness of trajectory and the time of convergence.

2.7 Summary

In this Chapter, an augmented version of image based visual servoing for a 6 DOFs robot is presented. A PD controller is used to create an acceleration profile for the robot controller. The stability of the visual servoing controller is proven using the Lyapanov theory for perturbed systems. Simulation and experimental tests are performed to validate the method. Results validate the efficiency of the controller and show the advantages of the proposed AIBVS over the classic IBVS in terms of smoother motion in the image space and 3D space. In the next chapter, the image moment features are adopted to the AIBVS visual servoing.

Chapter 3

AIBVS for Image Moment Features

3.1 Introduction

The controller presented in the previous chapter depicts significant improvements compared to conventional visual servoing in terms of conquering the IBVS drawbacks. However, the methodology presented, is only applied to point features. The use of point feature shows several drawbacks in practice. In some cases it is hard to define four feature points. In addition, the use of four feature points necessitates the use of an eight by six interaction matrix and creates the risk of getting stuck in local minima for the visual servoing system. In order to fully take the advantages of the AIBVS method, it should be adapted to image moment features. In this chapter, a detailed investigation on using image moment features on AIBVS controller is conducted.

Image moments are the general image features including point, line and segment form of features. Image moments were introduced to visual servoing in [6] and the general formulation of image moments interaction matrix was developed. However, the image moments were intuitively used to generate camera velocity screw in the previous researches [10, 99, 100].

In this section, the new AIBVS controller is modified to adapt to image moment features, in order to have the benefit of smooth and more reliable feature tracking responses. The kinematic equation of the new AIBVS model is derived and a PD controller is used to achieve the exponential convergence of the system. The general interaction matrix is developed for relating the image moments features to camera acceleration. Simulation results validates the performance of the AIBVS controller on image moment features.

3.2 Interaction Matrix Derivation

In this section, the analytical presentation of the interaction matrix is introduced for any order of image moment. The general definition of geometric image moment is given as follows

$$m_{ij} = \int\int_{R(t)} I(x,y) f(x,y) dxdy, \tag{3.1}$$

where $I(x,y)$ is the intensity of the image pixels, $f(x,y) = x^i y^j$, where $(i+j)$ is the image moments order, $R(t)$ is the area in the image where the object projects. This thesis only focuses on black and white images and thus the intensity within the segments is equal to one ($I(x,y) = 1$). In order to implement the AIBVS controller, it is required to find the relation between the second order time variation $\ddot{m}$ and the acceleration screw $\mathbf{A} = [\mathbf{a}, \boldsymbol{\alpha}]$, where $a = [a_x\ a_y\ a_z]$ is the acceleration vector and $\alpha = [\alpha_x\ \alpha_y\ \alpha_z]$ is the angular acceleration vector.

Using Green's theorem, Chaumette [6] developed the relation between $\dot{m}$ and the velocity screw $\mathbf{V} = [\mathbf{v}, \boldsymbol{\omega}]$. The same approach is used to construct the desired relation between the time variation of image moments with the acceleration screw. Thus, according to [6], one has

$$\dot{m}_{ij} = \int\int_{R(t)} \mathrm{div}[f(x,y)\dot{\mathbf{p}}] dxdy, \tag{3.2}$$

where $\dot{\mathbf{p}}$ is the velocity of the image point $\mathbf{p} = [x, y]$. Now considering

$$f_n(x, y) = \text{div}[f(x, y)\dot{\mathbf{p}}], \tag{3.3}$$

the second time derivative of m_{ij} could be calculated as follows

$$\ddot{m}_{ij} = \int\int_{R(t)} \text{div}[f_n(x, y)\ddot{\mathbf{p}}]dxdy. \tag{3.4}$$

In this equation, $f_n(x, y)$ is written as

$$f_n(x, y) = \frac{\partial f}{\partial x}\dot{x} + \frac{\partial f}{\partial y}\dot{y} + f(x, y)(\frac{\partial \dot{x}}{\partial x} + \frac{\partial \dot{y}}{\partial y}) \tag{3.5}$$

Substituting equation (3.5) into equation (3.4) and taking the divergence, the variation of image moment velocity in time is

$$\ddot{m}_{ij} = \int\int_{R(t)} \ddot{M}dxdy, \tag{3.6}$$

where $\ddot{M}$ is

$$\begin{aligned}
\ddot{M} = \; & \tfrac{\partial^2 f}{\partial x^2}\dot{x}\ddot{x} + \tfrac{\partial f}{\partial x}\tfrac{\partial \dot{x}}{\partial x}\ddot{x} + \tfrac{\partial f}{\partial x}\dot{x}\tfrac{\partial \ddot{x}}{\partial x} + \tfrac{\partial^2 f}{\partial y \partial x}\dot{y}\ddot{y} + \tfrac{\partial f}{\partial y}\tfrac{\partial \dot{y}}{\partial x}\ddot{y} + \tfrac{\partial f}{\partial y}\dot{y}\tfrac{\partial \ddot{y}}{\partial x} + \tfrac{\partial f}{\partial x}\ddot{x}(\tfrac{\partial \dot{x}}{\partial x} + \tfrac{\partial \dot{y}}{\partial y}) \\
& + f\tfrac{\partial \ddot{x}}{\partial x}(\tfrac{\partial \dot{x}}{\partial x} + \tfrac{\partial \dot{y}}{\partial y}) + f\ddot{x}(\tfrac{\partial^2 \dot{x}}{\partial x^2} + \tfrac{\partial^2 \dot{y}}{\partial y \partial x}) + \tfrac{\partial^2 f}{\partial x \partial y}\dot{x}\ddot{y} + \tfrac{\partial f}{\partial x}\tfrac{\partial \dot{x}}{\partial y}\ddot{y} + \tfrac{\partial f}{\partial x}\dot{x}\tfrac{\partial \ddot{y}}{\partial y} + \tfrac{\partial^2 f}{\partial y^2}\dot{y}\ddot{y} \\
& + \tfrac{\partial f}{\partial y}\tfrac{\partial \dot{y}}{\partial y}\ddot{y} + \tfrac{\partial f}{\partial y}\dot{y}\tfrac{\partial \ddot{y}}{\partial y} + \tfrac{\partial f}{\partial y}\ddot{y}(\tfrac{\partial \dot{x}}{\partial x} + \tfrac{\partial \dot{y}}{\partial y}) + f\tfrac{\partial \ddot{y}}{\partial y}(\tfrac{\partial \dot{x}}{\partial x} + \tfrac{\partial \dot{y}}{\partial y}) + f\ddot{y}(\tfrac{\partial^2 \dot{x}}{\partial x \partial y} + \tfrac{\partial^2 \dot{y}}{\partial y^2})
\end{aligned} \tag{3.7}$$

In equation (3.7), $\dot{x}$, $\ddot{x}$, $\dot{y}$ and $\ddot{y}$ can be expressed in terms of camera's velocity and acceleration using the following kinematic equations [71].

$$\dot{\mathbf{p}} = \mathbf{L_a}\mathbf{V}, \tag{3.8}$$

$$\ddot{\mathbf{p}} = \mathbf{L}_a\mathbf{A} + \mathbf{L}_v, \tag{3.9}$$

where

$$\mathbf{L}_a = \begin{bmatrix} -\frac{1}{Z} & 0 & \frac{x}{Z} & xy & -(1+x^2) & y \\ 0 & -\frac{1}{Z} & \frac{y}{Z} & 1+y^2 & -xy & x \end{bmatrix} \tag{3.10}$$

and $\mathbf{L}_v$ is obtained from equation (3.11)

$$\mathbf{L}_v = \begin{bmatrix} \mathbf{V}^{\mathrm{T}}\boldsymbol{\Omega}_x\mathbf{V} \\ \\ \mathbf{V}^{\mathrm{T}}\boldsymbol{\Omega}_y\mathbf{V} \end{bmatrix}, \tag{3.11}$$

where $\boldsymbol{\Omega}_x$ and $\boldsymbol{\Omega}_y$ can be calculated from

$$\boldsymbol{\Omega}_x = \begin{bmatrix} 0 & 0 & \frac{1}{Z^2} & -\frac{y}{Z} & \frac{3x}{2Z} & 0 \\ 0 & 0 & 0 & -\frac{x}{2Z} & 0 & -\frac{1}{2Z} \\ \frac{1}{Z^2} & 0 & \frac{2x}{Z^2} & \frac{2xy}{Z} & -\frac{1}{2Z} - \frac{2x^2}{Z} & \frac{y}{Z} \\ -\frac{y}{Z} & -\frac{x}{2Z} & \frac{2xy}{Z} & x + 2xy^2 & -\frac{y}{2} - 2x^2y & \frac{1}{2} - \frac{x^2}{2} + y^2 \\ \frac{3x}{2Z} & 0 & -\frac{1}{2Z} - \frac{2x^2}{Z} & -\frac{y}{2} - 2x^2y & 2x + 2x^3 & -\frac{3xy}{2} \\ 0 & -\frac{1}{2Z} & \frac{y}{Z} & \frac{1}{2} - \frac{x^2}{2} + y^2 & -\frac{3xy}{2} & -x \end{bmatrix} \tag{3.12}$$

and

$$\boldsymbol{\Omega}_y = \begin{bmatrix} 0 & 0 & 0 & 0 & \frac{2y}{Z} & \frac{1}{2Z} \\ 0 & 0 & \frac{1}{Z^2} & -\frac{3y}{2Z} & \frac{x}{Z} & 0 \\ 0 & \frac{1}{Z^2} & \frac{2y}{Z^2} & \frac{1}{2Z} + \frac{y^2}{Z} & -\frac{xy}{Z} & -\frac{x}{Z} \\ 0 & -\frac{3y}{2Z} & -\frac{1}{2Z} + \frac{y^2}{Z} & 2y + 2y^3 & -\frac{x}{2} - 2xy^2 & -\frac{3xy}{2} \\ \frac{1}{2Z} & \frac{x}{y} & -\frac{xy}{Z} & -\frac{x}{2} - 2xy^2 & y + 2x^2y & \frac{1}{2} + x^2 - \frac{y^2}{2} \\ \frac{1}{2Z} & 0 & -\frac{x}{Z} & \frac{3xy}{2} & \frac{1}{2} + x^2 - \frac{y^2}{2} & -y \end{bmatrix}. \tag{3.13}$$

The unknown depth Z which appears in $\mathbf{L}_a$ and $\mathbf{L}_v$, is dealt with the same way as in [6]. It is assumed that the object surface is continuous and the depth of each pixel can be written as a function of image coordinate of the point as

$$\frac{1}{Z} = \sum_{p\geq 0, q\geq 0} A_{pq} x^p y^q, \tag{3.14}$$

where A_{pq} is a constant and $p + q$ is the surface equation order. In case of a planar object, this equation could be written as zeroth and first order function of image coordinates such as

$$\frac{1}{Z} = Ax + By + C \tag{3.15}$$

Furthermore, constants A and B are equal to zero if the object surface is parallel to camera image plane and $Z = \frac{1}{C}$. In this thesis, the focus is only on planar objects. However, the interaction matrix could be derived for any continuous object surface, similarly.

Considering $f(x, y) = x^i y^j$ and substituting (3.8), (3.9) and (3.7) in (3.6), the relationship is developed between $\ddot{m}_{ij}$ and the camera acceleration $\mathbf{A}$ as follows.

$$\ddot{m}_{ij} = \mathbf{L}_{ma_{ij}}\mathbf{A} + L_{mv_{ij}} \tag{3.16}$$

where

$$\mathbf{L}_{ma_{ij}} = \begin{bmatrix} m_{ax} & m_{ay} & m_{az} & m_{\alpha x} & m_{\alpha y} & m_{\alpha z} \end{bmatrix} \tag{3.17}$$

and

$$\begin{aligned} m_{ax} &= -i(Am_{ij} + Bm_{i-1,j+1} + Cm_{i-1,j}) - Am_{ij} \\ m_{ay} &= -j(Am_{i+1,j-1} + Bm_{ij} + Cm_{i,j-1}) - Bm_{ij} \\ m_{az} &= (i + j + 3)(Am_{i+1,j} + Bm_{i,j+1} + Cm_{ij}) - Cm_{ij} \\ m_{\alpha x} &= (i + j + 3)m_{i,j+1} + jm_{i,j-1} \\ m_{\alpha y} &= -(i + j + 3)m_{i+1,j} - im_{i-1,j} \\ m_{\alpha z} &= im_{i-1,j+1} - jm_{i+1,j-1}. \end{aligned} \tag{3.18}$$

In equation (3.16), $L_{mv_{ij}}$ is called the velocity interaction term and it could be written in the form of

$$\mathbf{L}_{mv_{ij}} = \mathbf{V^T}\mathbf{O}_{m_{ij}}\mathbf{V} \tag{3.19}$$

where the elements of $\mathbf{O}_{m_{ij}}$ is given in the Appendix B.

3.2.1 Three Basic Image Moments

In this section, the interaction matrix of three basic image moments, the zeroth and first order moments, are introduced. These moments will be used to define

centroid and normal moments. Selecting $i = 0$ and $j = 0$, the zeroth order moment which is the area of the image could be calculated.

$$a = m_{00} \tag{3.20}$$

Selecting $i = 1$ and $j = 0$ or $i = 0$ and $j = 1$ the first order moment could be calculated which gives the centroid of the image if it is divided by the image area.

$$\begin{aligned} x_g &= \tfrac{m_{10}}{m_{00}} \\ y_g &= \tfrac{m_{01}}{m_{00}} \end{aligned} \tag{3.21}$$

Using the general form of interaction matrix $\mathbf{L}_{ma_{ij}}$ and $\mathbf{L}_{mv_{ij}}$ introduced in (3.18) and (3.19) the interaction matrix and $\mathbf{L}_{mv_{ij}}$ of these basic moments could be found as follows.

$$\begin{aligned} \mathbf{L}_{ma_{00}} &= \begin{bmatrix} -aA & -aB & a(\frac{3}{Z_g} - C) & 3sy_g & -3ax_g & 0] \end{bmatrix} \\ \mathbf{L}_{ma_{x_g}} &= \begin{bmatrix} -\frac{1}{Z_g} & 0 & x_{g_{az}} & x_{g_{\alpha z}} & x_{g_{\alpha z}} & y_g \end{bmatrix} \\ \mathbf{L}_{ma_{y_g}} &= \begin{bmatrix} 0 & -\frac{1}{Z_g} & y_{g_{az}} & y_{g_{\alpha z}} & y_{g_{\alpha z}} & -x_g \end{bmatrix} \end{aligned} \tag{3.22}$$

where $\frac{1}{Z_g} = Ax_g + By_g + C$ and

$$\begin{cases} x_{g_{az}} = xg/Z_g + 4(An_{20} + Bn_{11}) \\ y_{g_{az}} = yg/Z_g + 4(An_{11} + Bn_{02}) \\ x_{g_{\alpha x}} = -y_{g_{\alpha y}} = x_g y_g + 4n_{11} \\ x_{g_{\alpha y}} = -(1 + x_g^2 + 4n_{20}) \\ y_{g_{\alpha x}} = 1 + y_g^2 + 4n_{20} \end{cases} \tag{3.23}$$

where n_{ij} is the normalized moments of order 2, which is defined as $n_{ij} = \mu_{ij}/m_{00}$, μ_{ij} is called centered moments and the definition is given in equation (3.24). Accordingly, equation (3.19) becomes

$$\mathbf{L}_{mv_{00}} = \mathbf{V}^{\mathbf{T}} \mathbf{O}_{m_{00}} \mathbf{V} \tag{??}$$

The first order moments x_g and y_g are used to control the x and y motion of the robot camera which is mounted on the end-effector [10]. The zeroth order moment a is used to control the z motion of the camera [10].

3.3 Interaction Matrix of Central Moments

Centered moments are introduced to possess the property of invariance to x and y translation. Centered moments are also called central geometric moments [101] and are defined as;

$$\mu_{ij} = \int \int_{R(t)} f^{'}(x,y)dxdy, \tag{3.24}$$

where $f^{'}(x,y) = (x - x_g)^i(y - y_g)^j$.

The interaction matrix and the velocity interaction term for the central moments are derived similarly. Starting by taking the second time derivative of μ_{ij} in equation (3.24) results in

$$\ddot{\mu}_{ij} = \int \int_{R(t)} \ddot{M}' dxdy \tag{3.25}$$

where $\ddot{M}'$ can be derived as $\ddot{M}$ in (3.6). After performing the same mathematical manipulation, the relation between $\ddot{\mu}$ and camera acceleration is given as follow;

$$\ddot{\mu}_{ij} = \mathbf{L}_{\mu a_{ij}}\mathbf{A} + L_{\mu v_{ij}} \tag{3.26}$$

where

$$\mathbf{L}_{\mu a_{ij}} = \begin{bmatrix} \mu_{ax} & \mu_{ay} & \mu_{az} & \mu_{\alpha x} & \mu_{\alpha y} & \mu_{\alpha z} \end{bmatrix} \tag{3.27}$$

and

$$\begin{aligned}
\mu_{ax} = & -(i+1)A\mu_{ij} - iB\mu_{i-1,j+1} \\
\mu_{ay} = & -jA\mu_{i+1,j-1} - (j+1)B\mu_{ij} \\
\mu_{az} = & -A\mu_{\alpha y} + B\mu_{\alpha x} + (i+j+2)C\mu_{ij} \\
\mu_{\alpha x} = & (i+j+3)\mu_{i,j+1} + ix_g\mu_{i-1,j+1} \\
& +(i+j+3)y_g\mu i, j - 4in_{11}\mu_{i-1,j} - 4n_{02}\mu_{i,j-1} \\
\mu_{\alpha y} = & -(i+j+3)\mu_{i+1,j} - (2i+j+3)x_g\mu_{ij} \\
& -jy_g\mu_{i+1,j-1} + 4in_{20}\mu_{i-1,j} + 4jn_{11}\mu_{i,j-1} \\
\mu_{\alpha z} = & i\mu_{i-1,j+1} - j\mu_{i+1,j-1}.
\end{aligned} \tag{3.28}$$

The velocity interaction term, $L_{mv_{ij}}$, could be written in the form of

$$L_{\mu v_{ij}} = \mathbf{V^T O}_{\mu_{ij}} \mathbf{V} \tag{3.29}$$

where

$$\mathbf{O}_{\mu_{ij}} = \int\int_{R(t)} \overline{\mathbf{O}}_{\mu_{ij}} dxdy \tag{3.30}$$

where $\overline{\mathbf{O}}_{\mu_{ij}}$ is given in the Appendix B. One of the important image features used in visual servoing and computer vision is the picture orientation which is given by

$$\theta_z = \frac{1}{2}\arctan(\frac{2\mu_{11}}{\mu_{20} - \mu_{02}}) \tag{3.31}$$

It consists of three second-order centered image moments. According to (3.27) and (3.28), the interaction matrix of each moment and the overall interaction matrix could be calculated as

$$\mathbf{L}_{\mu a_{ij}} = \begin{bmatrix} \theta_{vx} & \theta_{vy} & \theta_{vz} & \theta_{\alpha x} & \theta_{\alpha y} & -1 \end{bmatrix} \tag{3.32}$$

where

$$\begin{cases} \theta_{vx} = a_m A + b_m B \\ \theta_{vy} = -c_m A - a_m B \\ \theta_{vz} = -A\theta_{\alpha y} + B\theta_{\alpha x} \\ \theta_{\alpha x} = -b_m x_g + a_m y_g + d_m \\ \theta_{\alpha y} = a_m x_g - c_m y_g + e_m \end{cases} \tag{3.33}$$

and

$$\begin{cases} a_m = \mu_{11}(\mu_{20} + \mu_{02})/D_m \\ b_m = (2\mu_{11}^2 + \mu_{02}(\mu_{02} - \mu_{20}))/D_m \\ c_m = (2\mu_{11}^2 + \mu_{20}(\mu_{20} - \mu_{02}))/D_m \\ d_m = 5(\mu_{12}(\mu_{20} - \mu_{02}) + \mu_{11}(\mu_{03} - \mu_{21}))/D_m \\ e_m = 5(\mu_{21}(\mu_{02} - \mu_{20}) + \mu_{11}(\mu_{30} - \mu_{12}))/D_m \\ D_m = (\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2 \end{cases} \tag{3.34}$$

3.4 Moment Features for Visual Servoing

Six different image moments are required to control the six DOFs of the camera. Four basic moments were introduced in [10] to control the camera's translation along X, Y, Z and rotation about Z. These moments are x and y image centroids, image area and image angle which are calculated as $x_g = m_{10}/m_{00}$, $y_g = m_{01}/m_{00}$, $m_a = m_{00}$ and $\theta = \frac{1}{2}\arctan(\frac{2\mu_{11}}{\mu_{20}-\mu_{02}})$. Finding the two image moment features relating to the other two DOFs which are the rotations about X and Y is the most challenging part of the moment feature selection. Hu's invariant moments[102] were used to develop the new moments for the fourth and fifth DOFs. Chaumette presented a set of moments consisting of the four basic moments and two moments combinations for these two DOFs [6]. These two moments are

$$\begin{cases} P_x = I_1/I_3^3 \\ P_y = aI_2/I_3^3 \end{cases} \tag{3.35}$$

where

$$\begin{cases} I_1 = c_1^2 + s_1^2 \\ I_2 = c_2^2 + S_2^2 \\ I_3 = \mu_{20} + \mu_{02} \end{cases} \quad \text{and} \quad \begin{cases} c_1 = \mu_{20} - \mu_{02} \\ c_2 = \mu_{03} - 3\mu_{21} \\ s_1 = 2\mu_{11} \\ s_2 = \mu_{30} - 3\mu_{12} \end{cases} \tag{3.36}$$

However, since the interaction matrix of P_x and P_y is zero for a centered symmetrical object, Chaumette presented another invariant to overcome this problem [6].

$$\begin{cases} S_x = (c_2c_3 + s_2s_3)/K \\ S_y = (s_2c_3 - c_2s_3)/K \end{cases} \tag{3.37}$$

where

$$\begin{cases} K = I_1 I_3^{(3/2)} \\ c_3 = c_1^2 - s_1^2 \\ s_3 = 2s_1c_1 \end{cases} \tag{3.38}$$

Later on, Liu[7] introduced two improved image moments for these two DOFs as follows

$$\begin{cases} M_x = 0.1 - (c_1c_2 + s_1s_2)/I_7^{\frac{9}{4}} \\ M_y = (s_1c_2 - c_1s_2)/I_7^{\frac{9}{4}} \end{cases} \tag{3.39}$$

where for small objects $I_7 = I_3$ and for large object $I_7 = c_1$.

3.5 Controller Design

In order to control all six DOFs of the camera, six moments from [99] will be adopted in AIBVS controller. Stacking the selected moments in a vector format, $\boldsymbol{\xi}_m$ is defined as the vector of image moments and the general system error is defined as follows.

$$\boldsymbol{\epsilon_m} = \boldsymbol{\xi}_{m_d} - \boldsymbol{\xi}_m \tag{3.40}$$

where $\boldsymbol{\xi}_{m_d}$ is the desired image moments vector. Recalling equation (3.16) and (3.26), the general equation of motion for the six selected image moments could be written similarly.

$$\ddot{\boldsymbol{\xi}} = \mathbf{L}_{ma_6}\mathbf{A} + \mathbf{L}_{mv_6} \tag{3.41}$$

where $\mathbf{L}_{ma_6}$ is the interaction matrix created by stacking each moment in interaction matrix and $\mathbf{L}_{mv_6}$ is the velocity interaction vector generated by stacking the velocity interactions corresponding to each moment. An augmented visual servoing controller [71] is designed to control the system. A PD compensator is used to decrease the error according to the following second order exponential decrease;

$$\ddot{\boldsymbol{\epsilon}} + \kappa_{m_v}\dot{\boldsymbol{\epsilon}} + \kappa_{m_p}\boldsymbol{\epsilon} = 0, \tag{3.42}$$

where κ_{m_p} and κ_{m_v} are positive constants used as the proportional and derivative gains of the controller. Thus, the proposed controlling law is

$$\mathbf{A}_c = \mathbf{L}_{ma_6}^{-1}(-\lambda_v\dot{\boldsymbol{\epsilon}} - \lambda_p\boldsymbol{\epsilon} - \mathbf{L}_{mv_6}), \tag{3.43}$$

where $\mathbf{A}_c$ is the acceleration screw command. The interaction matrix $\mathbf{L}_{ma_6}$ is a function of objects depth Z. Due to the lack of depth information in monocular visual servoing, different approaches could be used to calculate the interaction matrix. One popular approach is to use the interaction matrix for the desired moments for the whole visual servoing process. Thus

$$\mathbf{A}_c^* = \mathbf{L}_{ma_6}^{*-1}(-\lambda_v\dot{\boldsymbol{\epsilon}} - \lambda_p\boldsymbol{\epsilon} - \mathbf{L}_{mv_6}), \tag{3.44}$$

where $\mathbf{L}^{*-1}_{ma_6}$ is the interaction matrix of the desired position. The other approach is to use an estimation of the object depth and assume the object remains parallel to the camera in the whole process. For this method it could be written

$$\hat{\mathbf{A}}_c = \hat{\mathbf{L}}^{-1}_{a_6}(-\lambda_v \dot{\boldsymbol{\epsilon}} - \lambda_p \boldsymbol{\epsilon} - \mathbf{L}_{v_6}), \tag{3.45}$$

where $\hat{\mathbf{L}}_{a_6}$ is the interaction matrix calculated with the estimation of object depth. In Equations (3.44) and (3.45), it is assumed that the object is parallel to the camera frame at all times and thus in the interaction matrix $A = 0$ and $B = 0$. In this thesis, the later approach is utilized which also showed a better performance in the simulation results.

3.6 Experimental Results

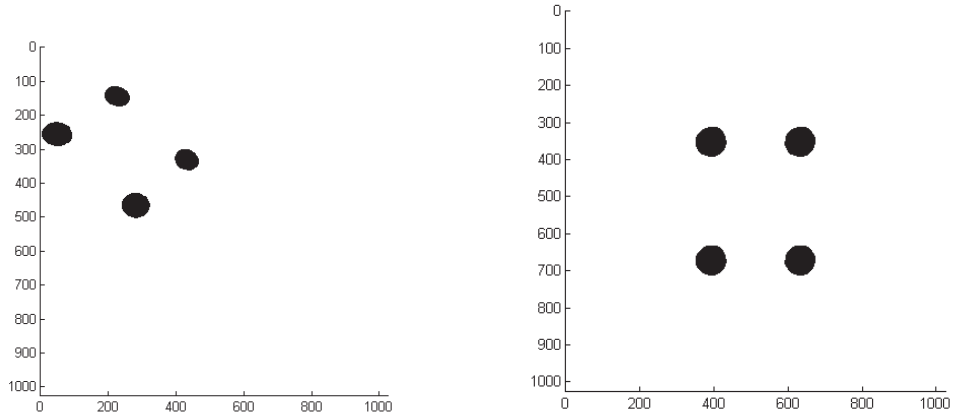
Four tests are carried out to validate the performance of the controller.

Test1:

In the first test, AIBVS controller is tested on a symmetrical image. The moment features used for this purpose are the four basic moment features given in section 3.4 and the Chaumette moments for a centered symmetrical shape given in (3.37) are used to control the rotations about X and Y axes. The image features are four circular points. The initial and desired images are given in Figures 3.1a and 3.1b. The initial displacement of the camera with respect to the desired pose is $T = [-9(cm), 9(cm), 0(cm), 45(deg), 20(deg), 20(deg))]$. The interaction matrix is updated at each iteration using the following equation.

$$\hat{\mathbf{L}}_{a_6} = \frac{1}{2}(\mathbf{L}^{||}{}_{a_6} + \mathbf{L}^{*}{}_{a_6}) \tag{3.46}$$

where $\mathbf{L}^{||}{}_{6a}$ is the interaction matrix at each instant assuming the camera is parallel to the object, $\mathbf{L}^{*}{}_{a_6}$ is the interaction matrix for the image when the camera is in its



(a) Initial image (b) Velocity field of v_y motion

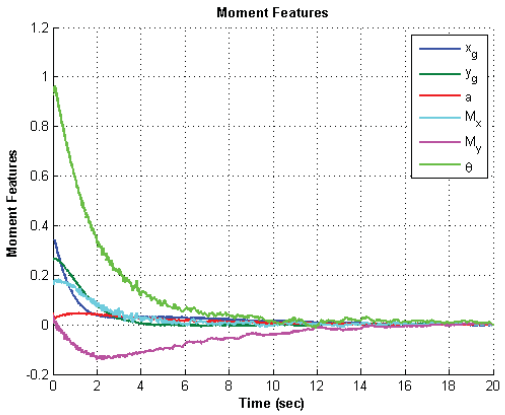
Figure 3.1: Initial and desired images for Test 1

desired position. The interaction matrix for the desired image in this test is given as follows.

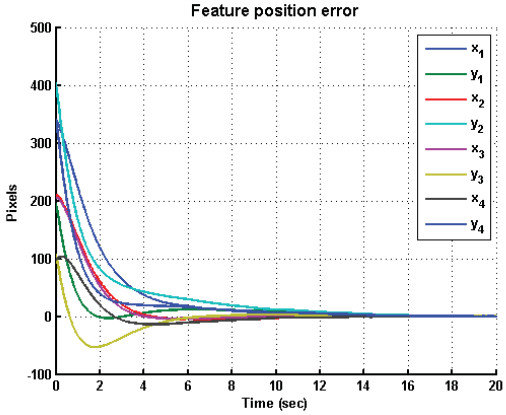
$$\mathbf{L}^* = \begin{bmatrix} -1 & 0 & 0 & 0 & -1.18 & 0 \\ 0 & -1 & 0 & 1.1 & 0 & 0 \\ 0 & 0 & 4.67 & 0.01 & -0.01 & 0 \\ 0 & 0 & 0 & -0.5564 & -0.01 & 0 \\ 0 & 0 & 0 & 0.002 & 0.97 & 0 \\ 0 & 0 & 0 & 0.001 & -0.0052 & -1 \end{bmatrix} \tag{3.47}$$

It can be seen that this matrix is almost diagonal. This makes the matrix invertible and far from singularity. However using the instant interaction matrix ($\mathbf{L}^{||}{}_{a_6}$) in each iteration could possibly make the resulting interaction matrix singular. However, singularity have not been experienced during this research. Furthermore, matrix $\mathbf{L}_{v_6}$ is also calculated for each instant. This matrix is a function of camera velocity and it is zero at the desired position.

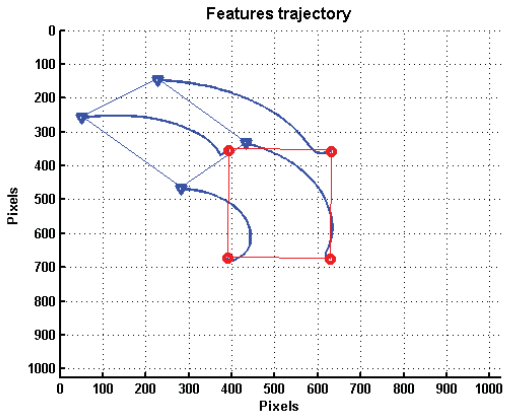
One important point in tuning the controller gains is that, since the units of moments are different from each other, using a scalar value as the gains in equation (3.45) does not lead to the good performance. Therefore, a diagonal matrix is used



(a) Feature error



(b) Point error



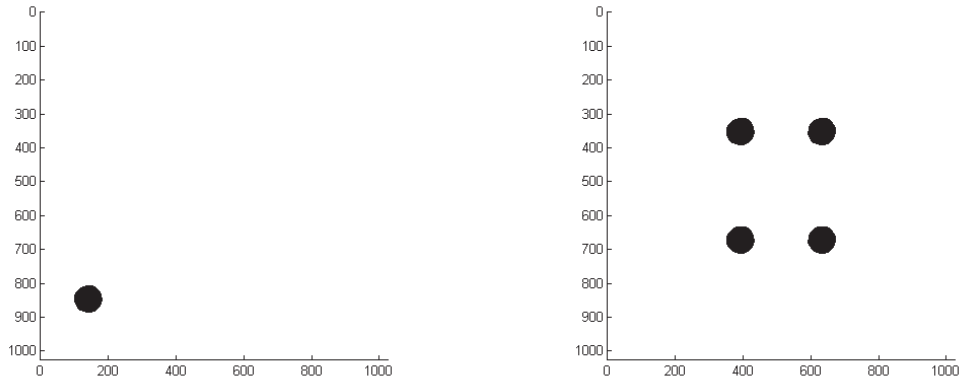
(c) 2D image trajectory of the points from

Figure 3.2: Test 1 results with Chaummete feature moments for asymmetrical images [6]

as the controller gain with different positive values on the diagonal elements. The diagonal element values are adjusted manually by a trial and error method to get the best possible results. The optimum gains could be found using optimization techniques. The results of this test are presented in Figures 3.2a to 3.2c. These figures show the stability of the system and the smooth elimination of the errors.

Test2:

In the second test, the ability of this algorithm is tested in stabilizing the system when some of the features are out of the field of view. The same test using



(a) Initial image (b) Desired image

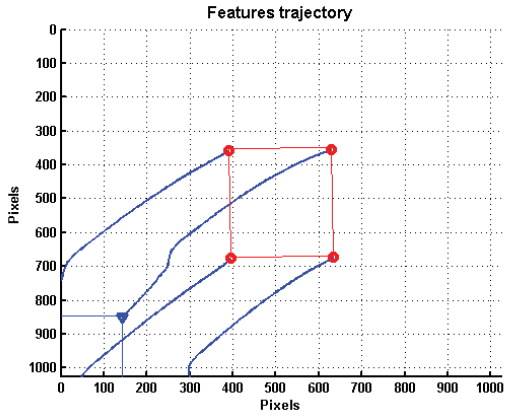
Figure 3.3: Initial and desired images for Test 2

the point features is performed to show the advantage of moment features over point features. The system is started in a situation where three of the feature points are out of the field of view. The initial position of the features are shown in Figure 3.3a. The desired position of the features is the same as that in the first test (Fig. 3.3b). The comparison results are shown in Figure 3.4.

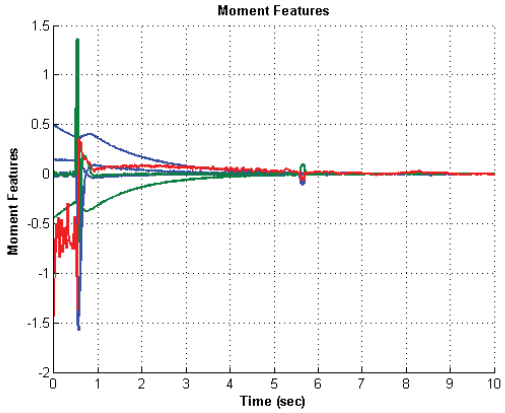
The results show that, using point features the visual servoing fails when some features are out of field of view. However, using image moment as image feature can solve the problem. When only one point feature is visible, the moment of this point feature can still be calculated. Thus no lack of data occurs. As the other points appear in the field of view, a sudden change happens in the image moment calculation. This explains the sudden jump of the moments error in Figure 3.4b.

Test 3:

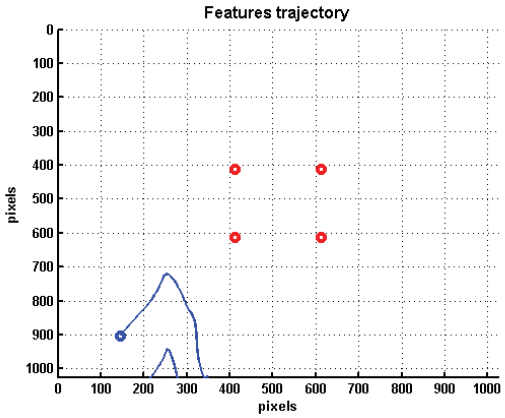
In the third test, the AIBVS controller is tested with an object with asymmetrical shape. The moment features used for this test consist of the four basic moments along with the Chuamette moments for asymmetrical object given in equation (3.35). The initial and desired image for this test are shown in Figures 3.5a and 3.5b. The



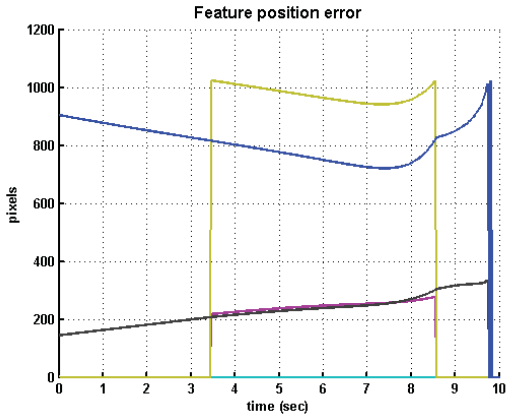
(a) Feature trajectory using image moment features from ▽ to ○



(b) Point errors



(c) Failed feature trajectory using point features from ▽ to ○

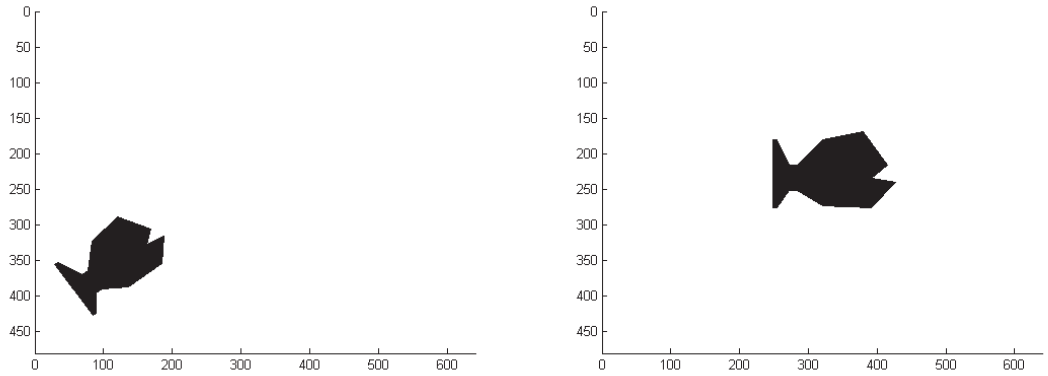


(d) Point errors

Figure 3.4: Test 2, Comparing moment feature with point features when they start beyond the field of view

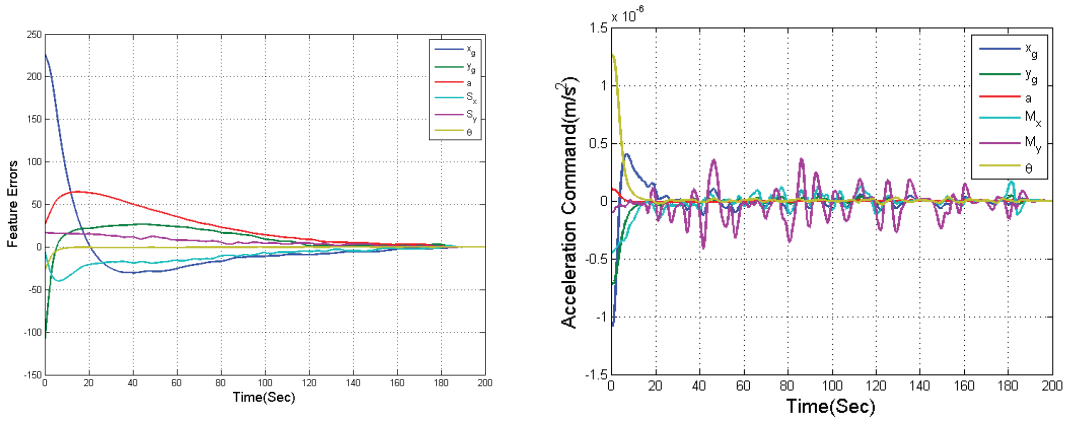
interaction matrix for the desired point is calculated as follows.

$$\mathbf{L}^* = \begin{bmatrix} -1 & 0 & 0 & 0 & -1.01 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 2.77 & -0.02 & 0.03 & 0 \\ 0 & 0 & 0 & 0.018 & 0.054 & 0 \\ 0 & 0 & 0 & -0.013 & -0.039 & 0 \\ 0 & 0 & 0 & 0.02 & -0.005 & -1 \end{bmatrix} \tag{3.48}$$



(a) Initial image (b) Desired image

Figure 3.5: Initial and desired images for Test 3



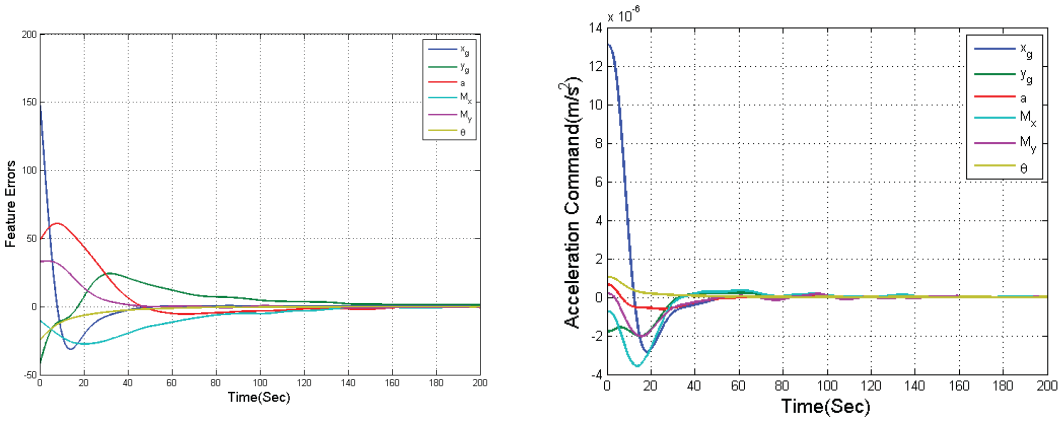
(a) Feature error (b) Acceleration command

Figure 3.6: Test 3, with Chaummete feature moments for symmetrical images [6]

The initial displacement of the camera with respect to the desired feature is $T = [17(cm), 7(cm), 0(cm), 30(deg), 30(deg), 30(deg))]$. The results of this test are shown in Figures 3.6a and 3.6b.

Test 4:

In Test four, the same four basic moments are used along with the Liu's moments as the fourth and fifth moments, which are given in equation (3.39). The same initial and desired conditions as those in the third test are used for this test.



(a) Feature error (b) Acceleration command

Figure 3.7: Test 4, with Liu's feature moments [7]

Using Liu's moments the interaction matrix is calculated as follows.

$$\mathbf{L}^* = \begin{bmatrix} -1 & 0 & 0 & 0 & -1.18 & 0 \\ 0 & -1 & 0 & 1.1 & 0 & 0 \\ 0 & 0 & 4.67 & 0.01 & -0.01 & 0 \\ 0 & 0 & 0 & 0.007 & 0.028 & 0 \\ 0 & 0 & 0 & -0.001 & -0.0189 & 0 \\ 0 & 0 & 0 & 0.017 & -0.017 & -1 \end{bmatrix} \tag{3.49}$$

The results of Test 4 are shown in Figures 3.7a and 3.7b. The results show the ability of the AIBVS controller using different sets of moments in fulfilling the visual servoing task.

3.7 Summary

In this chapter, a new AIBVS controller is proposed using image moment features in visual servoing tasks. The visual servoing kinematic model is developed and the interaction matrix relating the image moment features to the acceleration screw is derived. A PD control law is developed based on the system equation. The controller is tested on three different sets of moment features. The results show the

improved performance of the AIBVS controller by using moment features compared to the AIBVS controller by using point features.

Chapter 4

Catching Moving Objects Using AIBVS Controller and Navigation Guidance Technique

4.1 Introduction

One of the demands of visual servoing in robotic systems arises in the application with unpredicted environments, especially while dealing with non-stationary target objects [103]. A good example of such environments can be a production line with products moving on a conveyor, where the robot can manipulate the unsorted objects. Furthermore, it gives the robot the ability to catch the items moving on a conveyor. Other examples of such situations worth mentioning is catching moving object in space using space robots where lack of gravity can not guarantee that the object stays still.

Catching a moving object with a robot becomes a challenging problem in visual servoing which needs a well-designed trajectory planning algorithm. Various methods and strategies have been developed for this matter. These strategies can

also be classified under two main classes of visual servoing, Position Based and Imaged Based, which are listed as follows,

1. Position Based Methods
    (a) Trajectory regeneration methods[52–55]
    (b) Potential field methods[56]
    (c) Navigation guidance methods[57]
2. Imaged Based Methods
    (a) Potential Field methods

Trajectory regeneration methods are the most basic methods used for catching moving objects. The basic strategy of this approach is to plan an initial trajectory from the current position of the robot's end-effector to the potential catching position of the object and revising the trajectory as the object moves and changes its position and velocity [61]. Since the position of the object is changing, an object trajectory estimation is used to predict future position and velocity of the object.

Researchers have also taken advantage of trajectory planning methods in image space for guiding the robot to a stationary desired position[104]. Park et al. in [69] presents a trajectory planning algorithm in image space using stereo vision visual servoing without performing the 3D reconstruction. An imaged based trajectory planning is presented in [68] aiming at avoiding the obstacles without 3D reconstruction of environment. However, the research on the use of image based trajectory regeneration methods for catching moving objects is rarely reported in literature due to their low speed and high processing demand.

Potential field methods have been also used as a powerful trajectory planning tool in visual servoing. Potential field is utilized both in position based strategies[56] and image based strategies [58]. In [26] potential field along with a hybrid switching control strategy is used to avoid image singularities and local minima and to

guide the robot toward the target while keeping it away form obstacles and other constraints in real time. However, the summation of attracting and repulsive forces in potential field algorithm could bring the system to local minima[58].

Furthermore, high speed demand of catching moving objects needs for navigation guidance techniques. Navigation guidance is a basic technique for interception of free flying objects with fast maneuvering. The primary application of these methods is to guide the missiles to intercept targets. In these methods, interception happens by reducing the distance between interceptor and object and guiding the interceptor to a collision with the target by applying an acceleration vector to the interceptor, disregarding the interception condition[57]. Four various navigation guidance strategies has been reported; Proportional Navigation Guidance(PNG), Augmented Proportional Navigation Guidance(AIPNG), Ideal Proportional Navigation Guidance(IPNG) and Augmented Ideal Proportional Navigation Guidance (AIPNG)[61]. These navigation guidance methods were designed for target interception and not for smoothly catching objects. However, due to their high speed, researchers started to improve these methods and apply to such applications. Mehrandezh et al. used an ideal proportional navigation guidance method for a robotic interception task [105]. Later on Dongkyoung et al. proposed a modified technique for IPNG for smoothly catching fast maneuvering objects considering robotic torque and velocity constraints[106]. In [107], this method is expanded to 3D space by using a 5 DOFs robotic manipulator.

In this chapter, as an application to the developed AIBVS controller, this controller is tested on a visual servoing task for catching an object. A combination of navigation guidance technique with image based visual servoing system is practiced. The navigation guidance speed is used to track and catch a fast maneuvering moving object smoothly in visual servoing. The path created by the navigation guidance will be followed by a previously developed AIBVS controller, which can achieve smooth

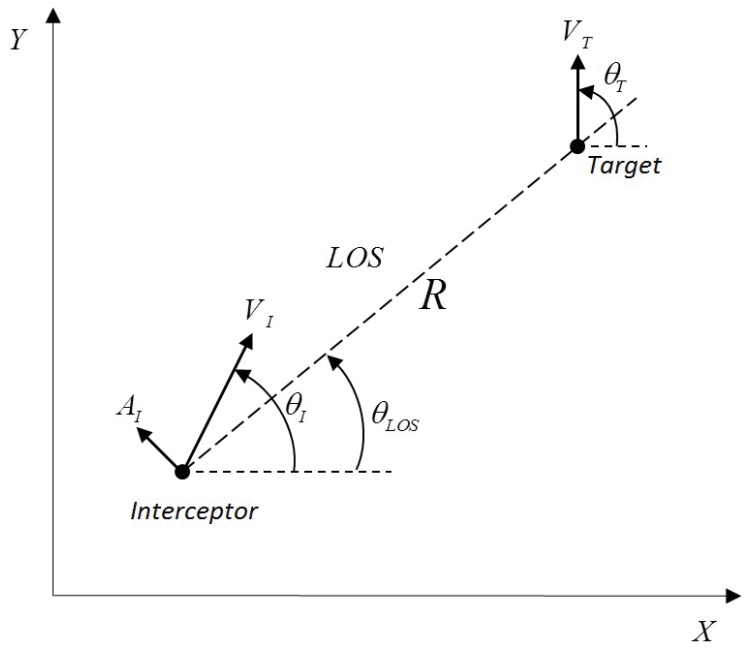


Figure 4.1: A schematic diagram of a interception geometry

and linear feature trajectory in image space and owns the robustness with respect to camera and robots calibration errors. Finally, the simulation results are presented to validate the effectiveness of the proposed controller.

4.2 Navigation Guidance Planning

Navigation guidance law produces an acceleration command according to interceptor and target velocity vectors. Figure 4.1 shows a schematic diagram of an interceptor following the target. In this figure, $\mathbf{V}_I$ and $\mathbf{A}_I$ are the interceptor velocity and acceleration vectors respectively. $\mathbf{V}_T$ and $\mathbf{A}_T$ are the target's velocity and acceleration vector respectively. $\boldsymbol{\theta}_I$ and $\boldsymbol{\theta}_T$ are the angle of interceptor and targets velocity. Line of Sight (LOS) is the line connecting the interceptor and the target, and $\boldsymbol{\theta}_{LOS}$ is the angle of LOS and R is the length of LOS.

In order to perform a smooth catching, the acceleration command is divided into two commands including tangential and normal commands. The following

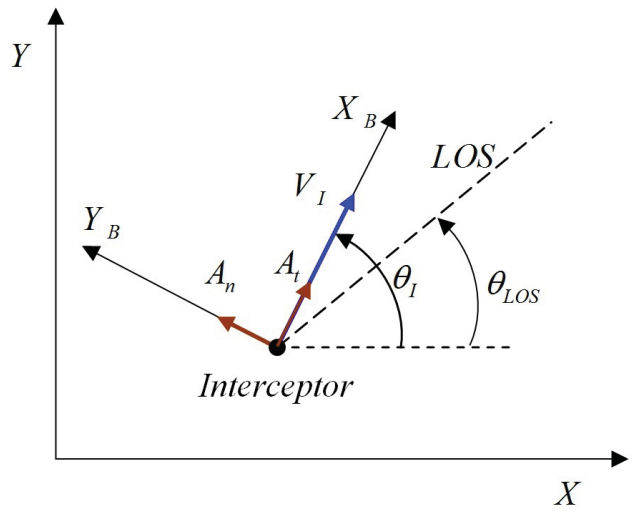


Figure 4.2: Applying acceleration commands to the interceptor

guidance laws will be used for this purpose [106].

$$A_{It} = k_d \dot{R} + k_p R, \tag{4.1}$$

$$A_{In} = k_1(\dot{\theta}_{Los} - \dot{\theta}_t) + k_2 \sin(\theta_{Los} - \theta_t), \tag{4.2}$$

where R is the distance between the object features and the target features and $\dot{R}$ is its derivative, k_d and k_p are the derivative and proportional gains for the tangential acceleration command, k_1 and k_2 are the gains used in the normal acceleration command. Thus, the desired motion of the features can be calculated as

$$\ddot{\xi}_d^2 = A_{It}^2 + A_{In}^2. \tag{4.3}$$

Equation (4.1) works as a PD controller to reduce the distance between the interceptor and the target. Equation (4.2) reduces the deviation of the interceptor velocity with respect to the target. These equations will be applied to the interceptor as shown in Figure 4.2. It is important to distinct between the target feature points and the desired feature points. The target points are the points when the catching happens if it matches the current points, but the desired points are the mid way points that the trajectory planner produces in each sequence to navigate the current

points toward the target points. There are four feature points in the image plane and thus a guidance law is written for each point.

The guidance law should be planned in the image space. The target features are stationary in the image space with respect to the camera frame. Thus, from an observer attached to the camera frame, the planned guidance law will affect the motion of interceptor. This is like fixing the interceptor and applying a negative acceleration to the target with the same amount of $\mathbf{A}_I$. Consequently, we can assume that a negative acceleration command is applied to the image feature instead of the desired image feature. We use this vector to calculate the required acceleration and apply the negative amount of this to the camera to perform the navigation motion. Figure 4.3 shows a diagram of this assumption.

Integrating the acceleration command twice, gives the desired position of the features for the next sequences .

$$\begin{aligned} \dot{\mathbf{s}}_{d4} &= \int_0^t \ddot{\mathbf{s}}_{d4}\, dt + \dot{\mathbf{s}}_{4i}, \\ \mathbf{s}_{d4} &= \int_0^t \dot{\mathbf{s}}_{d4}\, dt + \mathbf{s}_{4i}, \end{aligned} \tag{4.4}$$

where $\mathbf{s}_{4i}$ and $\dot{\mathbf{s}}_{4i}$ are the initial position and velocity of the object features.

Using these desired positions and their derivative, as the desired features in the AIBVS control law, the required end-effector acceleration command required to catch the object can be calculated.

The system block diagram is shown in Figure 4.4. The camera is mounted on the robot end-effector (eye-in-hand) and provides the current location of the object's features. The trajectory planner produces midpoint desired points along the desired trajectories. These mid-point desired features are used to produce the required end-effector's acceleration screw. The AIBVS controller is used for this purpose. Finally this acceleration is converted to joint acceleration and speeds using the inverse Jacobian matrix of the manipulator.

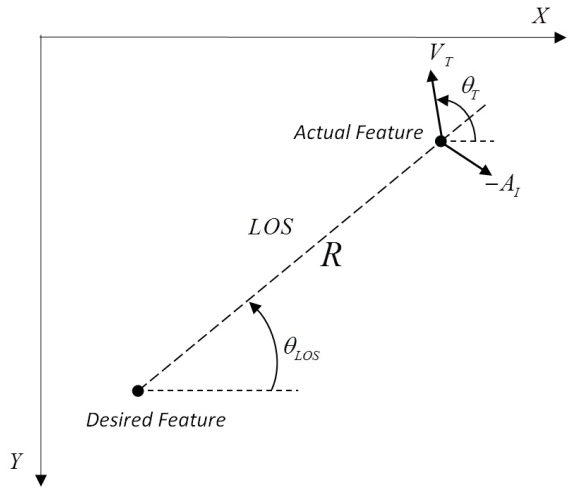


Figure 4.3: Interception plan in the image plane

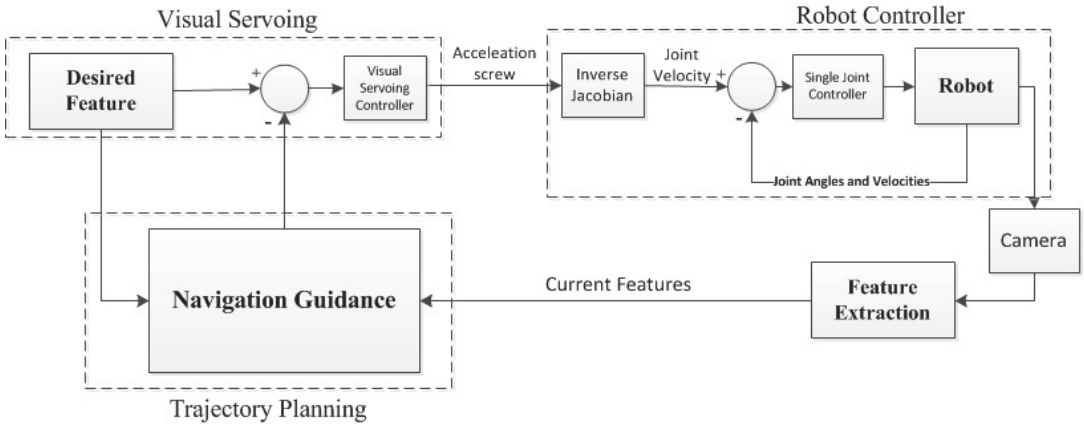


Figure 4.4: System Block Diagram

4.3 Simulation Results

To validate the effectiveness of the proposed algorithms, three different catching simulations have been carried out in this section, catching an object with constant velocity, catching an object moving in a sinusoidal path and catching an object thrown in the air. The experimental setup presented in the introduction is modeled in the simulation. The object is assumed to be a cube with a length of 10(cm) on each side. Each corner of the cube face makes a feature point.

Test 1: Object with Constant Velocity

For this test we assume the target object is moving with a constant velocity on y direction. Figures 4.5a and 4.5 show the results of this simulation.

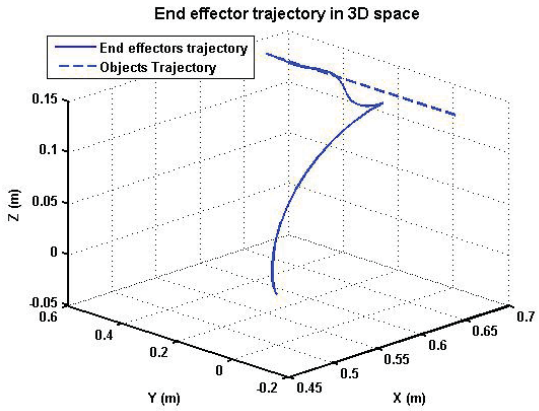
Figure 4.5a shows the 3D trajectory of the end-effector and the 3D trajectory of the target object. It is noticed that catching has happened smoothly. The feature error defined as the difference between the desired features and the current features, is decreasing gradually as shown in Figure 4.5b. The features trajectories in image plane are shown in figure 4.5c. Figures 4.5d and 4.5e show the tangential and normal acceleration command used to perform this action.

Test 2: Object with Sinusoidal Motion

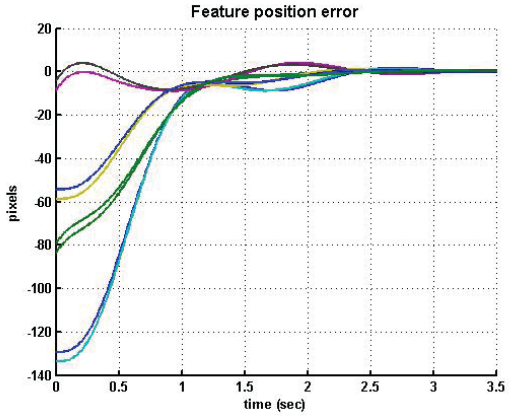
In this test, the catching algorithm is tested with an object with a sinusoidal motion. This test could better show the performance of the catching algorithm for following and smoothly catching the object with unstable motion (Figure 4.6a). However this test can reveal the deficiencies of this algorithm in catching object because of unnecessarily pursuing the object. As it can be seen in Figure 4.2, the robot's end-effector makes an unnecessary motion in order to follow the object as it goes up and down. This problem could be solved by estimating the objects path and choosing an interception point and velocity in the estimated path. The robot then should be guided to that point and velocity to perform the interception. This can be proposed as the future work in this area. Simulation results for this test is given in Figure 4.6.

Test 3: Thrown Object (Parabolic Motion)

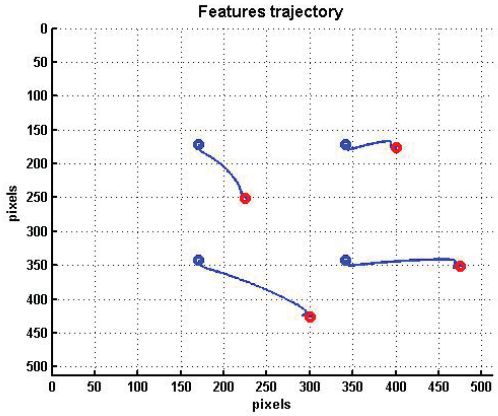
One of the most interesting goals in catching algorithms is to perform a catching for a thrown objects. This task requires a high speed data acquisition and high speed respond. A position based high speed catching have been reported in [108].



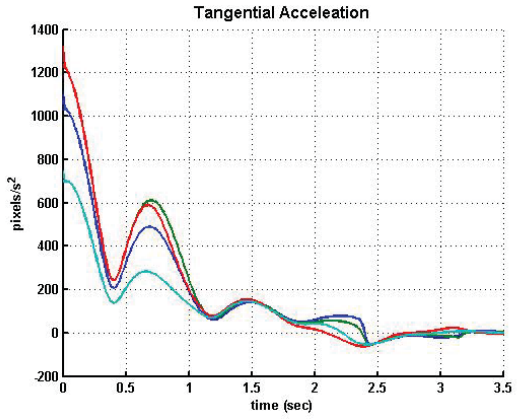
(a) 3D end-effector and object trajectory



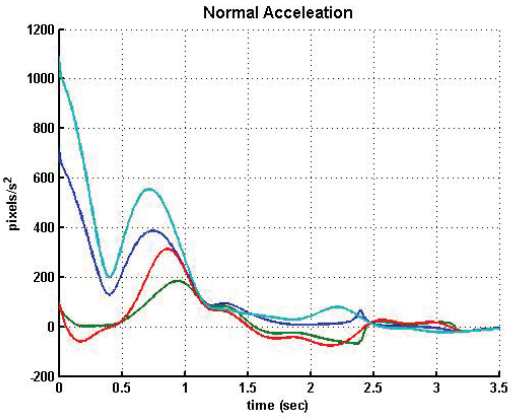
(b) Feature errors



(c) Feature Trajectory

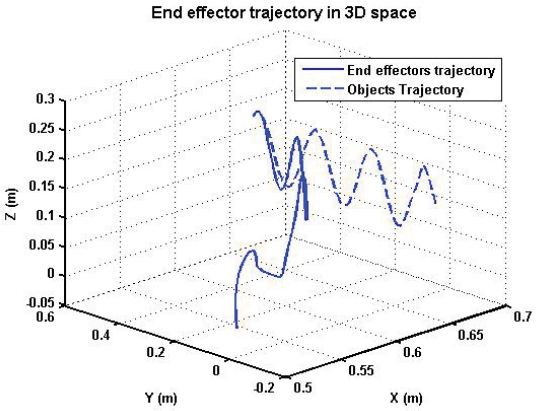


(d) Tangential Accelerations

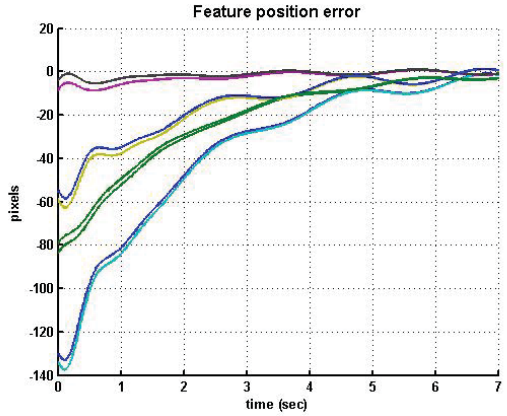


(e) Normal Accelerations

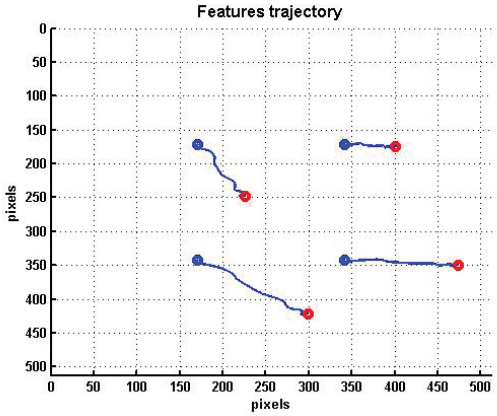
Figure 4.5: Results for catching an object with constant velocity



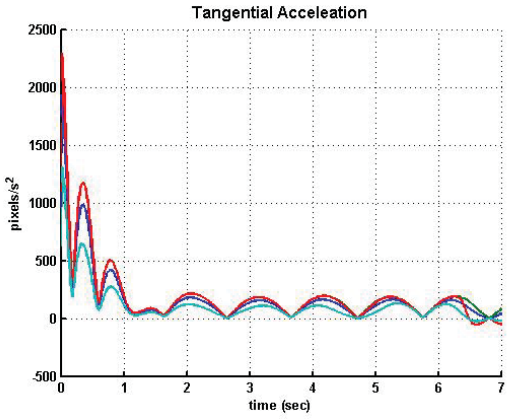
(a) 3D end-effector and object trajectory



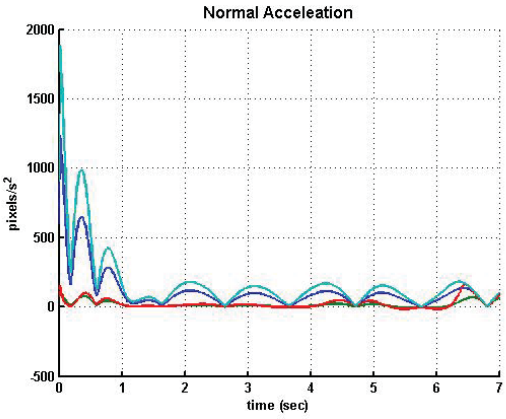
(b) 3D end-effector and object trajectory



(c) Feature Trajectory



(d) Tangential Accelerations



(e) Normal Accelerations

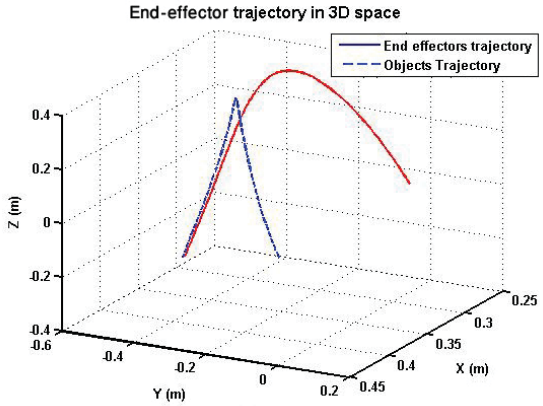
Figure 4.6: Results for catching an object with sinusoidal motion

Besides, the robot needs to be actuated fast enough to be able to follow the planned trajectory. A test has been done to catch a thrown ball and the result is shown in Figures 4.7. The unnecessary following of the object is again observed in this test, which causes a delays in catching and consumes a lot of energy. This also causes the robot to catch the object in the final reachable area in the robot's working space.

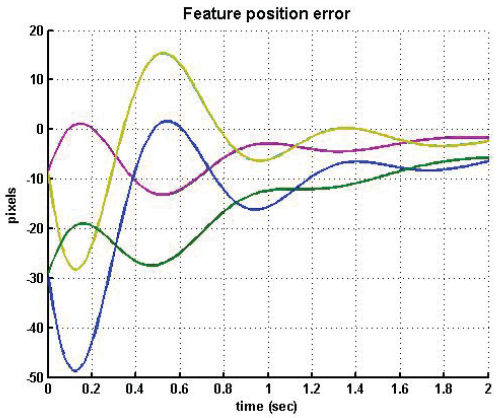
The results show that the proposed methodology succeeded in catching the object smoothly. However, the performance for more complicated motions of the object, such as sinusoidal motion, is not as good as that for simple motions. Yet, the methodology is fast enough to catch a thrown ball.

4.4 Summary

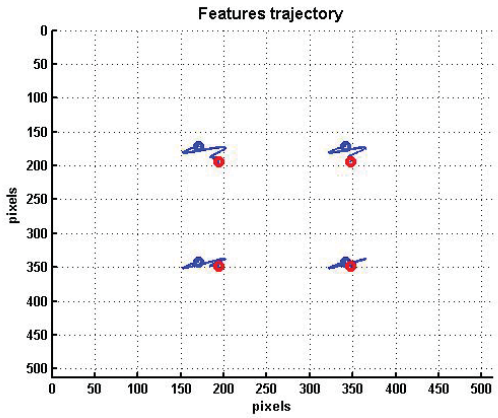
In this chapter, a navigation guidance algorithm is modified and applied to a robot to follow and catch a moving target using vision feedback information. The navigation guidance algorithm is combined with an AIBVS strategy to perform catching of a moving object with different motions. Simulation results exhibit the effectiveness of this method. However, it still needs some improvement in order to overcome some drawbacks such as unnecessary following of the moving objects which appears mostly in fast maneuvering targets. In future works, artificial intelligent algorithm will be used to estimate the trajectory of the object, to solve the problem of extra motion of the robot.



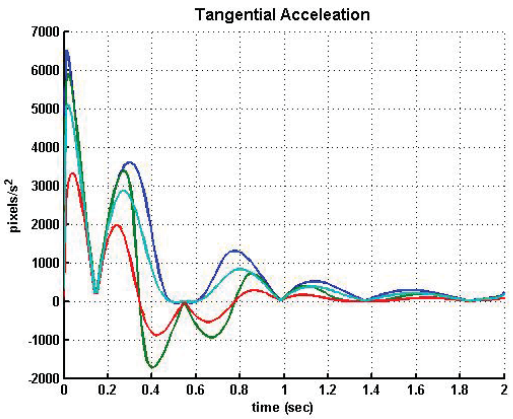
(a) 3D end-effector and object trajectory



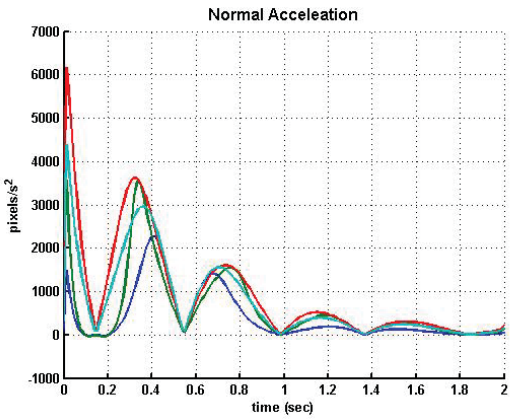
(b) Feature Errors



(c) Feature Trajectory



(d) Tangential Accelerations ▽ to ○



(e) Normal Accelerations ▽ to ○

Figure 4.7: Results for catching an object with parabolic motion

Chapter 5

Visual Servoing Using Trajectory Planning

5.1 Introduction

In the previous chapters, a visual servoing controller is developed to perform visual servoing tasks. However, as mentioned in the stability analysis, the developed controllers could only guarantee the stability of the system in a local region around the desired location. Moreover, despite a great amount of development in visual servoing technologies, in the last two decades, visual servoing still suffers from a number of problems which prevent it from wide industrial use. In [28], some potential problems of implementing visual servoing are presented. Overall, the most prominent deficiencies of visual servoing, preventing it from practical employment, are listed as follows,

1. Instability of system in long distance tasks
2. Interaction matrix singularity
3. Local minima
4. Instability of the system in tasks with rotation of 180^o about camera's center

5. Having no control on the speed of the robot during the visual servoing task
6. Unknown path of the robot prior to the tasks
7. Features leaving the Field of View (FOV).

Various methodologies have been presented in literature to overcome the deficiencies. Image moment features were introduced to deal with the interaction matrix singularity and local minima problem [6, 10, 72, 109]. Model predictive visual servoing controller was introduced to deal with the constraints of the system and to prevent the features from leaving the FOV [36]. AIBVS was developed to make the visual servoing smoother and reduce the risk of features leaving the field of view [70]. Although, lots of researches have been devoted to solve one or two of the above mentioned problems, a reliable and general solution to all of these problems can not be found in literature.

Combining visual servoing with trajectory planning techniques is a possible solution to overcome the above mentioned problems. Chesi et al. [65] proposed a trajectory planning method for position based visual servoing. Homogeneous forms were used to parametrize the path and an LMI optimization is carried out to calculate the parameters. Moreover other techniques were used in PBVS path planning[66]. An adaptive trajectory regeneration method was proposed in [110] for visual servoing in an unstructured environment. Later on navigation guidance techniques were integrated with visual servoing to achieve fast visual servoing [78, 107]. Potential field methods were used to perform on-line trajectory planing in robotic systems[56, 111]. Potential filed techniques were useful for trajectory planning in the presence of obstacles and when the system is subjected to constraints.

Although the reported trajectory planning techniques demonstrate good performance in executing visual servoing tasks, they were designed for position based visual servoing. Thus, they suffer from PBVS drawbacks such as sensitivity to model and camera calibration errors. This gap motivated the researchers[26, 58] to develop

a trajectory planning technique in an IBVS system.

Usually, in an IBVS trajectory planning, a reference path is produced by the trajectory planner considering the goals and constraints of the image and the robot. A small controlling error will be defined for each small segment of the trajectory to be followed by the IBVS controller. In such algorithms, the main challenge is to find a path in image space which corresponds to a feasible path in task space. The most basic method developed for solving this problem is using stereo vision and the epipolar geometry constraint between two camera images. Utilizing the privilege of epipolar geometry, an image trajectory is generated on both images in a way that corresponds to a feasible or even straight line trajectory in Cartesian space [68, 69]. High load of processing and decreased field of view is the problems of such solution.

In this chapter, a new image based trajectory planning algorithm is proposed to overcome the visual servoing deficiencies and develop a reliable algorithm to perform visual servoing tasks. In this method, the camera's velocity screw is separated into elements. Each velocity element is parameterized using a time based function which is refereed to as the velocity profiles. The velocity profile parameters are determined through an optimization process which minimizes the features errors. In order to facilitate and speed up the optimization technique, some new image features are introduced. A convexity analysis is performed to show the convexity of the optimization problem. Similar to other IBVS systems, depth estimation plays an important role in the performance of the proposed trajectory planning algorithm. A depth estimation technique is introduced. Having the initial depth, the object depth could be integrated during the visual servoing task. By integrating all these techniques, the proposed IBVS based trajectory planning can overcome the above mentioned deficiencies to a great extent.

The trajectory planning is developed in two stages. First a trajectory planning algorithm is developed for a 4 DOFs robot. This algorithm is then extended to a 6

DOFs robot. Due to the highly coupled behavior of the features in a 6 DOFs robot, the algorithm used for a 4 DOFs robot turns into a non-convex problem for a 6 DOFs robot. By decoupling the orientation planning from positioning problem, the optimization problem becomes a convex problem again.

Due to some uncertainties in the system or calibration errors, it is probable that the generated trajectory does not exactly take the robot to its desired location. However, it is observed that with such incomplete trajectories the robot is taken to a position which is close enough to the desired location. The desired location will then be reached using an AIBVS controller. In other words, the trajectory planning algorithm is switched to a controller at the end of its path to compensate for any inaccuracy of the system performance. In summary, the whole visual servoing procedure consist of 3 stages. The first stage is the depth estimation stage. The second stage is the trajectory planning stage. Finally, in the third stage the trajectory planning block switches to a visual servoing controller block. Each stage is elaborated in the following sections.

Simulation and experimental tests are performed to validate the proposed method. The results show that in the situations where the visual servoing task fails using traditional methods, the proposed method successfully perform it.

5.2 Trajectory Planning for a 4 DOFs Robot

In this section, the goal is to develop a trajectory planning algorithm for an imaged based visual servoing task. A 4 DOFs robot manipulator is used to perform such task. The robot end-effector has 3 linear motion in x_e, y_e and z_e axes and a rotation about z_e axis. The picture of the robot manipulator is illustrated in Figure 5.1. A pinhole CCD camera is attached to the end-effector of the robot to form an eye-in-hand configuration. The object is stationary in the workspace and

Figure 5.1: Denso robot

is characterized by 4 feature points on its four corners. A picture is taken from the object when the camera is located in a known relative position with respect to the object. The features coordinates in this picture are used as the target image features. The visual servoing task is complete when the image features match the target features.

The generated trajectory profile will be designed using the initial and target features positions and the velocity relation between the features and the camera. Thus, the velocity relation between the camera and the image features is required. For a 4 DOFs robot with a camera attached at its end effector, the velocity screw of the camera is in the following form.

$$\mathbf{V}_{c_4} = \begin{bmatrix} v_x \\ v_y \\ v_z \\ \omega_z \end{bmatrix}, \tag{5.1}$$

where, v_x, v_y, v_z and ω_z are linear velocities in x_c, y_c, z_c direction and angular

velocity about z_c, respectively. The features velocity relation to camera velocity screw for a 4 DOFs robot is given as follows [71].

$$\dot{\mathbf{p}} = \mathbf{L}_s \mathbf{V_c}, \tag{5.2}$$

where

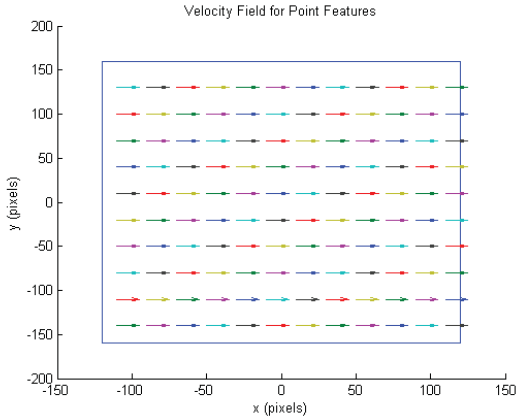
$$\mathbf{L}_s = \begin{bmatrix} -\frac{1}{Z} & 0 & \frac{x}{Z} & y \\ 0 & -\frac{1}{Z} & \frac{y}{Z} & -x \end{bmatrix} \tag{5.3}$$

is the interaction matrix and Z is the depth of the object with respect to the camera.

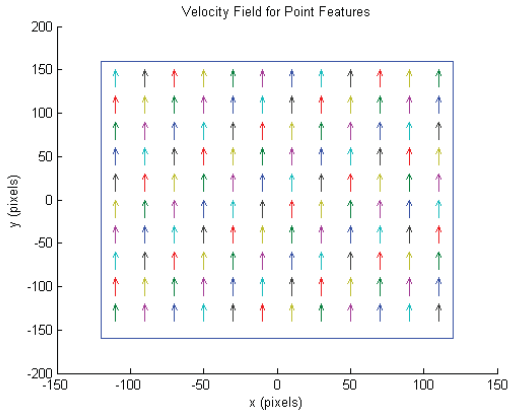
5.2.1 Path Planning

The robot's end-effector needs to have a movement in each DOFs to reach its desired destination. Each of these motions moves the image features on a specific path in image space. Knowing the relation between the velocity screw and features velocity, the features path could be calculated by integrating equation (5.2). Figure 5.2 shows the velocity field of a feature point created using each of the velocity screw elements.

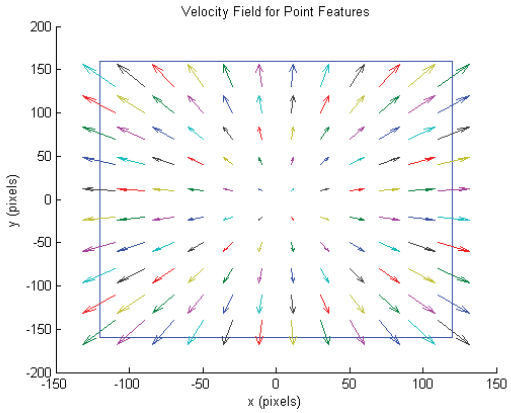
The first two elements of the velocity screw create linear motions in the same direction for all features (Figures 5.2 (a) and (b)). These two camera motions are used for displacing the features in x and y direction of the image plane. A camera motion in z_c direction creates an outward motion for the features which are in the direction of the line connecting the center of the image to the image feature. A negative motion in z_c direction will create an inward motion for the features. This motion could compensate the distances between the features. Subsequently, the fourth element of the velocity screw, which is rotation about camera's z_c axis, will cause the features to rotate about the center of image. The features velocity vector has a greater magnitude in the features further from the center of image with the same velocity of the camera. This motion is used to correct the angle of the features with respect to the image plane coordinates.



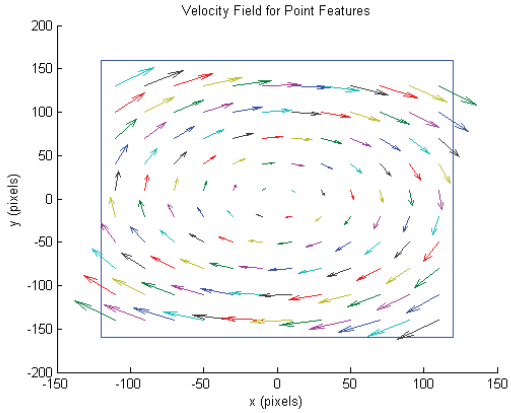
(a) Velocity field for v_x motion



(b) Velocity field of v_y motion



(c) Velocity field of v_z motion



(d) Velocity field of ω_z motion

Figure 5.2: Velocity field of the features subject to camera velocities

The concept behind the trajectory planning is that any target features could be reached by using a combination of shown feature motions. Four predefined velocity profiles are associated to each of the camera's velocity screw elements. The effect of the generated velocity screw can be calculated using equation (5.2). In other words, by superposing the velocity fields caused by each element of the velocity screw, the final position of the features could be calculated. The parameters of the camera velocity are then determined by minimizing the error between the image features and the target ones.

Having the velocity screw in equation (5.1), the features velocity in image space is given by

$$\begin{aligned} \dot{x}_i &= \tfrac{-1}{Z} v_x + \tfrac{x_i}{Z} v_z + y_i \omega_z, \\ \dot{y}_i &= \tfrac{-1}{Z} v_y + \tfrac{y_i}{Z} v_z - x_i \omega_z, \end{aligned} \tag{5.4}$$

where $\dot{x}_i$ and $\dot{y}_i$ are the velocities of the ith image feature in x and y direction, respectively. Consequently, the image feature position could be calculated as

$$\begin{aligned} x_{it} &= \textstyle\int_{t_0}^{t} (\tfrac{-1}{Z(t)} v_x(t) + \tfrac{x_i(t)}{Z(t)} v_z(t) + y_i(t)\omega_z(t)) dt + x_{i0}, \\ y_{it} &= \textstyle\int_{t_0}^{t} (\tfrac{-1}{Z(t)} v_y(t) + \tfrac{y_i(t)}{Z(t)} v_z(t) - x_i(t)\omega_z(t)) dt + y_{i0}, \end{aligned} \tag{5.5}$$

where x_{i_0} and y_{i_0} are the initial coordinates of the image features and x_{it} and y_{it} are the locations of the image features at time t. Knowing the position of the image feature, the cost function (i.e. objective function) is defined as the square of error between the image features at final time t_f and the target image features.

$$OF = (\boldsymbol{\xi}(t_f) - \boldsymbol{\xi}_d)^{\mathrm{T}} (\boldsymbol{\xi}(t_f) - \boldsymbol{\xi}_d) \tag{5.6}$$

where OF is the objective function, and $\boldsymbol{\xi}(t_f)$ and $\boldsymbol{\xi}_d$ are the feature vector at the end of trajectory and desired feature vector, respectively. Point features positions could be an acceptable feature set to solve this problem as given in equations 5.7 and 5.8.

$$\xi(t) = \begin{bmatrix} x_1(t) & y_1(t) & \dots & x_i(t) & y_i(t) \end{bmatrix}^{\mathrm{T}}, \tag{5.7}$$

and

$$\xi_d = \begin{bmatrix} x_{1_d} & y_{1_d} & \dots & x_{i_d} & y_{i_d} \end{bmatrix}^{\mathrm{T}}. \tag{5.8}$$

However, a better set of feature is introduced here to facilitate the optimization process. The new set of feature is given as follows,

$$\xi_{\mathbf{n}}(t) = \begin{bmatrix} x_c(t) & y_c(t) & p_z(t) & \theta_z(t) \end{bmatrix}^{\mathrm{T}}, \tag{5.9}$$

where $x_c(t)$ and $y_c(t)$ are the centers of the feature points, $p_z(t)$ is the perimeter of the lines connecting each consecutive feature point and $\theta_z(t)$ is the angle of the whole object picture relative to x coordinate of image. Considering Figure 5.3 as an image taken from the object, the selected new features can be calculated as follows;

$$\begin{aligned} x_c(t) &= \tfrac{\sum_{i=1}^{4} x_i(t)}{4} \\ y_c(t) &= \tfrac{\sum_{i=1}^{4} y_i(t)}{4} \\ p_z(t) &= \textstyle\sum_{i=1}^{4} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} \\ \theta_z(t) &= \tfrac{\theta_1 + \theta_2}{2} \end{aligned} \tag{5.10}$$

where x_i and y_i are the feature point coordinates in the image plane and θ_1 and θ_2 are shown in Figure 5.3. These features are referred to the selected features throughout this thesis. The new objective function is written as

$$OF_n = (\boldsymbol{\xi_n}(t_f) - \boldsymbol{\xi_{n}}_d)^{\mathrm{T}} (\boldsymbol{\xi_n}(t_f) - \boldsymbol{\xi_{n}}_d) \tag{5.11}$$

5.2.2 Parameterizing the Velocity Profile

A general predefined velocity profile is selected named $\mathbf{V}_{tp}(t)$. In a visual servoing task which deals with a stationary object, the robot starts from stationary situation and ends in a stationary situation. Thus, the selected profile needs to satisfy the following conditions.

$$\begin{aligned} \mathbf{V}_{tp}(0) &= 0 \\ \mathbf{V}_{tp}(t_f) &= 0 \end{aligned} \tag{5.12}$$

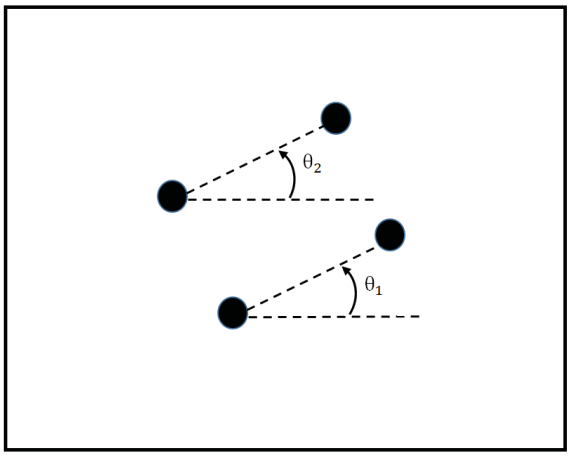


Figure 5.3: Calculating the angle feature from features points in an image

where t_f is the time which we planned to have the robot at the target position. Some examples of these functions could be a trapezoid function, a polynomial function or half cycle of a sinusoidal wave. However, more complicated trajectories with more parameters could be used such as higher order polynomial especially for the cases where other objective functions such as energy or path length are used for optimization.

For the 4 DOFs trajectory planning, half cycle of a sinusoidal profile is used to parameterize the velocity profile. Thus the general velocity profile can be shown as follows

$$\mathbf{V}_{tp}(t) = \mathbf{v}_m \sin(\tfrac{\pi t}{t_f}) \qquad 0 \leq t \leq t_f \tag{5.13}$$

where $\mathbf{v}_m$ is the vector of maximum speed that the camera reaches within the profile and is given as follows;

$$\mathbf{v}_m = \begin{bmatrix} v_{mx} \\ v_{my} \\ v_{mz} \\ v_{m\omega_z} \end{bmatrix} \tag{5.14}$$

where v_{mx}, v_{my}, v_{mz} and $v_{m\omega_z}$ are the maximum velocity of each velocity screw elements, respectively. The final time, t_f, is selected by the user depending on the

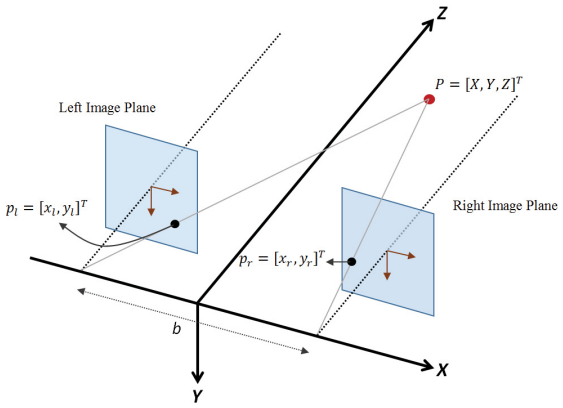


Figure 5.4: Stereo camera model

desired speed of the task. Thus, each profile have only one parameter to be designed and the overall number of design parameters of the system is four.

5.2.3 Depth Estimation

The estimation of the object depth is necessary for the trajectory planing due to its presence in equation (5.5). The motion of the camera in z_c direction is known from v_z element of the velocity screw which is given as a parameterized equation of time. Thus the depth Z can be calculated at any time t from

$$Z(t) = \int_{t_0}^{t} f_z(t)dt + Z_0 \tag{5.15}$$

where, Z_0 is the initial depth of the object with respect to the camera coordinates. If the initial depth of the object is estimated, accurately, the depth in the rest of the times could be calculated. Let us recall that using a stereo camera the depth of the object could be calculated [112–114]. This is done by applying the epipolar geometry constraint that exist between the features in images planes of each camera. In a simple case where the two cameras are mounted parallel to each other (Figure 5.4), the depth of the object with respect to the cameras can be calculated using the disparity of the images from equation (5.16).

$$Z_c = \frac{b}{x_l - x_r} \tag{5.16}$$

where Z_c is the depth of the object in the camera coordinates, x_r and x_l are the features x coordinates in left and right cameras, respectively and b is the distance between the cameras. We can conclude that by having two images of an object from a camera from which the second image is taken at a location with a displacement of b along X_c from the first location of the camera, the same equation could be used to calculated the object depth. Thus, by moving the camera along X_c by a small displacement b and using the initial and the final image feature positions and the depth of the object could be calculated from equation (5.16), This procedure takes about 1 second to complete which is feasible in experiment.

5.2.4 Features Motion Using Sinusoidal Profile

Letting $\mathbf{V}_c = \mathbf{V}_{tp}(t)$ and substituting $\mathbf{V}_{tp}$ from equation (5.13) into equation (5.4) the feature motion equation could be derived as follows;

$$\begin{aligned} \dot{x}_i &= -\tfrac{1}{Z_c} v_{mx} S + \tfrac{1}{Z_c} x_i v_{mz} S + y_i \omega_{mz} S \\ \dot{y}_i &= -\tfrac{1}{Z_c} v_{my} S + \tfrac{1}{Z_c} y_i v_{mz} S - x_i \omega_{mz} S, \end{aligned} \tag{5.17}$$

where $S = \sin(\frac{\pi t}{t_f})$ and Z_c is the depth of the object with respect to the camera. The depth could be calculated by substituting equation $v_z = v_{mz} \sin(\frac{\pi t}{t_f})$ into (5.15) which gives

$$Z_c = Z_0 - \frac{v_{mz} t f}{\pi} + \frac{v_{mz} t_f \cos(\frac{\pi t}{t_f})}{\pi} \tag{5.18}$$

Finding a closed form solution for $x_i(t)$ and $y_i(t)$ from equation (5.17) is nearly impossible due to the nonlinearity and the coupled nature of the differential equations. Therefore, numerical integration methods are used to integrate equation (5.17).

5.2.5 Convexity Analysis

An important point that needs to be considered, is that the trajectory planning procedure must be completed in a reasonable time. Otherwise, the method would be useless for real word applications because of the delay that is imposed to the system. One important factor that leads to fast convergence of the optimization problem is the convexity of the problem. In this section the convexity of the problem is investigated. To start with, let us review the following main theorems regarding convexity of a problem.

Theorem 1: If $f(x^*)$ is a local minimum for a convex function $f(x)$ defined on a convex feasible set S, it is also a global minimum [115].

Theorem 2: A function of n variable $f(x_1, x_2, ..., x_n)$ is defined on a convex set S is convex if and only if the Hessian matrix of the function is positive semidefinite or positive definite at all points in the set S [115].

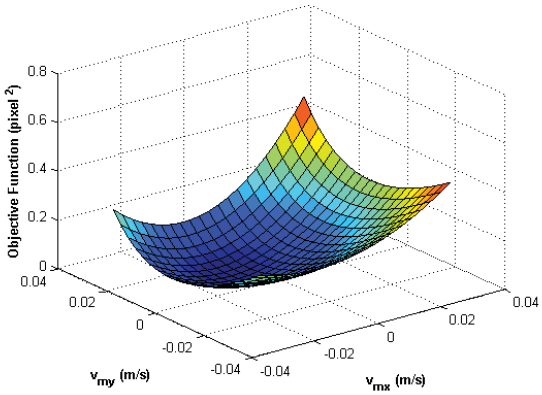
Proving the convexity of the objective function given in equation (5.11) requires the Hessian matrix of OF. Chinneck [116] introduced a method to discover the convexity of a program using numerical method. Accordingly, a code is generated to numerically calculate the Hessian matrix ([117]) of the objective function for a desired span of the desired parameters. The design parameter range depends on the physical limitations of the robot. In this case, the design parameters are the maximum velocity of the end-effector in the associated DOF. Knowing the speed limits of the robotic system, this could be identified. In our test the following ranges have been used.

$$\begin{aligned} -0.1 \leq v_{mx} \leq 0.1 \quad (m/sec) \\ -0.1 \leq v_{my} \leq 0.1 \quad (m/sec) \\ -0.1 \leq v_{mz} \leq 0.1 \quad (m/sec) \\ -0.3 \leq v_{m\omega_z} \leq 0.3 \quad (rad/sec) \end{aligned} \tag{5.19}$$

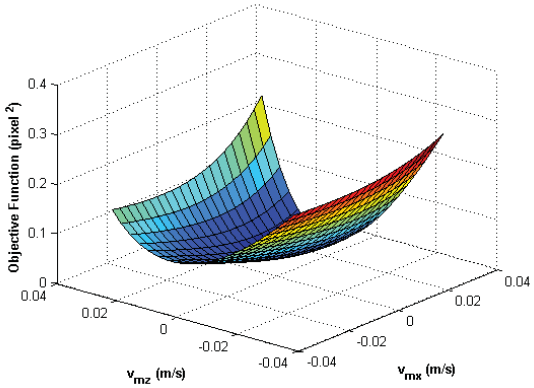
To demonstrate the results of this investigation, without the loss of generality, we chose the initial and desired locations such that the robot needs all the 4 DOFs motions to reach the desired position. The final time t_f is selected as 10 (sec). The changes to the objective function for different values of the design parameters are shown in Figure 5.5. To be able to show these variations in 3D plot format, the variation of the objective function is shown due the to changes in two parameters at each figure. All available combinations are presented. The variation of the objective function due to changes in v_{mx} and v_{my} are shown in Figure 5.5a. The variation of the objective function due to the changes in $v_{mx} - v_{mz}$, $v_{mx} - \omega_{mz}$ and $v_{mz} - \omega_{mz}$ are shown in Figures 5.5b, 5.5c and 5.5d, respectively. Because of the similarity in behavior of the system due to change in v_{mx} and v_{my}, all the diagrams related to changes in v_{my} are omitted here and one can refer to the figures showing the variations due to the changes in v_{mx}. The convexity of the objective function is clearly demonstrated in the Figures 5.5.

5.2.6 Constraints

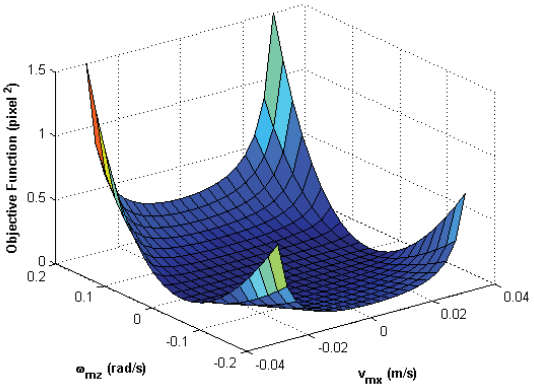
One of the main issues in conventional visual servoing is that it does not limit the robot within the system constraints. By just limiting the system within the constraints, the convergence of the system to the target point can not be guaranteed. The highly coupled nature of visual servoing system could cause the controlling law to take the robot toward and beyond its boundaries while IBVS is attempting to fix the camera's orientation. This can be easily observed in a visual servoing task using a conventional controller. Thus, limiting the system motion like a model predictive controller would do, is not sufficient to stabilize the system [36]. On the other hand, in a trajectory planning algorithm, the generated trajectory could be examined beforehand to guarantee that reaches the target while respecting the constraints. Two main constraints are considered in this paper. The first constraint



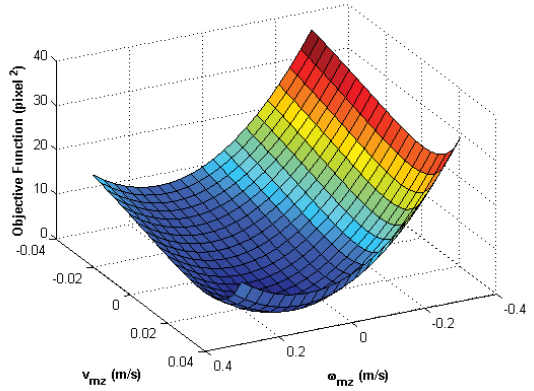
(a) v_{mx} verses v_{my}



(b) v_{mx} verses v_{mz}



(c) v_{mx} verses ω_{mz}



(d) v_{mz} verses ω_{mz}

Figure 5.5: Objective function variations

is associated with the robots working space. The second constraint is the robot joint limits. These constraints are discussed in details in the following sections.

It is good to note that, limiting the system to keep the features inside the field of view of the camera is vital to the success of the task in an IBVS conventional visual servoing. The proposed method integrates the equation of motion and predicts the features position at different time moments. Thus, it only requires the initial and the final positions of the features. Consequently, limiting the features inside the field of view is not necessary in this method.

Working Space Constraint

The planned trajectory is feasible only if it is inside the robot working space at all times. Every robot has its own working space. The typical working space of a serial manipulator is a part of sphere with the radius equal to the length of the arms when they are aligned in the same direction. This could be formulated in a polar system as follows,

$$\begin{aligned} X_c &= R_c\cos(\theta_c)\cos(\alpha_c) & & 0 < R_c \leq R_{c_{max}} \\ Y_c &= R_c\cos(\theta_c)\sin(\alpha_c) & \textit{and} \quad & \theta_{c_{min}} < \theta_c \leq \theta_{c_{max}} \\ Z_c &= R_c\sin(\theta_c) & & \alpha_{c_{min}} < \alpha_c \leq \alpha_{c_{max}} \end{aligned} \tag{5.20}$$

where, $\mathbf{P}_c = [X_c, Y_c, Z_c]^{\mathrm{T}}$ and $\mathbf{P}_{p_c} = [R_c, \theta_c, \alpha_c]^{\mathrm{T}}$ are the cameras coordinates in Cartesian and polar systems, $R_{c_{max}}$ is the maximum possible length of the robot's arm, $\theta_{c_{min}}$ and $\theta_{c_{max}}$ are the minimum and maximum angles of the robot's arm about its base X axis, $\alpha_{c_{min}}$ and $\alpha_{c_{max}}$ are the minimum and maximum angles of the robot's arm about its base Z axis.

Joints Space Constraint

Keeping the robot inside the working space is not enough to accomplish a visual servoing task. In addition to work space constraint, it is necessary to make sure the

robot respects its joint limits and does not collide with itself. These constraints can be formulated as follows.

$$\mathbf{q}_{min} \leq \mathbf{q} \leq \mathbf{q}_{max} \tag{5.21}$$

where $\mathbf{q}$ is the robot joint vector and $\mathbf{q}_{min}$ and $\mathbf{q}_{max}$ are defined as the robot's joint limits. The end-effector position is known at all time during the servoing. A function is required to transform the robot's end-effector coordinates to robot joints' value. This function is the inverse kinematic of the robot. The constraint could be written as

$$\mathbf{q}_{min} \leq \mathbf{I}(\mathbf{P_c}) \leq \mathbf{q}_{max} \tag{5.22}$$

where $\mathbf{I}(\mathbf{P}_c)$ is the inverse kinematic function of the robot.

5.3 Visual Servoing Controller

In the cases where there are some uncertainties in the system model, the generated trajectory locates the features with a small error with respect to the target position. To compensate for such errors, a visual servoing controller is required. For this matter, the AIBVS controller developed in section 2.3.1 and given in equation 2.21 will be used.

5.4 Experimental Results

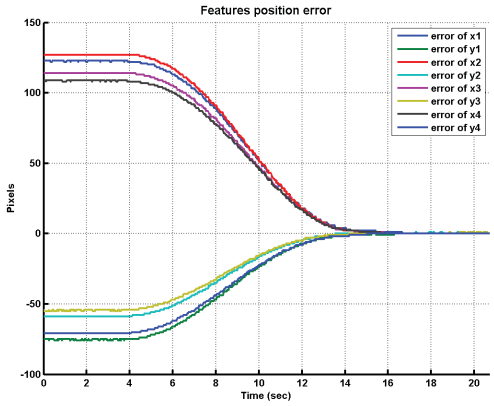
In this section, the results of the experimental test of the proposed algorithm are presented. The experimental setup is described in section 2.5. Since the algorithm is developed for a 4 DOFs robot, the robot joints 4 and 5 are set to zero and $90^o - q_2 + q_3$ to keep the camera looking down in the whole process. Thus, end-effector can move in X, Y, Z directions and it can rotate about the end-effector axis which we refer to as θ_z. The camera characteristics are given in Table 2.2.

First, the depth estimation algorithm moves the end-effector in X_c direction by 10 cm to take the stereoscopic image and estimates the depth of the object. Second, using the current image features, desired image features and the initial depth of the object, the trajectory planning algorithm generates the appropriate velocity screw through an optimization process to take the robot to the desired position. Due to the nonlinearity of the selected objective function and constraints, an interior point algorithm [118] is used to solve the optimization problem. The generated velocity is applied to the robot. As the robot finishes moving according to the generated velocity, the third stage of the algorithm starts. At the third stage, an AIBVS [70] controller is executed to compensate for any difference between the image features and the desired image features caused by the uncertainties in system model. As it is shown in the results, most of the tests may not require the last stage, since the trajectory planning exactly matches the features with the desired ones. Four different tests with different strategies have been performed to ensure the algorithms validity.

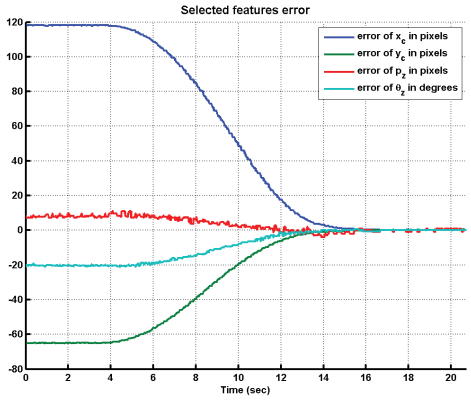
Test 1:

In the first test, our aim is to show the system performing a relativity simple visual servoing task. The initial and desired locations of the features are given in the Table 5.1.

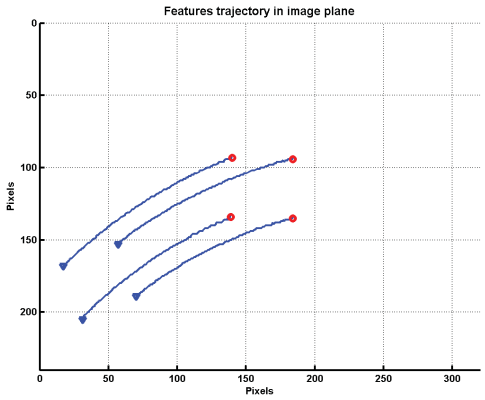
The trajectory planning algorithm generates the velocity profiles shown in Figure 5.6e. Applying the velocities to the robot, the robot is taken to the desired position. The features trajectory in image space and the camera trajectory in 3D space are shown in Figures 5.6c and 5.6d. The quarter of the sphere in this figure shows the workspace of the robot. The robot joint angles during the robot motion are shown in Figure 5.6f. Since, the system model is sufficiently accurate, the desired position is reached using the velocity profiles and the third stage of the algorithm



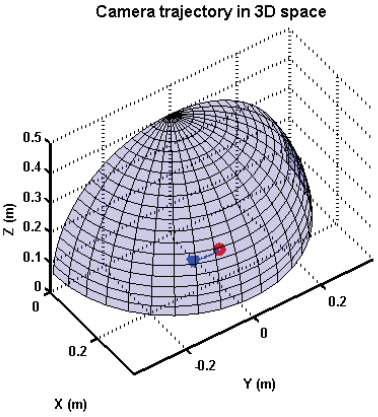
(a) Features position error



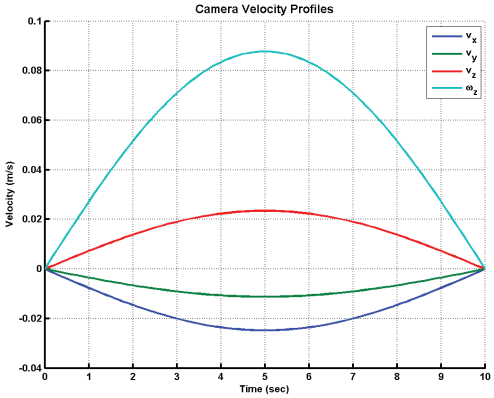
(b) Selected features error



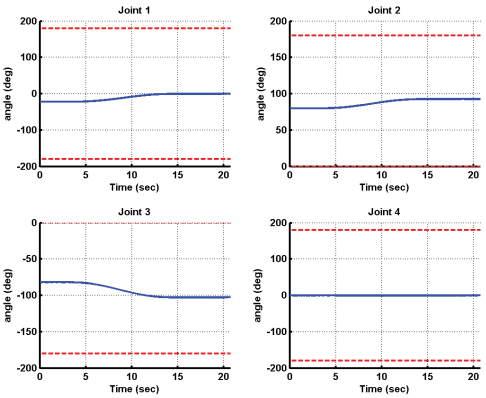
(c) Feature trajectory in image plane



(d) Camera 3D trajectory



(e) Generated velocity profile



(f) Robot joint angles

Figure 5.6: Results for Test 1, performing a basic visual servoing task

Table 5.1: Initial(I) and desired(D) location of the feature points in pixel for 4DOFs trajectory planning tests

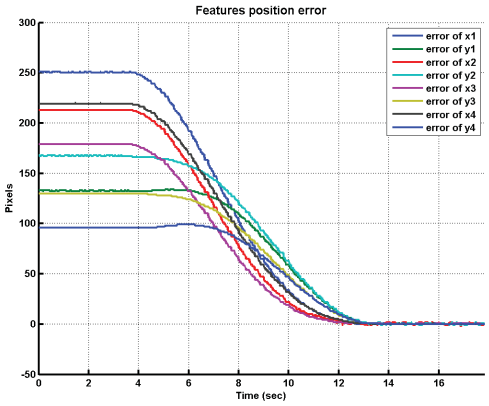
		Point1		Point2		Point3		Point4	
		(x	y)	(x	y)	(x	y)	(x	y)
Test 1	I	80	195	40	212	23	175	65	160
	D	182	141	136	141	136	98	183	98
Test 2	I	200	79	196	153	126	148	129	79
	D	197	154	123	147	128	78	200	80
Test 3	I	178	226	132	226	132	183	177	184
	D	151	143	191	142	192	180	151	181
Test 4	I	107	210	16	206	26	133	114	137
	D	291	212	203	229	187	154	276	136
Test 5	I	123	149	123	189	83	188	83	148
	D	104	212	77	202	121	161	112	69

is not required for this test. In the first stage of the algorithm, the robot moves the camera by 10 cm in x_c direction and the estimated depth is 0.49 m. The optimization process in this test takes less than a second to complete using a Intel Xeon E31220 3.10GHz CPU.

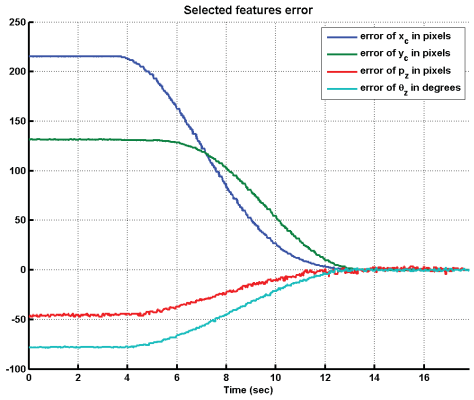
In the Tests 2, 3 and 4, it is intended to check and compare the performance of the proposed algorithm for the cases where the conventional image based controllers [12] cannot stabilize the system.

Test 2:

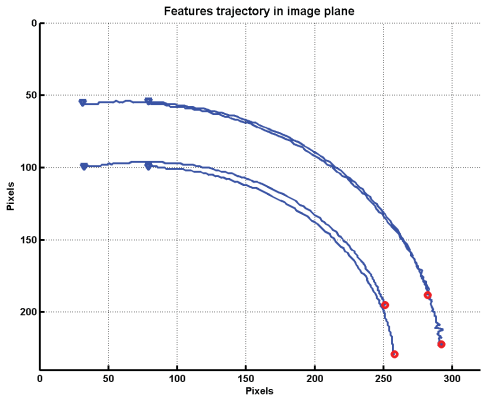
Test 2 is done on a case where the initial and target features are distant. The initial and desired locations of the features are shown in Table 5.1. The result of second test is presented in Figures 5.7. The results show that using the generated velocity profiles, the visual servoing task can be accomplished. Figure 5.7d shows the trajectory of the robot's end-effector within the robot's workspace. The same test has been done with a visual servoing controller. The results are shown in Figures 5.8.



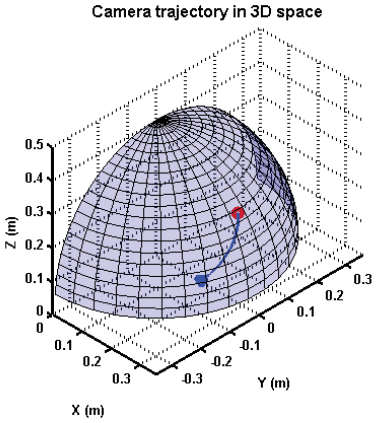
(a) Features position error



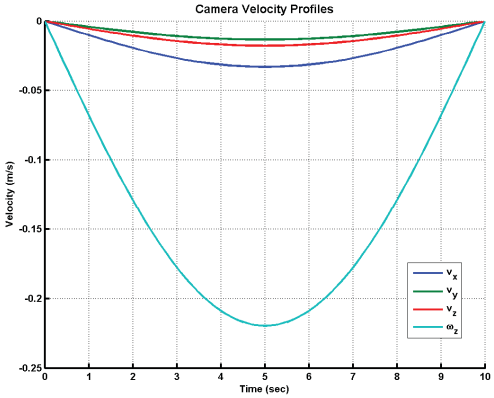
(b) Selected features error



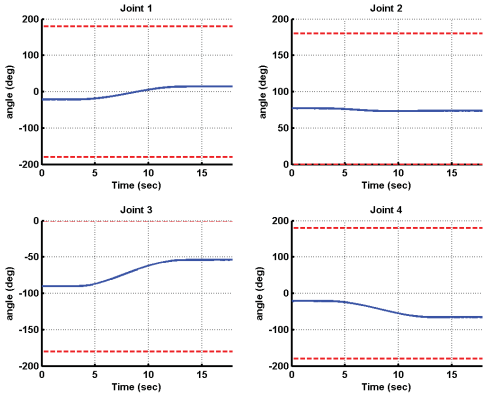
(c) Feature trajectory in image plane



(d) Camera 3D trajectory

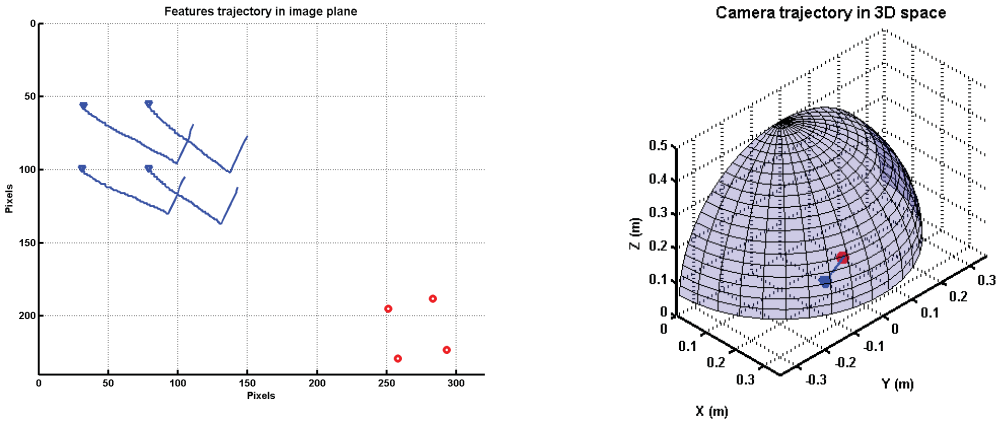


(e) Generated velocity profile



(f) Robot joint angles

Figure 5.7: Results for Test 2, reaching distant desired feature



(a) Features position error (b) Selected features error

Figure 5.8: Results for Test 2 using IBVS controller

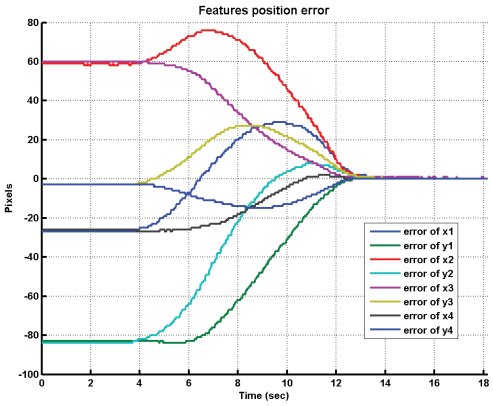
Test 3:

For the third test, another common problem of IBVS is investigated using the proposed method. The visual servoing fails when a 180 degrees rotation of the camera is required to reach its desired position [28]. A test is prepared including a 180 degrees rotation in the end effector motion. The initial and desired locations of the feature points are given in Table 5.1. The result of this test is shown in Figures 5.9.

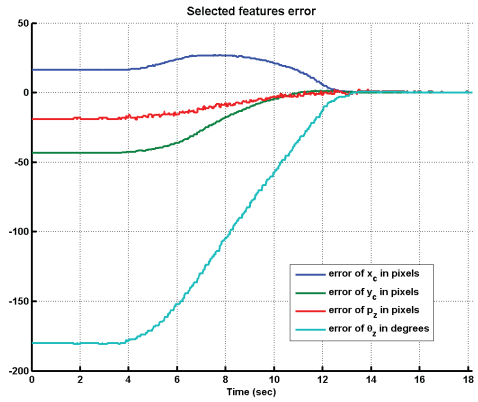
The same test is conducted using IBVS controllers. The results are shown in Figures 5.10. The results show that, similar to the previous test, the IBVS controller tries to match the features through the shortest path available which results in a motion of camera in the Z_c direction. This continues until the end-effector reaches its physical limits and the robot stops, as shown in Figure 5.10b.

Test 4:

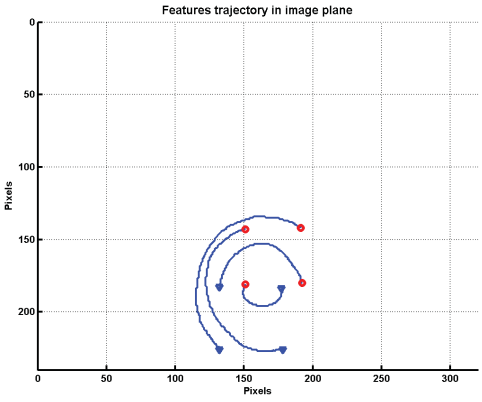
Another challenges in conventional visual servoing is the local minima problem. In an IBVS controller, the interaction matrix is an eight by six matrix. The inverse of this matrix, which is used to produce the controlling law, is an eight by six matrix



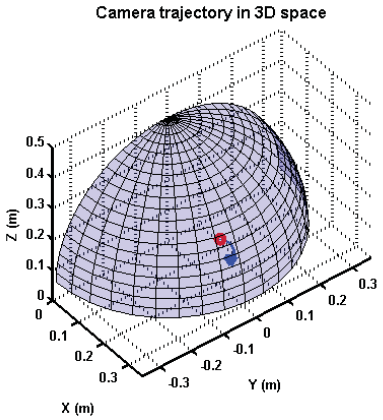
(a) Features position error



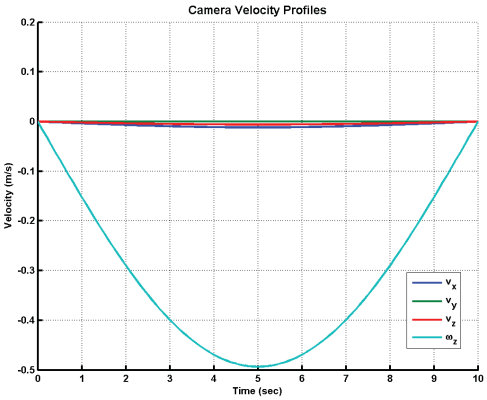
(b) Selected features error



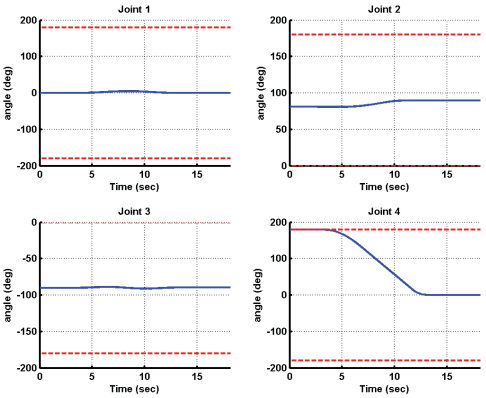
(c) Feature trajectory in image plane



(d) Camera 3D trajectory

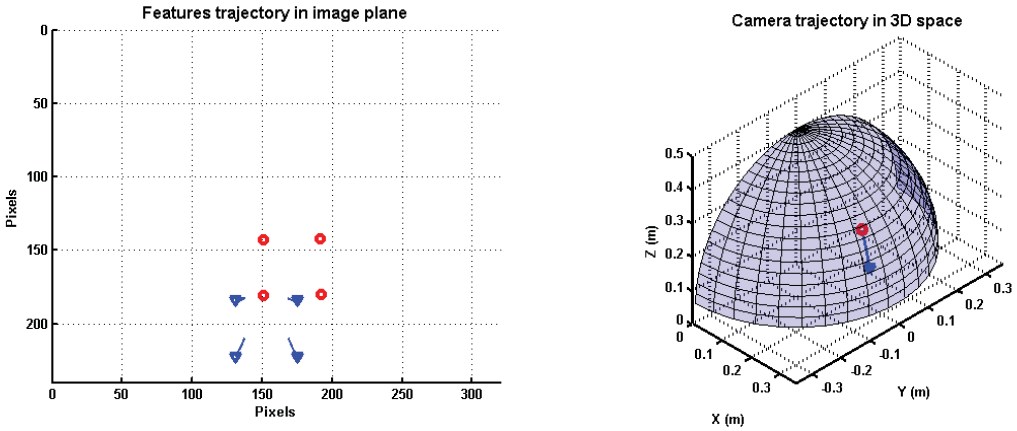


(e) Generated velocity profile



(f) Robot joint angles

Figure 5.9: Results for Test 3, performing a visual servoing task with 180^o about the camer z_c axis



(a) Features position error

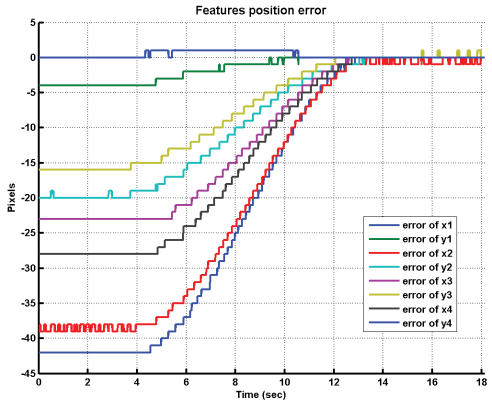
(b) Selected features error

Figure 5.10: Results for Test 3 using IBVS controller

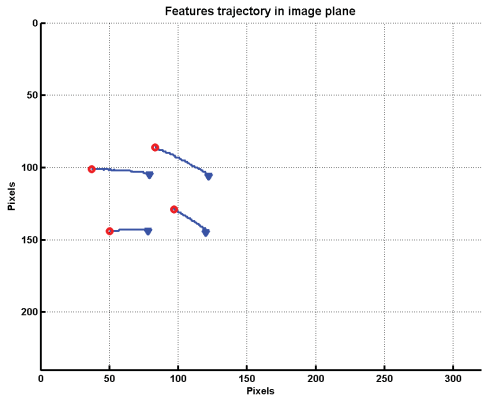
and has two vector of null space. If the features error vector is a factor of these null space vectors, the controller generates a zero velocity vector as the controlling command. This causes the system to get stuck in that spot. In the trajectory planning algorithm, the inverse of the interaction matrix is not used. consequently, the local minima problem is solved. The next test demonstrates this ability in the proposed algorithm. The initial and desired locations of the feature points are given in Table 5.1. The desired features are chosen so that the vector of feature position error is in the null space of the interaction matrix. The results are shown in Figures 5.11. We can see that the proposed algorithm produces a velocity profile to take the robot to the desired position while the IBVS controller produces a zero velocity vector.

5.5 Trajectory Planning for a 6 DOFs Robot

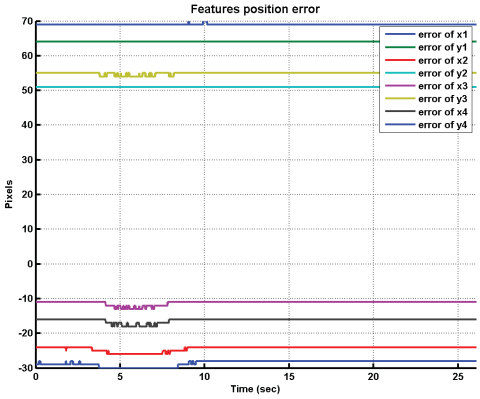
In this section, the developed trajectory planning algorithm for a 4 DOFs robot is modified to work on a 6 DOFs robot. A 6 DOFs robot manipulator is used to perform such task. In this case the robot end-effector has 3 linear motion in x_c, y_c and z_c axes and three rotation about x_c, y_c and z_c axis. The relation between the



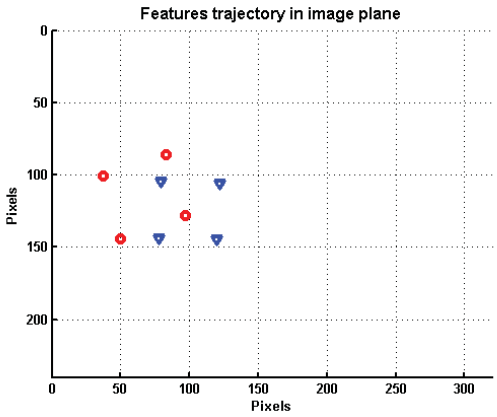
(a) Features position error



(b) Features trajectory in image space



(c) Features position error for IBVS



(d) Features trajectory in image space for IBVS

Figure 5.11: Results for Test 4, performing a visual servoing task with desired features are located at the null space of the interaction matrix

motion of the camera and features motion in the image is given by

$$\dot{\mathbf{p}} = \mathbf{L}_a \mathbf{V}, \tag{5.23}$$

where

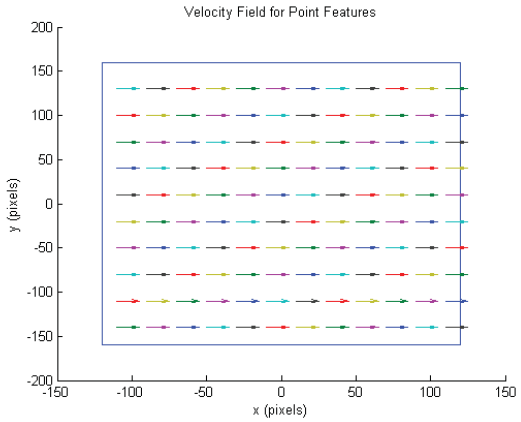
$$\mathbf{L}_a = \begin{bmatrix} -\frac{1}{Z} & 0 & \frac{x}{Z} & xy & -(1+x^2) & y \\ 0 & -\frac{1}{Z} & \frac{y}{Z} & 1+y^2 & -xy & -x \end{bmatrix} \tag{5.24}$$

is the interaction matrix, Z is the depth of the object with respect to the camera and $\mathbf{V} = [v_x \; v_y \; v_z \; \omega_x \; \omega_y \; \omega_z]^{\mathrm{T}}$ is the camera's velocity screw represented in camera frame.

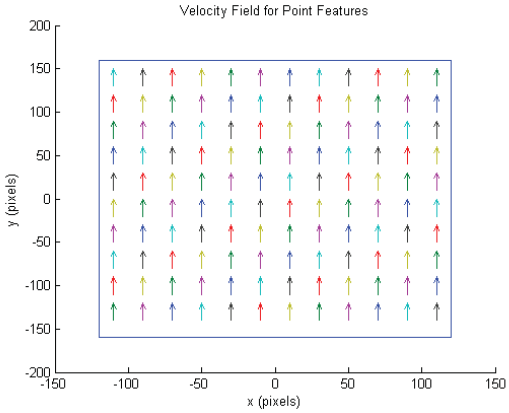
5.5.1 Path Planning

The robot could perform 6 degrees of motion to reach any desired pos (including position and orientation). The effect of each motion could be calculated using equation (5.24). Figure 5.12 shows how each motion affects the feature point position.

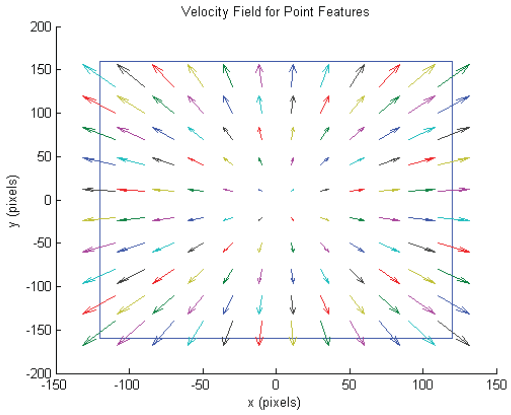
The first two elements of the velocity screw create linear motions in the same direction for all features (Figures 5.12a and 5.12b). These two camera motions are used for displacing the features in x and y direction of the image plane. A camera motion in z_c direction creates an outward motion for the features which is in the direction of line connecting the center of the image to the image feature (Figure 5.12c). It has to be noted that a negative motion in z_c direction will create an inward motion for the features. Thus, this motion could compensate the distances between the features. The fourth and fifth element of the velocity screw create a complicated motion in the features. It creates an inward motion for features in one side of the image and an outward motion for the features on the other side of the image (Figures 5.12d and 5.12e). The last element of the velocity screw rotates the features about the center of image (Figure 5.12f).



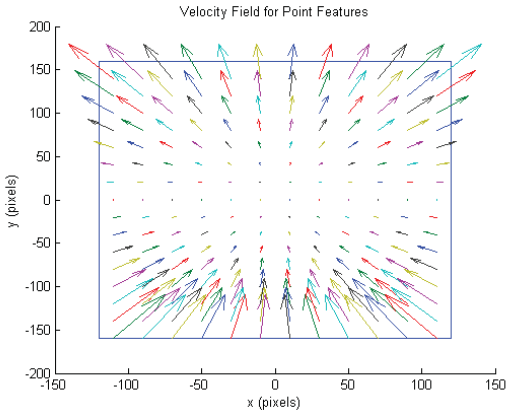
(a) Velocity field for v_x motion



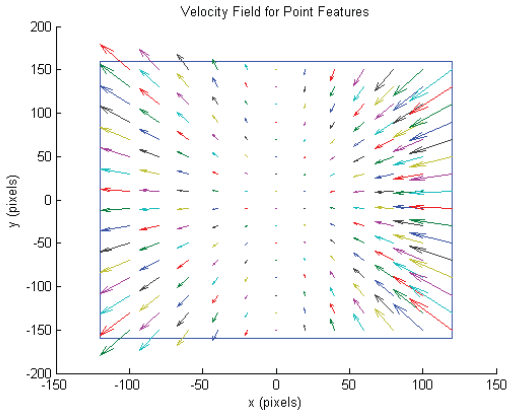
(b) Velocity field of v_y motion



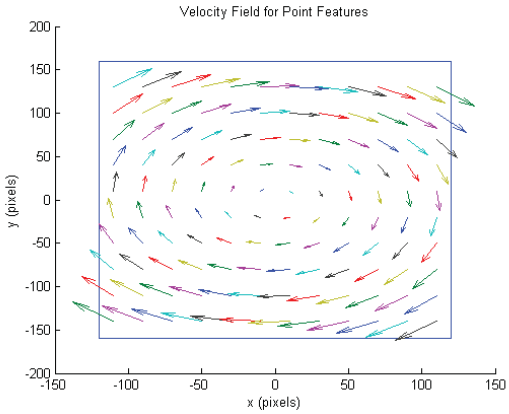
(c) Velocity field of v_z motion



(d) Velocity field of ω_z motion



(e) Velocity field of v_z motion



(f) Velocity field of ω_z motion

Figure 5.12: Velocity field of the features subject to camera velocities

The concept behind the trajectory planning is that any target features could be reached by using a combination of shown feature motions. Six basic velocity profiles are generated for each of the camera's velocity screw elements. The effect of the generated velocity screw can be calculated using equation (5.24). In other words, by superposing the velocity fields caused by each element of the velocity screw, the final position of the features could be calculated. The parameters of the camera velocity are then determined by minimizing the error between the image features and the target ones.

The features velocity in image space could be written as a function of velocity screw elements, given by

$$\begin{aligned} \dot{x}_i &= \tfrac{-1}{Z} v_x + \tfrac{x_i}{Z} v_z + x_i y_i \omega_x - (1 + x_i^2)\omega_y + y_i \omega_z \\ \dot{y}_i &= \tfrac{-1}{Z} v_y + \tfrac{y_i}{Z} v_z + (1 + y_i)^2 \omega_x - x_i y_i \omega_y - x_i \omega_z, \end{aligned} \tag{5.25}$$

where $\dot{x}_i$ and $\dot{y}_i$ are the velocities of the ith image feature in x and y direction, respectively. Consequently, the image feature position could be calculated as

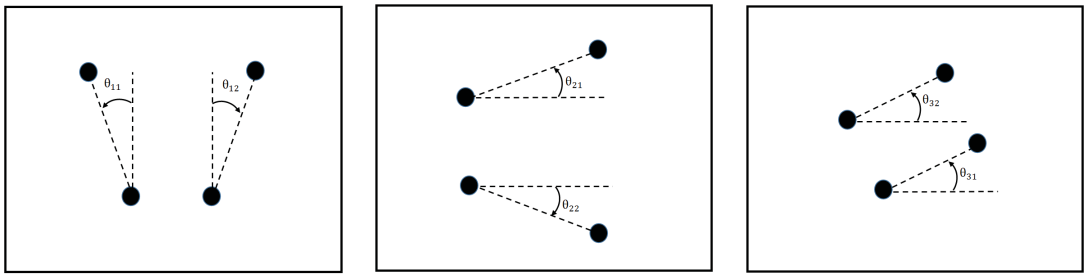
$$\begin{aligned} x_{it} &= \textstyle\int_{t_0}^{t} (\dot{x}_i(t))dt + x_{i0} \\ y_{it} &= \textstyle\int_{t_0}^{t} (\dot{y}_i(t))dt + y_{i0}, \end{aligned} \tag{5.26}$$

where x_{i_0} and y_{i_0} are the initial coordinates of the image features and x_i and y_i are the locations of the image features at time t. Thus, by knowing the initial position of the features and the velocity of the camera the position of the features can be calculated at each time.

Image Features

The interaction matrix achieved for point features (equation (5.24)) is highly nonlinear and coupled. In order to facilitate the optimization process some new features are presented in this thesis. The new set of image features is shown as

$$\mathbf{s_n} = \begin{bmatrix} x_c & y_c & p_z & \theta_x & \theta_y & \theta_z \end{bmatrix}^{\mathrm{T}} \tag{5.27}$$



(a) Velocity field for v_x motion (b) Velocity field of v_y motion (c) Velocity field of v_z motion

Figure 5.13: Calculating the three last features form the point features

where $x_c(t)$ and $y_c(t)$ are the centers of the feature points and $p_z(t)$ is the perimeter of the lines connecting each consecutive feature point which are given as

$$\begin{aligned} x_c(t) &= \frac{\sum_{i=1}^{4} x_i(t)}{4} \\ y_c(t) &= \frac{\sum_{i=1}^{4} y_i(t)}{4} \\ p_z(t) &= \sum_{i=1}^{4} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} \end{aligned} \tag{5.28}$$

where $\theta_x(t)$, $\theta_y(t)$ and $\theta_y(t)$ are defined based on the deformation that is made in the features by rotating the camera about x_c, y_c and z_c. These features are given by

$$\begin{aligned} \theta_x(t) &= \frac{\theta_{11}+\theta_{12}}{2} \\ \theta_y(t) &= \frac{\theta_{21}+\theta_{22}}{2} \\ \theta_z(t) &= \frac{\theta_{31}+\theta_{32}}{2} \end{aligned} \tag{5.29}$$

where θ_{11}, θ_{12}, θ_{21}, θ_{22}, θ_{31}, θ_{32} are shown in the Figure 5.13.

Parameterizing the Velocity Profile

A general predefined velocity profile named $\mathbf{V}_{tp}(t)$ is selected. In a visual servoing task dealing with a stationary object, the robot starts from stationary situation and ends in a stationary situation. Thus, the selected profile needs to

satisfy the following conditions.

$$\begin{aligned} \mathbf{V}_{tp}(0) &= 0 \\ \mathbf{V}_{tp}(t_f) &= 0 \end{aligned} \tag{5.30}$$

where t_f is the final time which we planned to have the robot at the target position.

In this thesis, half cycle of a sinusoidal profile is used to parameterize the velocity profile. The general velocity profile can be shown as follows

$$\mathbf{V}_{tp}(t) = \mathbf{v}_m \sin(\tfrac{\pi t}{t_f}) \qquad 0 \leq t \leq t_f \tag{5.31}$$

where $\mathbf{v}_m$ is the maximum speed that the camera reaches within the profile and is given as follows;

$$\mathbf{v}_m = \begin{bmatrix} v_{mx} & v_{my} & v_{mz} & v_{m\omega_x} & v_{m\omega_y} & v_{m\omega_z} \end{bmatrix}^{\mathrm{T}} \tag{5.32}$$

where v_{mx}, v_{my}, v_{mz}, $v_{m\omega_x}$, $v_{m\omega_y}$ and $v_{m\omega_z}$ are the maximum velocity of each velocity screw elements, respectively. The final time, t_f, is selected by the user depending on the desired speed of the task. Thus, each profile have only one parameter to be designed and the overall number of design parameters of the system is six.

5.5.2 Decoupling Orientation Planning from Position Planning

Testing the trajectory planning as explained above shows that the system is highly nonlinear and the optimization process is not convex. In some cases the process does not converges and in other cases there is no guarantee to converge in a reasonable time. Due to the important role that the convergence time plays in feasibility of the algorithm, it is proposed to decouple the orientation planing from position planning. Decoupled visual servoing was previously presented in [38]. In this thesis, decoupled trajectory planning is proposed.

The decoupling procedure is explained in follows. First the last three velocity screw elements are planned in the optimization process so that they take the last

three feature set elements to their desired values. Second, the first three elements of the velocity screw is planned to eliminate the error existing in the first three elements of the feature set. The last three joints of the robot is responsible for the fixing the orientation and the first three joints of the robot is responsible for positioning. As it is investigated in the next section, using the selected features and decoupling the planning process leads to a convex optimization process.

5.5.3 Optimization and Convexity Analysis

Let us define the objective function as the quadratic form of the selected features error, given by

$$OF = (\boldsymbol{\xi}_n(t_f) - \boldsymbol{\xi}_{nd})^{\mathrm{T}} \mathbf{Q} (\boldsymbol{\xi}(t_f) - \boldsymbol{\xi}_d), \tag{5.33}$$

where $\mathbf{Q}$ is a orthogonal matrix introducing the desired weight of each error in the optimization process. An important point that needs to be considered, is that the trajectory planning procedure must be completed in a reasonable time. Otherwise, the method would be useless for real word applications because of the delay that is imposed to the system. One important factor that leads to fast convergence of the optimization problem is the convexity of the problem. In this section the convexity of the problem is investigate. To start, let us review the following main theorems regarding convexity of a problem.

Theorem 1: If $f(x^*)$ is a local minimum for a convex function $f(x)$ defined on a convex feasible set S, it is also a global minimum [115].

Theorem 2: A function of n variable $f(x_1, x_2, ..., x_n)$ is defined on a convex set S is convex if and only if the Hessian matrix of the function is positive semidefinite or positive definite at all points in the set S [115].

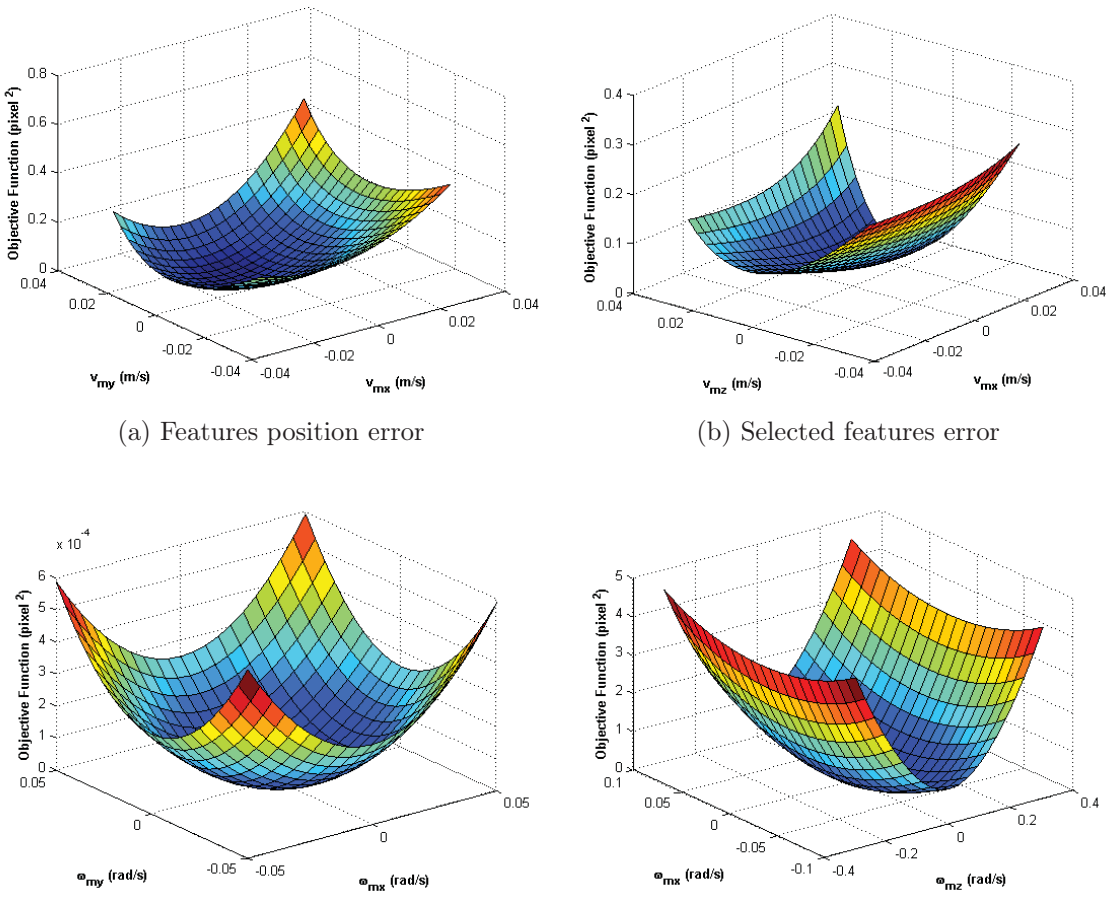
Proving the convexity of the objective function given in equation (5.33) requires the Hessian matrix of OF. Chinneck [116] introduced a method to discover

the convexity of a program using numerical method. Accordingly, a code is generated to numerically calculate the Hessian matrix [117] of the objective function for a desired span of the desired parameters. The design parameter range depends on the physical limitations of the robot. In this case, the design parameters are the maximum velocity of the end-effector in the associated DOF. Knowing the speed limits of the robotic system, this could be identified. In our test, the following ranges have been used.

$$\begin{aligned}
&-0.1 \leq v_{mx} \leq 0.1 \quad (m/sec) \\
&-0.1 \leq v_{my} \leq 0.1 \quad (m/sec) \\
&-0.1 \leq v_{mz} \leq 0.1 \quad (m/sec) \\
&-0.1 \leq v_{m\omega_x} \leq 0.1 \quad (rad/sec) \\
&-0.1 \leq v_{m\omega_y} \leq 0.1 \quad (rad/sec) \\
&-0.3 \leq v_{m\omega_z} \leq 0.3 \quad (rad/sec)
\end{aligned} \tag{5.34}$$

To demonstrate the results of this investigation, without the loss of generality, we chose the initial and desired locations such that the robot needs a motion in all the 6 DOFs to reach the desired position. The final time t_f is selected as 10 (sec). The changes to the objective function for different values of the design parameters are shown in Figure 5.14. To be able to show these variations in 3D plot format, the variation of the objective function is shown due to the changes in two parameters at each figure. All available combinations are presented. The variations of the objective function due to changes in $v_{mx} - v_{my}$ are shown in Figure 5.14a. The variation of the objective function due to the changes in $v_{mx} - v_{mz}$ is shown in Figure 5.14b. Because of the similarity in behavior of the system due to change in $v_{mx} - v_{my}$ all the diagrams related to changes in v_{my} are omitted here and one can refer to the figures showing the variations due to the changes in v_{mx}. Moreover, due to the fact that the trajectory planning is decoupled, the orientation never interfere with the

positioning. Thus, it is not required to check the convexity of the system due to a combined linear and angular motion. To check for the convexity of the system due to the angular motions the changes in the objective function is introduced due to the changes in $\omega_{mx} - \omega_{my}$ and $\omega_{mx} - \omega_{mz}$. These changes are shown in Figures 5.14c and 5.14c. The convexity of the objective function is clearly demonstrated in the Figures 5.14.



(a) Features position error

(b) Selected features error

(c) Feature trajectory in image plane

(d) Camera 3D trajectory

Figure 5.14: Convexity of the objective function

5.5.4 Constraints

One of the main issues in conventional visual servoing is that it does not limit the robot within the system constraints. In addition, by just limiting the system within the constraints the convergence of the system to the target point can not be guaranteed. The highly coupled nature of visual servoing system could cause the controlling law to take the robot toward and beyond its boundaries while IBVS is attempting to fix the camera's orientation. This can be easily observed in a visual servoing task using a conventional controller. Thus, limiting the system motion like a model predictive controller would do, is not sufficient to stabilize the system [36]. On the other hand, in a trajectory planning algorithm, the generated trajectory could be examined beforehand to guarantee that the target can be reached while respecting the constraints. Two main constraints are considered in this thesis. The first constraint is associated with the robot's working space. The second constraint is the robot joint limits. These constraints are discussed in details in the following sections.

It is good to note that, limiting the system to keep the features inside the field of view of the camera is vital to the success of the task in an IBVS conventional visual servoing. The proposed method integrates the equation of motion and predicts the features position at different time moments. Thus, it only requires the initial and the final positions of the features. Consequently, limiting the features inside the field of view is not necessary in this method.

Working Space Constraint

The planned trajectory is feasible only if it is inside the robot working space at all times. Every robot has its own working space. The typical working space of a serial manipulator is a part of sphere with the radius equal to the length of the arms when they are aligned in the same direction. This could be formulated in a

polar system as follows,

$$\begin{aligned} X_c &= R_c\cos(\theta_c)\cos(\alpha_c) & & 0 < R_c \leq R_{c_{max}} \\ Y_c &= R_c\cos(\theta_c)\sin(\alpha_c) & \textit{and} \quad & \theta_{c_{min}} < \theta_c \leq \theta_{c_{max}} \\ Z_c &= R_c\sin(\theta_c) & & \alpha_{c_{min}} < \alpha_c \leq \alpha_{c_{max}} \end{aligned} \tag{5.35}$$

where, $\mathbf{P}_c = [X_c, Y_c, Z_c]^{\mathrm{T}}$ and $\mathbf{P}_{p_c} = [R_c, \theta_c, \alpha_c]^{\mathrm{T}}$ are the cameras coordinates in Cartesian and polar systems, $R_{c_{max}}$ is the maximum possible length of the robot's arm, $\theta_{c_{min}}$ and $\theta_{c_{max}}$ are the minimum and maximum angles of the robot's arm about its base X axis, $\alpha_{c_{min}}$ and $\alpha_{c_{max}}$ are the minimum and maximum angles of the robot's arm about its base Z axis.

Joints Space Constraint

Keeping the robot inside the working space is not enough to accomplish a visual servoing task. In addition to work space constraint, it is necessary to make sure the robot respects its joint limits and does not collide with itself. These constraints can be formulated as follows.

$$\mathbf{q}_{min} \leq \mathbf{q} \leq \mathbf{q}_{max} \tag{5.36}$$

where $\mathbf{q}$ is the robot joint vector and $\mathbf{q}_{min}$ and $\mathbf{q}_{max}$ are defined as the robot's joint limits. The end-effector position is known at all time during the servoing. A function is required to transform the robot's end-effector coordinates to robot joints' value. This function is the inverse kinematic of the robot. The constraint could be written as

$$\mathbf{q}_{min} \leq \mathbf{I}(\mathbf{P_c}) \leq \mathbf{q}_{max} \tag{5.37}$$

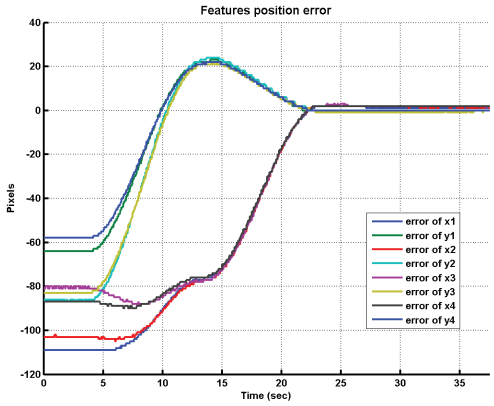
where $\mathbf{I}(\mathbf{P}_c)$ is the inverse kinematic function of the robot.

5.6 Experimental Results

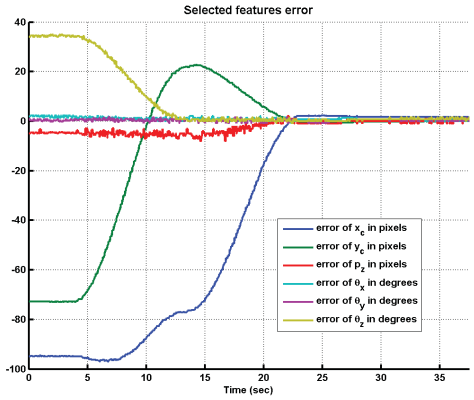
In this section, the results of the experimental tests of the proposed algorithm are presented. Each complete test consists of four stages. First, the depth estimation algorithm moves the end-effector in X_c direction by 10 cm to take the stereoscopic image and estimates the depth of the object. Second, using the current image features, desired image features and the initial depth of the object, the trajectory planning algorithm generates the appropriate angular velocity though optimization to reorient the camera to a parallel plane as the object plane. This is done by matching the three last selected features. In the third stage, the positioning trajectory is generated by matching the first three selected features. Due to the nonlinearity of the selected objective function, an interior point algorithm [118] is used to solve the optimization problem. The generated velocity is applied to the robot. As the robot finishes moving according to the generated velocity, the fourth stage of the algorithm starts. At the fourth stage, an AIBVS [70] controller is executed to compensate for any difference between the image features and the desired image features caused by the uncertainties in system model. As it is shown in the results, most of the tests may not require the last stage, since the trajectory planning exactly matches the features with the desired ones. Four different tests with different strategies have been performed to ensure the algorithms validity.

Test 1:

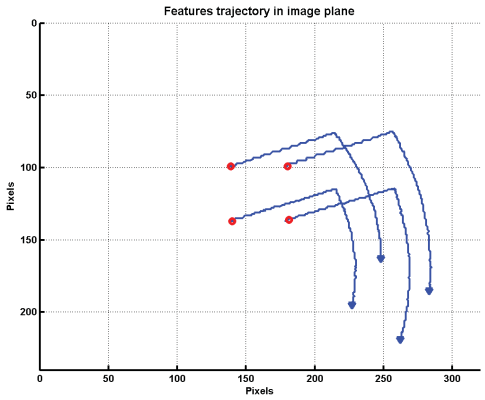
In the first test, our aim is to show the performance of the system on performing a relativity simple visual servoing task. The initial and desired locations of the features are given in the Table 5.2. The trajectory planning algorithm generates the velocity profiles shown in Figure 5.15e. Applying the velocities to the robot, the robot is taken to the desired position. The first sin cycle is related to the orientation planning and the second part is related to the positioning. The features trajectory in



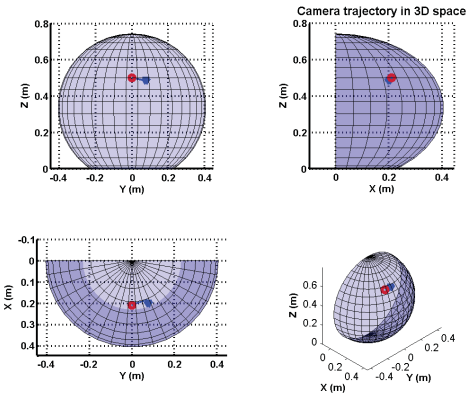
(a) Features position error



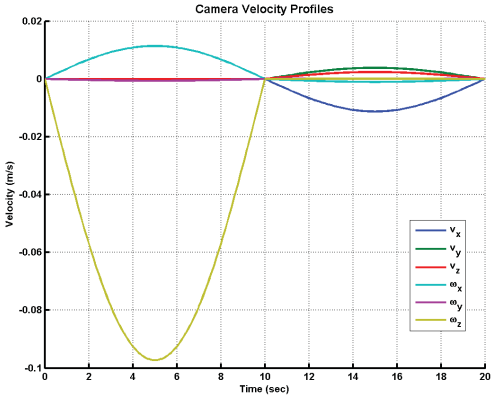
(b) Selected features error



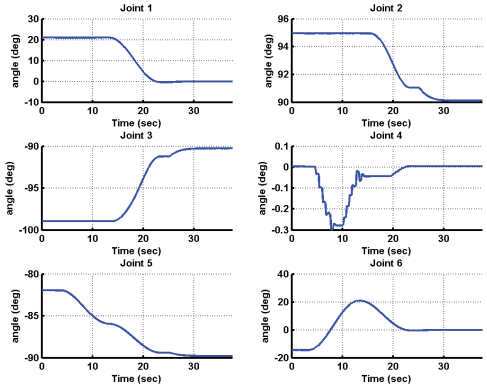
(c) Feature trajectory in image plane



(d) Camera 3D trajectory



(e) Generated velocity profile



(f) Robot joint angles

Figure 5.15: Results for Test 1, performing a basic visual servoing task

Table 5.2: Initial(I) and desired(D) location of the feature points in pixel for 6DOFs trajectory planning tests

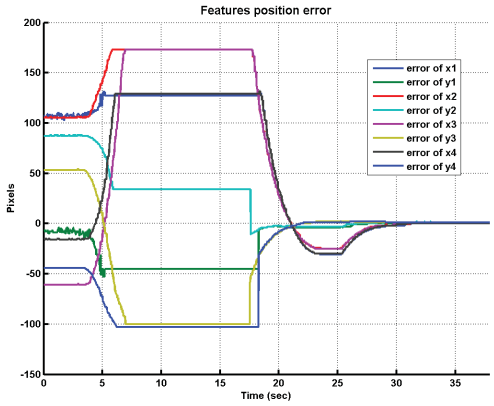
		Point1		Point2		Point3		Point4	
		(x	y)	(x	y)	(x	y)	(x	y)
Test 1	I	248	163	283	185	262	219	227	195
	D	138	99	179	99	179	136	138	137
Test 2	I	32	106	75	12	242	83	155	181
	D	139	100	179	98	180	135	139	136
Test 3	I	137	99	178	99	179	136	138	137
	D	190	154	129	154	128	98	190	98
Test 4	I	107	210	16	206	26	133	114	137
	D	291	212	203	229	187	154	276	136
Test 5	I	123	149	123	189	83	188	83	148
	D	104	212	77	202	121	161	112	69

image space and the camera trajectory in 3D space are shown in Figures 5.15c and 5.15d. The half sphere in this figure shows the workspace of the robot. The robot joint angles during the robot motion are shown in Figure 5.15f. Since, the system model is sufficiently accurate, the desired position is reached using the velocity profiles and the fourth stage of the algorithm is not required for this test. In the first stage of the algorithm, the robot moves the camera by 10 cm in X_c direction and the depth estimation is 0.4 m. The optimization process in this test takes less than two second to complete using a Intel Xeon E31220 3.10GHz CPU.

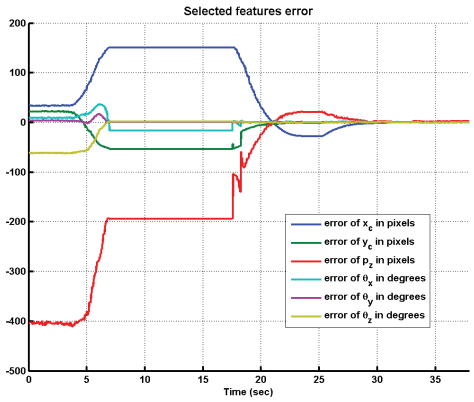
Test 2:

In the second test some of prominence of the proposed method to IBVS controller are shown. A relatively complicated task is chosen for this matter. The initial and final position of the robot is given in Table 5.2. The results of this test are given in Figures 5.16

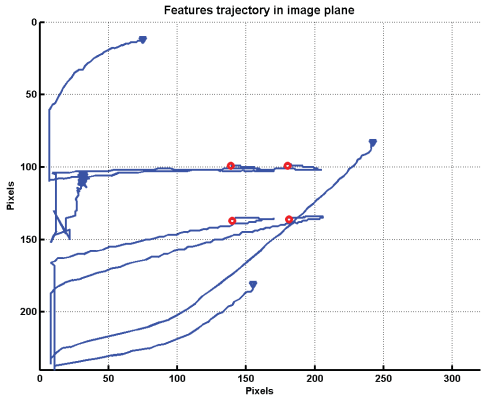
The optimization process creates the velocity profile given in Figure 5.16e. The first part of the velocity profile is to orient the camera to be parallel to the object's feature plane. These velocity profiles only moves the three last joints. This



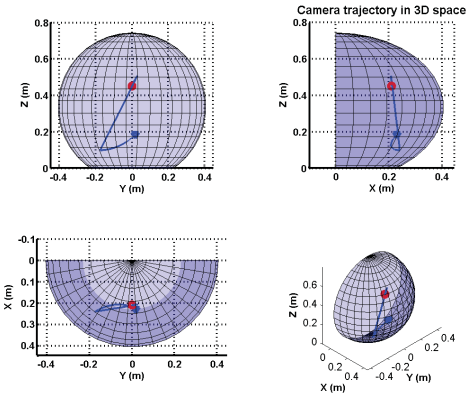
(a) Features position error



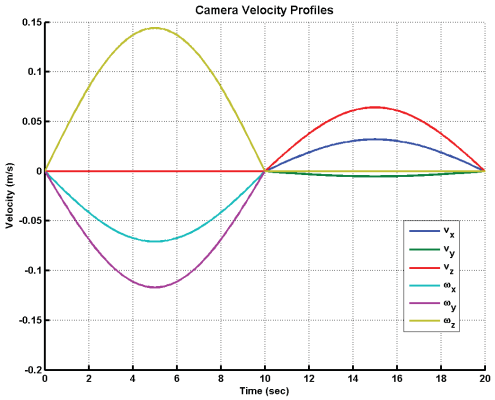
(b) Selected features error



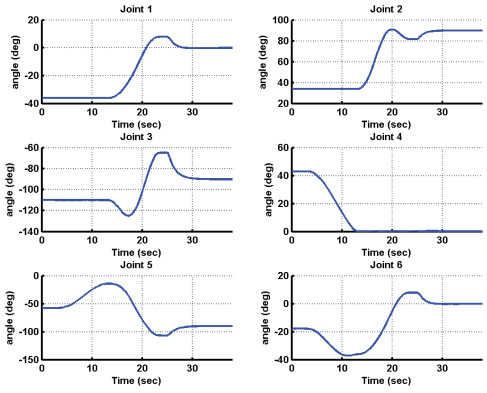
(c) Feature trajectory in image plane



(d) Camera 3D trajectory



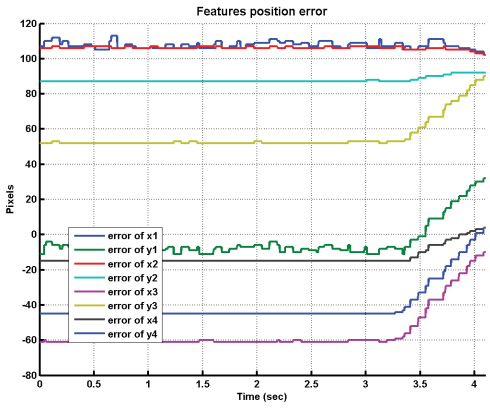
(e) Generated velocity profile



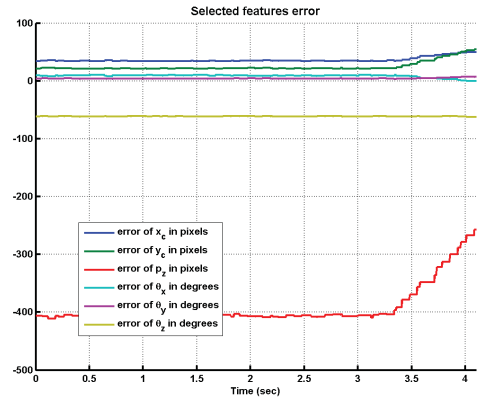
(f) Robot joint angles

Figure 5.16: Results for Test 2, performing a complicated visual servoing task including big orientation changes

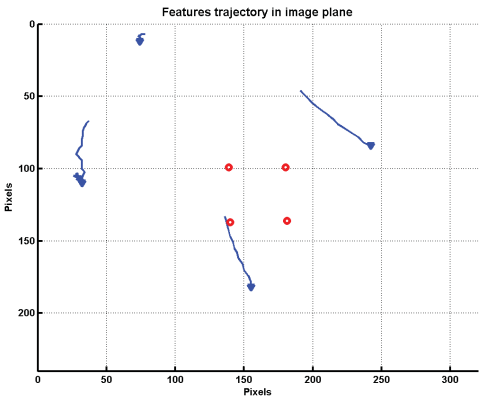
cause the features to move out of the FOV. However since this algorithm is an off line planning it only depends on the initial and desired location of the features. Within the algorithm it is assumed that the camera FOV is unlimited. The features eventually return to the real FOV of the camera as the robot completes the created path. The constant lines in the feature error and selected features error in Figures 5.16a and 5.15b are for the time when the features are out of FOV. It is shown that the task is completed keeping the robot in its workspace. The joint angles are also shown in Figure 5.16f. The same task is done using an IBVS controller. The results are given in Figures 5.17



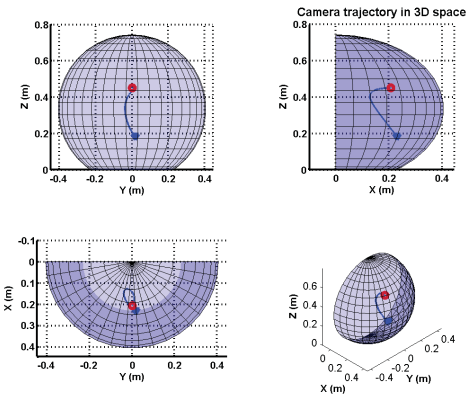
(a) Features position error



(b) Selected features error



(c) Feature trajectory in image plane



(d) Camera 3D trajectory

Figure 5.17: Results for Test 2 using IBVS controller

As shown in Figure 5.17c, the rotation required for this task takes the features

out of the field of view. The IBVS controller depends on the features position at each instant. As soon as the features run out of the field of view the controller have false data from the features position and it causes the task to fail.

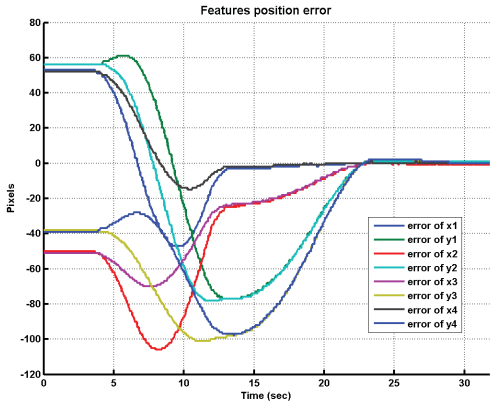
Test 3:

For the third test, another common problems of IBVS is investigated using the proposed method. The visual servoing fails when a 180 degrees rotation of the camera is required to reach its desired position [28]. A test is prepared including a 180 degrees rotation in the end effector motion. The initial and desired locations of the feature points are given in Table 5.2. The result of this test is shown in Figures 5.18.

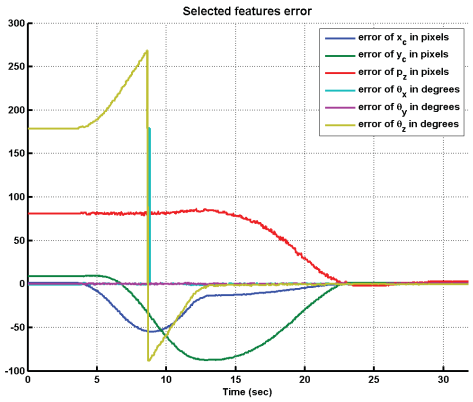
The same test is conducted using IBVS controllers. The results are shown in Figures 5.19. The results show that, similar to the previous test, the IBVS controller tries to match the features through the shortest path available which results in a motion of camera in the z_c direction. This continues until the end-effector reaches its physical limits and the robot stops, as shown in Figure 5.19d.

Test 4:

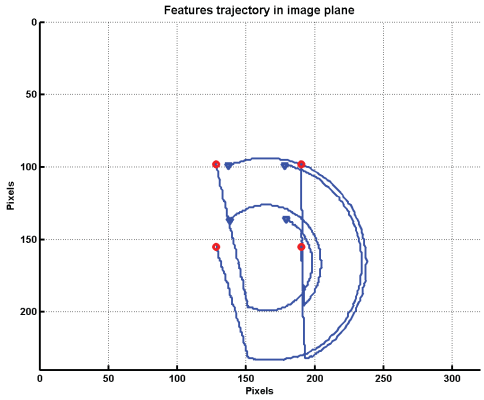
Another challenges in conventional visual servoing is the local minima problem. In an IBVS controller, the interaction matrix is an eight by six matrix. The inverse of this matrix, which is used to produce the controlling law, is an eight by six matrix and has two vector of null space. If the features error vector is a factor of these null space vectors, the controller generates a zero velocity vector as the controlling command. This causes the system to get stuck in that spot. In the trajectory planning algorithm, the inverse of the interaction matrix is not used. consequently, the local minima problem is solved. The next test demonstrates this ability in the proposed algorithm. The initial and desired locations of the feature points are given



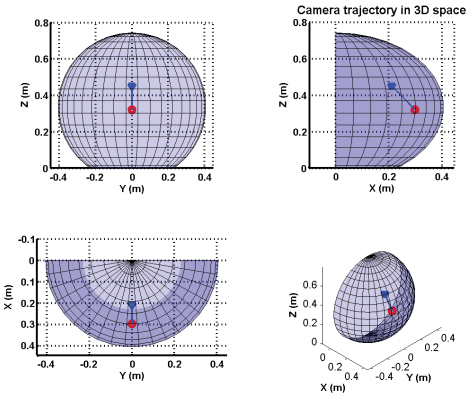
(a) Features position error



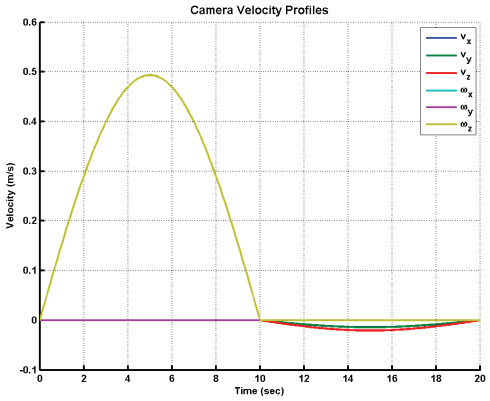
(b) Selected features error



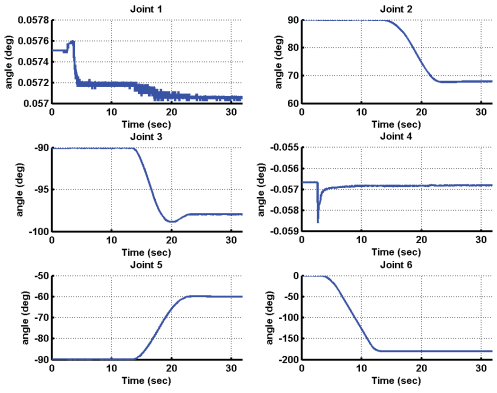
(c) Feature trajectory in image plane



(d) Camera 3D trajectory

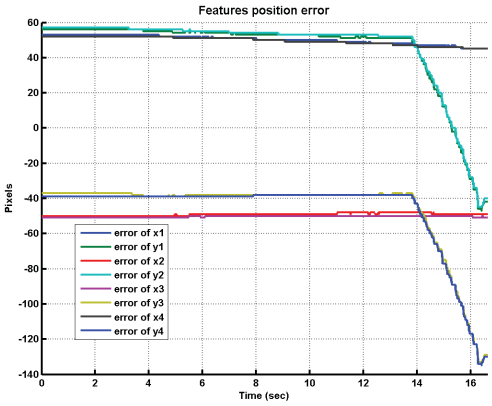


(e) Generated velocity profile

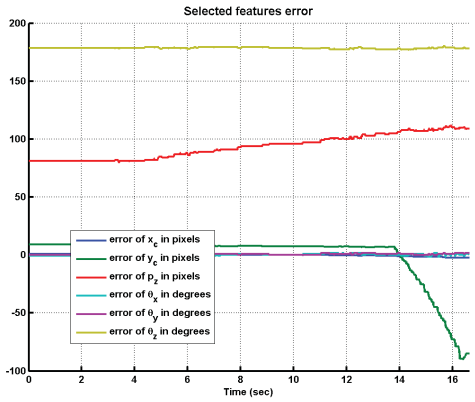


(f) Robot joint angles

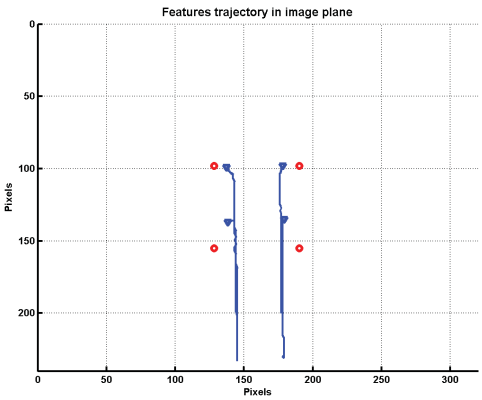
Figure 5.18: Results for Test 3, perforimg visual servoing task including 180^o rotation about camera's z_c axis



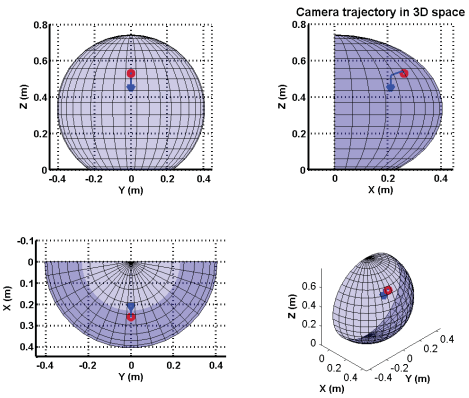
(a) Features position error



(b) Selected features error



(c) Feature trajectory in image plane



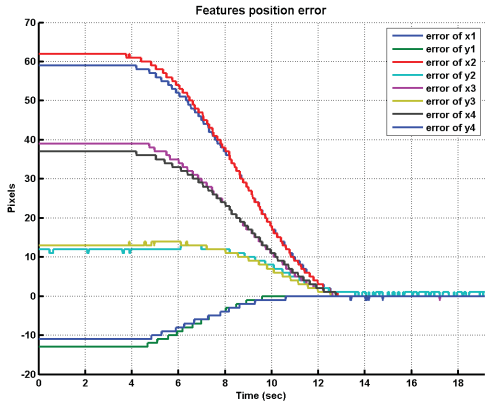
(d) Camera 3D trajectory

Figure 5.19: Results for Test 2 using IBVS controller

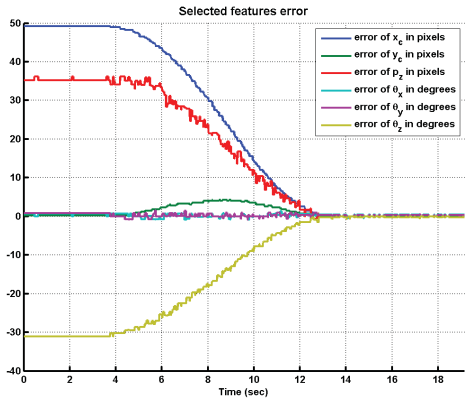
in Table 5.2. The desired features are chosen so that the vector of feature position error is in the null space of the interaction matrix. The results are shown in Figures 5.20. We can see that the proposed algorithm produces a velocity profile to take the robot to the desired position while the IBVS controller produces a zero velocity vector.

5.7 Summary

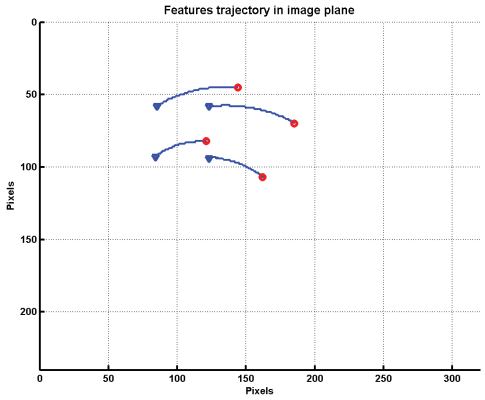
In this Chapter, a novel visual servoing technique is proposed. This technique is performed by planning a trajectory from the initial robot's position to a position where the image features match the desired ones. The trajectory is based on optimizing a predefined path which satisfies the system's initial and final conditions. The trajectory parameters are identified through an optimization procedure by minimizing the error between the image features and the desired ones. In order to speed up the optimization process, new features are introduced. This method successfully worked on a 4 DOFs robot. Due to the complexity of planning for a 6 DOFs robot, the planning procedure is decoupled to two stages of orientation planning and position planning. This is necessary to have a convex problem. A depth estimation method is proposed to provide the object depth to the trajectory planning algorithm. After performing the velocity profile generated from the trajectory planning algorithm, AIBVS controller is used to compensate for any probable errors appeared in matching the features with the desired ones. Experimental tests validate the proposed method and exhibit its advantages over IBVS controllers. The results show that in cases where the IBVS controller is unable to complete the visual servoing task, the proposed algorithm is successful.



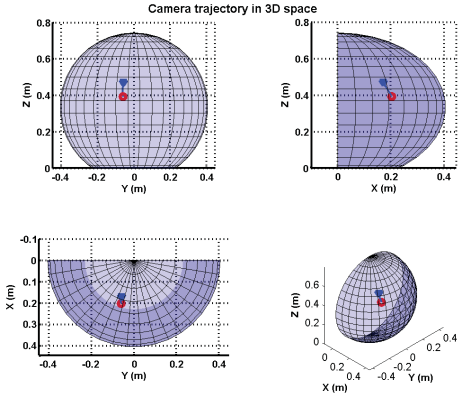
(a) Features position error



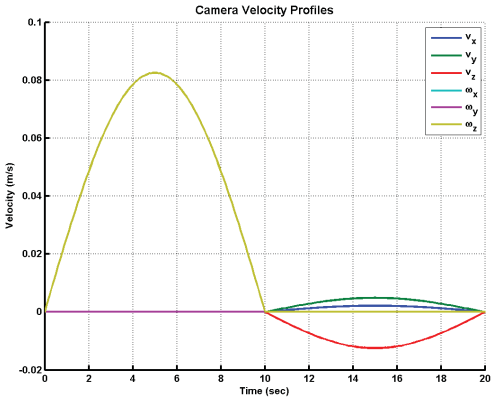
(b) Selected features error



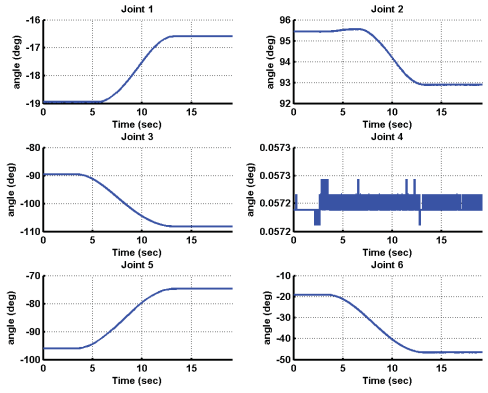
(c) Feature trajectory in image plane



(d) Camera 3D trajectory

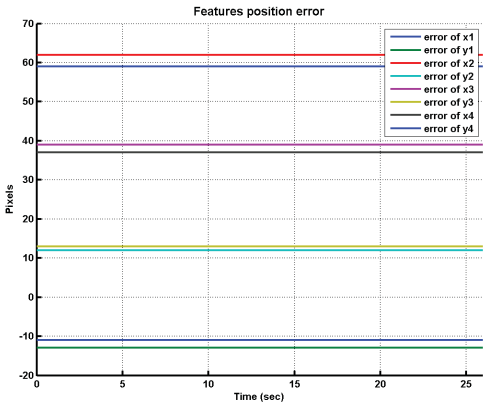


(e) Generated velocity profile

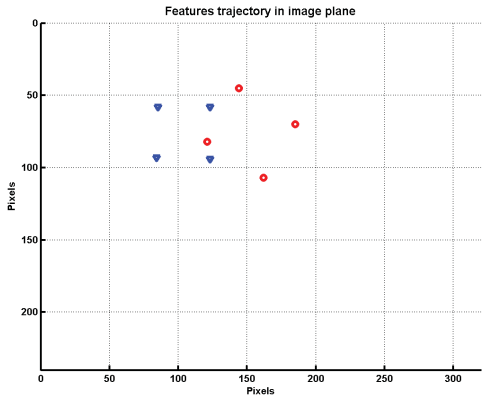


(f) Robot joint angles

Figure 5.20: Results for Test 4, performing a visual servoing task with desired features are located at the null space of the interaction matrix



(a) Features position error



(b) Selected features error

Figure 5.21: Results for Test 4 using IBVS controller

Chapter 6

Conclusion and Future Works

6.1 Summary of the thesis

Vision system is a relatively new technology that has improved the intelligence of automatic machines and robots. Vision system could be used in different applications such as surveillance, quality control, tracking, etc. In this thesis, the focus is on the use of vision system in a robotic system and guiding the robot based on the visual feedback. This is also called visual servoing. Researchers have introduced various methods to perform visual servoing tasks. In this thesis, existing problems and gaps of this field of research are discovered and some methodologies are proposed to solve them. This thesis aims in solving the existing problems by proposing new controlling and trajectory planning algorithms.

1. **Augmented image based visual servoing (AIBVS)**

First, an augmented image based visual servoing controller for a 6 DOFs robot is developed. Point features are used in this visual servoing system. In order to control all 6 DOFs, four feature points are chosen. A PD controller is used to create an acceleration profile for the robot controller. The visual servoing controller is designed so that the system error is reduced exponentially. The stability of the

visual servoing controller is proven using stability theorem of a perturb system. The visual servoing system is divided to two part naming the nominal system and the perturbed system. A Lyapanov function is introduced for the nominal part. The gain condition is extracted to have a stable system. The robot is controlled using a single joint control system. A PID controller is designed for each joint. Extensive experimental results validate the efficiency of the controller and show the advantages of the proposed AIBVS over the classic IBVS in terms of smoother motion in the image space and 3D space. The AIBVS controller moves the features in a more linear path than the IBVS controller. This improves the controller by reducing the risk of features leaving the field of view. This controller was also tested on a object catching application[78].

2. **Augmented image based visual servoing for image moment features**

   The AIBVS controller was developed for point features. However, in some cases point features are hard to extract from the image. Moreover, using point features could cause the system to get stuck in local minima. This controller is improved to work with image moment features. Image moments are the general image features that include points, lines and segments form of features. The visual servoing kinematic models are developed and the interaction matrix relating the image moment features to the acceleration screw is derived. The interaction matrix for the six chosen features are extracted. The first three features used in the controller are the x and y component of the image center and the area of the image. These three features control the X, Y and Z motion of the camera. To control the rotation of the camera three moment were chosen which are a combination of the second and third order moments. A PD control law is developed based on the system equation. The controller is validated in experiment.

3. **Catching moving object using visual servoing and navigation guidance techniques**

One of the fundamental capabilities of vision system is the ability to track and catch moving objects. Several applications could be named such as sorting objects moving on a conveyor. Therefore, a catching technique is developed in this thesis by utilizing the combination of AIBVS controller and navigation guidance technique. The navigation guidance algorithm generates the desired feature position for the visual servoing controller. The AIBVS controller follows the generated path to reduce its distance from the object and finally catch it. The navigation guidance technique is modified to create a smooth catching process. Simulation results validate the proposed algorithm. Three tests have been performed for catching an object with different types of motion such as linear motion, sinusoidal motion and a thrown object motion.

4. **Visual servoing using trajectory planning techniques**

A big drawback of visual servoing controller is that it can not guarantee global stability of the system. A lot of examples could be found that the system could not reach its desired position. The best solution to this problem is trajectory planning. In this thesis, a new trajectory planning algorithm is presented to overcome the mentioned problems. This technique is performed by planning a trajectory from the initial robot's position to a position where the image features match the desired ones. The trajectory is based on a predefined path which satisfies the system's initial and final conditions. In this project half cycle of sine wave is used as the predefined trajectory. The trajectory parameters are identified through an optimization procedure by minimizing the error between the image features and the desired ones. First, the planning algorithm is developed for a 4 DOFs robot. In order to speed up the optimization process, four new features are introduced. The first three feature corresponds to the x_c, y_c and z_c motion of the camera. The forth feature corresponds to the rotation of the camera about its z_c axis. Using these features the optimization problem becomes a convex problem. Applying the same algorithm to a

6 DOFs robot does not create a convex optimization problem. A decoupled planning algorithm was introduced to overcome the problem. The positioning problem was decoupled from the orientation planning. Six new features were presented to make possible the decoupling. Since the trajectory planning algorithm is highly dependent to the initial depth of the object, a depth estimation method is proposed to provide the object depth to the trajectory planning algorithm. The depth estimation algorithm is based on the constraint that exist between the location of the projection of a 3D point in two different images. After the trajectory is applied to the system a visual servoing controller is used to compensate for any probable errors appeared in matching the features with the desired ones. The AIBVS controller designed in this thesis is used for this matter. Experimental tests validate the proposed method and exhibit its advantages over IBVS controllers. Experimental tests show that in cases where visual servoing controller could not complete the visual servoing the trajectory planning algorithm completes the task. The most important task among all is the visual servoing tasks which include a rotation of 180 degrees about its center. This task was successfully performed using the trajectory planning algorithm.

6.2 Future Works

This thesis focuses on developing an AIBVS controller and a trajectory planning algorithm to stabilize the visual servoing system. The general visual task considered in the tests could be applied to pick and place applications. Future works include applying the developed visual servoing controller to the other robotic manufacturing tasks and solving the problems arising from these application. These applications could be named as welding, quality checking and vehicle navigation. Applications such as welding and quality control requires specific type of features. These new features lead to new structure of interaction matrix and thus introduces

new stability analysis problems.

In this project, the AIBVS controller has been developed for point features and image moment feature. However, while testing the proposed moments features proposed in the literature, some problems showed up. One of the problems is that the reverse contribution of the first and fourth moments and also the second and fifth moments on the acceleration command cause the system to stop without reaching the desired location. As a solution to this problem, other image moments could be explored and be used in AIBVS.

The trajectory planning algorithm presented in this thesis works with feature positions. A proposition for further work on this topic is adapting the trajectory planning algorithm to image moments features and other type of image features.

Bibliography

[1] A. Mohebbi, "Image based visual servoing using improved image moments in 6-dof robot systems," Master's thesis, Department of Mechanical and Industrial Engineering, Concordia University, Mar. 2008.

[2] J. Markoff, "Google cars drive themselves, in traffic," *Newyork Times*, p. A1, Oct., 9, 2010.

[3] Y. Shi, B. Liang, X. Wang, W. Xu, and H. Liu, "Study on intelligent visual servoing of space robot for cooperative target capturing," in *Information and Automation, ICIA, International Conference on*, Jun. 2012, pp. 733–738.

[4] S. Liu, "Image based visual servoing using improved image moments in 6-dof robot systems," Master's thesis, Department of Mechanical and Industrial Engineering, Concordia University, 2008.

[5] E. Malis, F. Chaumette, and S. Boudet, "$2\frac{1}{2}$d visual servoing," *Robotics and Automation, IEEE Transactions on*, vol. 15, no. 2, pp. 238–250, Apr. 1999.

[6] F. Chaumette, "Image moments: a general and useful set of features for visual servoing," *Robotics, IEEE Transactions on*, vol. 20, no. 4, pp. 713–723, Aug. 2004.

[7] S. Liu, W.-F. Xie, and C.-Y. Su, "Image-based visual servoing using improved image moments," in *Information and Automation, ICIA, International Conference on*, 2009, pp. 577–582.

[8] K. Hashimoto, "A review on vision based control of robot manipulators," *Advanced Robotics*, vol. 17, pp. 969–991, 2003.

[9] S. Hutchinson, G. Hager, and P. Corke, "A tutorial on visual servo control," *Robotics and Automation, IEEE Transactions on*, vol. 12, no. 5, pp. 651–670, Oct. 1996.

[10] P. Corke and S. Hutchinson, "A new partitioned approach to image based visual servo control," *Robotics and Automation, IEEE Transactions on*, vol. 17, no. 4, pp. 507–515, Aug. 2001.

[11] S. Hutchinson, G. Hager, and P. Corke, "A tutorial on visual servo control," *Robotics and Automation, IEEE Transactions on*, vol. 12, no. 5, pp. 651–670, Oct. 1996.

[12] F. Chaumette and S. Hutchinson, "Visual servo control. i. basic approaches," *Robotics Automation Magazine, IEEE*, vol. 13, no. 4, pp. 82–90, Dec. 2006.

[13] P. Corke and G. Hager, "Vision based robot control," in *Control Problems in Robotics and Automation*, ser. Lecture Notes in Control and Information Sciences, B. Siciliano and K. Valavanis, Eds. Springer Berlin Heidelberg, 1998, vol. 230, pp. 177–192.

[14] Z. Li, W. Xie, and X. Tu, "Switching control of image based visual servoing with laser pointer in robotic assembly systems," in *Systems, Man and Cybernetics, 2007. ISIC. IEEE International Conference on*, Oct. 2007, pp. 2383–2389.

[15] Y. Song, M. Li, L. Sun, and J. Ji, "Global visual servoing of miniature mobile robot inside a micro-assembly station," in *Mechatronics and Automation, IEEE International Conference*, vol. 3, 2005, pp. 1586–1591 Vol. 3.

[16] H. Chen, J. Li, X. Zhang, and Z. Deng, "Application of visual servoing to an x-ray based welding inspection robot," in *Control and Automation, ICCA, International Conference on*, vol. 2, Jun. 2005, pp. 977–982 Vol. 2.

[17] Q. Chen, S. Zhu, X. Wang, and W. Wu, "Analysis on an uncalibrated image based visual servoing for 6 dof industrial welding robots," in *Mechatronics and Automation, ICMA, International Conference on*, Aug. 2012, pp. 2013–2018.

[18] A. Ghanbari, W. Wang, C. Hann, J. Chase, and X. Chen, "Cell image recognition and visual servo control for automated cell injection," in *Autonomous Robots and Agents, ICARA, 4th International Conference on*, Feb. 2009, pp. 92–96.

[19] A. Krupa, C. Doignon, J. Gangloff, and M. de Mathelin, "Combined image based and depth visual servoing applied to robotized laparoscopic surgery," in *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, vol. 1, 2002, pp. 323–329.

[20] G.-Q. Wei, K. Arbter, and G. Hirzinger, "Real-time visual servoing for laparoscopic surgery. controlling robot motion with color image segmentation," *Engineering in Medicine and Biology Magazine, IEEE*, vol. 16, no. 1, pp. 40–45, Jan. 1997.

[21] R. Rao, V. Kumar, and C. Taylor, "Visual servoing of a ugv from a uav using differential flatness," in *Intelligent Robots and Systems, IROS, Proceedings, IEEE/RSJ, International Conference on*, vol. 1, Oct. 2003, pp. 743–748.

[22] Y. Shirai and H. Inoue, "Guiding a robot by visual feedback in assembling tasks," *Pattern Recognition*, vol. 5, no. 2, pp. 99–108, 1973.

[23] J. Hill and W. T. Park, "Real time control of a robot with a mobile camera," in *9th ISIR*.

[24] L. Weiss, A. C. Sanderson, and C. P. Neuman, "Dynamic sensor-based control of robots with visual feedback," *IEEE Journal on Robotics and Automation*, vol. RA-3, no. 5, Oct. 1987.

[25] K. Hashimoto, *Visual Servoing: Real-Time Control of Robot Manipulators Based on Visual Sensory Feedback*, ser. World Scientific series in robotics and automated systems. World Scientific, 1993.

[26] L. Deng, F. Janabi-Sharifi, and W. Wilson, "Hybrid motion control and planning strategies for visual servoing," *Industrial Electronics, IEEE Transactions on*, vol. 52, no. 4, pp. 1024–1040, Aug. 2005.

[27] F. Janabi-Sharifi, *Visual Servoing: Theory and Applications*. Opto-Mechatronic Systems Handbook, 2002.

[28] F. Chaumette, "Potential problems of stability and convergence in image based and position based visual servoing," in *The Confluence of Vision and Control*, D. Kriegman, G. . Hager, and A. Morse, Eds. LNCIS Series, No. 237, Springer-Verlag, 1998, pp. 66–78.

[29] L. Deng, F. Janabi, and W. Wilson, "Stability and robustness of visual servoing methods," in *Robotics and Automation, ICRA, IEEE International Conference on*, vol. 2, 2002, pp. 1604–1609 vol.2.

[30] L. Deng, "Comparison of image based and position based robot visual servoing methods and improvements," Ph.D. dissertation, Waterloo, Ont., Canada, Canada, 2004.

[31] E. Malis, "Visual servoing invariant to changes in camera-intrinsic parameters," *Robotics and Automation, IEEE Transactions on*, vol. 20, no. 1, pp. 72–81, Feb. 2004.

[32] N. Gans and S. Hutchinson, "A switching approach to visual servo control," in *Intelligent Control, IEEE International Symposium*, 2002, pp. 770–776.

[33] J. Wang and H. Cho, "Micropeg and hole alignment using image moments based visual servoing method," *Industrial Electronics, IEEE Transactions on*, vol. 55, no. 3, pp. 1286–1294, Mar. 2008.

[34] Q. Yancheng, W. Changhong, W. Qiyong, and M. Guangcheng, "A new robust visual servoing algorithm: theory and experiment," in *Control Applications, CCA, IEEE Conference on*, vol. 2, Jun. 2003, pp. 1459–1462 vol.2.

[35] G. Morel, P. Zanne, and F. Plestan, "Robust visual servoing: bounding the task function tracking errors," *Control Systems Technology, IEEE Transactions on*, vol. 13, no. 6, pp. 998–1009, Nov. 2005.

[36] G. Allibert, E. Courtial, and F. Chaumette, "Predictive control for constrained image-based visual servoing," *Robotics, IEEE Transactions on*, vol. 26, no. 5, pp. 933–939, Oct. 2010.

[37] J.-K. Kim, D.-W. Kim, S.-J. Choi, and S.-C. Won, "Image based visual servoing using sliding mode control," in *SICE-ICASE, International Joint Conference*, Oct. 2006, pp. 4996–5001.

[38] W.-F. Xie, Z. Li, X.-W. Tu, and C. Perron, "Switching control of image-based visual servoing with laser pointer in robotic manufacturing systems," *Industrial Electronics, IEEE Transactions on*, vol. 56, no. 2, pp. 520–529, Feb. 2009.

[39] R. Tsai and R. Lenz, "A new technique for fully autonomous and efficient 3d robotics hand/eye calibration," *Robotics and Automation, IEEE Transactions on*, vol. 5, no. 3, pp. 345–358, Jun. 1989.

[40] R. Tsai, "A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses," *Robotics and Automation, IEEE Journal of*, vol. 3, no. 4, pp. 323–344, Aug. 1987.

[41] G. Chesi and K. Hashimoto, "Static-eye against hand-eye visual servoing," in *Decision and Control, CDC, 41st IEEE Conference on*, vol. 3, Dec. 2002, pp. 2854–2859 vol.3.

[42] G. Flandin, F. Chaumette, and E. Marchand, "Eye-in-hand/eye-to-hand cooperation for visual servoing," in *Robotics and Automation, ICRA, IEEE International Conference on*, vol. 3, 2000, pp. 2741–2746 vol.3.

[43] V. Lippiello, B. Siciliano, and L. Villani, "Eye-in-hand/eye-to-hand multi-camera visual servoing," in *Decision and Control, European Control Conference, CDC-ECC, 44th IEEE Conference on*, Dec. 2005, pp. 5354–5359.

[44] E. Malis, "Vision based control invariant to camera intrinsic parameters: stability analysis and path tracking," in *Robotics and Automation, ICRA, IEEE International Conference on*, vol. 1, 2002, pp. 217–222 vol.1.

[45] H. Sutanto, R. Sharma, and V. Varma, "Image based autodocking without calibration," in *Robotics and Automation, IEEE International Conference on*, vol. 2, Apr. 1997, pp. 974–979 vol.2.

[46] J. Qian and J. Su, "Online estimation of image jacobian matrix by kalman-bucy filter for uncalibrated stereo vision feedback," in *Robotics and Automation, ICRA, IEEE International Conference on*, vol. 1, 2002, pp. 562–567 vol.1.

[47] D. Kragic and H. I. Christensen, "Survey on visual servoing for manipulation," Computational Vision and Active Perception Laboratory, Tech. Rep., 2002.

[48] O.-E. Ng and V. Ganapathy, "A novel modular framework for stereo vision," in *Advanced Intelligent Mechatronics, AIM, IEEE/ASME International Conference on*, Jul. 2009, pp. 857–862.

[49] E. Malis, F. Chaumette, and S. Boudet, "Multi-cameras visual servoing," in *Robotics and Automation, ICRA, IEEE International Conference on*, vol. 4, 2000, pp. 3183–3188 vol.4.

[50] H. Li, M. Jin, and L. Zou, "A new binocular stereo visual servoing model," in *Computational Intelligence and Industrial Application, PACIIA, Pacific-Asia Workshop on*, vol. 1, Dec. 2008, pp. 461–465.

[51] Y. Zhao, W.-F. Xie, and X.-W. Tu, "Multiple cameras visual servoing used for large scale 3d positioning," in *Intelligent Control and Automation, WCICA, 9th World Congress on*, Jun. 2011, pp. 462–467.

[52] G. Buttazzo, B. Allotta, and F. Fanizza, "Mousebuster: a robot for real-time catching," *Control Systems, IEEE*, vol. 14, no. 1, pp. 49–56, Feb. 1994.

[53] D. Fernandes and P. Lima, "A testbed for robotic visual servoing and catching of moving objects," in *Electronics, Circuits and Systems, 1998 IEEE International Conference on*, vol. 2, 1998, pp. 475–478 vol.2.

[54] E. Croft, R. Fenton, and B. Benhabib, "Time-optimal interception of objects moving along topologically varying paths," in *Systems, Man and Cybernetics, Intelligent Systems for the 21st Century, IEEE International Conference on*, vol. 5, Oct. 1995, pp. 4089–4094.

[55] A. Namiki and M. Ishikawa, "Vision-based online trajectory generation and its application to catching," in *Springer Tracts in Advanced Robotics*, vol. 5, 2003.

[56] A. Masoud and M. Bayoumi, "Intercepting a maneuvering target in a multidimensional stationary environment using a wave equation potential field strategy," in *Intelligent Control, IEEE International Symposium on*, 1994, pp. 243–248.

[57] J. M. Borg, M. Mehrandezh, R. G. Fenton, and B. Benhabib, "Navigation-guidance-based robotic interception of moving objects in industrial settings," *Journal of Intelligent Robotics Systems*, vol. 33, no. 1, pp. 1–23, 2002.

[58] Y. Mezouar and F. Chaumette, "Path planning for robust image based control," *Robotics and Automation, IEEE Transactions on*, vol. 18, no. 4, pp. 534–549, Aug. 2002.

[59] L. Ge and Z. Jie, "A real-time stereo visual servoing for moving object grasping based parallel algorithms," in *Industrial Electronics and Applications, ICIEA, 2nd IEEE Conference on*, May 2007, pp. 2886–2891.

[60] M. Asada, T. Tanaka, and K. Hosoda, "Visual tracking of unknown moving object by adaptive binocular visual servoing," in *Multisensor Fusion and Integration for Intelligent Systems, MFI, IEEE/SICE/RSJ International Conference on*, 1999, pp. 249–254.

[61] M. Keshmiri, M. Keshmiri, and A. Mohebbi, "Augmented online point to point trajectory planning, a new approach in catching a moving object by a manipulator," in *Control and Automation, ICCA, 8th IEEE International Conference on*, Jun. 2010, pp. 1349–1354.

[62] J. Latombe, *Robot Motion Planning*, ser. Kluwer international series in engineering and computer science: Robotics. Springer, 1990.

[63] S. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.

[64] Y. Mezouar and F. Chaumette, "Optimal camera trajectory with image-based control," *The International Journal of Robotics Research*, vol. 22, no. 10-11, pp. 781–803, 2003.

[65] G. Chesi, "Visual servoing path planning via homogeneous forms and lmi optimizations," *Robotics, IEEE Transactions on*, vol. 25, no. 2, pp. 281–291, Apr. 2009.

[66] G. Chesi and Y. Hung, "Global path-planning for constrained and optimal visual servoing," *Robotics, IEEE Transactions on*, vol. 23, no. 5, pp. 1050–1060, Oct. 2007.

[67] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, 2004.

[68] K. Hosoda, K. Sakamoto, and M. Asada, "Trajectory generation for obstacle avoidance of uncalibrated stereo visual servoing without 3d reconstruction," in *Intelligent Robots and Systems, IEEE/RSJ International Conference on*, vol. 1, Aug. 1995, pp. 29–34.

[69] J. S. Park and M. J. Chung, "Path planning with uncalibrated stereo rig for image based visual servoing under large pose discrepancy," *Robotics and Automation, IEEE Transactions on*, vol. 19, no. 2, pp. 250–258, Apr. 2003.

[70] M. Keshmiri, W.-F. Xie, and A. Mohebbi, "Augmented image-based visual servoing of a manipulator using acceleration command," *Industrial Electronics, IEEE Transactions on*, vol. 61, no. 10, pp. 5444–5452, Oct. 2014.

[71] M. Keshmiri and W.-F. Xie, "Augmented imaged based visual servoing controller for a 6 dof manipulator using acceleration command," in *Decision and Control, CDC, IEEE 51st Annual Conference on*, Dec. 2012, pp. 556–561.

[72] ——, "Augmented image based visual servoing using image moment features," in *ASME International Mechanical Engineering Congress & Exposition, IMECE*, Nov. 2014.

[73] ——, "Visual servoing using an optimized trajectory planning technique for a 4 dofs robotic manipulator," *Submitted to Journal of Intelligent & Robotic Systems*, 2014.

[74] ——, "Visual servoing of a robotic manipulator using an optimized trajectory planning technique," in *Electrical and Computer Engineering, CCECE, 27th Annual IEEE Canadian Conference on*, May 2014.

[75] ——, "Visual servoing using a trajectory optimization technique," *Submitted to Mechatronics, IEEE/ASME Transactions on*, 2014.

[76] A. Hajiloo, M. Keshmiri, and W.-F. Xie, "Real-time model predictive visual servoing controller," *To be submitted to Industrial Electronics, IEEE Transactions on*, 2014.

[77] A. Mohebbi, M. Keshmiri, and W.-F. Xie, "Tracking and catching moving objects through image based robot visual servoing: Binocular vs. monocular," *Submitted to Journal of Intelligent & Robotic Systems*, 2014.

[78] M. Keshmiri and W. F. Xie, "Catching moving objects using a navigation guidance technique in a robotic visual servoing system," in *American Control Conference, ACC*, Jun. 2013, pp. 6302–6307.

[79] A. Mohebbi, M. Keshmiri, and W.-F. Xie, "Eye-in-hand image based stereo visual servoing for tracking and grasping moving objects," in *Control Conference, CCC, 32nd Chinese*, Jul. 2014.

[80] ——, "An acceleration command approach to stereo image based robot visual servoing," in *19th IFAC World Congress*, Aug. 2014.

[81] M. Keshmiri, A. F. Jahromi, A. Mohebbi, M. H. Amoozgar, and W.-F. Xie, "Modeling and control of ball and beam system using model based and non-model based control approaches," *International Journal of Smart Sensing and Intelligent systems*, vol. 5, no. 1, pp. 14–35, Mar. 2012.

[82] X.-M. Ma, W.-F. Xie, M. Keshmiri, and A. Mohebbi, "Wavelet-based linearization for single-degree-of-freedom nonlinear systems," in *Intelligent Robotics and Applications*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2012, vol. 7506, pp. 99–110.

[83] H. Wang, Y.-H. Liu, and D. Zhou, "Adaptive visual servoing using point and line features with an uncalibrated eye-in-hand camera," *Robotics, IEEE Transactions on*, vol. 24, no. 4, pp. 843–857, Aug. 2008.

[84] Y. Fang, X. Liu, and X. Zhang, "Adaptive active visual servoing of nonholonomic mobile robots," *Industrial Electronics, IEEE Transactions on*, vol. 59, no. 1, pp. 486–497, Jan. 2012.

[85] J.-K. Kim, D.-W. Kim, S.-J. Choi, and S.-C. Won, "Image based visual servoing using sliding mode control," in *SICE-ICASE, International Joint Conference*, Oct. 2006, pp. 4996–5001.

[86] H. Fakhry and W. Wilson, "A modified resolved acceleration controller for position based visual servoing," *Mathematical and Computer Modelling*, vol. 24, no. 5, pp. 1–9, Sep. 1996.

[87] V. Mark W. Spong, Hutchinson, *Robot Modeling and Control*, illustrated ed. New York, USA: John Wiley & Sons Inc., 2006.

[88] X. Liang, X. Huang, M. Wang, and X. Zeng, "Adaptive task-space tracking control of robots without task-space and joint-space-velocity measurements," *Robotics, IEEE Transactions on*, vol. 26, no. 4, pp. 733–742, Aug. 2010.

[89] 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 Association for Computing Machinery, ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981.

[90] J. L. Meriam and L. G. Kraige, *Engineering mechanics. Vol. 2, Dynamics*, 6th ed. Hoboken, NJ, USA: John Wiley & Sons Inc., 2008.

[91] J. Craig, *Introduction to Robotics: Mechanics and Control*, ser. Addison-Wesley series in electrical and computer engineering: control engineering. Pearson Education, Incorporated, 2005.

[92] H. Khalil, *Nonlinear Systems*. Prentice Hall, 2002.

[93] T. A. Clarke and J. G. Fryer, "The development of camera calibration methods and models," *The Photogrammetric Record*, vol. 16, no. 91, pp. 51–66, 1998.

[94] Z. Zhang, "Flexible camera calibration by viewing a plane from unknown orientations," in *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, vol. 1, 1999, pp. 666–673 vol.1.

[95] A. Fetic, D. Juric, and D. Osmankovic, "The procedure of a camera calibration using camera calibration toolbox for matlab," in *MIPRO, 35th International Convention*, May 2012, pp. 1752–1757.

[96] P. I. Corke and B. Armstrong-Helouvry, "A meta-study of puma 560 dynamics: A critical appraisal of literature data," *Robotica*, vol. 13, no. 03, pp. 253–258, 1995.

[97] P. Corke, "A robotics toolbox for MATLAB," *IEEE Robotics and Automation Magazine*, vol. 3, no. 1, pp. 24–32, Mar. 1996.

[98] ——, "Machine vision toolbox," *IEEE Robotics and Automation Magazine*, vol. 12, no. 4, pp. 16–25, Nov. 2005.

[99] Y. Zhao, W.-F. Xie, and S. Liu, "Image based visual servoing using improved image moments in 6-dof robot systems," *International Journal of Control, Automation and Systems*, vol. 11, no. 3, pp. 586–596, 2013.

[100] Y.-M. Zhao, W.-F. Xie, S. Liu, and T. Wang, "Neural network-based image moments for robotic visual servoing," *Journal of Intelligent & Robotic Systems*, pp. 1–18, 2014.

[101] J. Flusser, B. Zitova, and T. Suk, *Moments and Moment Invariants in Pattern Recognition*. Wiley Publishing, 2009.

[102] M.-K. Hu, "Visual pattern recognition by moment invariants," *Information Theory, IRE Transactions on*, vol. 8, no. 2, pp. 179–187, Feb. 1962.

[103] N. Houshangi, "Control of a robotic manipulator to grasp a moving target using vision," in *Robotics and Automation Proceedings, IEEE International Conference on*, vol. 1, May 1990, pp. 604–609.

[104] J. Feddema and O. Mitchell, "Vision-guided servoing with feature based trajectory generation for robots," *Robotics and Automation, IEEE Transactions on*, vol. 5, no. 5, pp. 691–700, Oct. 1989.

[105] M. Mehrandezh, M. Sela, R. Fenton, and B. Benhabib, "Robotic interception of moving objects using ideal proportional navigation guidance technique," *Robotics and Autonomous Systems*, vol. 28, no. 4, pp. 295–310, 1999.

[106] D. Chwa, J. Kang, and J. Y. Choi, "Online trajectory planning of robot arms for interception of fast maneuvering object under torque and velocity constraints," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 35, no. 6, pp. 831–843, Nov. 2005.

[107] M. Keshmiri and M. Keshmiri, "Performance comparison of various navigation guidance methods in interception of a moving object by a serial manipulator considering its kinematic and dynamic limits," in *Methods and Models in Automation and Robotics, MMAR, 15th International Conference on*, 2010, pp. 212–217.

[108] A. Namiki, T. Senoo, S. Mizusawa, and M. Ishikawa, "High-speed visual feedback control for grasping and manipulation," in *Visual Servoing via Advanced Numerical Methods*, ser. Lecture Notes in Control and Information Sciences, G. Chesi and K. Hashimoto, Eds. Springer Berlin / Heidelberg, 2010, vol. 401, pp. 39–53.

[109] O. Tahri and F. Chaumette, "Point based and region based image moments for visual servoing of planar objects," *Robotics, IEEE Transactions on*, vol. 21, no. 6, pp. 1116–1127, 2005.

[110] M. Keshmiri, M. Keshmiri, and A. Mohebbi, "Augmented online point to point trajectory planning, a new approach in catching a moving object by a manipulator," in *Control and Automation, ICCA, 8th IEEE International Conference on*, 2010, pp. 1349–1354.

[111] S. Masoud and A. Masoud, "Motion planning in the presence of directional and regional avoidance constraints using nonlinear, anisotropic, harmonic potential fields: a physical metaphor," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 32, no. 6, pp. 705–723, 2002.

[112] S. Zhang, C. Wang, and S. Chan, "A new high resolution depth map estimation system using stereo vision and kinect depth sensing," *Journal of Signal Processing Systems*, pp. 1–13, 2013.

[113] C. Strecha and L. Gool, "Motion âĂŤ stereo integration for depth estimation," in *Computer Vision, ECCV*, ser. Lecture Notes in Computer Science, A. Heyden, G. Sparr, M. Nielsen, and P. Johansen, Eds. Springer Berlin Heidelberg, 2002, vol. 2351, pp. 170–185.

[114] B. Barrois and C. Wohler, "3d pose estimation of vehicles using stereo camera," in *Transportation Technologies for Sustainability*, M. Ehsani, F.-Y. Wang, and G. Brosch, Eds. Springer New York, 2013, pp. 1–24.

[115] J. Arora, *Introduction to Optimum Design*. Elsevier Science, 2004.

[116] J. W. Chinneck, "Discovering the characteristics of mathematical programs via sampling," *Optimization Methods and Software*, vol. 17, no. 2, pp. 319–352, 2002.

[117] G. J. Scolnik H., "A new method to compute second derivatives," vol. 1, no. 6.

[118] R. Byrd, M. Hribar, and J. Nocedal, "An interior point algorithm for large-scale nonlinear programming," *SIAM Journal on Optimization*, vol. 9, no. 4, pp. 877–900, 1999.

Appendix A

Denso VS-6556G

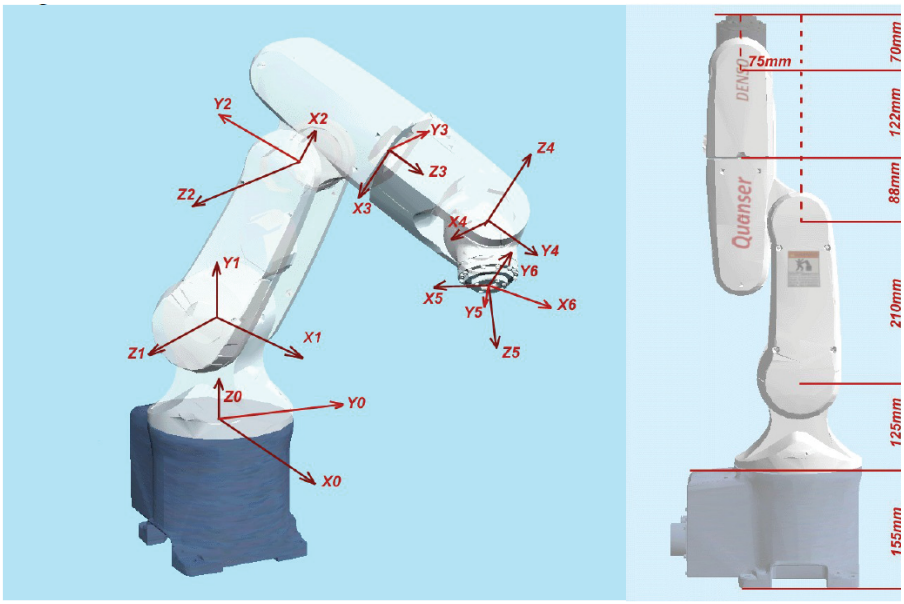
This appendix is devoted to presenting some general specifications of the DENSO VS-6556G robot.

The Denso VS-6556G is a 6 DOF robotic arm. The Denso arm consists of six joints and corresponding six encoders that measure the angular position of the six motors. The encoders and motors specifications are summarized in Table A.1 The encoders resolution, motors gear ratios, motors torque constants, and joints hard stop limits are listed in this Table.

Figure A.1b, demonstrate the robot joint coordinate systems including the world frame 0 and the joint frames which are used to define the forward kinematics, inverse kinematics and the Jacobian matrix. Link lengths are also illustrated in Figure A.1, where the robot is in a completely straightened up situation. In this configuration, all the joints encoder values are zero and the axes in frames 1, 3, 4, 5, and 6 are parallel to their counter part axes in global frame. The joints 2, 3, and 5 are zero when the robot is completely straightened up as depicted in Figure A.1. Figure A.2 demonstrates the Denso robot workspace from right and top view.

Table A.1: Denso Robot Specifications

Motor / Encoder Starting with Base	*Motor Gear Ratio*	*Encoder Calibration (count/deg)*	*Torque Constant N.m/Amp*	*Joint maximum stop limit (deg)*	*Joint minimum stop limit (deg)*
1	120:1	43690.666670	0.38	160	-160
2	160:1	58254.222220	0.38	120	-120
3	120:1	43690.666670	0.22	160	20
4	100:1	36408.888890	0.21	160	-160
5	100:1	36408.888890	0.21	120	-120
6	100:1	36408.888890	0.21	360	-360



(a) Denso robots joint frames

(b) Denso robot's arm lengths

Figure A.1: Denso robot joint frames and links length

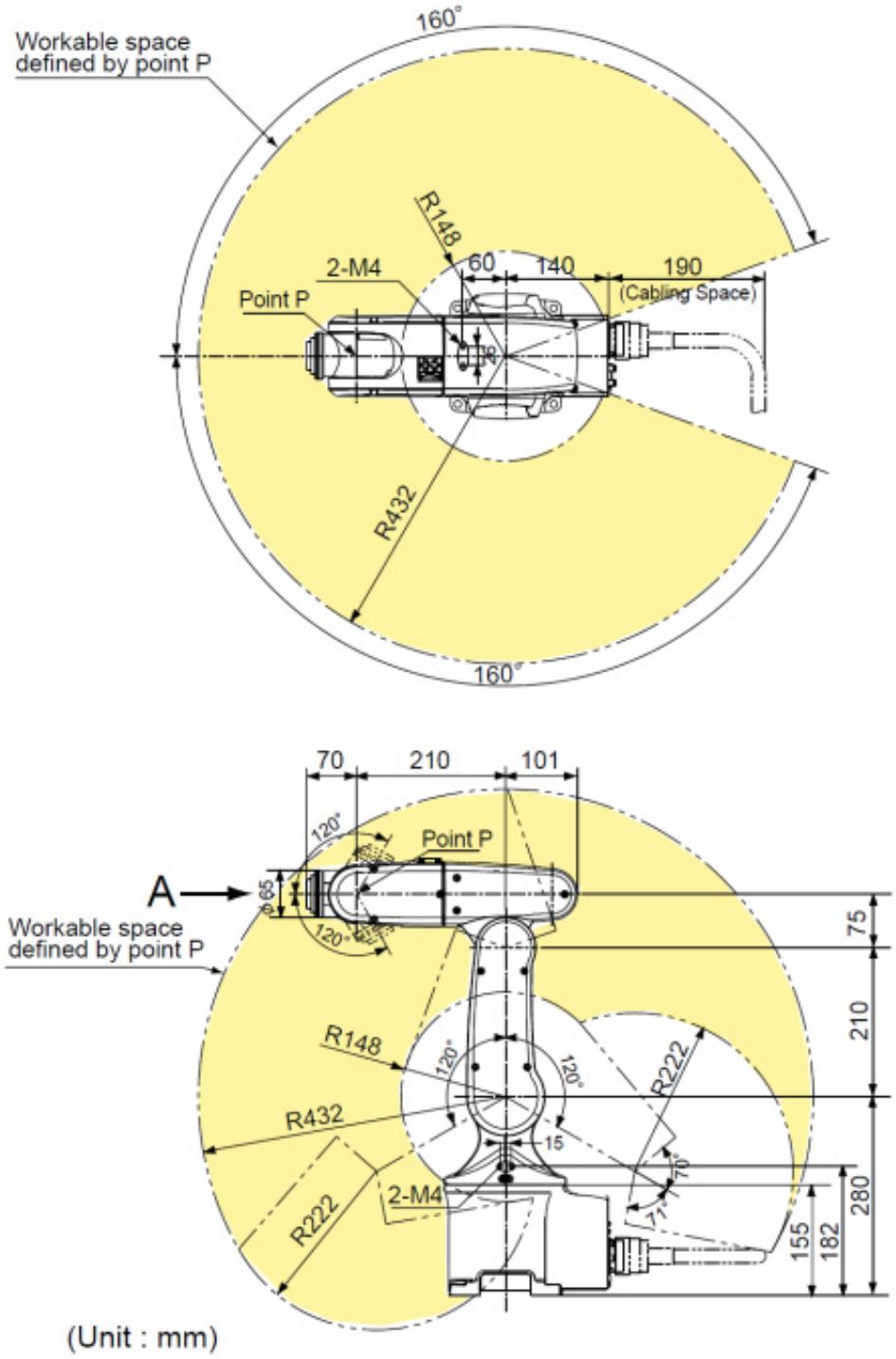


Figure A.2: Denso robot workspace

Appendix B

Image Moment Velocity Interaction Matrix

The elements of $\mathbf{O}_{m_{ij}}$ matrix introduced in equation (3.19) is given by the followings $o_{m_{11}} = o_{m_{12}} = 0$

$o_{m_{13}} = -(i+1)2ACm_{ij} - (2+i)A^2m_{i+1,j} - iC^2m_{i-1,j} - (i+1)2ABm_{i,j+1} - iB^2m_{i-1,j+2} - 2iBCm_{i-1,j+1}$

$o_{m_{14}} = -Am_{ij+1} - iAm_{i,j+1} - iBm_{i-1j+2} - iCm_{i-1,j+1}$

$o_{m_{15}} = (3 + 2i + j)Cm_{ij} + (5 + 2i + j)Am_{i+1,j} + (4 + 2i + j)Bm_{i,j+1}$

$o_{m_{16}} = (1 + j)Bm_{ij} + jCm_{i,j-1} + jAm_{i+1,j-1}$

$o_{m_{21}} = o_{m_{22}} = 0$

$o_{m_{23}} = -(i + j)2BCm_{ij} - (2 + j)B^2m_{i,j+1} - jC^2m_{i,j-1} - (1 + j)2ABm_{i+1,j} - jA^2m_{i+2,j-1} - 2jACm_{i+1,j-1}$

$o_{m_{24}} = -(3 + i + 2j)Cm_{ij} - (4 + i + 2j)Am_{i+1,j} - (5 + i + 2j)Bm_{i,j+1}$

$o_{m_{25}} = (1 + j)Bm_{i+1,j} + jAm_{i+2,j-1} + jCm_{i+1,j-1}$

$o_{m_{26}} = -(1 + i)Am_{ij} - iCm_{i-1,j} - iBm_{i-1,j+1}$

$o_{m_{33}} = -4C^2m_{ij} + (8+2i+2j)(A^2m_{i+2,j} + B^2m_{i,j+2} + C^2m_{i,j}) + (12+4i+4j)(ACm_{i+1,j} +$

$BCm_{i,j+1} + ABm_{i+1,j+1}) + 4ABm_{i+1,j+1}$

$o_{m_{34}} = (8 + 2i + 2j)(Cmi, j + 1 + Bm_{i,j+2} + Am_{i+1,j+1}) - 2Cm_{i,j+1}$

$o_{m_{35}} = -(8 + 2i + 2j)(Cmi + 1, j + Bm_{i+1,j+1} + Am_{i+2,j}) + 2Cm_{i+1,j}$

$o_{m_{36}} = (1 + i)Am_{i,j+1} - (1 + j)Bm_{i+1,j} - j(Am_{i+2,j-1} + Cm_{i+1,j-1})$

$o_{m_{44}} = (3 + i + 2j)m_{ij} + (8 + 2i + 2j)m_{i,j+2}$

$o_{m_{45}} = -(8 + 2i + 2j)m_{i+1,j+1} - 0.5im_{i-1,j+1} - 0.5jm_{i+1,j-1}$

$o_{m_{46}} = 0.5im_{i-1,j} - (2.5 + 0.5i + 1.5j)m_{i+1,j} + im_{i-1,j+2}$

$o_{m_{55}} = (3 + 2i + j)m_{ij} + (8 + 2i + 2j)m_{i+2,j}$

$o_{m_{56}} = 0.5jm_{i,j-1} - (2.5 + 1.5i + 0.5j)m_{i,j+1} + jm_{i+2,j-1}$

$o_{m_{66}} = -(2 + i + j)m_{ij}$. and

$$o_{m_{pq}} = o_{m_{qp}} \tag{1}$$

which indicated that the matrix $\mathbf{O}_{m_{ij}}$ is a symmetric matrix.

The elements of $\overline{\mathbf{O}}_{\mu_{ij}}$ matrix introduced in equation (3.30) is given by the followings $\overline{\mathbf{O}}_{\mu_{ij_{11}}} = 2Ai(x - xg)^{i-1}(y - yg)^j/Z_g m_{00}$

$\overline{\mathbf{O}}_{\mu_{ij_{12}}} = (x - xg)^{i-1}(y - yg)^{j-1}(Ajx - Ajx_g + Biy - Biy_g))/Z_g m00$

$\overline{\mathbf{O}}_{\mu_{ij_{13}}} = -2A(x - xg)^i(y - yg)^j(C + Ax + By) - (i(x - xg)^{i-1}(y - yg)^j(A^2m_{00}^3x^2 - 3m_{20}A^2m_{00}^2 + 2A^2m_{00}m_{10}^2 + 10m_{20}A^2m_{00} - 6A^2m_{10}^2 + 2ABm_{00}^3xy - 4m11ABm_{00}^2 + 2m_{01}ABm_{00}m_{10} + 13m_{11}ABm_{00} - 6m_{01}ABm_{10} + 2ACm_{00}^3x - 2ACm_{00}^2m_{10} + 6ACm_{00}m_{10} + B^2m_{00}^3y^2 - m_{20}B^2m_{00}^2 + 3m_{20}B^2m_{00} + 2BCm_{00}^3y - 2m_{01}BCm_{00}^2 + 5m_{01}BCm_{00} + 2C^2m_{00}^2))/m_{00}^3 - (Aj(x - xg)^i(y - yg)^{j-1}(7Am_{00}m_{11} - 6Bm_{01}^2 - 6Am_{01}m_{10} + 7Bm_{00}m_{20} + Cm_{00}m_{01} - 2Am_{00}^2m_{11} + 2Bm_{00}m_{01}^2 - 2Bm_{00}^2m_{20} + 2Am_{00}m_{01}m_{10}))/m_{00}^3$

$\overline{\mathbf{O}}_{\mu_{ij_{14}}} = (i(x - xg)^{i-1}(y - yg)^j((2m_{01}(6Am_{10} - 3Cm_{00} + Cm_{00}^2 - Am_{00}m_{10}))/m_{00}^3 - 2y(C + Ax + By) + (4Am_{11}(m_{00} - 5))/m_{00}^2 + (2Bm_{20}(m_{00} - 3))/m_{00}^2))/2 - Ay(x - xg)^i(y - yg)^j - (Aj(x - xg)^i(y - yg)^{j-1}(7m_{00}m_{20} + m_{00}m_{01}^2 - m_{00}^2m_{20} + m_{00}^2 - 6m_{01}^2))/m_{00}^3$

$\overline{\mathbf{O}}_{\mu_{ij_{15}}} = ((x - xg)^i(y - yg)^j(6C + 10Ax + 8By))/2 + (j(x - xg)^i(y - yg)^{j-1}(Bm_{00}^3y^2 +$

$7Am_{00}m_{11}-6Am_{01}m_{10}-6Am_{00}^2m_{11}+4Bm_{00}m_{01}^2-5Bm_{00}^2m_{20}-Cm_{00}^2m_{01}+Cm_{00}^3y+Am_{00}^3xy+5Am_{00}m_{01}m_{10}))/m_{00}^3+(i(x-xg)^{i-1}(y-yg)^j(Am_{00}^2-6Am_{10}^2+2Am_{00}^3x^2+10Am_{00}m_{20}+3Bm_{00}m_{11}+3Cm_{00}m_{10}+5Am_{00}m_{10}^2-7Am_{00}^2m_{20}-6Bm_{00}^2m_{11}-2Cm_{00}^2m_{10}+2Cm_{00}^3x+2Bm_{00}^3xy+4Bm_{00}m_{01}m_{10}))/m_{00}^3$

$\overline{\mathbf{O}}_{\mu_{ij_{16}}}=B(x-xg)^i(y-yg)^j+(j(x-xg)^i(y-yg)^{j-1}(Am_{10}-Am_{00}m_{10}-Bm_{00}m_{01}+Am_{00}^2x+Bm_{00}^2y))/m_{00}^2-(Aim_{01}(x-xg)^{i-1}(y-yg)^j)/m_{00}^2$

$\overline{\mathbf{O}}_{\mu_{ij_{22}}}=(2Bj(x-xg)^i(y-yg)^{j-1})/Z_gm_{00}^2$

$\overline{\mathbf{O}}_{\mu_{ij_{23}}}=-2B(x-xg)^i(y-yg)^j(C+Ax+By)-(j(x-xg)^i(y-yg)^{j-1}(A^2m_{00}^3x^2-m_{20}A^2m_{00}^2+3m_{20}A^2m_{00}+2ABm_{00}^3xy-4m_{11}ABm_{00}^2+2m_{10}ABm_{00}m_{01}+13m_{11}ABm_{00}-6m_{10}ABm_{01}+2ACm_{00}^3x-2m_{10}ACm_{00}^2+5m_{10}ACm_{00}+B^2m_{00}^3y^2-3m_{20}B^2m_{00}^2+2B^2m_{00}m_{01}^2+10m_{20}B^2m_{00}-6B^2m_{01}^2+2BCm_{00}^3y-2BCm_{00}^2m_{01}+6BCm_{00}m_{01}+2C^2m_{00}^2))/m_{00}^3-(Bi(x-xg)^{i-1}(y-yg)^j(7Am_{00}m_{20}-6Am_{10}^2+7Bm_{00}m_{11}-6Bm_{01}m_{10}+Cm_{00}m_{10}+2Am_{00}m_{10}^2-2Am_{00}^2m_{20}-2Bm_{00}^2m_{11}+2Bm_{00}m_{01}m_{10}))/m_{00}^3$

$\overline{\mathbf{O}}_{\mu_{ij_{24}}}=-((x-xg)^i(y-yg)^j(6C+8Ax+10By))/2-(i(x-xg)^{i-1}(y-yg)^j(Am_{00}^3x^2+7Bm_{00}m_{11}-6Bm_{01}m_{10}+4Am_{00}m_{10}^2-5Am_{00}^2m_{20}-6Bm_{00}^2m_{11}-Cm_{00}^2m_{10}+Cm_{00}^3x+Bm_{00}^3xy+5Bm_{00}m_{01}m_{10}))/m_{00}^3-(j(x-xg)^i(y-yg)^{j-1}(Bm_{00}^2-6Bm_{01}^2+2Bm_{00}^3y^2+3Am_{00}m_{11}+10Bm_{00}m_{20}+3Cm_{00}m_{01}-6Am_{00}^2m_{11}+5Bm_{00}m_{01}^2-7Bm_{00}^2m_{20}-2Cm_{00}^2m_{01}+2Cm_{00}^3y+2Am_{00}^3xy+4Am_{00}m_{01}m_{10}))/m_{00}^3$

$\overline{\mathbf{O}}_{\mu_{ij_{25}}}=Bx(x-xg)^i(y-yg)^j-(j(x-xg)^i(y-yg)^{j-1}((2m_{10}(6Bm_{01}-3Cm_{00}+Cm_{00}^2-Bm_{00}m_{01}))/m_{00}^3-2x(C+Ax+By)+(2Am_{20}(m_{00}-3))/m_{00}^2+(4Bm_{11}(m_{00}-5))/m_{00}^2))/2+(Bi(x-xg)^{i-1}(y-yg)^j(7m_{00}m_{20}+m_{00}m_{10}^2-m_{00}^2m_{20}+m_{00}^2-6m_{10}^2))/m_{00}^3$

$\overline{\mathbf{O}}_{\mu_{ij_{26}}}=(Bjm_{10}(x-xg)^i(y-yg)^{j-1})/m_{00}^2-(i(x-xg)^(i-1)(y-yg)^j(Bm_{01}-Am_{00}m_{10}-Bm_{00}m_{01}+Am_{00}^2x+Bm_{00}^2y))/m_{00}^2-A(x-xg)^i(y-yg)^j$