MODEL FREE VISUAL SERVOING IN MACRO AND MICRO DOMAIN ROBOTIC APPLICATIONS

by

ERGL OZGUR

Submitted to the Graduate School of Engineering and Natural Sciences
in partial fulfillment of
the requirements for the degree of
Master of Science

Sabanci University
Spring 2007

MODEL FREE VISUAL SERVOING IN MACRO AND MICRO DOMAIN ROBOTIC APPLICATIONS

Erol ÖZGÜR

APPROVED BY

Assoc. Prof. Dr. Mustafa ÜNEL (Thesis Advisor)	....
Prof. Dr. Asif ŞABANOVIÇ	....
Assist. Prof. Dr. Hakan ERDOĞAN	....
Assist. Prof. Dr. Kemalettin ERBATUR	....
Assist. Prof. Dr. Volkan PATOĞLU	....

DATE OF APPROVAL: ....

to my family...

sevgili aileme...

Autobiography

Erol Ozgur was born in Razgrad, Bulgaria in 1982. He received his B.S. degree in Computer Engineering from Gebze Institute of Technology, Kocaeli, Turkey in 2005. His research interests include computer vision and vision guided control of various robotic systems using visual feedback strategies.

Publications:

- H. Bilen, M. Hocaoglu, E. Ozgur, M. Unel, A. Sabanovic. Experimental Comparison of Calibrated and Uncalibrated Visual Servoing in Microsystems, *will appear in the Proceedings of IEEE IROS'07.*

- E. Ozgur, M. Unel. Positioning and Trajetory Following Tasks in Microsystems Using Model Free Visual Servoing, *will appear in the Proceedings of IEEE IECON'07.*

- E. Ozgur, M. Unel. Image Based Visual Servoing Using Bitangent Points Applied to Planar Shape Alignment, *will appear in the Proceedings of IASTED RA'07.*

- E. Ozgur, M. Unel. Object Recognition by Matching Multiple Concavities, *IEEE Conference on Signal Processing and Communications Applications, SIU06.*

Acknowledgments

First of all, I would like to express my deepest gratitude to my thesis advisor Assoc. Prof. Dr. Mustafa Unel, who brought me to this talented position by supervising my thesis and providing me with great moral support. His careness, interest and his admirable research enthusiasm and capabilities, amazed and trailed me along all the way here.

Among all members of the Faculty of Engineering and Natural Sciences, I would like to thank Prof. Dr. Asif Sabanovic, Assist. Prof. Dr. Kemalettin Erbatur, Assist. Prof. Dr. Volkan Patoglu, Assist. Prof. Dr. Hakan Erdogan and Assist. Prof. Dr. Ahmet Onat for spending their valuable time to serve as my jurors.

I would like to thank each of my friends who were next to me and anyone who contributed in anyway to this thesis.

Finally, I would like to thank my family for all their love, support and patience throughout my life.

MODEL FREE VISUAL SERVOING IN MACRO AND MICRO DOMAIN ROBOTIC APPLICATIONS

Erol ÖZGÜR

Electronics Engineering and Computer Sciences, MS Thesis, 2007

Thesis Supervisor: Assoc. Prof. Dr. Mustafa ÜNEL

Keywords: Model free, visual servoing, shape alignment, bitangents, microsystems

Abstract

This thesis explores model free visual servoing algorithms by experimentally evaluating their performances for various tasks performed both in macro and micro domains. Model free or so called uncalibrated visual servoing does not need the system (vision system + robotic system) calibration and the model of the observed scene, since it provides an online estimation of the composite (image + robot) Jacobian. It is robust to parameter changes and disturbances. A model free visual servoing scheme is tested on a 7 DOF Mitsubishi PA10 robotic arm and on a microassembly workstation which is developed in our lab. In macro domain, a new approach for planar shape alignment is presented. The alignment task is performed based on bitangent points which are acquired using convex-hull of a curve. Both calibrated and uncalibrated visual servoing schemes are employed and compared. Furthermore, model free visual servoing is used for various trajectory following tasks such as square, circle, sine etc. and these reference trajectories are generated by a linear interpolator which produces midway targets along them. Model free visual servoing can provide more flexibility in microsystems, since the calibration of the optical system is a tedious and error prone process, and recalibration is required at each focusing level of the optical system. Therefore, micropositioning and three different trajectory following tasks are also performed in micro world. Experimental results validate the utility of model free visual servoing algorithms in both domains.

MAKRO VE MİKRO DÜNYADAKİ ROBOTİK UYGULAMALARDA MODELDEN BAĞIMSIZ GÖRSEL GERİ BESLEMELİ KONTROL

Erol ÖZGÜR

Elektronik Mühendisliği ve Bilgisayar Bilimi, Yüksek Lisans Tezi, 2007

Tez Danışmanı: Doç. Dr. Mustafa ÜNEL

Anahtar Kelimeler: Model bağımsız, görsel geri beslemeli kontrol, şekil hizalama, iki noktada teğetler, mikrosistemler

Özet

Bu çalışmada, makro ve mikro düzeylerde gerçekleştirilmiş değişik görevler için deney sonuçlarını değerlendirerek, modelden bağımsız görsel geri beslemeli kontrol algoritmaları üzerine performans araştırması yapılmıştır. Modelden bağımsız yada bir diğer adıyla kalibre edilmemiş görsel geri beslemeli kontrol, komposit (imge+robot) Jakobyan'ı çevrimiçi olarak kestirebildiğinden sistemin kalibre edilmesine (görme sistemi + robotik sistem) ve incelenen ortamın modeline ihtiyaç duymaz. Parametre değişimlerine ve bozucu dış etkilere karşı gürbüzdür. Modelden bağımsız görsel geri beslemeli kontrol yöntemi 7 serbestlik derecesine sahip Mitsubishi PA10 robotik kol ve mikromontaj iş istasyonu üzerinde test edilmiştir. Makro dünyada, düzlemsel şekil hizalama için yeni bir yaklaşım sunulmuştur. Şekil hizalama işlemi, bir eğrinin dışbükey zarfı (convex-hull) kullanılarak elde edilen, iki noktada teğetler (bitangents) yardımıyla gerçekleştirilmiştir. Hizalama işlemi kalibre edilmiş ve kalibre edilmemiş görsel geri beslemeli kontrol yaklaşımları kullanılarak gerçekleştirilmiş ve sonuçlar karşılaştırılmıştır. Buna ilave olarak, modelden bağımsız görsel geri beslemeli kontrol kare, çember ve sinüs gibi değişik yörünge takibi görevleri için denenmiştir ve bu yörüngeler kendileri boyunca ara hedefler üreten bir doğrusal aradeğerleyici kullanılarak oluşturulmuştur. Modelden bağımsız görsel geri beslemeli kontrol metodunun, kalibrasyonu oldukça usandırıcı ve hata olasılığı yüksek olan ve ayrıca her farklı yakınlaştırma seviyesinde sistemin yeniden kalibre edilmesini gerektiren optik sistemlerde kullanımı oldukça rahatlık sağlamaktadır. Bu nedenle makro dünyada yapılanların dışında, mikrokonumlandırma ve üç farklı yörünge takip görevi de mikro dünyada gerçekleştirilmiştir. Sunulan deneysel sonuçlar, modelden bağımsız görsel geri beslemeli kontrol algoritmalarının makro ve mikro düzeylerde gerçekleştirilen görevlerde kullanılmasında sağladığı faydaları ortaya koymuştur.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation for Visual Servoing

Today's manufacturing robots can perform assembly and manipulation of parts with certain speed and precision, but they have a distinct disadvantage in that they cannot "see" what they are doing, when compared to humans. Consequently, in the domain of applications, a significant engineering effort is expended in setting up a desirable work environment for these blind machines, which necessitates the design and manufacture of specialized mechanisms, such as task based end-effectors.

Once the desired work environment has been composed, the spatial coordinates of all relevant points must then be taught. Even so, due to low robot accuracy manual teaching is often required. The reason that causes this low accuracy is the obtained end-effector pose from measured joint angles using the kinematic model of the robot. Discrepancies between the model and the actual robot lead to tool-tip pose errors. By integrating sensory capabilities to robotic systems these errors can be removed and substantial increase in the versatility and application domain of robotic systems can be ensured [1].

Vision is a useful robotic sensor since it mimics the human sense of vision and allows for noncontact measurement of the environment. Since the early work of Shirai and Inoue [2] who describe how a visual feedback loop can be used to correct the position of a robot to increase task accuracy, considerable effort has been devoted to the visual control of robotic manipulators. Typically visual sensing and manipulation are combined in an open-loop fashion, looking then moving. To increase the accuracy of these subsystems is to use a visual-feedback control loop that will increase the overall accuracy of the system. Taken to the extreme, machine vision can provide closed-loop position control for a robot end-effector -this is referred to as *visual servoing*. This term appears to have been first introduced by Hill and Park [3] in 1979 to distinguish their approach from earlier works where the system alternated between picture taking and moving. Prior to the introduction of this term, the less specific term visual feedback was generally used.

A visually guided robotic system does not need to know a priori the coordinates of workpieces or other objects in its workspace. In a manufacturing environment visual servoing could thus eliminate robot teaching and allow tasks that were not strictly repetitive, such as assembly without precise fixturing and with components that were unoriented.

Visual servoing schemes can be classified on the basis of the knowledge that system structure and parameters are available or not. If these parameters are known, one can use a “*calibrated visual servoing*” approach, while if they are only roughly known an “*uncalibrated visual servoing*” or so called “*model free visual servoing*” approach can be used.

1.2 Why Model Free Visual Servoing?

In most of the previous work on visual servoing, it is assumed that the system structure and parameters were known, or that the parameters could be identified in an off-line process. Such systems, however is not robust to disturbances [4], changes of the parameters and have found limited use outside of the laboratories since they require complete information on the system model and geometry of the robotic workspace.

Obtaining these parameters require calibration methods. These methods are often difficult to understand, inconvenient to use in many robotic environments, and may require the minimization of several, complex, non-linear equations which is not guaranteed to be numerically robust or stable. Moreover, calibrations are typically only accurate in a small subspace of the workspace; accuracy degenerates quickly as the calibration area is left and for a mobile system it is not feasible to recalibrate at each time the system moves.

To overcome these problems, some adaptive visual servoing methods consisting of on-line estimators and feedback controllers have been proposed for controlling robotic systems with visual feedback from cameras whose relations with robotic manipulator are not known, i.e *uncalibrated visual servoing* problem. These adaptive visual servoing methods have the following common features:

- The estimator does not need *a priori* knowledge on the system parameters nor on the kinematic structure of the system. That is, we need not to devote ourselves to tedious calibration process, or to separate the unknown parameters from the system equations, which depends on the detailed knowledge on the kinematic structure of the system.
- There is no restriction on a camera-manipulator system: the number of cameras, kinds of image features, structure of the system (*eye-in-hand* or *eye-to-hand*), the number of inputs and outputs (SISO or MIMO). Proposed methods are applicable to any kind of systems.
- The aim of the estimator is not to obtain the true parameters but to ensure asymptotical convergence of the image features to the desired values under the proposed controller. Therefore, the estimated parameters do not necessarily converge to the true values.

Most of the previous works on uncalibrated visual servoing focus on the Image-Jacobian based scheme. The Image Jacobian model was first introduced by Weiss [5] and used to linearly describe the differential relation between visual feedback space and the robot motion space. In literature, researches on the online estimation of the Jacobian have been extensively studied. Hosoda and Asada have estimated the Jacobian matrix using an extended least squares algorithm with exponential data weighting [4]. Jagersand employed a Broyden's method in the Jacobian estimation [6]. Piepmeier used a recursive least squares (RLS) estimate and a dynamic Quasi-Newton method for model free visual servoing [7]- [8]. Qian exploited the Kalman filtering technique to estimate the Jacobian elements [9]. Lv has employed the Kalman filtering with fuzzy logic adaptive controller to ensure stable Jacobian estimation [10].

1.3 Contribution of The Thesis

This thesis explores model free visual servoing algorithms by experimentally evaluating their performances for various tasks performed both in macro and micro domains. In macro domain, a new approach [11] for planar shape alignment is presented. The alignment task is performed based on bitangent points which are acquired using convex-hull of a curve. Both calibrated and uncalibrated visual servoing schemes are employed and compared. Furthermore, model free visual servoing is used for square, circle and sine trajectory following tasks both in macro and micro domains and it has been shown that it can provide more flexibility in microsystems, since the calibration of the optical system is a tedious and error prone process, and recalibration is required at each focusing level of the optical system.

The remaining of this thesis is organized as follows: Chapter 2 summarizes visual servoing fundamentals. Chapter 3 presents the theory of both model based and model free visual servoing methods. Chapter 4 develops a novel approach for planar shape alignment in the context of visual servoing. Chapter 5 is on the experimental results performed both on a robotic arm and on a microassembly workstation. Finally, Chapter 6 concludes the thesis with some remarks and future works.

Chapter 2

Visual Servoing Fundamentals

In this chapter, a short review of visual servoing is presented. Visual servoing concerns several fields of research including vision, robotics and control. Visual servoing can be useful for a wide range of applications and it can be used to control many different dynamic systems like manipulator arms, mobile robots, aircrafts, etc. Visual servoing systems are generally classified depending on the number of cameras, on the position of the camera with respect to the robot, on the design of the error function to minimize in order to reposition the robot.

2.1 Background

2.1.1 Camera Configurations

Single camera vision systems are generally used since they are cheaper and easier to build than multi-camera vision systems. On the other hand, using two cameras in a stereo configuration make several computer vision problems easier. If the camera(s) are mounted on the robot end-effector, the system is called "*eye-in-hand*". In contrast, if the camera observe the robot from a stationary pose, the system can be called "*eye-to-hand*" (see Figure 2.1). There exist hybrid systems where one camera is in-hand and another camera is fixed somewhere to observe the scene [12]. Figs. 2.2-2.3 show various camera configurations on 7 DOF PA10 robot.

In visual control systems, if the camera only observes the target object it is referred as *endpoint open-loop* (EOL) system and camera that observes both the target object and the robot end-effector is referred as endpoint closed-loop (ECL) system (see Figure 2.4).

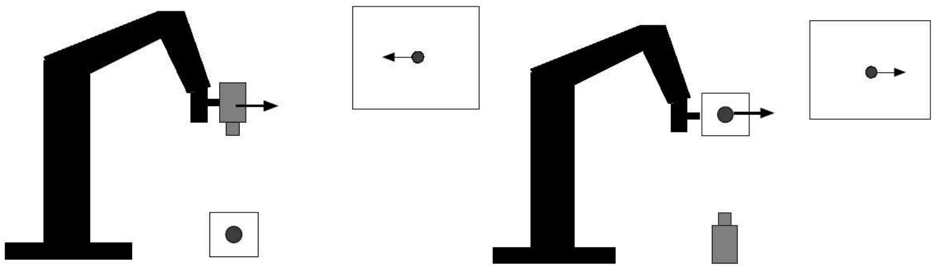


Figure 2.1: Eye-in-hand and eye-to-hand camera configurations

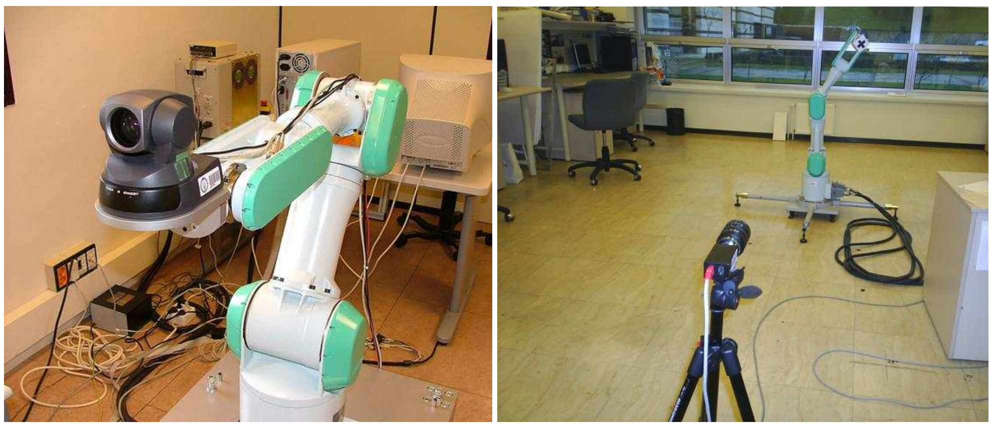


Figure 2.2: Eye-in-hand and eye-to-hand configuration on PA10

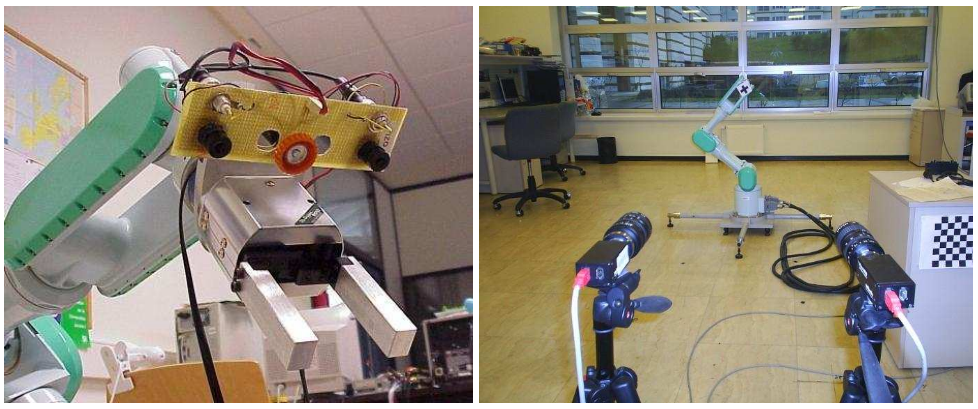


Figure 2.3: Stereo camera configurations on PA10

2.1.2 Camera Model

A "pinhole" camera performs the perspective projection of a 3D point onto the image plane. The image plane is a matrix of light sensitive cells. The resolution of the image is the size of the matrix. The single cell is called a "pixel". For each pixel of coordinates $[u, v]^T$, the camera measures the intensity of the light. For example,

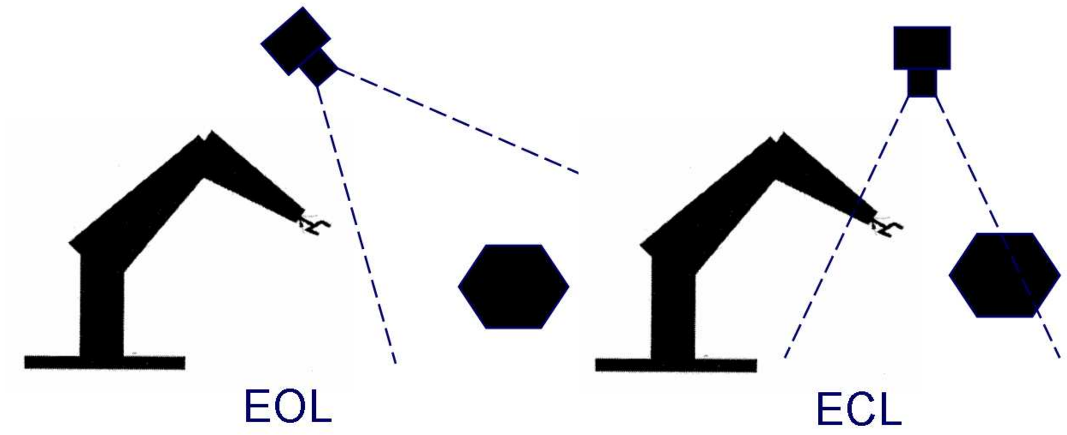


Figure 2.4: Endpoint open-loop and endpoint closed-loop systems

a 3D point, with homogeneous coordinates $P = [X, Y, Z, 1]^T$ project to an image point with homogeneous coordinates $p = [u, v, 1]^T$ (see Figure 2.5):

$$p \propto \begin{pmatrix} K & 0 \end{pmatrix} P \tag{2.1}$$

where K is a matrix containing the intrinsic parameters of the camera:

$$K = \begin{pmatrix} fk_u & fk_u cot(\phi) & u_0 \\ 0 & \frac{fk_v}{sin(\phi)} & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.2}$$

where u_0 and v_0 are the pixels coordinates of the principal point, k_u and k_v are the scaling factors along the $\vec{u}$ and $\vec{v}$ axes (in pixels/meters), ϕ is the angle between these axes and f is the focal length. For most of the commercial cameras, it is a reasonable approximation to suppose square pixels (i.e. $\phi = \frac{\pi}{2}$ and $k_u = k_v$).

The intrinsic parameters of the camera are often only roughly known. Precise calibration of the parameters is a tedious procedure which needs a specific calibration grid [13]. It is thus preferable to estimate the intrinsic parameters without knowing the model of the observed object. If several images of any rigid object are available it is possible to use a self-calibration algorithm [14] to estimate the camera intrinsic parameters.

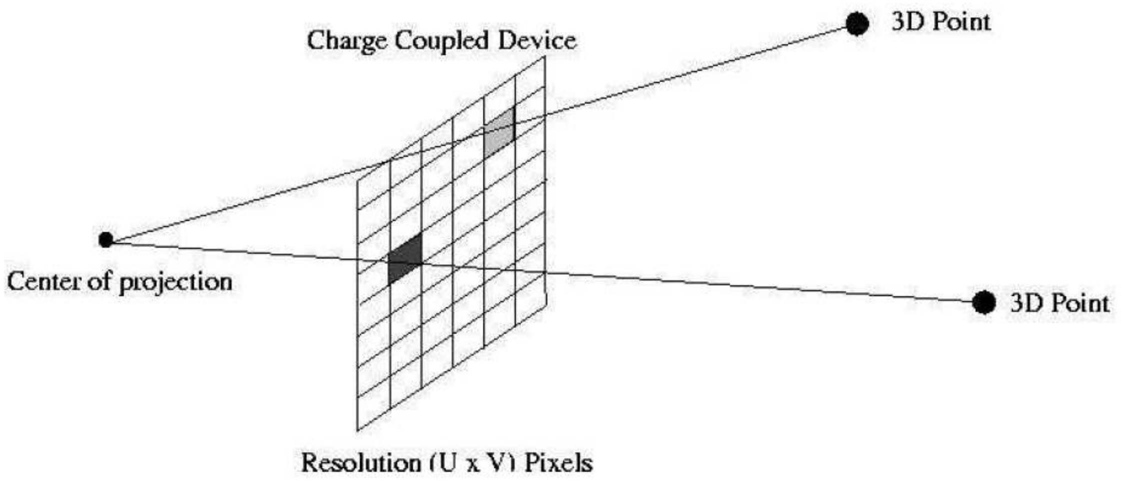


Figure 2.5: Camera Model

2.1.3 Image Features

In the computer vision literature, an *image feature* is defined as any meaningful, detectable part that can be extracted from an image e.g. an edge or a corner. Typically, an image feature will correspond to the projection of a physical feature of some object (e.g. the robot tool) onto the image plane. An *image feature parameter* is defined to be any real-valued quantity that can be calculated from one or more image features. Some of the parameters that have been used for visual servoing include the image plane coordinates of points in the image [15], the distance between two points in the image plane and the orientation of the line connecting those two points, perceived edge length [5], the area of projected surfaces, the centroid and higher order moments of a projected surface, the parameters of line and the parameters of an ellipse in the image plane [15].

2.1.4 Feature Extraction and Tracking

A vision system is required to extract the information needed to perform the servoing task. For this purpose, many reported implementations plan the vision problem to be simple: *e.g.* painting objects white, using artificial targets, and so forth.

In less structured situations, vision has typically relied on the extraction of sharp contrast changes, referred to as “corners” or “edges”, to point the presence of object boundaries or surface markings in an image. The most known algorithms have been proposed by Harris [16] to extract corners and by Canny [17] to get edges from the image.

Processing the entire image to extract these features necessitates the use of extremely high-speed hardware in order to work with a sequence of images at video rate. However not all pixels in the image are of interest, and computation time can be greatly reduced if only a small region around each image feature is processed. Thus, a favorable technique for making vision cheap and tractable is to use *window-based* tracking techniques [18]. Window-based methods have several advantages, among them: computational simplicity, little requirement for special hardware, and easy reconfiguration for different applications. However, that initial positioning of of each window typically presupposes an automated or human-supplied solution to a potentially complex vision problem.

2.1.5 Visual Task Function

In general, the task in vision based control is to control a robotic manipulator to manipulate its environment using vision as opposed to just observing the environment. A visual task is also referred to as a visual task function or a control error function as defined in [19]. For a given visual task, a set of s visual features have to be chosen for achieving the task. These visual features must be tracked over the entire course of the task because the differences between their references, which are determined before the task is initiated, and these visual features are defined as error functions which are inputs to visual controller.

Representing some desired set of features by s^* and the set of current features with s, the objective of visual servoing is to regulate the task function to zero. When the task is completed, the following equality holds:

$$e(s - s^*) = 0 \tag{2.3}$$

Visual features are selected depending on a priori knowledge that we have about the goal of the task.

2.2 Vision Based Control Architectures

A fundamental classification of visual servoing approaches is presented by Sanderson and Weiss [20]. First classification depends on the design of the control scheme. Two different control schemes are generally used for the visual servoing of a robot. The first control scheme is called "*direct visual servoing*" where the vision-based controller directly computes the joint inputs by eliminating robot controller. The second control scheme can be called, contrary to the first one, "*indirect visual servoing*" where the vision-based controller computes set-point inputs to the joint-level controller, thus making use of joint feedback to internally stabilize the robot. For several reasons, most of the visual servoing structures proposed in the literature follows an indirect control scheme which is called "*dynamic look-and-move*". Firstly, the relatively low sampling rates available from vision make direct control of a robot end-effector with complex, nonlinear dynamics an extremely challenging control problem. Using internal feedback with a high sampling rate generally presents the visual controller with idealized axis dynamics. Secondly, many robots already have an interface for accepting Cartesian velocity or incremental position commands. This simplifies the construction of the visual servo system, and also makes the methods more portable.

The second major classification of visual servoing systems builds on the definition of error signal which is computed in 3D task space coordinates or directly in terms of image features. These visual servoing schemes are called *position-based* control and *image-based* control, respectively. So, general classification of vision based control architectures are given as follows:

- Dynamic Position Based Look-and-Move
- Position Based Direct Visual Servoing
- Dynamic Image Based Look-and-Move
- Image Based Direct Visual Servoing

In the order given, Figs. 2.6-2.9 depict these architectures.

In *position-based* control, features are extracted from the image and used in conjunction with geometric model of the target and the known camera model to

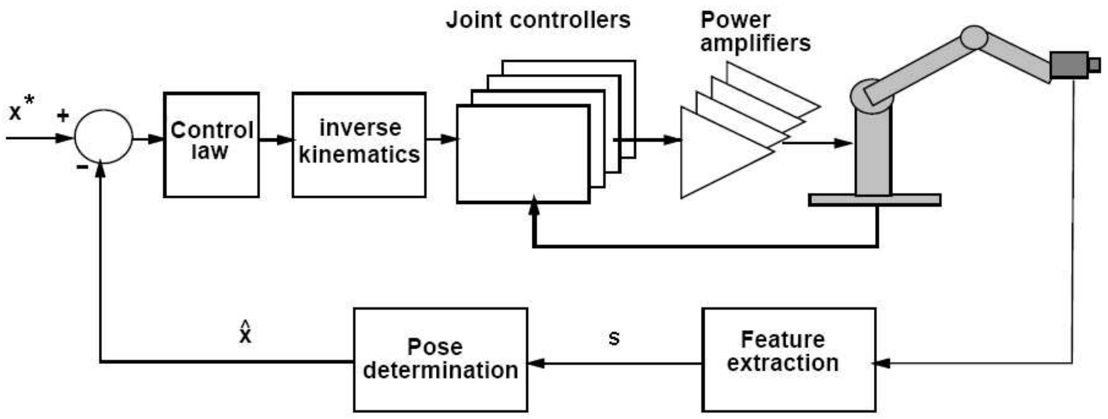


Figure 2.6: Dynamic position based look and move

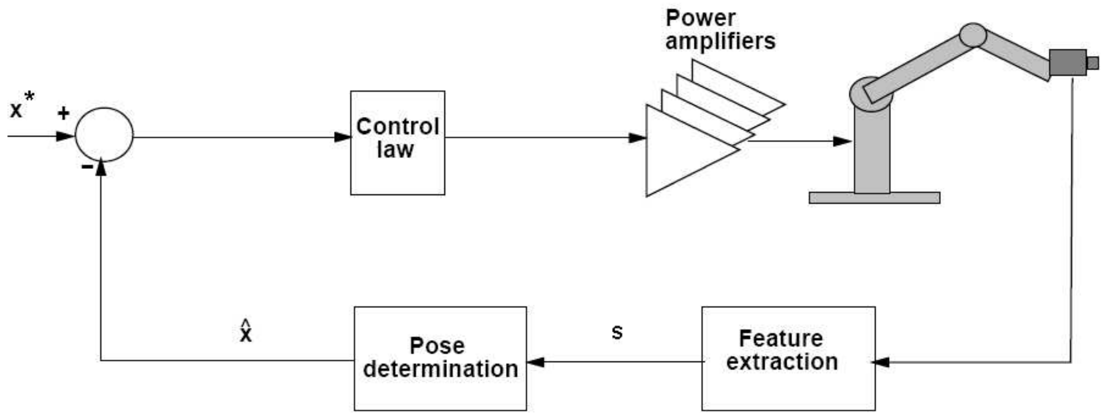


Figure 2.7: Position based direct visual servoing

estimate the pose of the target with respect to camera. Feedback is computed by reducing errors in estimated pose space. In *image-based* servoing, control values are computed on the basis of image features directly. The image-based approach may reduce computational delay, eliminate the necessity for image interpretation and eliminate errors due to sensor modeling and camera calibration. However it does present a significant challenge to controller design since the plant is nonlinear and highly coupled.

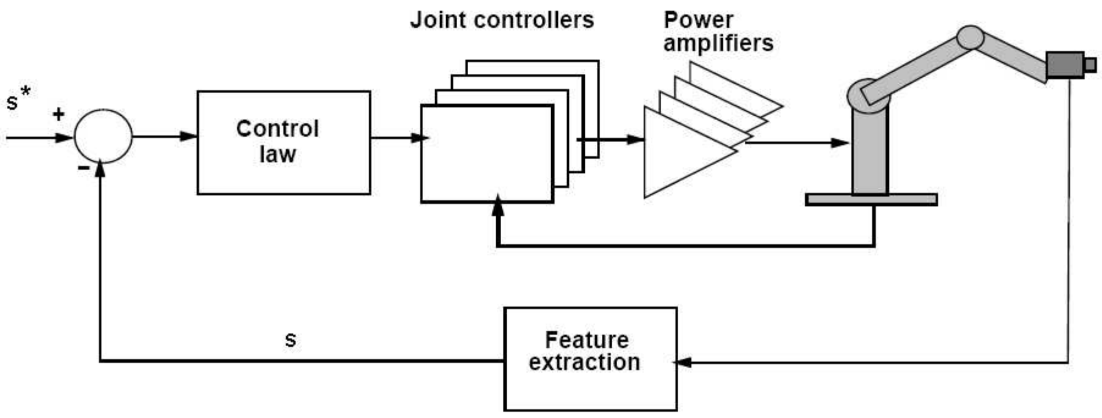


Figure 2.8: Dynamic image based look and move

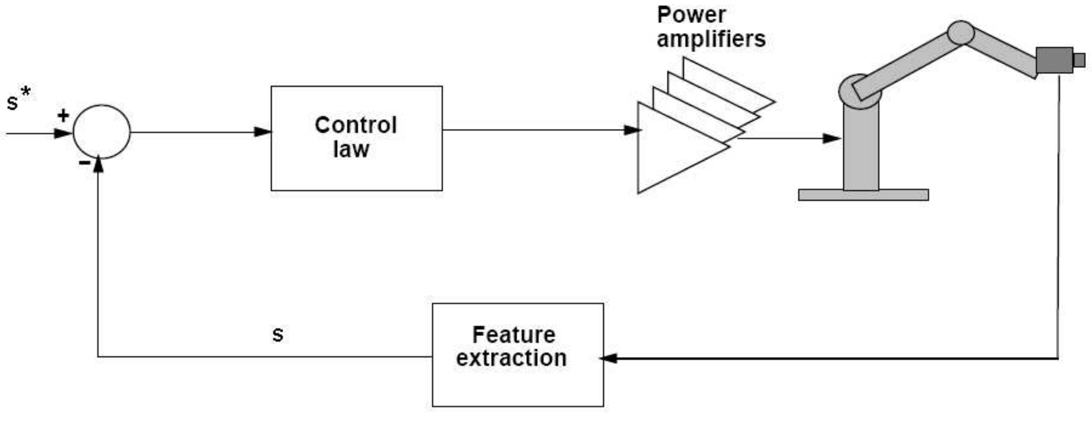


Figure 2.9: Image based direct visual servoing

Chapter 3

Model Based Versus Model Free Visual Servoing

This chapter presents image based, calibrated and uncalibrated, vision guided robotic control methods with a fixed imaging system. These control methods are referred to as model based and model free approaches. Since they are image based visual servo systems the error signal is defined directly in terms of image feature parameters and the motion of the manipulator causes changes to the image observed by the vision system. Thus, specification of an image based visual servo task involves determining an appropriate error function e, such that when the task is achieved, $e = 0$. This can be done by directly using the projection equations, or via "teach-by-showing" method in which the robot is moved to a goal position and the corresponding image is used to compute a vector of desired image feature parameters, s^*. Although the error, e, is defined on the image parameter space, the manipulator control input is typically defined either in joint coordinates or in task space coordinates. Therefore, it is necessary to relate changes in the image feature parameters to changes in the position of the robot. To capture these relationships an *image Jacobian* was first introduced by Weiss [5], who referred to it as the *feature sensitivity matrix.* It is also called an *interaction matrix* [15].

Let $s = [s_1, s_2, \ldots, s_m]^T$ $(s \in \Re^m)$ and $r = [t_x, t_y, t_z, \alpha_x, \alpha_y, \alpha_z]^T$ $(r \in \Re^6)$ denote vectors of image feature parameters obtained from visual sensors and the pose (position + orientation) of the end-effector of the robot, respectively. The relation between s and r is given as $s = s(r(t))$ and its differentiation with respect to time yields,

$$\dot{s} = \frac{\partial s}{\partial r}\dot{r} = J_I \dot{r} \tag{3.1}$$

where $J_I \in \Re^{m\times 6}$ is the image Jacobian, and

$$J_I \triangleq \frac{\partial s}{\partial r} = \begin{pmatrix} \frac{\partial s_1}{\partial r_1} & \cdots & \frac{\partial s_1}{\partial r_6} \\ \vdots & \ddots & \vdots \\ \frac{\partial s_m}{\partial r_1} & \cdots & \frac{\partial s_m}{\partial r_6} \end{pmatrix} \tag{3.2}$$

The relationship given by (3.1) describes how image feature parameters change with respect to changing manipulator pose and the $\dot{r}$ is the camera velocity screw, V_c. Let $\theta \in \Re^n$ denote the vector of joint variables of a robot (see Fig. 3.1).

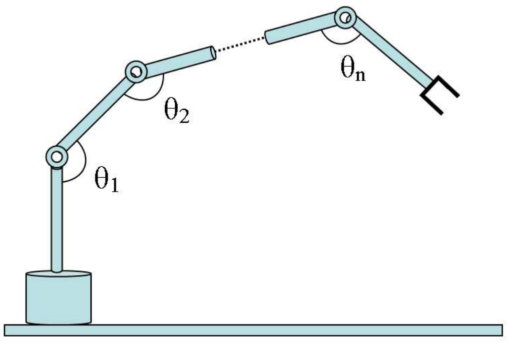


Figure 3.1: Joint variables of a robot

The differential relation between θ and r with respect to time implies

$$\dot{r} = \frac{\partial r}{\partial \theta}\dot{\theta} = J_R(\theta)\dot{\theta} \tag{3.3}$$

where $J_R(\theta) = \partial r/\partial\theta \in \Re^{6\times n}$ is the robot Jacobian which describes the relation between the robot joint velocities and the velocities of its end-effector in Cartesian space. The composite Jacobian is defined as

$$J \triangleq J_I J_R \tag{3.4}$$

where $J \in \Re^{m\times n}$ is a matrix which is the product of image and robot Jacobians. Thus, the relation between joint coordinates and image features is given by

$$\dot{s} = J\dot{\theta} \tag{3.5}$$

3.1 Model Based Visual Servoing

Model based approach needs system parameters which are acquired by calibrating the visual sensor and robotic manipulator, in order to evaluate the available analytical model of the image Jacobian for an image feature.

3.1.1 Image Jacobian for a Point Feature

Let $P = (X, Y, Z)^T$ be a point rigidly attached to the end effector. The velocity of the point P, expressed relative to the camera frame, is given by

$$\dot{P} = V + \Omega \times P \tag{3.6}$$

where $V = (V_x, V_y, V_z)^T$ is translational velocity and $\Omega = (\Omega_x, \Omega_y, \Omega_z)^T$ is rotational velocity. Equation (3.6) can be written in matrix form as follow:

$$\dot{P} = V - [P]_x \Omega \tag{3.7}$$

where $[P]_x$ is the *skew-symmetric matrix* associated with vector P and note that $[a]_x b = [-b]_x a$.

$$[P]_x = \begin{pmatrix} 0 & -Z & Y \\ Z & 0 & -X \\ -Y & X & 0 \end{pmatrix} \tag{3.8}$$

A single point feature vector s in a fixed-camera system is given as

$$s = \begin{pmatrix} x \\ y \end{pmatrix} \tag{3.9}$$

where x and y are normalized, unity focal length, image coordinates of P in camera frame obtained using the following perspective projection equations,

$$x = \frac{X}{Z}, \quad y = \frac{Y}{Z} \tag{3.10}$$

Inserting (3.10) into (3.9) and differentiating with respect to time,

$$\dot{s} = \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} \frac{d}{dt}\left(\frac{X}{Z}\right) \\ \frac{d}{dt}\left(\frac{Y}{Z}\right) \end{pmatrix} = \begin{pmatrix} \frac{\dot{X}Z - X\dot{Z}}{Z^2} \\ \frac{\dot{Y}Z - Y\dot{Z}}{Z^2} \end{pmatrix} = \begin{pmatrix} \frac{\dot{X}}{Z} - \frac{X}{Z}\frac{\dot{Z}}{Z} \\ \frac{\dot{Y}}{Z} - \frac{Y}{Z}\frac{\dot{Z}}{Z} \end{pmatrix} = \begin{pmatrix} \frac{\dot{X}}{Z} - x\frac{\dot{Z}}{Z} \\ \frac{\dot{Y}}{Z} - y\frac{\dot{Z}}{Z} \end{pmatrix} \tag{3.11}$$

$$\Rightarrow \dot{s} = \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} \frac{1}{Z} & 0 & -\frac{x}{Z} \\ 0 & \frac{1}{Z} & -\frac{y}{Z} \end{pmatrix} \underbrace{\begin{pmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{pmatrix}}_{\dot{P}} \tag{3.12}$$

Combining (3.7) and (3.12), and rearranging, one gets

$$\dot{s} = \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} \frac{1}{Z} & 0 & -\frac{x}{Z} \\ 0 & \frac{1}{Z} & -\frac{y}{Z} \end{pmatrix} \left(\begin{pmatrix} V_x \\ V_y \\ V_z \end{pmatrix} + \begin{pmatrix} 0 & Z & -Y \\ -Z & 0 & X \\ Y & -X & 0 \end{pmatrix} \begin{pmatrix} \Omega_x \\ \Omega_y \\ \Omega_z \end{pmatrix} \right) \tag{3.13}$$

$$\dot{s} = \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \underbrace{\begin{pmatrix} \frac{1}{Z} & 0 & \frac{-x}{Z} & -xy & (1+x^2) & -y \\ 0 & \frac{1}{Z} & \frac{-y}{Z} & -(1+y^2) & xy & x \end{pmatrix}}_{\triangleq \hat{J}_I} \underbrace{\begin{pmatrix} V_x \\ V_y \\ V_z \\ \Omega_x \\ \Omega_y \\ \Omega_z \end{pmatrix}}_{\dot{r}} \tag{3.14}$$

where

$$x = \frac{x_p - x_c}{f_x}, \quad y = \frac{y_p - y_c}{f_y} \tag{3.15}$$

and (x_p, y_p) are pixel coordinates of the image point and (x_c, y_c) are the coordinates of the principal point (image center), and (f_x, f_y) are effective focal lengths of the vision sensor, respectively. From (3.15), to derive image Jacobian using pixel coordinates, we proceed as follows:

$$x_p = f_x x + x_c, \quad y_p = f_y y + y_c \tag{3.16}$$

$$\Rightarrow \dot{x}_p = f_x \dot{x}, \quad \dot{y}_p = f_y \dot{y} \tag{3.17}$$

and defining s with new image feature parameters $s = [x_p, y_p]^T$, (3.17) can be rewritten in matrix form as below,

$$\Rightarrow \dot{s} = \begin{pmatrix} \dot{x}_p \\ \dot{y}_p \end{pmatrix} = \begin{pmatrix} f_x & 0 \\ o & f_y \end{pmatrix} \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} \tag{3.18}$$

$$\Rightarrow \dot{s} = \begin{pmatrix} \dot{x}_p \\ \dot{y}_p \end{pmatrix} = \underbrace{\begin{pmatrix} f_x & 0 \\ o & f_y \end{pmatrix} \begin{pmatrix} \frac{1}{Z} & 0 & \frac{-x}{Z} & -xy & (1+x^2) & -y \\ 0 & \frac{1}{Z} & \frac{-y}{Z} & -(1+y^2) & xy & x \end{pmatrix}}_{\triangleq J_I} \underbrace{\begin{pmatrix} V_x \\ V_y \\ V_z \\ \Omega_x \\ \Omega_y \\ \Omega_z \end{pmatrix}}_{\dot{r}} \tag{3.19}$$

$$\Rightarrow \dot{s} = J_I \dot{r} \tag{3.20}$$

where J_I is the pixel-image Jacobian. In eye-to-hand case, the image jacobian has to consider the mapping from the camera frame onto the robot control frame. This relationship is given by the robot-to-camera transformation, denoted by:

$$\dot{r} = V_c = T V_R \tag{3.21}$$

where V_R is the end-effector velocity screw in robot control frame. The robot-to-camera velocity transformation $T \in \Re^{6\times 6}$ is defined as below

$$T = \begin{pmatrix} R & [t]_x R \\ 0_3 & R \end{pmatrix} \tag{3.22}$$

where $[R, t]$ are being the rotational matrix and the translation vector that map camera frame onto robot control frame and $[t]_x$ is the skew symmetric matrix associated with vector t.

Substituting (3.21) into (3.20), an expression that relates the image motion to the end-effector velocity is acquired:

$$\dot{s} = \underbrace{J_I T}_{\triangleq \bar{J}_I} V_R = \bar{J}_I V_R \tag{3.23}$$

where $\bar{J}_I$ is the new image Jacobian which directly relates the changes of the image features to the end-effector velocity in robot control frame. Note that if k feature points are taken into account, e.g. $s = [x_1, y_1 \dots x_k, y_k]^T$, $\bar{J}_I$ is given by the following stacked image Jacobian

$$\bar{J}_I = \begin{pmatrix} \bar{J}_I^1 \\ \vdots \\ \bar{J}_I^k \end{pmatrix} \tag{3.24}$$

3.1.2 Visual Control Design

The results of the previous section shows how to relate robot end-effector motion to perceived motion in a camera image. However, visual servoing applications typically require the reverse -computation of $\dot{r}$ given $\dot{s}$ as input. Suppose that the goal of particular task is to reach a constant desired image feature parameter vector $s^* \in \Re^m$ and and the error $e \in \Re^m$ is defined on image plane as

$$e = s - s^* \tag{3.25}$$

Then the visual control problem can be formulated as follows: design an end-effector velocity screw $\dot{r}$ in such a way that the error disappears, i.e. $e \rightarrow 0$.

By imposing $\dot{e} = -\Lambda e$ and solving (3.23), a simple proportional control law for the end-effector motion with an exponential decrease of the error function, is obtained as follow:

$$V_R = -\bar{J}_I^{\dagger}\Lambda(s - s^*) \tag{3.26}$$

where $\Lambda \in \Re^{6\times6}$ is a positive constant gain matrix, $\bar{J}_I^{\dagger}$ is the pseudo-inverse of the image Jacobian and $V_R = \begin{pmatrix} V_x & V_y & V_z & \Omega_x & \Omega_y & \Omega_z \end{pmatrix}^T$.

3.2 Model Free Visual Servoing

Model free approach estimates the composite Jacobian dynamically by assuming that elements of composite Jacobian are unknown. Dynamic Broyden's Update and Recursive Least Square methods for composite Jacobian estimation are proposed by Piepmeier in [7], [8]. Design of Optimal and Dynamic Gauss-Newton visual controllers with dynamic Jacobian estimation schemes are also presented in [21] and [8], respectively. Using these controllers, the robot can be servoed to both static and moving targets, even with uncalibrated robot kinematics and camera models.

The control methods are completely independent of robot type, camera type, and camera location. In other words, they are independent of the system model.

3.2.1 Problem Formulation

A stationary vision system is assumed that can sense sufficient end-effector and target features to locate both bodies in space. This renders the target features, $s^*(t)$, as functions of only time t, and the end-effector features, $s(\theta)$, as functions of only the robot joint angles, $\theta \in \Re^n$. It is important to note that t and θ are independent variables, since as time varies, the joint angles can be held constant, and conversely, at every given time, the joint angles can take on any values. There are no assumptions yet about target tracking. To optimally track the target, a constraint relationship is imposed between θ and t so joint angles are selected as a function of time, $\theta(t) = g(s^*(t))$. This establishes an optimal end-effector trajectory $s(\theta(t))$ to follow the moving target. The constraint is established by minimizing the tracking error, $e \in \Re^m$, as seen in the image plane

$$e(\theta, t) = s(\theta) - s^*(t) \tag{3.27}$$

The combined transformations of forward kinematics and imaging geometry render $e(\theta, t)$ a highly nonlinear function. This multivariate optimization problem is solved at each increment by a dynamic quasi-Newton controller with a dynamic Jacobian estimator.

3.2.2 Visual Controllers

A. Dynamic Gauss-Newton Controller

The imposed trajectory $\theta(t)$ that causes the end-effector to follow the target is established by minimizing the squared image error

$$E(\theta, t) = \frac{1}{2} e^T(\theta, t) e(\theta, t) \tag{3.28}$$

which can also be modified by a weighting matrix, but is omitted for simplicity. The Taylor series expansion about (θ, t) is

$$E(\theta + h_\theta, t + h_t) = E(\theta, t) + E_\theta h_\theta + E_t h_t + \dots \tag{3.29}$$

where E_θ and E_t are partial derivatives, and $h_\theta = \theta_k - \theta_{k-1}$ and $h_t = t_k - t_{k-1}$ are increments of θ and t. For a fixed sampling period h_t, E is minimized by solving

$$\frac{\partial E(\theta + h_\theta, t + h_t)}{\partial \theta} = 0 \tag{3.30}$$

which in turn implies

$$E_\theta + E_{\theta\theta} h_\theta + E_{t\theta} h_t + O(h^2) = 0 \tag{3.31}$$

where $O(h^2)$ indicates 2^{nd} order terms in h_t and h_θ. Dropping these terms and recalling the definition of the joint-to-image feature error composite Jacobian J as

$$J \equiv \tfrac{\partial e}{\partial \theta},$$

one can proceed as follows:

$$E_\theta = \frac{\partial E}{\partial \theta} = \frac{\partial}{\partial \theta}\left(\frac{1}{2} e^T e\right) = \frac{1}{2}\frac{\partial e^T}{\partial \theta} e + \frac{1}{2} e^T \frac{\partial e}{\partial \theta} = \frac{\partial e^T}{\partial \theta} e = J^T e \tag{3.32}$$

Define $\frac{\partial J^T}{\partial \theta} e$ by S, namely

$$S \equiv \frac{\partial J^T}{\partial \theta} e \tag{3.33}$$

It follows that

$$E_{\theta\theta} = \frac{\partial}{\partial \theta}(E_\theta) = \frac{\partial}{\partial \theta}(J^T e) = \frac{\partial J^T}{\partial \theta} e + J^T \underbrace{\frac{\partial e}{\partial \theta}}_{J} = S + J^T J = J^T J + S \tag{3.34}$$

and

$$E_{t\theta} = \frac{\partial}{\partial t}(E_\theta) = \frac{\partial}{\partial t}(J^T e) = J^T \frac{\partial e}{\partial t} \tag{3.35}$$

Hence,

$$h_\theta = -(J^T J + S)^{-1} J^T (e + \frac{\partial e}{\partial t} h_t) \tag{3.36}$$

Adding θ to both sides of this equation gives what is referred to as a dynamic Newtons method

$$\theta + h_\theta = \theta - (J^T J + S)^{-1} J^T \left(e + \frac{\partial e}{\partial t} h_t\right) \tag{3.37}$$

To compute the terms S and J analytically requires a calibrated system model. The term S is difficult to estimate, but as θ approaches the solution, it approaches zero and it is often dropped to give what is sometimes called a Gauss-Newton method. It can be shown that for a small enough time increment, h_t, the method is well defined for all θ and converges linearly to a finite steady-state error. When an estimated Jacobian, $\hat{J}$, is used, the algorithm becomes a dynamic quasi-(Gauss)Newton method, such that at the k^{th} increment

$$\theta_{k+1} = \theta_k - (\hat{J}_k^T \hat{J}_k)^{-1} \hat{J}_k^T (e_k + \frac{\partial e_k}{\partial t} h_t) \tag{3.38}$$

where $h_t = t_k - t_{k-1}$. The qualifier dynamic specifically refers to the presence of the error velocity term $(\partial e_k / \partial t)$ which is used to linearly predict the error vector at the next time increment as $e_{k+1} \approx e_k + (\partial e_k/\partial t) h_t$, assuming the robot remains at its current position. Then control is defined as

$$u_{k+1} = \dot{\theta}_{k+1} = -K_p \hat{J}_k^{\dagger} (e_k + \frac{\partial e_k}{\partial t} h_t) \tag{3.39}$$

where K_p and $\hat{J}_k^{\dagger}$ are some positive proportional gain and the pseudo-inverse of the estimated Jacobian at k^{th} iteration, respectively.

B. Optimal Controller

Equation (3.5) can be discretized as

$$s(\theta_{k+1}) = s(\theta_k) + T \hat{J}_k u_k \tag{3.40}$$

where T is the sampling time of the vision sensor and $u_k = \dot{\theta}_k$ is the velocity vector of the end effector. [21] presents an optimal control strategy based on the minimization of the following objective function which penalizes the pixelized position errors and the control energy or input u_k

$$E_{k+1} = (s_{k+1} - s_{k+1}^*)^T Q (s_{k+1} - s_{k+1}^*) + u_k^T L u_k \tag{3.41}$$

where Q and L are the weighting matrices. The resulting optimal control input u_k can be derived as

$$u_k = -(T \hat{J}_k^T Q T \hat{J}_k + L)^{-1} T \hat{J}_k^T Q (s_k - s_{k+1}^*) \tag{3.42}$$

Since there is no standard procedure to compute the weighting matrices Q and L, they are adjusted to obtain desired transient and steady state response.

3.2.3 Dynamic Jacobian Estimation

A. Dynamic Broyden's Update Method

The affine model of the error function $e(\theta, t)$ is a first-order Taylor series approximation denoted as $m(\theta, t)$ and $m_k(\theta, t)$ is an expansion of $m(\theta, t)$ about the k^{th} data point as follows:

$$m_k(\theta, t) = e(\theta_k, t_k) + \hat{J}_k(\theta - \theta_k) + \frac{\partial e_k}{\partial t}(t - t_k) \tag{3.43}$$

Requiring that the affine model (3.43) correctly specifies the error at $(\theta, t) = (\theta_{k-1}, t_{k-1})$ gives,

$$m_k(\theta_{k-1}, t_{k-1}) = e(\theta_{k-1}, t_{k-1}) \tag{3.44}$$

Next, writing $m_k(\theta, t)$ for increment $(\theta, t) = (\theta_{k-1}, t_{k-1})$ yields

$$m_k(\theta_{k-1}, t_{k-1}) = e(\theta_k, t_k) + \hat{J}_k(\theta_{k-1} - \theta_k) + \frac{\partial e_k}{\partial t}(t_{k-1} - t_k) \tag{3.45}$$

Substituting (3.44) into (3.45)

$$e(\theta_{k-1}, t_{k-1}) = e(\theta_k, t_k) + \hat{J}_k(\theta_{k-1} - \theta_k) + \frac{\partial e_k}{\partial t}(t_{k-1} - t_k) \tag{3.46}$$

and rearranging (3.46) yields the so-called secant equation

$$\hat{J}_k h_\theta + \frac{\partial e_k}{\partial t} h_t = \Delta e \tag{3.47}$$

where $\Delta e = e_k - e_{k-1}$. Broydens method requires that (3.47) holds. Subtracting $\hat{J}_{k-1} h_\theta$ from each side, rearranging, and transposing gives

$$h_\theta^T \Delta \hat{J}^T = \left(\Delta e - \frac{\partial e_k}{\partial t} h_t - \hat{J}_{k-1} h_\theta\right)^T \tag{3.48}$$

where $\Delta \hat{J} = \hat{J}_k - \hat{J}_{k-1}$. The Jacobian update $\Delta \hat{J}$ is selected to minimize the Frobenius norm $\|\Delta \hat{J}\|_F = \left(\sum (\Delta \hat{J})_{ij}^2\right)^{(1/2)}$ subject to the constraint (3.48), where $(\Delta \hat{J})_{ij}$ indexes $\Delta \hat{J} \in \Re^{m \times n}$. By stacking the elements into a vector and rewriting (3.48) accordingly, the problem is cast into a familiar form with a minimum norm solution. The stacked form of (3.48) can be written as,

$$\begin{pmatrix} h_\theta^T & \cdots & 0 \\ 0 & \ddots & 0 \\ 0 & \cdots & h_\theta^T \end{pmatrix} \begin{pmatrix} \left(\Delta \hat{J}^T\right)_1 \\ \vdots \\ \left(\Delta \hat{J}^T\right)_m \end{pmatrix} = \begin{pmatrix} (\phi)_1 \\ \vdots \\ (\phi)_m \end{pmatrix} \tag{3.49}$$

where $\left(\Delta \hat{J}^T\right)_i$ is the i^{th} column of $\Delta \hat{J}^T$, and $(\phi)_i$ is the i^{th} element of

$$\left(\Delta e - \tfrac{\partial e_k}{\partial t} h_t - \hat{J}_{k-1} h_\theta\right)^T.$$

Note that (3.49) is in the form $Ax = b$, and that the norm $\|x\|^2$ is equal to $\|\Delta \hat{J}\|_F^2$. The minimum norm solution is $x = A^T(AA^T)^{-1}b$ which minimizes $\|x\|$ subject to $Ax = b$. Unstacking the result gives the dynamic Broyden update,

$$\hat{J}_k = \hat{J}_{k-1} + \frac{\left(\Delta e - \hat{J}_{k-1} h_\theta - \frac{\partial e_k}{\partial t} h_t\right) h_\theta^T}{h_\theta^T h_\theta} \tag{3.50}$$

The qualifier dynamic specifically refers to the presence of the error velocity term $(\partial e_k / \partial t)$.

B. Recursive Least Square Method

An exponentially weighted recursive least square (RLS) algorithm [22] that minimizes a cost function G_k based on the change in the affine model of error over time is used to estimate composite Jacobian J.

$$G_k = \sum_{i=0}^{k-1} \lambda^{k-i-1} \|\Delta m_{ki}\|^2 \tag{3.51}$$

where

$$\Delta m_{ki} = m_k(\theta_i, t_i) - m_i(\theta_i, t_i) \tag{3.52}$$

In light of (3.43), (3.52) becomes

$$\Delta m_{ki} = e(\theta_k, t_k) - e(\theta_i, t_i) - \frac{\partial e_k}{\partial t}(t_k - t_i) - \hat{J}_k h_{ki} \tag{3.53}$$

where $h_{ki} = \theta_k - \theta_i$, the weighting factor λ satisfies $0 < \lambda < 1$, and the unknown variables are the elements of $\hat{J}_k$. Minimizing G_k is equivalent to minimizing the Frobenius norm of the term $(\Delta M)^T \Lambda (\Delta M)$ where ΔM is a $k \times m$ matrix, whose i^{th} column is $m_k(\theta_{i-1}, t_{i-1}) - m_{i-1}(\theta_{i-1}, t_{i-1})$ and Λ is a $k \times k$ diagonal matrix with λ^{k-i} at the i^{th} diagonal element.

Solution of the minimization problem yields the following recursive update rule for the composite Jacobian:

$$\hat{J}_k = \hat{J}_{k-1} + (\Delta e - \hat{J}_{k-1} h_\theta - \frac{\partial e_k}{\partial t} h_t)(\lambda + h_\theta^T P_{k-1} h_\theta)^{-1} h_\theta^T P_{k-1} \tag{3.54}$$

where

$$P_k = \frac{1}{\lambda}(P_{k-1} - P_{k-1} h_\theta (\lambda + h_\theta^T P_{k-1} h_\theta)^{-1} h_\theta^T P_{k-1}) \tag{3.55}$$

and $h_\theta = \theta_k - \theta_{k-1}$, $h_t = t_k - t_{k-1}$, $\Delta e = e_k - e_{k-1}$, and $e_k = s_k - s_k^*$, which is the difference between the end-effector position and the target position at k^{th} iteration. The term $\frac{\partial e_k}{\partial t}$ predicts the change in the error function for the next iteration, and in the case of a static camera it can directly be estimated from the target image feature vector with a first-order difference:

$$\frac{\partial e_k}{\partial t} \cong -\frac{s_k^* - s_{k-1}^*}{h_t} \tag{3.56}$$

The weighting factor is $0 < \lambda \leq 1$ and when close to 1 results in a filter with a longer memory.

Chapter 4

Shape Alignment

Shape alignment is one of the central problems in vision research and has played a key role in many particular domain of applications such as object recognition [23], [24] and tracking [25]. In the domain of visual servoing, most of the current alignment systems are based on known geometrical shaped objects such as industrial parts or those that have good features like corners, straight edges which are feasible to extract and track in real time [26]. The alignment of smooth free-form planar objects in unknown environments presents a challenge in visually guided assembly tasks.

It is proposed in [11] to use bitangent points in aligning planar shapes by employing both calibrated [27] and uncalibrated image based visual servoing [7] schemes. In literature the use of bitangents in recognizing planar objects by affine invariant alignment was first considered in [28] and for servoing purposes bitangent lines (lines joining corresponding features on the superposition of two views of a scene) were utilized to align the orientation between two cameras at different locations in space by [29]. In [30] similar principles were applied on landing and surveillance of aerial vehicles using vanishing points and lines. In order to acquire bitangent points, convex-hull [31] of a curve is used. Bitangent points are then employed in the construction of a feature vector.

4.1 Invariants

In mathematics a quantity is said to be invariant if its value does not change following a given operation. There are two types of well known invariants, which are algebraic and geometric, respectively.

4.1.1 Algebraic Invariance

Algebraic invariance refers to combinations of coefficients from certain functions that remain constant when the coordinate system in which they are expressed is translated, or rotated. An example of this kind of invariance is seen in the behavior of the conic sections. The general equation of a conic section is

$$ax^2 + bxy + cy^2 + dx + ey + f = 0$$

Each of the equations of a circle, or an ellipse, a parabola, or hyperbola represents a special case of this equation. One combination of coefficients, $(b^2 - 4ac)$, from this equation is called the discriminant. For a parabola, the value of the discriminant is zero, for an ellipse it is less than zero, and for a hyperbola is greater than zero. However, regardless of its value, when the axes of the coordinate system in which the figure is being graphed are rotated through an arbitrary angle, the value of the discriminant $(b^2 - 4ac)$ is unchanged. Thus, the discriminant is said to be invariant under a rotation of axes. In other words, knowing the value of the discriminant reveals the identity of a particular conic section regardless of its orientation in the coordinate system. Still another invariant of the general equation of the conic sections, under a rotation of axes, is the sum of the coefficients of the squared terms $(a + c)$, i.e. trace of the 2×2 matrix of 2^{nd} degree terms.

4.1.2 Geometric Invariance

In geometry, the invariant properties of points, lines, angles, and various planar and solid objects are all understood in terms of the invariant properties of these objects under such operations as translation, rotation, reflection, and magnification. For example, the area of a triangle is invariant under translation, rotation and reflection, but not under magnification. On the other hand, the interior angles of a triangle are invariant under magnification, and so are the proportionalities of the lengths of its sides.

4.1.3 Invariance of Features

Extraction upon features related to a model object, a similarity may be used to compare the shape features. The similarity measure is referred to as a shape mea-

	Euclidean	Similarity	Affine	Projective
Transformation				
Rotation	o	o	o	o
Translation	o	o	o	o
Uniform scaling		o	o	o
Non-uniform scaling			o	o
Shear			o	o
Perspective Projection				o
Composition of projections				o
Invariants				
Length	o			
Angle	o	o		
Ratio of lengths	o	o		
Parallelism	o	o	o	
Incidence	o	o	o	o
Cross-Ratio	o	o	o	o

Table 4.1: Geometric transformations versus invariant feature properties.

sure. The shape measure should be invariant under certain class of geometric transformation of the object. In the simple scenario, shape measures are invariant to translation, rotation and scale. In this case, the shape measures are invariant under similarity transformation. When included the invariance of shape measures to shear effect, the shape measures are said to be invariant under affine transformation. Finally in the complicated case, shape measures are invariant under perspective transformation, a special projective transformation, when included the effect caused by perspective projection. Table 4.1 tabulates the geometric transformations versus the invariant feature properties.

4.2 Bitangents

A line that is tangent to a curve at two points is called a bitangent and the points of tangency are called bitangent points. (See Fig. 4.1).

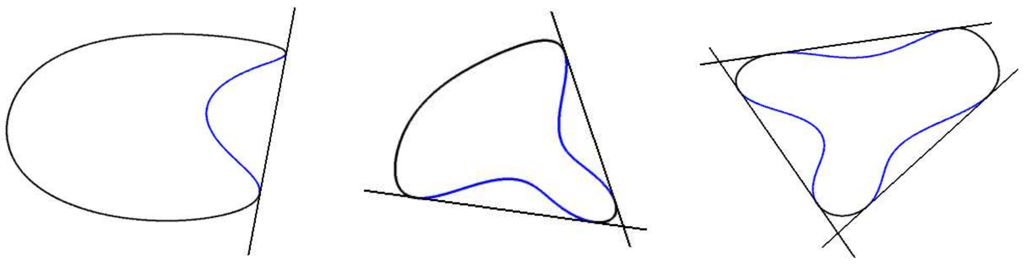


Figure 4.1: Some curves and their bitangents.

4.2.1 Properties of Bitangent Points

It is well known [33] that these bitangent points directly map to one another under projective transformations since they are projective invariants. They are also called *contact points*. Bitangent points are local features, so they are robust to occlusion, clutter and can be easily computed and tracked in real-time. They can also be extended to wide range of differing feature types.

4.2.2 Convex-Hull

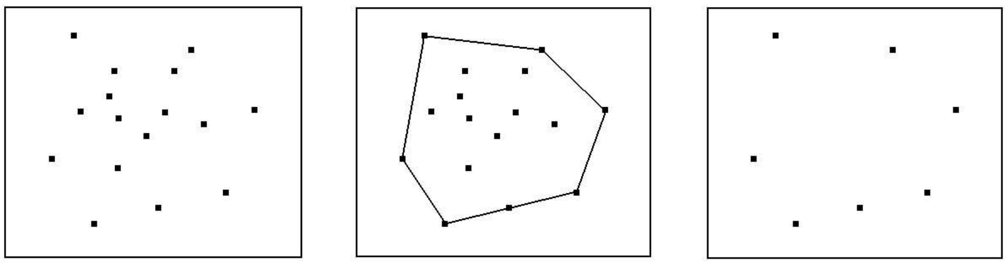
Computing a convex hull (or just "hull") is one of the first sophisticated geometry algorithms, and there are many variations of it. The most common form of this algorithm involves determining the smallest convex set (called the "convex hull") containing a discrete set of points. There are numerous applications for convex hulls: collision avoidance, hidden object determination, and shape analysis.

The most popular algorithms are the "Graham scan" algorithm [34] and the "divide-and-conquer" algorithm [35]. Implementations of both these algorithms are readily available and both are O(n log n) time algorithms.

In the example below (see Fig. 4.2) the convex hull of the points is the line that contains them. Informally, we can say that it is a rubber band wrapped around the "outside" points.

4.2.3 Computation of Bitangent Points

Computation of bitangent points of a curve is presented as a block diagram in Fig. 4.3. Block-I receives a sequence of images from a camera and tracks a region in a specified window using a tracking algorithm such as ESM algorithm [36]. Block-



(a) (b) (c)

Figure 4.2: (*a*) Randomly scattered points, (*b*) illustration of convex-hull and (*c*) points that are on convex-hull

II applies Canny edge detection algorithm to the specified region and extracts the curve boundary data. Finally, Block-III employs a Convex-hull algorithm [31] to find convex hull of the curve. Fig. 4.4 depicts various curves with different number of concavities. The convex-hull algorithm yields convex portion of the original data. Initial and final points of each convex portion are bitangent points.

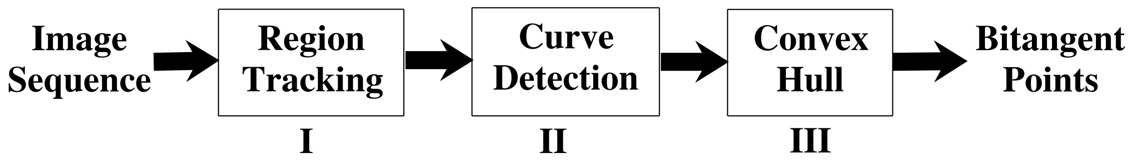


Figure 4.3: Block diagram representation of the algorithm for extracting bitangents.

If we apply the same algorithm to each concave portion of the curve we get two more tangent points for that concavity (see Fig. 4.5).

4.3 Bitangent Points In Computer Vision

They can be used in image alignment, 3D reconstruction by allowing computation of homography and fundamental matrices, motion tracking, and object recognition [32].

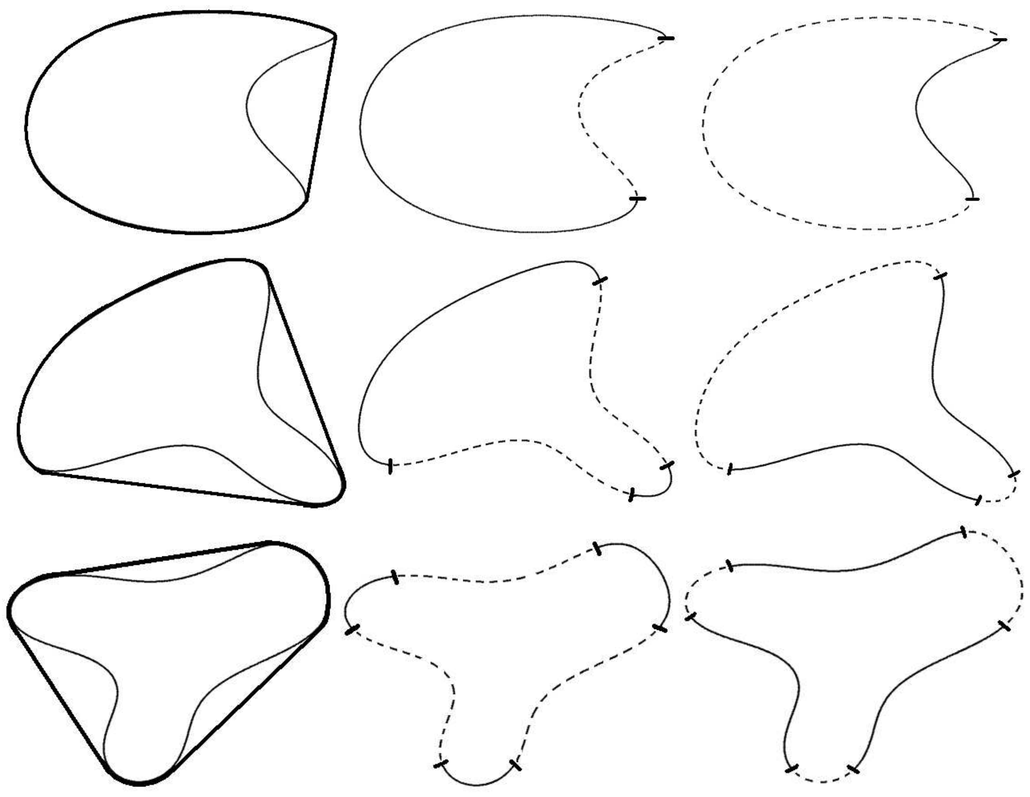

(a) (b) (c)

Figure 4.4: Acquiring the bitangent points of curves with one, two and three concavities. (a) superimposed (solid) convex-hull on curves, (b) overlapped data between the convex-hull and the curves, and the *bitangent* points, (c) concave data portions

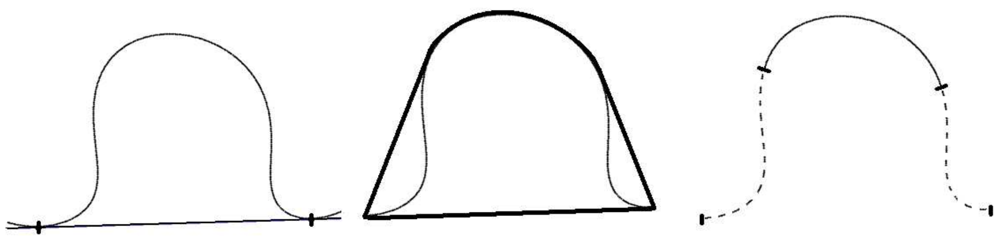


Figure 4.5: Concave portion of a curve and the tangent points.

4.3.1 Projective Equivalence and Peq-Points

The projective transformations are defined as non-linear mappings of the form,

$$\bar{x} = \frac{t_{11}x + t_{12}y + t_{13}}{t_{31}x + t_{32}y + t_{33}} \ , \ \bar{y} = \frac{t_{21}x + t_{22}y + t_{23}}{t_{31}x + t_{32}y + t_{33}} \tag{4.1}$$

It is not easy to derive a projection matrix similar to the euclidean and affine matrices. We need to introduce another transformation called *homogenous transformation*. The homogenous transformation converts *Cartesian coordinates* (x, y) into *homogenous coordinates* (wx, wy, w). In this transformation each x, y coordinate is multiplied by a constant w, and w is appended as the third component of the vector. Similarly, the inverse homogenous transform converts the homogenous coordinates (x_h, y_h, w) into the Cartesian coordinates $(\frac{x_h}{w}, \frac{y_h}{w})$, by dividing each of the two components with the third component.

$$x = \frac{x_h}{w}\ ,\ y = \frac{y_h}{w} \qquad \bar{x} = \frac{\bar{x}_h}{\bar{w}}\ ,\ \bar{y} = \frac{\bar{y}_h}{\bar{w}} \tag{4.2}$$

Substituting $x, y, \bar{x}, \bar{y}$, and rearranging we get:

$$\frac{\bar{x}_h}{\bar{w}} = \frac{(t_{11}\frac{x_h}{w} + t_{12}\frac{y_h}{w} + t_{13})w}{(t_{31}\frac{x_h}{w} + t_{32}\frac{y_h}{w} + t_{33})w} = \frac{t_{11}x_h + t_{12}y_h + t_{13}w}{t_{31}x_h + t_{32}y_h + t_{33}w} \tag{4.3}$$

$$\frac{\bar{y}_h}{\bar{w}} = \frac{(t_{21}\frac{x_h}{w} + t_{22}\frac{y_h}{w} + t_{23})w}{(t_{31}\frac{x_h}{w} + t_{32}\frac{y_h}{w} + t_{33})w} = \frac{t_{21}x_h + t_{22}y_h + t_{23}w}{t_{31}x_h + t_{32}y_h + t_{33}w} \tag{4.4}$$

$$\begin{aligned} \bar{x}_h &= t_{11}x_h + t_{12}y_h + t_{13}w \\ \bar{y}_h &= t_{21}x_h + t_{22}y_h + t_{23}w \\ \bar{w} &= t_{31}x_h + t_{32}y_h + t_{33}w \end{aligned}$$

We can write the projection matrix T from above equations as in the following form:

$$\underbrace{\begin{bmatrix} \bar{x}_h \\ \bar{y}_h \\ \bar{w} \end{bmatrix}}_{\bar{C}_h} = \underbrace{\begin{bmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \end{bmatrix}}_{T} \underbrace{\begin{bmatrix} x_h \\ y_h \\ w \end{bmatrix}}_{C_h} \tag{4.5}$$

Two corresponding *peq-points* ($\underline{\text{p}}$rojective $\underline{\text{eq}}$uivalent points) of two curves, such as $C = (x_i, y_i)$ and $\bar{C} = (\bar{x}_i, \bar{y}_i)$, respectively, will be defined by the condition that,

$$\begin{bmatrix} \bar{x}_{i_h} \\ \bar{y}_{i_h} \\ \bar{w} \end{bmatrix} = \underbrace{\begin{bmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \end{bmatrix}}_{T} \begin{bmatrix} x_{i_h} \\ y_{i_h} \\ w \end{bmatrix} \tag{4.6}$$

4.3.2 Comparison and Recognition via Canonical Models

If $C = (x_i, y_i)|_{i=1}^n$ and $\bar{C} = (\bar{x}_i, \bar{y}_i)|_{i=1}^n$ are points of projective equivalent curves, the known mappings of any four or more peq-points of $C = (x_i, y_i)|_{i=1}^n$ to any four or more peq-points of $\bar{C} = (\bar{x}_i, \bar{y}_i)|_{i=1}^n$, such as

$$(x_i, y_i) \xrightarrow{T} (\bar{x}_i, \bar{y}_i) \;\; for \;\; i = 1, 2, 3, 4, \ldots, n$$

will define the projective transformation matrix, T. The matrix T is 3×3, which has 9 unknowns. We can arbitrarily fix one of them and determine the remaining 8 unknowns. Rearranging equations in (4.1) we get:

$$t_{11}x + t_{12}y + t_{13} - t_{31}x\bar{x} - t_{32}y\bar{x} - t_{33}\bar{x} = 0$$

$$t_{21}x + t_{22}y + t_{23} - t_{31}x\bar{y} - t_{32}y\bar{y} - t_{33}\bar{y} = 0$$

There are 9 unknowns $(t_{11}, \ldots, t_{33})$. One pair of corresponding peq-points (x, y) and $(\bar{x}, \bar{y})$ give rise to two such equations in 9 unknowns. n such points will imply $2n$ equations, which can be solved for 9 unknowns.

$$t_{11}x_1 + t_{12}y_1 + t_{13} - \bar{x}_1 t_{31} x_1 - \bar{x}_1 t_{32} y_1 - \bar{x}_1 t_{33} = 0$$

$$t_{11}x_2 + t_{12}y_2 + t_{13} - \bar{x}_2 t_{31} x_2 - \bar{x}_2 t_{32} y_2 - \bar{x}_2 t_{33} = 0$$

$$\vdots$$

$$t_{11}x_n + t_{12}y_n + t_{13} - \bar{x}_n t_{31} x_n - \bar{x}_n t_{32} y_n - \bar{x}_n t_{33} = 0$$

$$t_{21}x_1 + t_{22}y_1 + t_{23} - \bar{y}_1 t_{31} x_1 - \bar{y}_1 t_{32} y_1 - \bar{y}_1 t_{33} = 0$$

$$t_{21}x_2 + t_{22}y_2 + t_{23} - \bar{y}_2 t_{31} x_2 - \bar{y}_2 t_{32} y_2 - \bar{y}_2 t_{33} = 0$$

$$\vdots$$

$$t_{21}x_n + t_{22}y_n + t_{23} - \bar{y}_n t_{31} x_n - \bar{y}_n t_{32} y_n - \bar{y}_n t_{33} = 0$$

These equations can be written in matrix form as:

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -\bar{x}_1 x_1 & -\bar{x}_1 y_1 & -\bar{x}_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -\bar{x}_2 x_2 & -\bar{x}_2 y_2 & -\bar{x}_2 \\ \vdots & & & & & & & & \\ x_n & y_n & 1 & 0 & 0 & 0 & -\bar{x}_n x_n & -\bar{x}_n y_n & -\bar{x}_n \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -\bar{y}_1 x_1 & -\bar{y}_1 y_1 & -\bar{y}_1 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -\bar{y}_2 x_2 & -\bar{y}_2 y_2 & -\bar{y}_2 \\ \vdots & & & & & & & & \\ 0 & 0 & 0 & x_n & y_n & 1 & -\bar{y}_n x_n & -\bar{y}_n y_n & -\bar{y}_n \end{bmatrix} \begin{bmatrix} t_{11} \\ t_{12} \\ t_{13} \\ t_{21} \\ t_{22} \\ t_{23} \\ t_{31} \\ t_{32} \\ t_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \Rightarrow CQ = 0 \tag{4.7}$$

where C is a $2n \times 9$ matrix, Q is a 9×1 vector, and 0 is also $2n \times 1$ vector. This is a homogenous system which has multiple solutions. Therefore, we can arbitrarily fix one of the unknowns, and determine the remaining ones. Let $t_{33} = 1$, and rewrite above equation as:

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -\bar{x}_1 x_1 & -\bar{x}_1 y_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -\bar{x}_2 x_2 & -\bar{x}_2 y_2 \\ \vdots & & & & & & & \\ x_n & y_n & 1 & 0 & 0 & 0 & -\bar{x}_n x_n & -\bar{x}_n y_n \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -\bar{y}_1 x_1 & -\bar{y}_1 y_1 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -\bar{y}_2 x_2 & -\bar{y}_2 y_2 \\ \vdots & & & & & & & \\ 0 & 0 & 0 & x_n & y_n & 1 & -\bar{y}_n x_n & -\bar{y}_n y_n \end{bmatrix} \begin{bmatrix} t_{11} \\ t_{12} \\ t_{13} \\ t_{21} \\ t_{22} \\ t_{23} \\ t_{31} \\ t_{32} \end{bmatrix} = \begin{bmatrix} \bar{x}_1 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_n \\ \bar{y}_1 \\ \bar{y}_2 \\ \vdots \\ \bar{y}_n \end{bmatrix} \Rightarrow DR = S \tag{4.8}$$

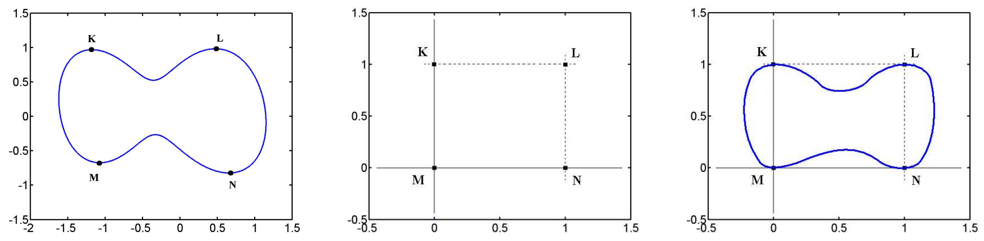
We can determine the unknown vector R by using pseudo-inverse, if at least 4 or more points and their corresponding point coordinates are known, namely

$$R = D^{\dagger} S = (D^T D)^{-1} D^T S \tag{4.9}$$

Once R is computed, the projection matrix T is automatically known.

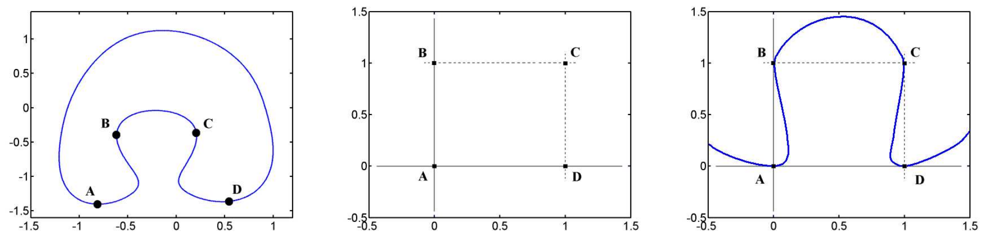
The canonical projective transformation matrix T_c that transforms the curve $C = (x_i, y_i)|_{i=1}^{n}$ to the corresponding canonical curve $C_c = (X_i, Y_i)|_{i=1}^{n}$, namely

$$C = (x_i, y_i)|_{i=1}^{n} \xrightarrow{T_c} C_c = (X_i, Y_i)|_{i=1}^{n} \tag{4.10}$$



(a) (b) (c)

Figure 4.6: (a) A curve with its four bitangent points, (b) the frame of unit square, (c) canonic projection of the curve.



(a) (b) (c)

Figure 4.7: (a) A curve with its bitangent and tangent points, (b)the frame of unit square, (c) canonic projection of the curve.

will be defined by mapping 4 peq-points, either 4 bitangent points or 2 bitangent and 2 tangent points, of $C = (x_i, y_i)|_{i=1}^{n}$ to the corners of the unit square as shown in Fig. 4.6 and 4.7.

In light of equations (4.8) and (4.9), substituting $\{X_i, Y_i\}$ for $i = 1, 2, 3, 4$, with the corners of unit square $\{(0,0), (0,1), (1,1), (1,0)\}$, we get:

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & 0 & 0 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & 0 & 0 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x_3 & -y_3 \\ x_4 & y_4 & 1 & 0 & 0 & 0 & -x_4 & -y_4 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & 0 & 0 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2 & -y_2 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -x_3 & -y_3 \\ 0 & 0 & 0 & x_4 & y_4 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} t_{c_{11}} \\ t_{c_{12}} \\ t_{c_{13}} \\ t_{c_{21}} \\ t_{c_{22}} \\ t_{c_{23}} \\ t_{c_{31}} \\ t_{c_{32}} \end{bmatrix} = \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \\ Y_1 \\ Y_2 \\ Y_3 \\ Y_4 \end{bmatrix} \Rightarrow DR = S \Rightarrow R = D^{-1}S \tag{4.11}$$

The canonical projective transformation matrix T_c can then be defined by the components of vector R with $t_{c_{33}} = 1$.

Note that this construction of canonical curves allows us to introduce multiple canonical curves, each of which can be defined by mapping 4 bitangent points of pairs of concavities to the corners of the unit square. This way, we will have more flexibility for the comparison and recognition of shapes with some similarity metrics.

4.3.3 Recognition with Invariants

The recognition of a partially occluded object is a major problem in computer vision. This problem has not been sufficiently resolved yet, although many people have been working on it for about last twenty years. Invariant measures like the cross-ratio (geometric invariant under perspective transformations) of four points on a line as shown in Fig. 4.8 can contain shape information and be used for identification of objects seen by a camera. For simplicity, we will restrict ourselves to planar objects here, so that the mapping from scene points to image ones is one to one. Figure 4.8 shows two planar curves with their tangent points (B, C) and $(\bar{B}, \bar{C})$, respectively, and the intersection points (A, D) and $(\bar{A}, \bar{D})$, respectively, which are acquired by intersecting the lines constructed from these tangent points with the curves, and their projection points (a, b, c, d) and $(\bar{a}, \bar{b}, \bar{c}, \bar{d})$, respectively.

The ratio of ratios of lengths on the line, called the cross-ratio, is given by

$$I = \frac{(C-A)(D-B)}{(C-B)(D-A)} = \frac{(c-a)(d-b)}{(c-b)(d-a)} \tag{4.12}$$

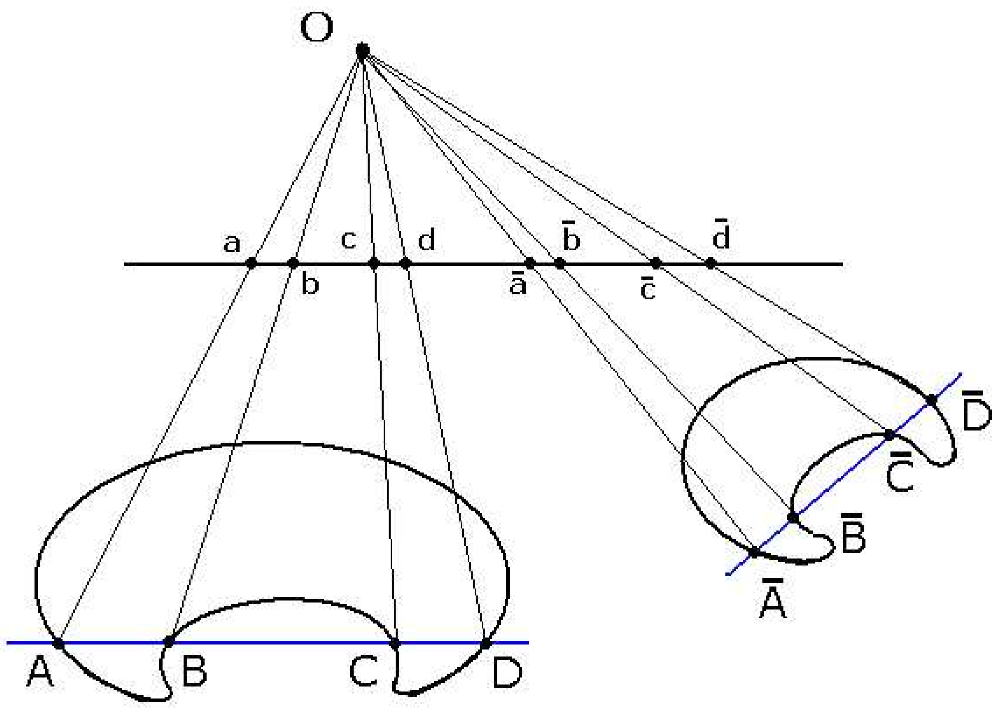


Figure 4.8: A one-dimensional construction of perspective viewing. The optical center of the camera is O. Under perspective projection, the length, and ratios of lengths, on a line are not invariant, but ratios of ratios are.

and

$$\bar{I} = \frac{(\bar{C} - \bar{A})(\bar{D} - \bar{B})}{(\bar{C} - \bar{B})(\bar{D} - \bar{A})} = \frac{(\bar{c} - \bar{a})(\bar{d} - \bar{b})}{(\bar{c} - \bar{b})(\bar{d} - \bar{a})} \tag{4.13}$$

where $(. - .)$ denotes euclidean distance between the points. If $I = \bar{I}$, then one can conclude that these curves might be equivalent. In summary, the cross-ratio is an invariant of any sets of four collinear points in projective correspondence. It is unaffected by the relative position of the line or the position of the optical center.

4.4 Bitangent Points In Visual Servoing

In this scenario bitangent points will be used in "teaching-by-show" method for shape alignment purpose. Let $C = (x_k, y_k)|_{k=1}^{n}$ and $C^* = (x_k^*, y_k^*)|_{k=1}^{n}$ are the extracted contours of the planar shape at initial and desired poses of the robot and $({}^b x_k, {}^b y_k)|_{k=1}^{m}$ and $({}^b x_k^*, {}^b y_k^*)|_{k=1}^{m}$ are corresponding bitangent points of them, respectively. Then these points can be used to construct the visual feature vectors s and

s^* as follows:

$$s = \begin{pmatrix} {}^b x_1 \\ {}^b y_1 \\ \vdots \\ {}^b x_m \\ {}^b y_m \end{pmatrix}, \quad s^* = \begin{pmatrix} {}^b x_1^* \\ {}^b y_1^* \\ \vdots \\ {}^b x_m^* \\ {}^b y_m^* \end{pmatrix}$$

Once the visual feature vectors are obtained, one can compute the control law as explained in Section 3.1.2 for steering the robotic system in order to align planar shapes.

Chapter 5

Experimental Results

This section presents experimental results for shape alignment, positioning and trajectory following tasks. The reference trajectories, which are pursued by the end-effector in experiments, are generated using a linear interpolator. This linear interpolator produces midway targets along the reference trajectory within the image and in the experiments performed. The interpolation speed was defined as follows,

$$\nu_{mid} = n_p \times fps \tag{5.1}$$

where ν_{mid}, n_p and *fps* denote velocity of the midway target in *pixel/s*, shift distance in pixels, and the number of retrieved frames in a second of the camera, respectively.

5.1 Experiments On A Robotic Arm

5.1.1 Shape Alignment

In this section, experimental results on planar shape alignment using model free visual servoing are presented. For comparison purposes, a calibrated approach using bitangents is also provided.

Experiments were conducted with a 7 DOF Mitsubishi PA10 robot arm and a Unibrain Fire-i400 digital camera. The camera is mounted on a tripod in eye-to-hand configuration in order to observe the motion of the end-effector. The images were digitized at 320×240 resolution. The system setup is shown in Fig.5.1. The visual control and image processing modules are implemented in VC++ 6.0 using OpenCV library and run on P4 2.26GHz with 1GB ram personal computer.

Fig. 5.2 shows a test shape, which is on a plane and rigidly attached to the end-effector. Bitangent points of the shape are acquired using the proposed algorithm in

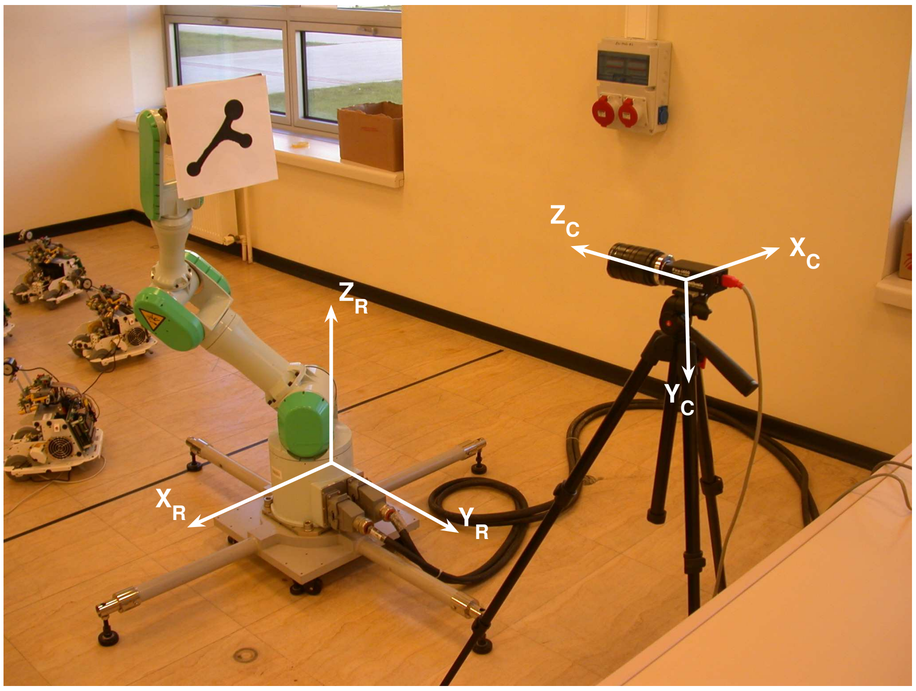


Figure 5.1: System setup with drawn robot control frame and camera frame.

Section 4.2. For visual servoing purposes, either bitangent points or their midpoints, see points denoted by 1, 2 and 3 in Fig. 5.2, can be used. Unlike bitangent points, which are projective invariant, midpoints are affine invariant. If the scene's depth is much less than its distance from the camera, a weak-perspective projection can be assumed. Throughout the experiments weak-perspective assumption is made and the visual feature vector s is constructed from the midpoints as follows:

$$s = [x_1, y_1, x_2, y_2, x_3, y_3]^T.$$

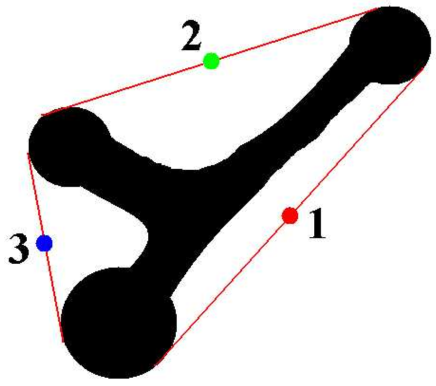


Figure 5.2: Test shape and its feature points.

In the case of perspective projection, i.e. if the weak-perspective assumption does not hold, one can use bitangent points to construct the visual feature vector.

For alignment task, desired pose of the shape is obtained during an off-line stage by moving the robot in xz-plane of the robot control frame (see Fig. 5.1) with some translational velocities V_x, V_z and rotational velocity Ω_y for a certain time interval. Consequently, the desired feature vector s^* is constructed from this reference pose.

A. Calibrated Visual Servoing Results

The parameters $f_x = 1000$, $f_y = 1000$, $x_c = 160$, $y_c = 120$ are obtained by a coarse calibration of the camera and $Z = 2000\ mm$. The robot base frame is positioned at $z = 2000\ mm$ in z-axis and $y = 1000\ mm$ in y-axis away from the camera frame. Thus, we have

$$R = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & -1 & 0 \end{pmatrix}, \quad t = \begin{pmatrix} 0 \\ 1000 \\ 2000 \end{pmatrix}$$

where R is the rotational matrix and t is the translational vector that are used for the construction of robot-to-camera transformation matrix T which is defined in (3.22). The gain matrix Λ in (3.26) is tuned as $\Lambda_i = 0.3$ for $i = 1, 2, .., 6$. The control input is defined as

$$\mathbf{u} = \begin{pmatrix} V_x & V_z & \Omega_y \end{pmatrix}^T$$

where $\mathbf{u}$ consists of the 1^{st} and 3^{rd} components of V_R (end-effector velocity screw) for the motion in xz-plane and 5^{th} component of V_R for the rotation around y-axis in robot control frame, respectively. Fig. 5.3 depicts the initial and the desired images. Fig. 5.4 shows feature trajectories. Alignment errors and control signals are plotted in Figs. 5.5-5.6. The norm of the resulting alignment error is found to be less than 1 pixel.

B. Uncalibrated Visual Servoing Results

Here we do not need the calibration parameters since the composite Jacobian $J \in \Re^{6\times3}$ is estimated in a recursive manner using RLS. Only 3 joints, namely the 2^{nd}, the 4^{th} and the 6^{th} joints of PA10 robot are used to steer the end-effector by locking

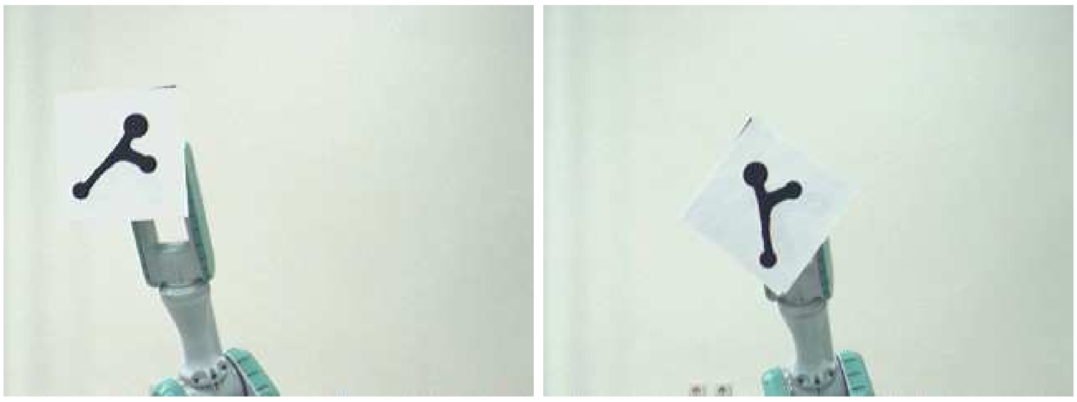


Figure 5.3: Initial and desired images

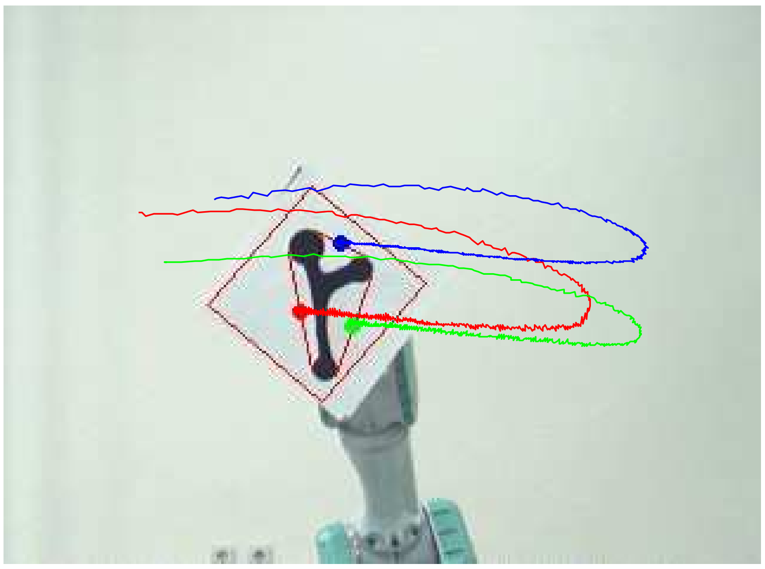


Figure 5.4: Feature trajectories on the image plane

the remaining 4 joints. The control parameters are set as $\lambda = 0.96$ and $K_p = 0.6$ for Dynamic Gauss-Newton controller with RLS Jacobian estimation. The control input is defined as

$$\mathbf{u} = \begin{pmatrix} \Omega_2 & \Omega_4 & \Omega_6 \end{pmatrix}^T$$

where Ω_2, Ω_4 and Ω_6 are the joint velocities. Figs. 5.7-5.8 depict the initial and the desired images, and the feature trajectories on the image plane. Alignment errors and the control signals are plotted in Figs. 5.9-5.10, respectively. The norm of the resulting alignment error is found to be less than 1.5 pixel.

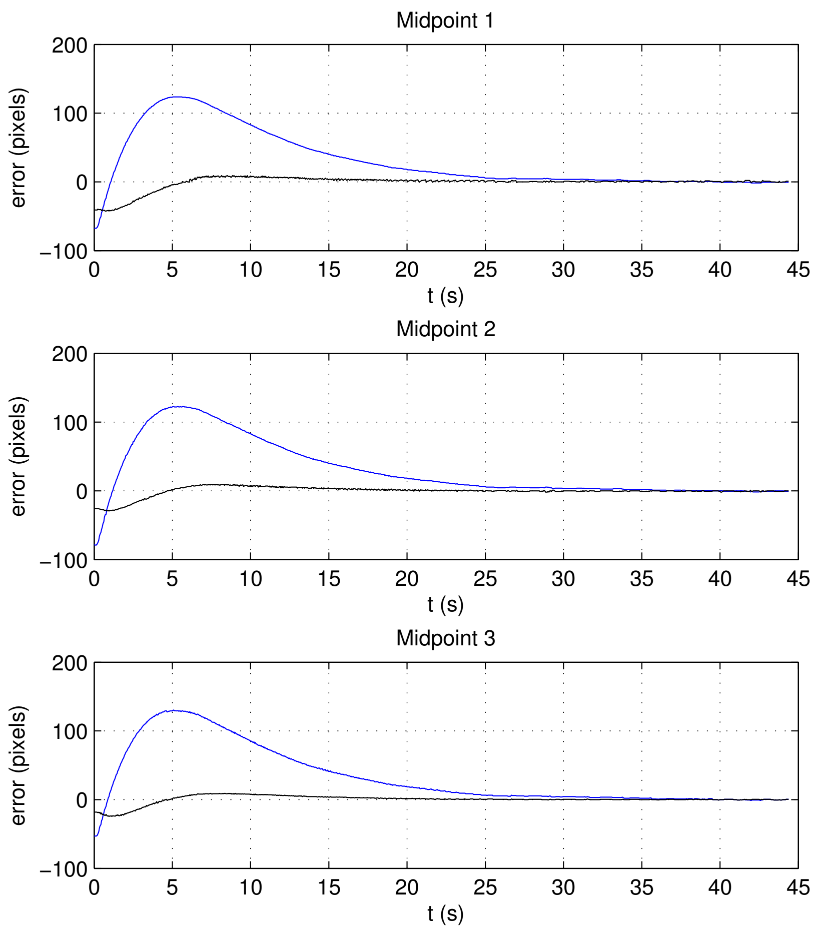


Figure 5.5: Alignment errors

C. Discussion

In both visual servoing approaches it is observed that alignment task errors are less than 1.5 pixels, which corresponds to 5 mm in robot workspace. It can be seen that calibrated approach draws more smoother trajectories while the uncalibrated one shows ambiguous behaviour until the Jacobian converges and the end-effector moves towards the desired pose. Computation times of region tracking, curve detection and bitangent extraction modules are approximately 13 ms, 5 ms and 4 ms, respectively.

5.1.2 Trajectory Following

In this section, the model free visual servoing approach for trajectory following tasks was accomplished with dynamic Gauss-Newton controller and tested in square,

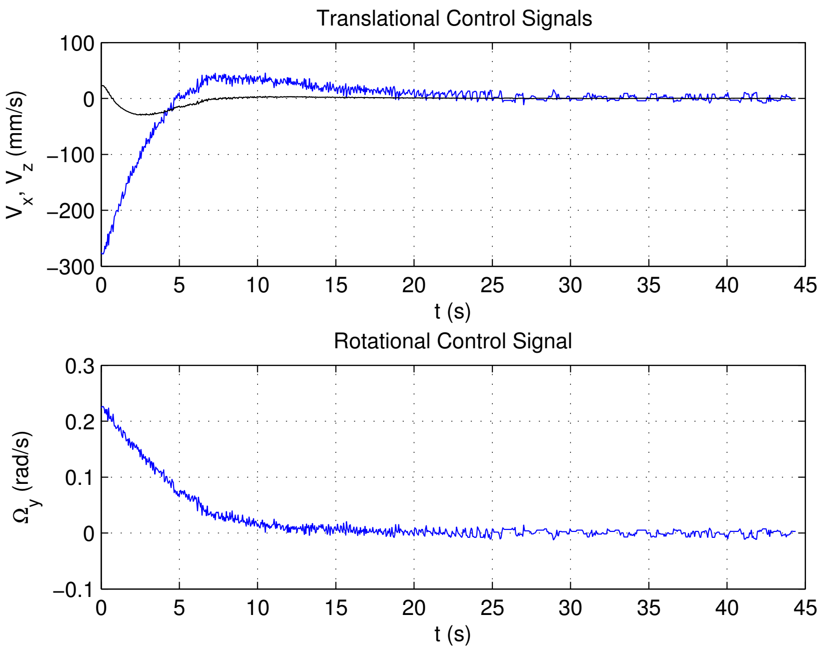


Figure 5.6: Control signals V_x, V_z and Ω_y

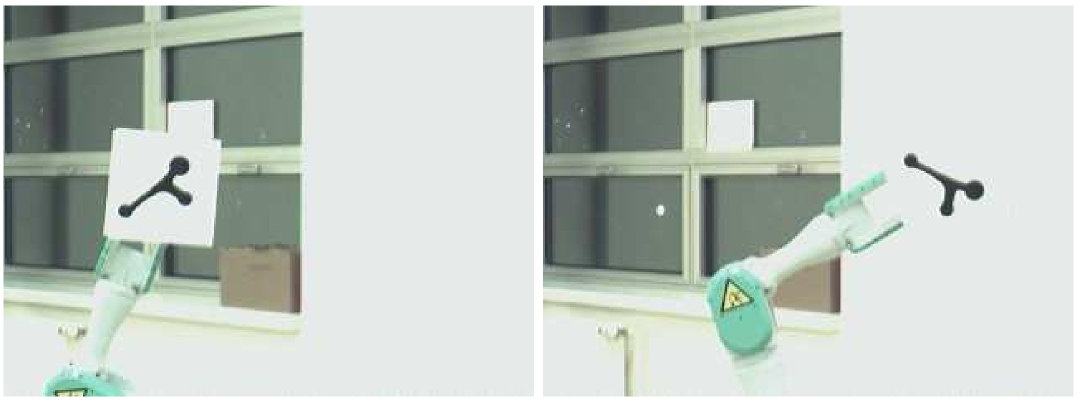


Figure 5.7: Initial and desired images

circular and sinusoidal paths.

All experimental results were evaluated in terms of accuracy and precision. Accuracy and precision values were determined as the mean and the standard deviation of the error-norms. A linear interpolator was used to generate midway targets to make the end-effector track them along these reference trajectories. The tracking performances for these three trajectories are depicted in Table 5.1. The tracking error was computed as the distance between the end-effector and the current midway target at each frame. Figs. 5.11, 5.12 and 5.13 show results of trajectory following experiments and the error-norms versus time graphs.

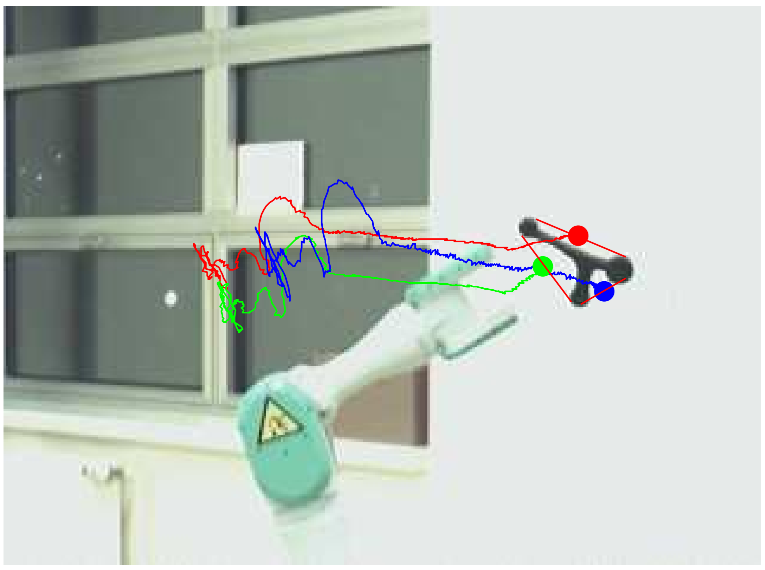


Figure 5.8: Feature trajectories on the image plane

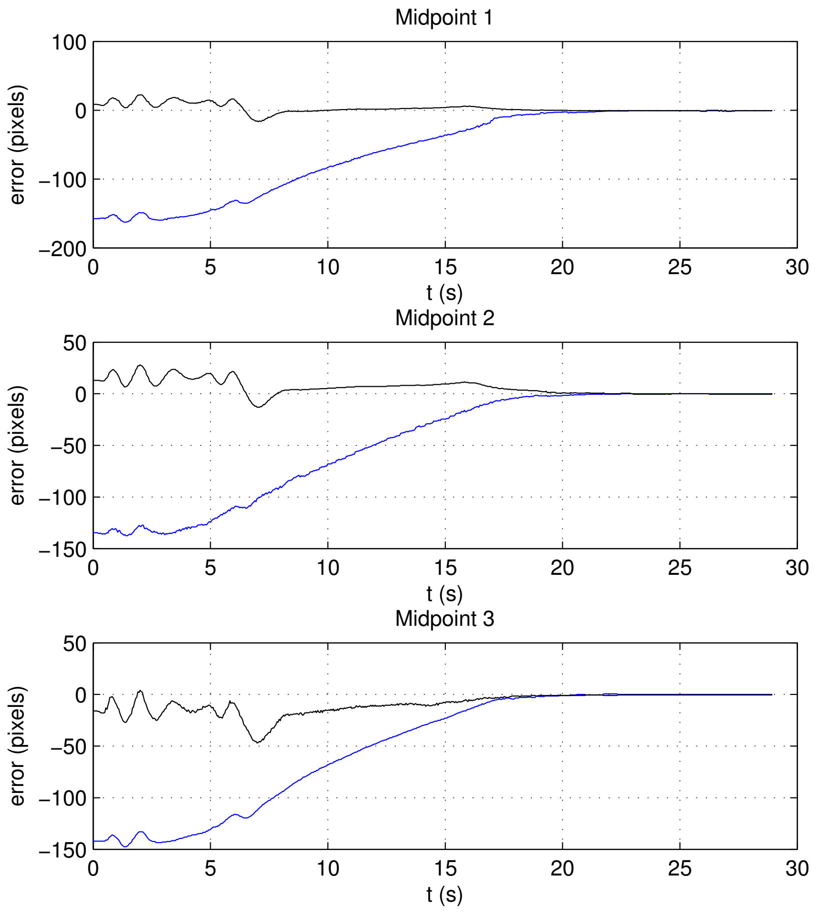


Figure 5.9: Alignment errors

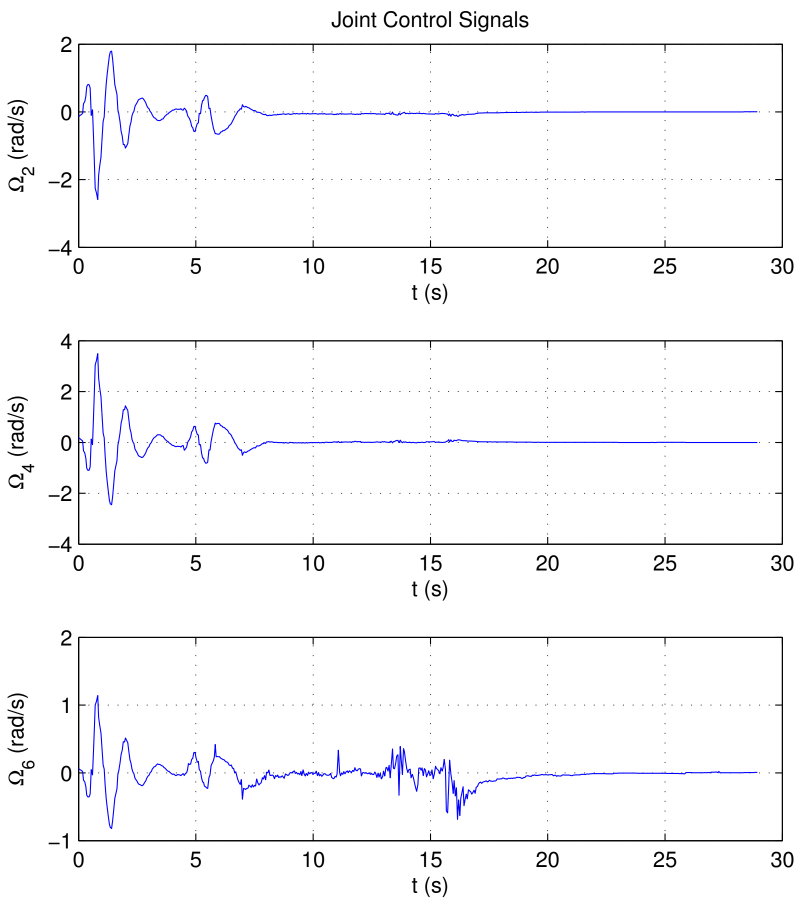
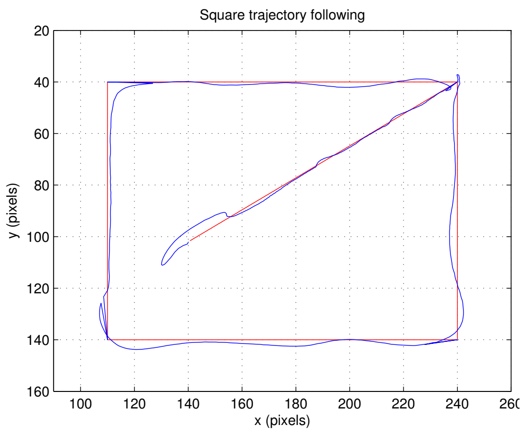


Figure 5.10: Control signals Ω_2, Ω_4 and Ω_6



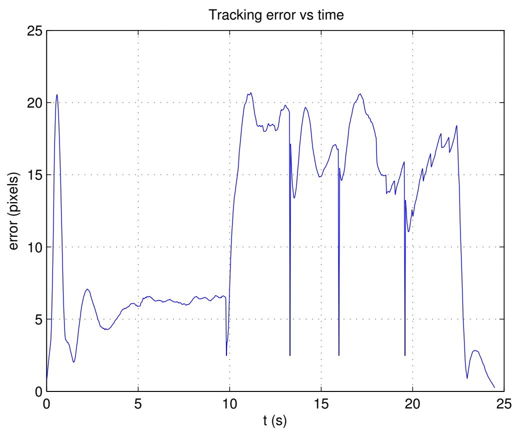


Figure 5.11: Square trajectory and tracking error for dynamic Gauss-Newton controller with RLS Jacobian estimation

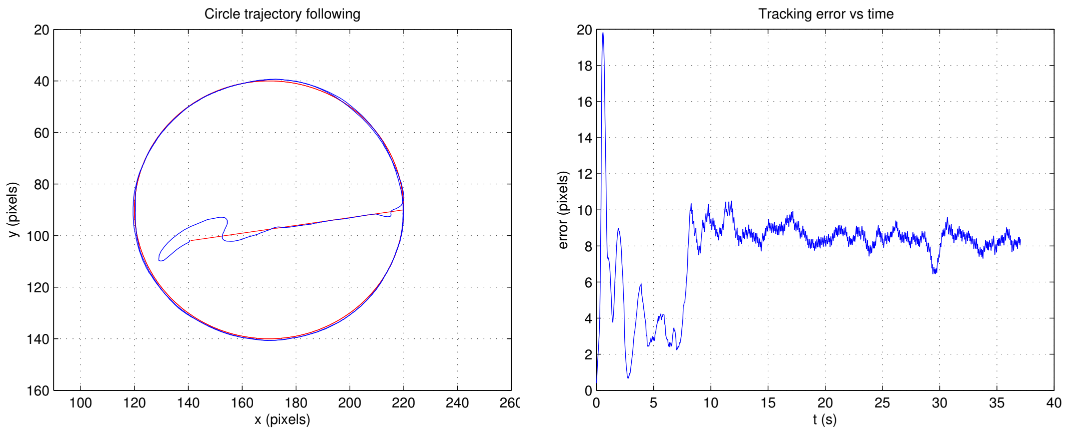


Figure 5.12: Circle trajectory and tracking error for dynamic Gauss-Newton controller with RLS Jacobian estimation

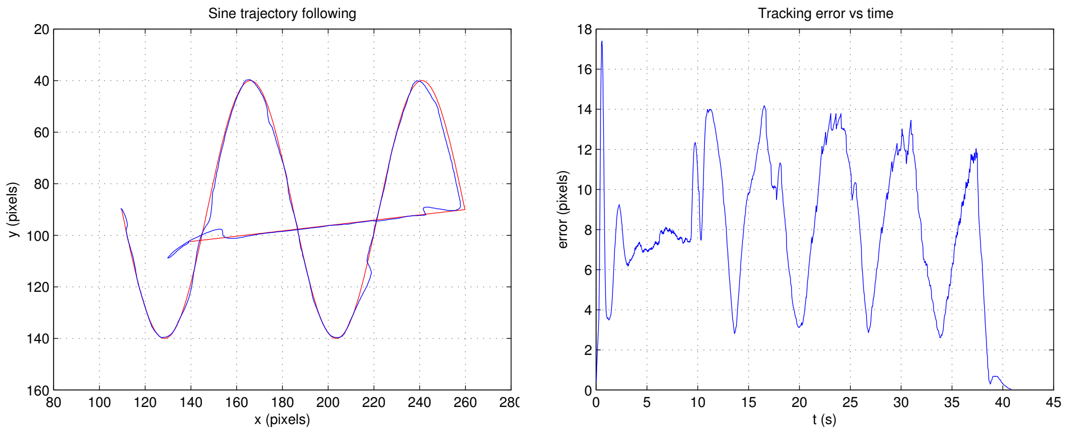


Figure 5.13: Sine trajectory and tracking error for dynamic Gauss-Newton controller with RLS Jacobian estimation

Table 5.1: Results for trajectory tracking on PA10

	Square		Circle		Sine	
	Acc.	Prec.	Acc.	Prec.	Acc.	Prec.
pixels	11.23	6.22	7.65	2.48	8.11	3.55
cm	3.7	2	2.6	0.8	2.7	1.2

A. Discussion

It can be seen from the presented results and graphs that model free visual servoing has performed the trajectory following tasks with centimeter accuracies. On the average, the tasks were achieved with 10 *pixels* accuracy which corresponds to 3.5 *cm* in robot workspace.

5.2 Experiments On A Microassembly Workstation

Our microassembly workstation consists of a Nikon SMZ1500 optical stereomicroscope that has a CCD camera module adapter onto which a Basler A602fc camera with $9.9\mu m \times 9.9\mu m$ cell sizes is mounted. The microscope has 1.6X objective and additional zoom. Zoom levels can be varied between 0.75X-11.25X, implying $15:1$ zoom ratio. Fig. 5.14 shows the complete microassembly system. The gripper that was used in the experiments is a Zyvex microgripper with an opening gap of 100 μm and it is rigidly fastened to a PI M-111.1 high-resolution micro-translation stage with 50 nm incremental motion in x, y and z positioning axes (see Fig. 5.15). The controllers for linear stages were implemented on dSpace ds1005 motion control board which steers the microgripper. The visual tracking algorithm (ESM) accomplished to track a 50×50 window up to 250 pixels/sec velocity at 33 Hz.

5.2.1 Tasks

Micro tasks were conducted on our microassembly station and visual feedback has been provided through coarse visual path of the microscope. In experiments, visual servoing was accomplished with dynamic Gauss-Newton and Optimal controllers for micropositioning and trajectory following tasks at 1X and 4X zoom levels. Fig. 5.16 depicts the microgripper for two different zoom levels.

Last two columns of Table 5.2 show the area in mm^2 of the microscopic view and the effective pixel size (resolution) for the zoom levels indicated in the first column. All experimental outcomes were assessed in terms of accuracy and precision. Accuracy and precision values were determined as the mean and the standard deviation of the error-norms. To estimate initial microscopic system Jacobian, each linear stage is successively moved by a small amount and the change of microgripper position in image is used to build its components. The microgripper is then servoed in workspace for a while to ensure convergence of the Jacobian to its true values.

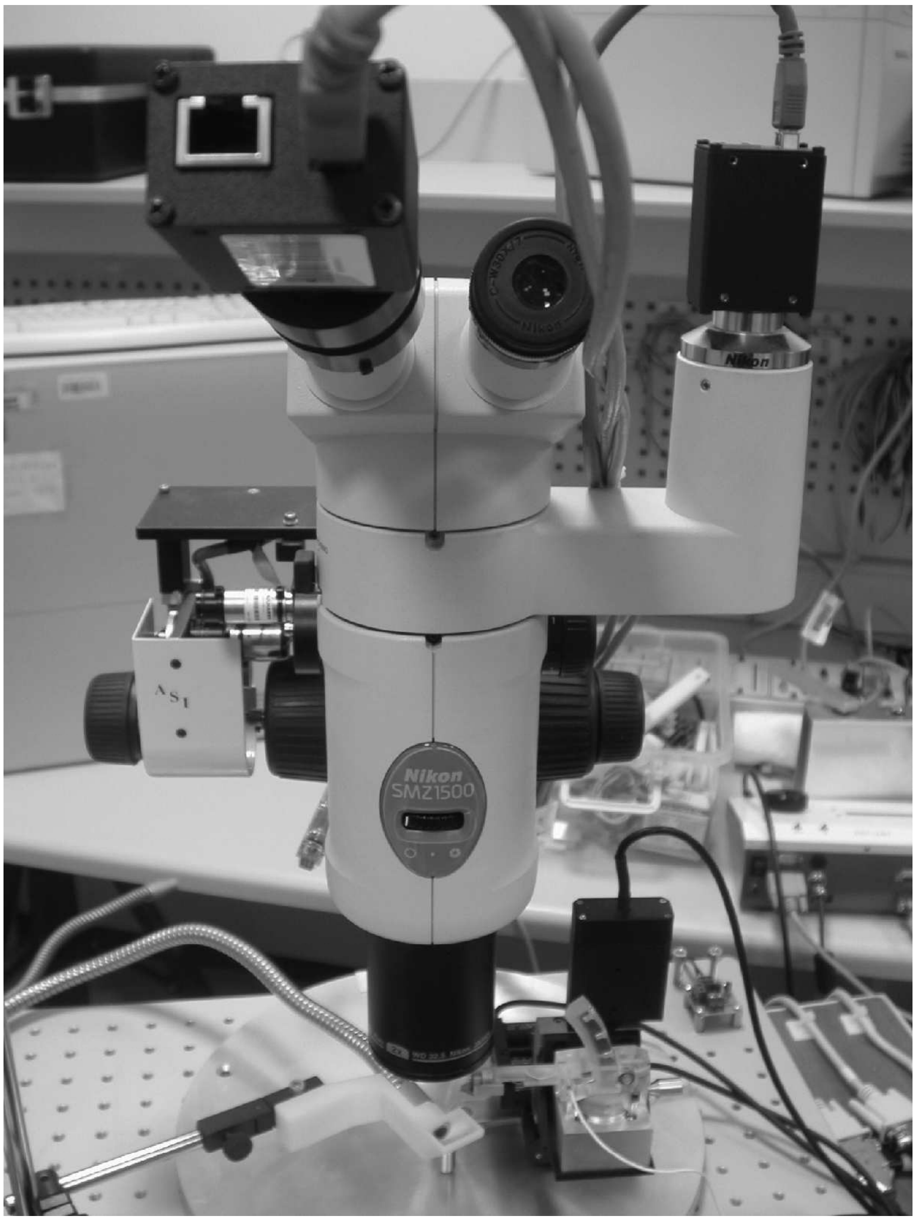


Figure 5.14: Microassembly workstation and attached visual sensors

Table 5.2: System Parameters

Z	$Area$ (mm^2)	ΔP (μm)
1X	4×3	6.18
4X	1×0.75	1.55

5.2.2 Micropositioning

In this task the microgripper was sent to a desired position from an arbitrary initial position by giving step inputs of 50 pixels both in x and y directions as references.

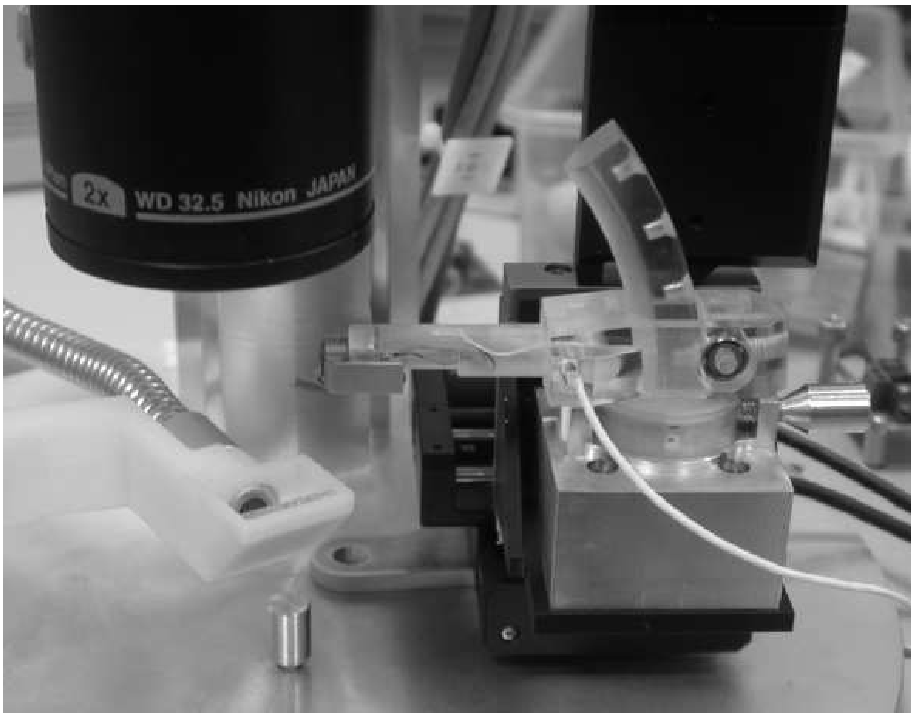


Figure 5.15: Microgripper mounted on linear stages in assembly workspace

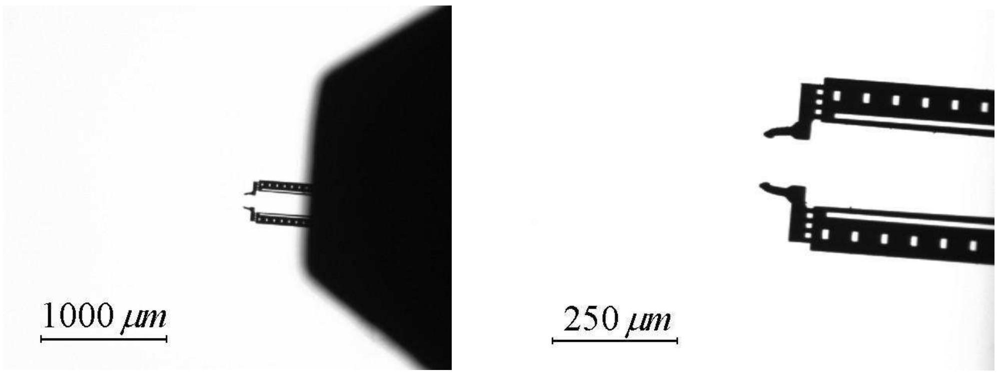


Figure 5.16: Views of microgripper at 1X and 4X

This corresponds to 70.8 pixels from the initial position. Results of these experiments for the Dynamic Gauss-Newton and the Optimal control, are tabulated in Tables 5.3 and 5.4 where Z, $\{K_p, Q, L\}$, $Step$, t_s, $Acc.$ and $Prec.$ represent zoom level, control gains, step input, settling time, accuracy and precision, respectively. The positioning errors were calculated after the response was settled and remained in 3% of its final value. Figs. 5.17 and 5.18 demonstrate the step responses and the corresponding Optimal control signals for a trial under 1X and 4X zoom levels .

Table 5.3: Dynamic Gauss-Newton control results for micropositioning

Z	K_p	*Step* (pix)	t_s (*sec*)	*Acc.* (μm)	*Prec.* (μm)
1X	4	50	1.6	4.37	1.32
4X	2	50	3	2.81	1.44

Table 5.4: Optimal control results for micropositioning

Z	Q	L	*Step* (pix)	t_s (*sec*)	*Acc.* (μm)	*Prec.* (μm)
1X	0.9	0.05	50	1.6	8.60	3.65
4X	0.6	0.4	50	1.6	4.74	1.92

5.2.3 Trajectory Following

Apart from micropositioning, the same model free visual servoing was tested in trajectory following tasks with square, circle and sine trajectories. A linear interpolator was used to generate midway targets to make the microgripper pursue them along these reference trajectories. The upshots for these trials are depicted in Tables 5.5 and 5.6. The tracking error was computed as the distance between the microgripper and the current midway target at each frame. Figs. 5.19, 5.20 and 5.21 depict results of trajectory following experiments and the error-norms versus time graphs. Performance versus microassembly tasks for two controllers are also depicted in Figs. 5.22 and 5.23 where each ellipse defines the accuracy (center of the ellipse) and the precision (half length of the major axis of the ellipse) of the performed task.

Table 5.5: Dynamic gauss-newton control results for trajectory following

Z	(μm)	Square	Circle	Sine
1X	Acc.	3.78	24.87	11.36
	Prec.	3.43	4.62	5.87
4X	Acc.	1.45	6.08	2.86
	Prec.	1.45	2.95	1.85

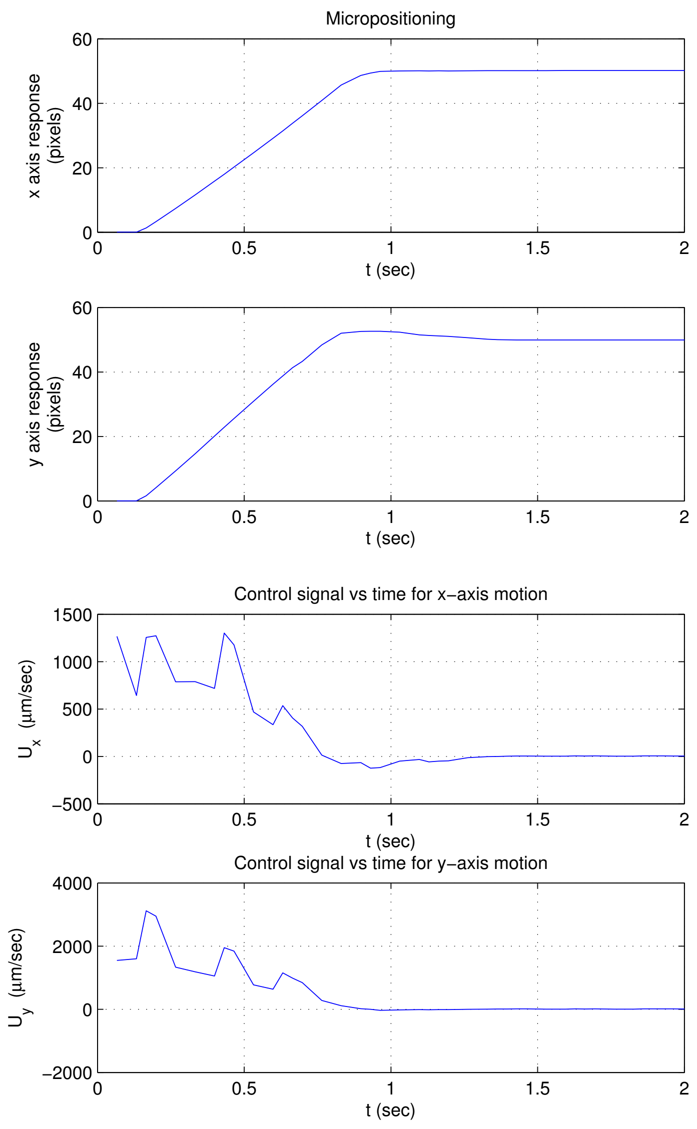


Figure 5.17: Step responses and optimal control signals at 1X

Table 5.6: Optimal control results for trajectory following

Z	(μm)	Square	Circle	Sine
1X	Acc.	8.65	21.05	6.14
	Prec.	2.70	2.90	2.74
4X	Acc.	1.64	3.30	1.17
	Prec.	1.12	1.17	0.57

5.2.4 Discussions

It can be seen from the presented tables and graphs that model free visual servoing has performed the micropositioning and trajectory following task with micron accu-

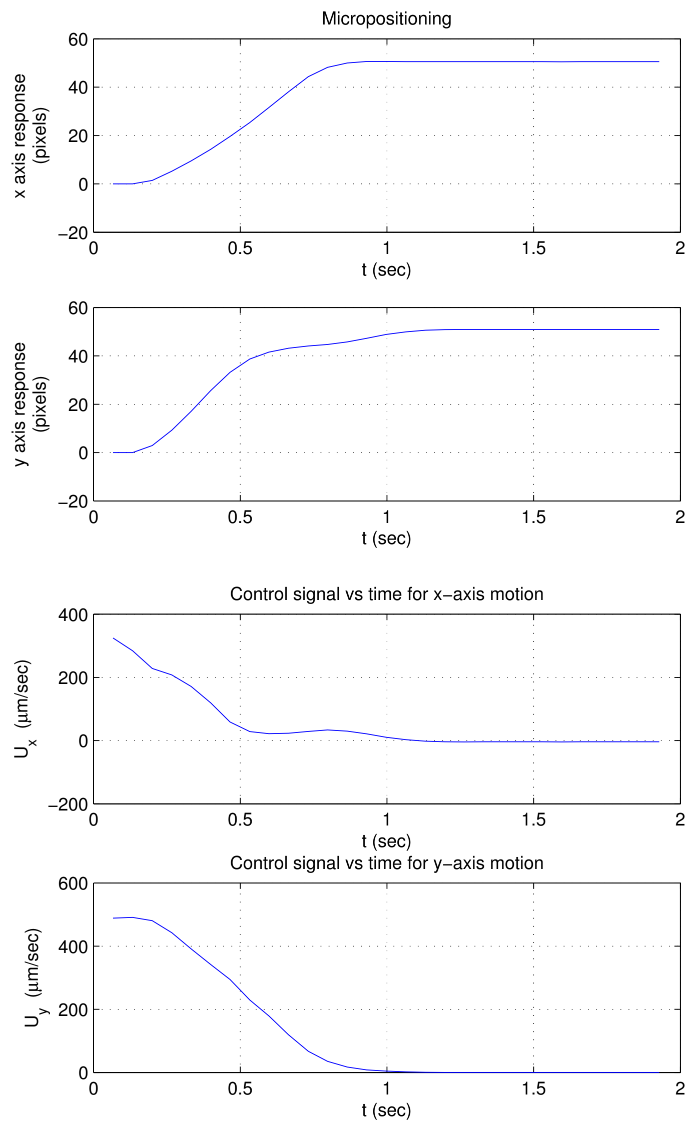


Figure 5.18: Step responses and optimal control signals at 4X

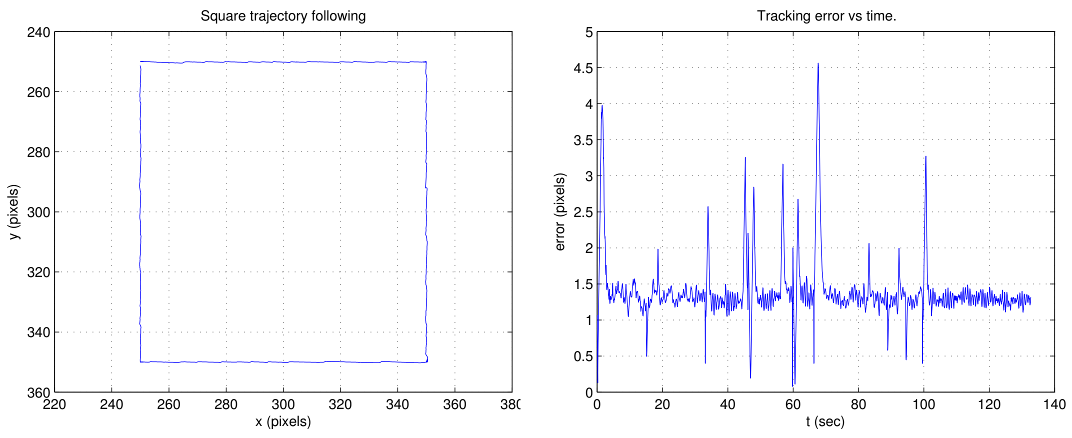
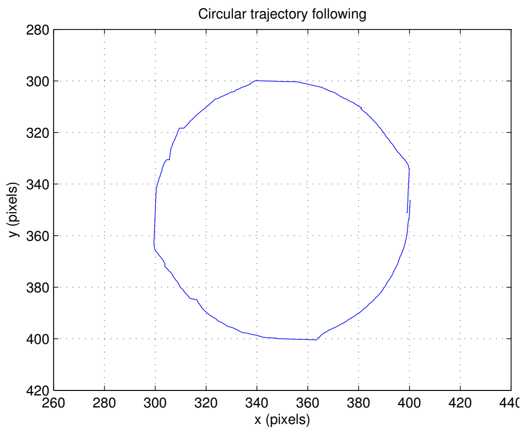


Figure 5.19: Square trajectory and the tracking error using optimal control at 1X



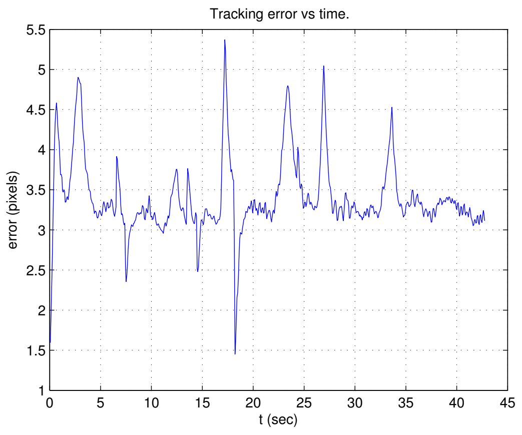
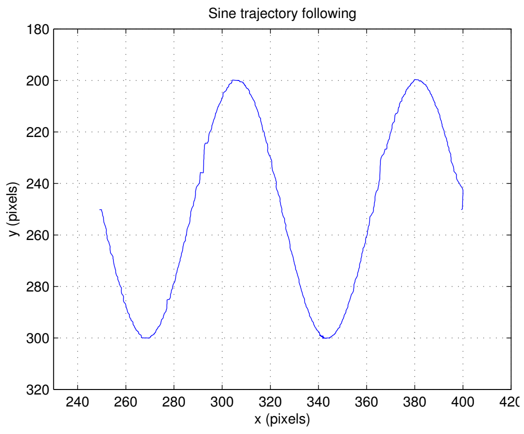


Figure 5.20: Circle trajectory and the tracking error using optimal control at 1X



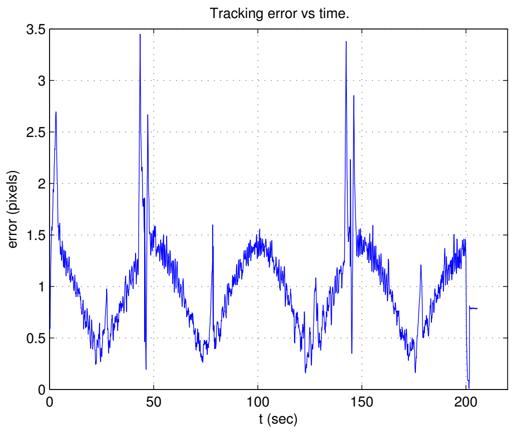


Figure 5.21: Sine trajectory and the tracking error using optimal control at 1X

racies. On the average, the tasks were achieved with 5 μm and 3 μm accuracies for positioning and with 12 μm and 3 μm accuracies for trajectory following at 1X and 4X zoom levels, respectively. Upon inspection of controllers, we see that the performance of Dynamic Gauss-Newton is better than Optimal control in linear motions (positioning and square trajectory following) while Optimal controller performs better than the previous one in nonlinear motions (circle and sine trajectory following). Also the precision results in both zoom levels for Dynamic Gauss-Newton control are worse than those of Optimal control. If time considerations are important for the tasks, uncalibrated visual servoing approach is more sluggish than the calibrated one.

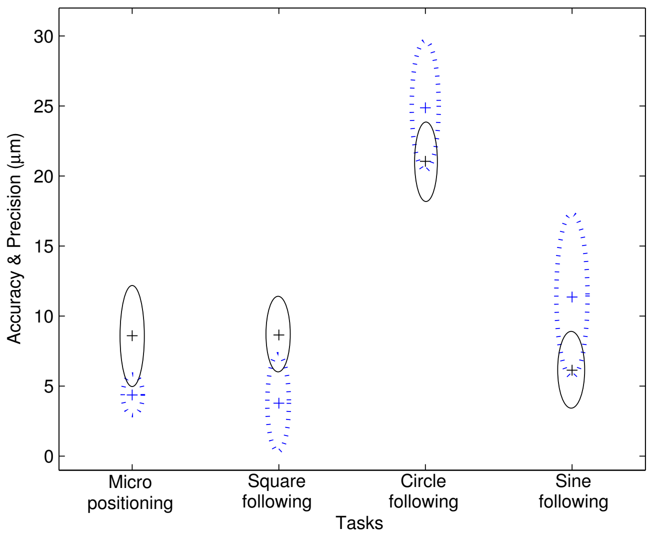


Figure 5.22: Accuracy & precision ellipses for Dynamic Gauss-Newton (dotted) and Optimal (solid) controllers at 1X

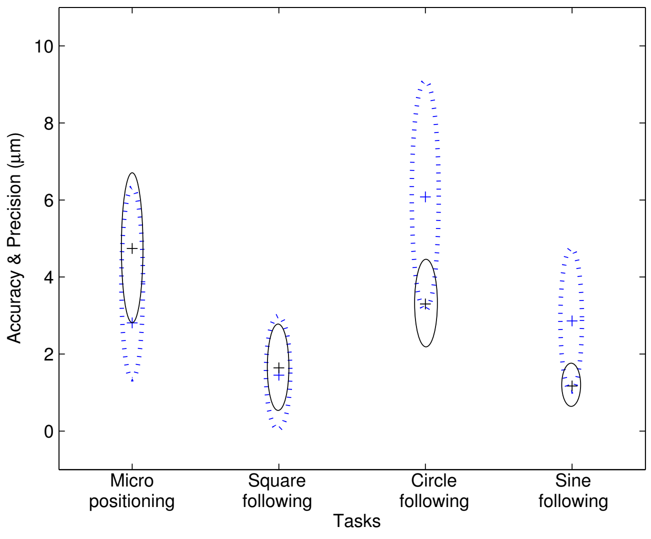


Figure 5.23: Accuracy & precision ellipses for Dynamic Gauss-Newton (dotted) and Optimal (solid) controllers at 4X

Chapter 6

Conclusion

In this thesis, performance of model free visual servoing algorithms are experimentally evaluated for various tasks performed both in macro and micro domains.

Particularly, in Chapter 2 the fundamentals of visual servoing such as camera model, system configurations, control architectures etc. are summarized. In Chapter 3 the theory of both model based and model free visual servoing approaches are presented. In model based approach, which needs camera parameters, kinematic model of robot, and the model of the observed scene, the analytical model of image Jacobian for a specific feature is derived and visual control law is designed. In model free approach, which does not necessitate system parameters, the dynamic composite Jacobian estimation methods are presented and their usage in dynamic Gauss-Newton and Optimal controllers are shown. A novel approach for planar shape alignment using bitangents is developed in Chapter 4. Bitangent points, which are acquired using convex-hull of a curve, are used to design image based visual servoing schemes, both calibrated and uncalibrated, for a fixed camera system. The assumption is that the curve has at least one concavity on its boundary shape. How bitangent points can be used for object recognition and comparison before initiating visual servoing is also discussed in the same chapter. Chapter 5 was on the experimental results performed both on a robotic arm and on a microassembly workstation.

In macro domain, shape alignment and trajectory following experiments are performed on 7 DOF Mitsubishi PA10 robotic arm, respectively. Alignment tasks are performed with *milimeter* accuracies. In trajectory following tasks, the average accuracy is around *a few centimeters.*

In micro domain, the use of model free visual servoing approach is also investigated and experimentally validated in micropositioning and trajectory following tasks. Model free visual servoing has the advantages of carrying out a task without requiring a model of the system and adapting itself to different operating modes through a dynamic estimation of the composite Jacobian. Experimental results show that positioning and trajectory following tasks can be performed in a robust manner with *micron* accuracies. The performance of model free visual servoing approach has been evaluated with two different controllers. It has been observed that for linear motion Dynamic Gauss-Newton controller shows slightly better performance, while Optimal controller does better job for the rest of the trajectories.

As a future work, 3D alignment tasks can be tackled using the idea of convex-hull in 3D and the use of a hybrid visual feature vector which includes both 2D and 3D features in the design of the model free visual servoing methods.

Bibliography

[1] Peter I. Corke, Visual Control of Robots: high-performance visual servoing, Wiley & Sons Inc. 1996.

[2] Y. Shirai and H. Inoue, Guiding a robot by visual fcedback in assembling tasks, Pattern Recognition, 5, 1973, 99-108.

[3] J. Hill and W. T. Park, Real time control of a robot with a mobile camera, in Proc. 9th ISIR, Washington, D.C., Mar. 1979, 233-246.

[4] K. Hosoda, M. Asada, Versatile visual servoing without knowledge of true Jacobian, in Proc. IEEE/RSJ *Int. Conf. on Intelligent Robots and Systems*, 1994, 186-193.

[5] L. Weiss, A. C. Sanderson, and C. Neuman, Dynamic sensor-based control of robots with visual feedback, IEEE *Journal of Robotics and Automation*, 3(5), 1987, 404-417.

[6] M. Jagersand, 0. Fuentes, and R. Nelson, Experimental evaluation of uncalibrated visual servoing for precision manipulation, in Proc. IEEE *Int. Conf. on Robotics and Automation*, 1997, 2874-2880.

[7] J. A. Piepmeier, H. Lipkin, Uncalibrated Eye-in-Hand Visual Servoing, *The International Journal of Robotics Research*, 2003.

[8] J. A. Piepmeier, G. V. McMurray, and H. Lipkin, Uncalibrated dynamic visual servoing, IEEE *Trans. on Robotics and Automation*, 20(1), 2004, 143-147.

[9] J. Qian and J. Su, Online Estimation of Image Jacobian Matrix by Kalman-Bucy Filter for Uncalibrated Stereo Vision Feedback, *International Conference on Robotics and Automation*, 2002.

[10] Xiadong Lv and Xinhan Huang, Fuzzy Adaptive Kalman Filtering based Estimation of Image Jacobian for Uncalibrated Visual Servoing, Proc. IEEE/RSJ *International Conference on Intelligent Robots and Systems*, 2006.

[11] E. Ozgur, M. Unel, Image Based Visual Servoing Using Bitangent Points Applied to Planar Shape Alignment, IASTED *Robotics and Applications*, 2007.

[12] J. T. Feddema, C. S. George Lee, and O. R. Mitchell, Weighted Selection of Image Features for Resolved Rate Visual Feedback Control. *IEEE Trans. on Robotics and Automation*, 7(1), 1991, 31-47.

[13] O. Faugeras and G. Toscani, The Calibration Problem for Stereo, *In IEEE Int. Conf. on Computer Vision and Pattern Recognition*, 1986, 15-20.

[14] O. Faugeras, Q.-T. Luong, and S. J. Maybank, Camera Self-Calibration: Theory and Experiments, *ECCV '92*, 588, 1992, 321-334.

[15] B. Espiau, F. Chaumette, and P. Rives, A New Approach to Visual Servoing In Robotics, *IEEE Trans. Robot. Automat.*, 8, 1992, 313-326.

[16] C. Harris and M. Stephens, A Combined Corner and Edge Detector, *In 4th Alvey Vision conference*, 1988, 147-151.

[17] J. F. Canny, A Computational Approach to Edge Detection, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(6), 1986, 679-698.

[18] G. D. Hager, The "X-Vision" System: A General Purpose Substrate for Real-Time Vision Based Robotics, In Proc. *Workshop on Vision for Robots*, 1995, 56-63.

[19] F. Chaumette, P. Rives, B. Espiau, Positioning of a Robot with Respect to an Object, Tracking it and Estimating its Velocity by Visual Servoing, *IEEE International Conference on Robotics and Automation Proceedings*, 3, 1991, 2248 - 2253.

[20] A. C. Sanderson and L. E.Weiss, Image Based Visual Servo Control Using Relational Graph Error Signal, *In Proc. of the Int. Conf. on Cybernetics and Society*, 1980, 1074-1077.

[21] Stephen J. Ralis, Barmeshwar Vikramaditya, and Bradley J. Nelson, Micropositioning of A Weakly Calibrated Microassembly System Using Coarse-to-Fine Visual Servoing Strategies, *IEEE Trans. On Electronics Packing Manufacturing*, 2000.

[22] S. Haykin, Adaptive Filter Theory. Englewood Cliffs, NJ: Prentice- Hall, 1991.

[23] N.J. Ayache and O.D. Faugeras, HYPER: A new approach for the recognition and positioning of two-dimensional objects, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8(1), 1986, 4454.

[24] C. A. Rothwell, A. Zisserman, D. A. Forsyth and J. L. Mundy, Planar Object Recognition using Projective Shape Representation, *International J. of Computer Vision*, 16, 1995, 57-59.

[25] Hemant D. Tagare, Shape-based non-rigid correspondence with application to heart motion analysis, *IEEE Trans. Medical Imaging*, 18(7), 1999, 570578.

[26] G. D. Hager. A modular system for robust positioning using feedback from stereo vision. *IEEE Transactions on Robotics and Automation*, 13(4), 1997, 582-595.

[27] S. Hutchinson, G. D. Hager, and P. I. Corke, A tutorial on visual servo control, IEEE*Trans. on Robotics and Automation*, 12(5), 1996, 651-670.

[28] Y.Lamdan, J.T.Schwartz, and H.J. Wolfson, Object recognition by affine invariant matching, *In Proc. CVPR*, pages 1988, 335-344.

[29] Jacopo Piazzi, Domenico Prattichizzo, Noah J. Cowan, Auto-epipolar Visual Servoing, *International Conference on Intelligent Robots and Systems*, 2004.

[30] Patrick Rives, Jose R. Azinheira, Linear Structures Following by an Airship using Vanishing Point and Horizon Line in a Visual Servoing Scheme, *Internatlonal Conference on Robotics & Automation*, 2004.

[31] J.Sklansky, Measuring concavity on a rectangular mosaic, IEEE *Trans Comput.* 21, 1972, 1355-1364.

[32] E. Ozgur, M. Unel. Object Recognition by Matching Multiple Concavities, *IEEE Conference on Signal Processing and Communications Applications, SIU06*, 2006.

[33] J. L. Mundy, Andrew Zisserman, *Geometric invariance in computer vision*, The MIT Press, 1992.

[34] R.L. Graham, An efficient algorithm for determining the convex hull of a finite planar set, Info. Proc. Lett. 1, 1972, 132-133.

[35] F.Preparata and S. Hong, Convex hulls of finite sets of points in two and three dimensions, Common. ACM 20, 1977, 87-93.

[36] S. Benhimane and E. Malis, Real-time image-based tracking of planes using efficient second-order minimization, IEEE/RSJ *International Conference on Intelligent Robots Systems*, Sendai, Japan, October 2004.