RANDOMIZED PATH PLANNING FOR VISUAL SERVOING

by

Moslem Kazemi
M.A.Sc., Industrial Systems Engineering, University of Regina, 2004
B.A.Sc., Computer Engineering, Sharif University of Technology, 2000

A Thesis submitted in partial fulfillment
of the requirements for the degree of

Doctor of Philosophy

in the
School of Engineering Science
Faculty of Applied Sciences

SIMON FRASER UNIVERSITY
Fall 2012

Approval

Name: **Moslem Kazemi**

Degree: **Doctor of Philosophy, Engineering Science**

Title of Thesis: ***Randomized Path Planning for Visual Servoing***

Examining Committee:

Chair: **Daniel Lee**
Professor

Kamal K. Gupta
Senior Supervisor
Professor

Mehran Mehrandezh
Supervisor
Associate Professor, University of Regina

Parvaneh Saeedi
Supervisor
Assistant Professor

Ahmad Rad
Internal Examiner
Professor
Engineering Science

Francois Chaumette
External Examiner
Professor, INRIA, France

Date Defended: November 20, 2012

Partial Copyright Licence

Abstract

Visual servoing has been introduced as a promising solution for sensor-based robotic applications. The basic visual servoing task is to guide the motion of a robot with respect to a target object based on the feedback obtained through a vision system. Despite their popularity, Image-Based Visual Servoing (IBVS) schemes suffer from stability and convergence issues. Moreover, in IBVS techniques, there is no direct control over the image/camera/robot trajectories induced by the servoing loop in the image and physical spaces. Therefore, these trajectories might violate the image and/or physical constraints usually encountered in visual servoing tasks. Incorporating path planning strategies into the visual servo loop is a promising effort towards accounting for a variety of constraints.

In this thesis, we propose a general and global path planning framework for image-based control built on the efficiency and success of randomized sampling-based path planning techniques. The proposed planner explores the camera planning space for permissible camera trajectories satisfying image constraints (e.g., camera field of view and occlusions) and simultaneously tracks these trajectories in the robot configuration space to check for robot kinematic constraints and collision with obstacles. The exploration in camera planning space follows a tree-based randomized planning scheme and a local controller is used to track camera trajectories in the robot configuration space. The proposed framework yields global trajectories for the whole robotic system. The solution trajectory is projected into the image space to obtain the corresponding feature trajectories pertinent to a target object. An image-based visual servoing scheme is then adopted to execute the solution feature trajectories.

We implemented the proposed framework on a 6 degrees of freedom (DOF) robotic arm and a 9-DOF wheeled mobile manipulator. The effectiveness of the proposed planning scheme in accounting for a variety of image and physical constraints is shown through a number of real world experiments. We also provide an empirical study on the performance of the image-based trajectory tracking scheme under modeling and calibration uncertainties.

To my wife, the love of my life, Sara.

Acknowledgments

First, I would like to thank my senior supervisor, Prof. Kamal Gupta, for the liberty he gave me to develop my ideas and pursuit my interests while providing me with continuous support and encouragements throughout my Ph.D. I am grateful for his patience in helping me with difficulties and challenges I encountered while working on my thesis.

I would like to thank my supervisor, Prof. Mehran Mehrandezh, for his support of my ideas and his help with understanding various concepts in control, and his patience in educating me. I am also grateful for his help with improving my writing and presentation skills.

I would like also to thank my Ph.D. defense committee, Prof. Parvaneh Saeedi, Prof. Ahmad Rad, and Prof. Francois Chaumette for their positive and constructive comments on my thesis manuscript and during my defense. As a graduate student, I am grateful of Prof. Chaumette's contribution to the field of Visual Servoing. I owe my knowledge of Visual Servoing to his strong and rigorous publications in the area. I have been always fascinated by reading his works.

I would like also to thank my colleague and fellow students at the Robotic Algorithm and Motion Planning laboratory, Bruno L'Esperance, Zhenwang Yao, Liila Torabi, Yifeng Huang, Yi Li, and Vinay Pilania. I am grateful for their support and encouragements. I enjoyed the chats, coffee times, and parties we had, and certainly will miss them all.

I would like to thank the staff at the School of Engineering Science, who made the school days more enjoyable by providing a supportive and warm atmosphere. I am grateful for their help with my needs for travel support, research tools and equipments, and official stuff. I would like also to thank the Dean of Graduate Studies for their support in form of fellowships and scholarships during my Ph.D. program, without which I would have had a hard time supporting my family.

I would like also to thank my parents, sisters, and brothers for their remote support and encouragements. Their positive views of me and my ideas and their high hopes in my future

kept me encouraged and moving throughout my studies.

Last, but not the least, I would like to thank my wife, Sara, and my daughters, Kiana and Rosha. I cannot even begin to express how grateful I am for Sara's continuous support and encouragements during my studies. Her patience with many challenges we faced while being away from our families and relatives was very encouraging to me. I am certain that without her understanding and support I would not have reached the end of my Ph.D. program. She is the love of my life and I am grateful for having her.

Contents

List of Tables

List of Figures

List of Symbols

$\boldsymbol{\omega}$	Camera angular velocity
$\boldsymbol{\xi}^*(t)$	Desired (planned) camera trajectory (wheeled mobile manipulator)
Γ	Camera trajectory
$\mathbf{h}$	Camera orientation (unit quaternion)
$\mathbf{p}$	Camera position
$\mathbf{q}$	Robot joint configuration/trajectory
$\mathbf{q}^*(t)$	Desired robot trajectory
$\mathbf{q}_a^*(t)$	Desired (planned) trajectory of robotic arm (wheeled mobile manipulator)
$\mathbf{q}_d$	Robot desired configuration
$\mathbf{q}_i$	Robot initial configuration
$\mathbf{q}_p^*(t)$	Desired (planned) trajectory of mobile platform (wheeled mobile manipulator)
$\mathbf{r}$	Robot end-effector pose
$\mathbf{s}$	Vector of image features
$\mathbf{s}^*(t)$	Desired feature trajectories
$\mathbf{s}_d$	Vector of desired image features
$\mathbf{s}_i$	Vector of initial image features
$\mathbf{v}$	Camera linear velocity
$\mathbf{x}$	Camera planning vector

$\mathbf{x}^*(t)$	Desired camera/end-effector trajectory
$\mathbf{x}_d$	Desired camera planning vector
$\mathbf{x}_i$	Initial camera planning vector
$\mathbf{x}_{near}$	Camera planning vector with closest distance to randomly sampled node
$\mathbf{x}_{new}$	New camera planning vector generated as a result of tree extension
$\mathbf{x}_{rand}$	A randomly sampled camera planning vector
$\mathbf{L_s}$	Image interaction matrix
$\mathbf{s}(t)$	Image feature trajectory
$\mathcal{E}_q$	Robot local trajectory
$\mathcal{E}_x$	Camera local trajectory
$\mathcal{F}_c$	Camera coordinate frame
$\mathcal{F}_i$	Camera frame at desired location
$\mathcal{F}_i$	Camera frame at initial location
$\mathcal{F}_o$	Object coordinate frame
$\mathcal{P}_j$	j-th 3D target point
$\mathcal{T}$	Camera tree
A	Camera intrinsic matrix
f	Focal length of camera
p_u	Image resolution (pixels/meter) along image u-axis
p_v	Image resolution (pixels/meter) along image v-axis
p_{greedy}	Probability of greedy extension toward goal
u_0	u-axis coordinate of center of image along the image plane
u_j	Image pixel coordinates of j-th 3D point along image u-axis

v_0 v-axis coordinate of center of image along the image plane

v_j Image pixel coordinates of j-th 3D point along image v-axis

X_j x-axis coordinate of j-th 3D point in camera frame

x_j x-axis coordinate of j-th 3D point projected onto image plane

Y_j y-axis coordinate of j-th 3D point in camera frame

y_j y-axis coordinate of j-th 3D point projected onto image plane

Z_j Depth of j-th 3D point in camera frame

Chapter 1

Introduction

1.1 Visual Servoing: Basic Techniques and Issues

The role of vision as a sensor for autonomous machines to interact with complex, unknown, and dynamic environments is paramount. Visual servoing has been introduced as a promising approach for sensor-based robotic tasks such as positioning a robot with respect to a target and tracking a moving target via estimating its 3D motion, i.e. egomotion analysis using vision. The basic visual servoing task is to guide the motion of a robot with respect to a target object based on the feedback obtained through a vision system [40]. Figure 1.1 shows an example of a visual servoing task where the aim is to reach the desired view of the target features, i.e., black dots (Figure 1.1(b)) from the current location of the robot (Figure 1.1(a)).

Usually an error function $\mathbf{e}$ (also called *task function* [30]) is defined as

$$\mathbf{e}(t) = \mathbf{s}(\mathbf{r}(t)) - \mathbf{s}_d \tag{1.1}$$

where $\mathbf{s}$ and $\mathbf{s}_d$ denote the current and desired features, respectively, extracted from the image feedback, and $\mathbf{r}$ is the robot/camera location. The visual servoing objective is to regulate this error to zero. The term $\mathbf{s}(\mathbf{r}(t))$ shows the fact that the features change as the robot/camera location change with respect to the target object. In visual servoing this change is captured in form of *image interaction matrix*, $\mathbf{L}_\mathbf{s}$ i.e.

$$\mathbf{L}_\mathbf{s} = \frac{\partial \mathbf{s}}{\partial \mathbf{r}} \tag{1.2}$$

The existing visual servoing techniques are classified into different categories based on the definition of error function, the underlying control architecture, and the robot-camera

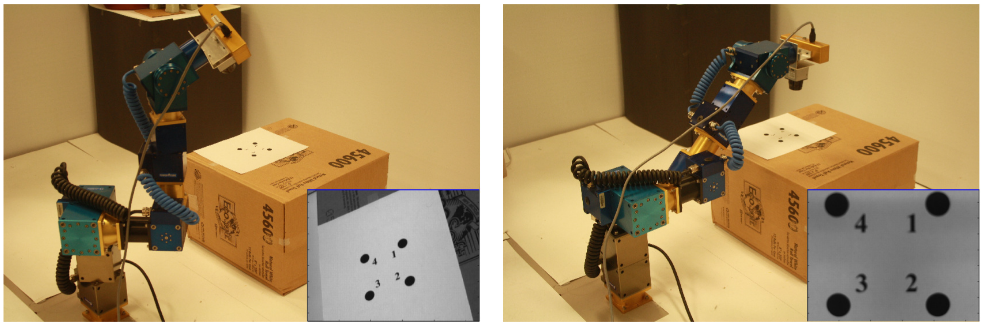


Figure 1.1: The visual servoing task is to guide the motion of a robot with respect to a target object based on the feedback obtained through a vision system. An error function is defined between the current and the desired features extracted from the images taken at the current (left) and the desired (right) robot/camera locations. The visual servoing task is to regulate this error to zero.

configuration (i.e., eye-in-hand vs. eye-to-hand configuration[1]). For a detailed review on existing techniques and their classification see [11], [12], and [40].

In contrast to position-based visual servoing (PBVS), where the control is performed in the task space based on the 3-D information retrieved from image, in IBVS techniques, the feedback is defined based on image features and the control loop is closed directly within the image. This results in a more robust control in presence of calibration and modeling errors [62], and hence adding to the popularity of IBVS methods.

In [10], through simple, yet, effective examples, Chaumette outlined the potential problems of stability and convergence of IBVS techniques: singularities in image Jacobian leading to an unstable behavior, and reaching local minima due to the existence of multiple camera poses yielding almost the same terminal image of the target. Moreover, in IBVS techniques there is no direct control over the image/camera/robot trajectories induced by the servoing loop in the image and physical spaces. Therefore, these trajectories might violate the image and/or physical constraints.

The image constraints normally encountered in practice include: field of view limits, i.e. the target may become invisible because it is outside of the camera's field of view; occlusion constraints, i.e., the target may be occluded due to obstacles, robot body, or self-occlusion; the physical constraints include: robot kinematics such as joint limits, singularities in robot Jacobian, robot/camera dynamic constraints, and collision with obstacles or self-collision.

[1] In an eye-in-hand configuration the camera is mounted on the end-effector of the robot and robot's motion results in camera's motion while in an eye-to-hand configuration, the camera is stationary and looks at the end-effector of the robot and robot's motion does not affect the camera pose [40].

These constraints are discussed with more details in Chapter 2 (also see [43]).

Since Chaumette's article [10] on the stability and convergence problems of IBVS techniques, research efforts in visual servoing have been devoted to incorporate the above image and physical constraints into the reactive visual servoing loop. Earlier works include *partitioned* and *switched* strategies. In contrast with partitioned strategies (e.g., [64, 33]), where certain degrees of freedom are controlled via IBVS while others are controlled via PBVS, switched strategies consist of a set of visual servo controllers along with a switching rule (e.g., [27, 32]). Partitioned techniques take advantage of both IBVS and PBVS techniques in avoiding some of the above constraints, while switched strategies enlarge the stability region of classical visual servoing techniques by switching between a set of unstable controllers to make the overall system stable.

Pertinent literature on partitioned and switched strategies indicated that only a small subset of the above mentioned constraints can be incorporated (see the review in [43]). Incorporating a global and general path planning strategy into the visual servo loop seems a promising effort towards addressing all the aforementioned constraints, especially in complex visual servoing scenarios.

1.1.1 Path Planning for Robust Visual Servoing

The main idea of path planning for visual servoing [69, 43] is to plan and generate *feasible* image feature trajectories while accounting for the constraints, and then to servo the robot along the planned trajectories. Overall, this results in a more robust servoing process with respect to violation of image and physical constraints. Here we briefly discuss the major works done in this area. The reader is referred to the comprehensive survey in Chapter 2 for more details.

Avoiding field of view limits and robustness to camera calibration and modeling errors motivated a number of techniques aimed at interpolating a path *directly* in the image space between the initial and desired images without using any knowledge of camera calibration or target model. Various results from projective geometry have been applied in this context including: epipolar geometry [38, 75], projective homography [70, 82, 51, 1, 8], and projective invariance [62]. The main advantage of these approaches is their insensitivity to camera calibration and/or object model errors. One of the difficulties in these techniques is that the planned path in the image may not correspond to a feasible camera motion. Moreover, since the planning is done directly in the image space, satisfying physical constraints (e.g., joint limits and obstacles) will be very challenging through such approaches lending themselves to be ineffective in complex visual servoing scenarios.

Potential fields (as in [46]) have been employed in the context of visual servoing in face of constraints (e.g., field of view and joint limits [69], or obstacle avoidance behavior [27]) by applying repulsive potentials in the image, joint, and/or task spaces. One of the main advantages of potential field-based approaches is that they can be employed in real-time applications. As an inherent deficiency of potential field-based path planning methods, the above strategies are prone to getting trapped in local minima. As a remedy, global navigation functions [55] could be employed instead. For example, a global stabilizing strategy using navigation functions is presented in [22], which guarantees convergence to a visible goal from almost every initial visible configuration while maintaining viability of all features along the way. However, one should note that constructing such navigation functions is limited to very simple scenarios only.

There is also a substantial body of literature aimed at finding globally optimal paths with respect to various costs (e.g., distance from the image boundary, length of the path traversed by the robot, energy expenditure, etc.): planning closed-form collineation paths corresponding to minimum energy and minimum acceleration camera paths [70], optimization over polynomial parametrization of the scaled camera paths [18, 16], convex optimization using Linear Matrix Inequality (LMI) test [14], or techniques based on optimal control theory such as the visual motion planning using Lagrange Multipliers [91], and geodesic techniques [7, 59, 37, 79]. Although the above techniques provide a better insight into the complexity of the problem in finding optimal paths, they are more or less limited to simple scenarios. Introducing global image/physical constraints greatly adds to the complexity of the optimization problem and hence, accounting for such constraints through the above frameworks is either impossible or highly detrimental to their time complexity.

The convergence problems of potential field-based techniques on one hand and the expensive cost of the above optimization-based techniques on the other hand motivates the need for general and, yet global, path planning approaches such as randomized sampling-based techniques [55, 56]. For example, in [3] a probabilistic roadmap approach has been utilized to plan minimal-occlusion paths for a camera with respect to a target object, which requires explicit computation of the boundary of visible and occluded regions. They also employed a dynamic collision checking strategy to check for the field of view limits along the edges of the roadmap.

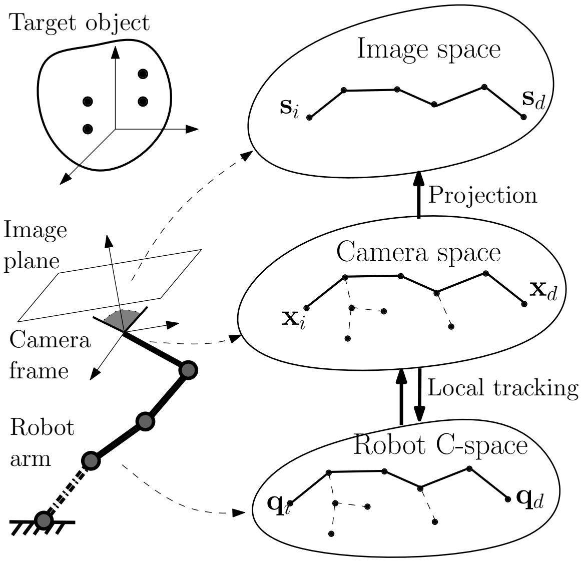


Figure 1.2: Our proposed planning framework alternatively explores the camera planning space and the robot configuration space (C-space) for feasible robot/camera paths to obtain feasible feature trajectories in image space.

1.2 Randomized Path Planning for Visual Servoing

We incorporated sampling-based global path planning with visual servoing for a robotic arm equipped with an in-hand camera [41]. The proposed planner explores the camera task space (i.e., space of camera poses) for camera paths satisfying field of view limits and occlusion constraints, and utilizes a local planner to track these paths in the robot's joint space to ensure feasible motions of the robot while accounting for robot's joint limits and collision with obstacles. The result is a search tree as in [50], which alternatively explores the camera and joint spaces (see Fig.1.2). The solution camera trajectories are then projected into the image space to obtain the corresponding image feature trajectories, which require proper time-scaling and smoothing to be executed using an image-based visual servoing technique.

We extended the work in [41] and proposed a kinodynamic planning approach [42], which performs kinodynamic planning for the camera in its state space while taking its dynamic constraints into account. Unlike the kinematic path planning approach in [41], which requires an additional (cubic) spline interpolation step to properly time scale the feature trajectories and obtain smooth trajectories (e.g., see [69]), the solution trajectories obtained using the proposed kinodynamic planning framework are (by construction) C^1-smooth with bounded acceleration and require no further post-processing to be executed smoothly using an IBVS technique (see Figure 1.3).

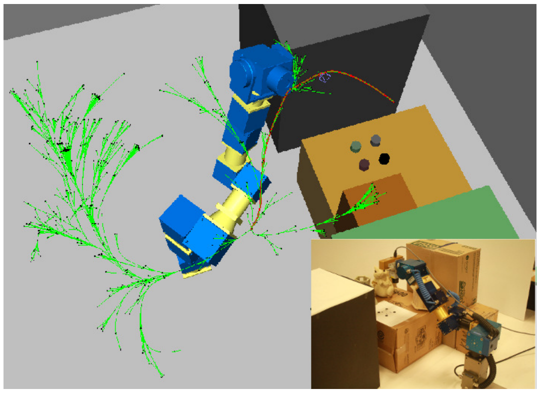


Figure 1.3: Our proposed randomized kinodynamic planner extends a search tree in camera state space.

1.3 Contributions

The main contribution of this thesis is to incorporate randomized path planning techniques with image-based control of robotic arms and wheeled mobile manipulators. To the best of our knowledge, this is the first attempt on incorporating randomized path planning approaches with image-based control. In contrast to previous works, the proposed planning framework provides a general and efficient planning strategy, which incorporates a variety of image and physical constraints encountered in visual servoing tasks.

The contributions of the thesis are detailed as follows:

- We carried out a comprehensive literature survey on existing path planning techniques for visual servoing. The survey covers both basic and state of the art strategies in the area, explains how each strategy works, and also discusses its benefits and drawbacks in comparison with other techniques. This survey has been published as a book chapter in [43] providing a unique and comprehensive background on path planning for visual servoing.

- Inspired by the work of Yao and Gupta on path planning with general end-effector constraints [90], we devised and implemented a kinematic planning strategy through which feasible image feature trajectories are generated by alternate exploration of camera workspace (i.e., space of camera poses) and robot configuration space. The feature trajectories correspond to feasible camera-robot trajectories that satisfy image and physical constraints imposed by the visual servoing tasks. The (kinematic) solution trajectories are non-smooth and require proper time-scaling/smoothing so they can be executed using an image-based visual servoing strategy. This work has been published in [41].

- We extended the above kinematic planning approach by accounting for camera dynamics via a randomized kinodynamic planning strategy in camera state space, i.e. space of camera poses and velocities [42]. The solution trajectories obtained using the proposed kinodynamic planning approach are (by construction) C^1-smooth trajectories with bounded acceleration and require no time-scaling/smoothing to be executed. This thesis presents the development and implementation of the proposed randomized kinodynamic planning strategy on a 6-DOF robotic arm with an in-hand camera. The robotic arm is servo controlled along the planned feature trajectories using an image-based visual servoing technique. We have performed a number of experiments on a real 6-DOF robotic arm to show the effectiveness of the proposed strategy in accounting for image and physical constraints to accomplish visual servoing tasks in complex environments. We also carried out an empirical study on the effect of modeling/calibration uncertainties on the performance of the image-based trajectory tracking controller in contrast with a joint space trajectory tracking strategy. This work has been published in [45].

- We also extended the proposed planning framework to wheeled mobile manipulators. Although the extension of the proposed planning framework to wheeled mobile manipulators is straightforward and (almost) follows the same strategy developed for robotic arms, we adapted a decoupled control strategy to execute the solution trajectories: the on-board robotic arm is servo controlled along the planned feature trajectories using an image-based visual servoing technique while the mobile platform is controlled along its planned trajectory in the workspace. To the best of our knowledge, this is considered the first attempt to integrate the planning and image-based control for wheeled mobile manipulators. We performed preliminary experiments by executing planned trajectories on a 9-DOF wheeled mobile manipulator to accomplish a visual servoing task. This work has been published in [44].

1.4 Outline of Thesis

This thesis is structured as follows:

- **Chapter 2:** provides a comprehensive literature survey on existing path planning techniques for visual servoing. It enumerates the image and physical constraints encountered in visual servoing tasks and explains their challenges. It categorizes the existing path planning techniques into four groups: (1) direct path planning in image space, (2) potential field-based strategies, (3) global parameterization and optimization techniques, and (4) global path planning. This survey has been published as a book chapter in [43].

- **Chapter 3:** explains the details of our proposed planning framework. It talks about the solution methodology for incorporating path planning with image-based control. We discuss the proposed randomized kinematic planning strategy and its extended kinodynamic version. We also explain how each of the image and physical constraints is accounted for through the proposed planning framework. We implemented the proposed planning framework on a 6-DOF robotic arm (Chapter 4) as well as a 9-DOF wheeled mobile manipulator (Chapter 5), a non-holonomic mobile platform with an on-board robotic arm.

- **Chapter 4:** provides and discusses the implementation of the proposed kinodynamic planning framework on a 6-DOF robotic arm with an in-hand camera. We explain the implementation of the image-based visual servoing technique to track the planned feature trajectories along with our system/software architecture. This chapter also presents the results of our extensive experiments on our 6-DOF robotic arm to accomplish a number of visual servoing tasks in complex environments. The effectiveness of the proposed planning framework in accounting for image and physical constraints is demonstrated through the results presented in this chapter. Moreover, we also discuss the effect of modeling/calibration uncertainties on the performance of the proposed framework in tracking the planned trajectories through the results of a number of simulated experiments presented in this chapter.

- **Chapter 5:** extends the implementation of the proposed framework to the case of a 9-DOF wheeled mobile manipulator. We explain the extension of the planning framework by introducing a null-space optimization technique combined with a weighted pseudo-inverse inverse kinematic control to benefit from the redundancy of such systems towards increasing the manipulability and achieving coordination along the solution trajectories. We also explain the details of a decoupled control strategy that was adapted and implemented on our 9-DOF wheeled mobile manipulator to execute the solution trajectories.

- **Chapter 6:** concludes the thesis and discusses future directions to this line of research.

Chapter 2

Literature Survey

This chapter provides a comprehensive technical review of existing major approaches to path planning for visual servoing. We provide an overview of each planning approach, explain the associated set of constraints and assumptions, and discuss the underlying path planning techniques along with the issues regarding their integration with reactive visual servo controllers.

2.1 Introduction

The existing visual servoing techniques are classified into different categories based on the definition of error function, the underlying control architecture, and the robot-camera configuration (i.e., eye-in-hand vs. eye-to-hand configuration). For a detailed review on existing techniques and their classification see [11], [12], and [40]. In summary, the existing approaches can be classified into two main categories: (1) Position-Based Visual Servoing (PBVS) where the feedback is defined in terms of the 3D Cartesian information derived from the image(s), and (2) Image-Based Visual Servoing (IBVS) where the feedback is defined directly in the image in terms of image features.

IBVS techniques have better local stability and convergence in presence of camera calibration and modeling errors. However, they suffer from global convergence problems, and, hence, will break down, in particular when the initial and desired camera poses are distant [10]. For example some of the image features might leave the camera's field of view and consequently result in failure of the servoing task. Moreover, there is no direct control on the robot/camera motion induced by the image-based control law. This might result in infeasible maneuvers due to the robot's joint limits and/or collision with workspace obstacles.

Amalgamation of path planning techniques with reactive image-based visual servoing strategies can robustify existing image based tracking systems by accounting for critical *constraints* and *uncertainties* in robotics applications where a high disparity between the initial and desired views of a target is inevitable (e.g., target interception, space docking, reaching and grasping, etc.). The main idea of path planning for visual servoing is to plan and generate *feasible* image trajectories while accounting for certain constraints, and then to servo the robot along the planned trajectories.

In this survey we provide a comprehensive technical review on existing and recent approaches to path planning for visual servoing. For each approach the set of constraints and the assumptions are explained and the underlying path planning technique is discussed along with the issues regarding its integration with the reactive image-based controllers.

In Section 2.2 we study the two sets of critical constraints in visual servoing context: (1) image/camera, and (2) robot/physical constraints. The existence of such constraints motivates the need for path planning techniques aimed at making the servoing process more robust especially in complex visual servoing scenarios. In Section 2.3 a comprehensive overview of these approaches and their categorization based on the underlying path planning techniques are provided. In Section 2.4 we discuss the effect of uncertainties on visual servoing and report on some recent works aimed at path planning under uncertainty for visual servoing. Finally, we conclude the survey in Section 2.5.

2.2 Constraints in Visual Servoing

In [10], through simple, yet, effective examples, Chaumette outlined the potential problems of stability and convergence in both IBVS and PBVS techniques imposed by a number of constraints. Overall one can divide these constraints into two main categories: (1) Image/Camera, and (2) Robot/Physical constraints. These two categories are detailed as follows.

2.2.1 Image/Camera Constraints

The image/camera constraints are mainly due to the sensing limits of the vision system or the inter-relationship between the optical flow (i.e., rate of change) of the features $\dot{\mathbf{s}}$ in the image space and the camera's Cartesian velocity $\dot{\mathbf{r}}$ defined through the *image Jacobian* (also called *interaction* matrix) $\mathbf{L_s}$ related to image features [40]:

$$\mathbf{L_s} = \frac{\partial \mathbf{s}}{\partial \mathbf{r}} \tag{2.1}$$

These constraints are: (1) *field of view limits*, (2) *image local minima*, and (3) *singularities in image Jacobian.*

Field of View Limits The camera as a sensing system have certain limitations. For example the 3D target features projected into the image plane of the camera are visible if their projections fall inside the boundary of the image. The limits of the image are usually represented by a rectangular region which determines the visible region of the image plane. Although in IBVS context the control is directly defined in the image, there is still the possibility that the features leave the camera's field of view, in particular when the initial and desired poses of the camera are distant [10].

Image Local Minima As shown in [10], in IBVS context, image local minima might occur due to the existence of unrealizable image motions which do not belong to the range space of image Jacobian $\mathbf{L_s}$. Hence, there does not exist any camera motion able to produce such unrealizable motions in the image. In general, determining the image local minima is difficult without considering the specific target location and the initial and desired relative camera-target locations, which in turn, leads to an exhaustive search for local minima in the image for each instance of a visual servoing task. As demonstrated in [10], using a nominal value of image Jacobian estimated at the desired location might be of help to avoid local minima in visual servoing tasks. But this may lead to peculiar trajectories of features in the image, which in turn, might violate field of view limits. One should note that the PBVS techniques are known to be free of image local minima since the task function is defined in the Cartesian space.

Singularities in Image Jacobian At these singularities, certain camera motions cannot be achieved by the motion of image features in the image space. Several cases of image singularities have been considered in [10]: the image Jacobian $\mathbf{L_s}$ is known to be singular if the vector of image features $\mathbf{s}$ consists of the image of (1) three collinear points, or (2) three points belonging to a cylinder containing the camera optical center. Although using more than three non-coplanar points will avoid such singularities, the image Jacobian may still become singular no matter how many feature points (irrespective of their arrangements) are used to define the task function. For example, a visual servoing task involving a 180 degrees rotation around the optical axis results in a singular image Jacobian. As shown in [10] using line features instead of points helps to avoid such singularities, however, it does not completely eliminate the singularities in the image space. *Motion Perceptibility* [85] has been proposed as a measure of closeness to image singularities.

2.2.2 Robot/Physical Constraints

Motion of the robot/camera system induced by the visual servo control loop, especially in IBVS, may also violate certain constraints imposed by the robot and/or physical obstacles in the workspace. These are: (1) *robot kinematics* such as joint limits and singularities in robot Jacobian, (2) *robot dynamics*, (3) *collision* with obstacles or self-collision, and (4) *occlusion* due to obstacles, robot body, or self-occlusion by the target.

Over the past three decades a great deal of research in robotics community has been devoted to planning feasible paths avoiding robot kinematics and/or dynamics constraints and collision with physical obstacles or self-collision in various environments (see e.g. [55] and [56]). Path planning approaches have also considered occlusion constraints in applications that require target visibility, e.g. [57] and [71].

Since Chaumette's article [10] on the convergence and stability problems of classical visual servoing techniques, most of the efforts in visual servoing community have been devoted to taking the above image/camera and/or robot/physical constraints into account and incorporating them into the reactive visual servoing control loop.

First a number of researchers proposed *partitioned* (or decoupled) control schemes in which certain degrees of freedom are controlled in the manner of IBVS while others are controlled in the manner of PBVS, thereby taking advantage of each individual technique's benefit in avoiding some of the above constraints (see e.g. [21], [25], [61], [64], [72]). Each of these partitioned approaches has its own benefits and drawbacks in accounting for the aforementioned constraints. A performance test has been presented in [33] comparing the efficiency of some of the above partitioned strategies with IBVS technique.

Later on, inspired by the theory of hybrid systems, a number of researchers proposed *hybrid* (or switched) strategies consisting a set of visual servo controllers along with a switching rule to switch between them if required (see e.g., [15], [27], [32], [36]). Using switched strategies it might be possible to enlarge the stability region of classical visual servoing techniques and to switch between a set of unstable controllers to make the overall system stable.

Each of the above partitioned or hybrid strategies deals with only a subset of the above mentioned constraints. Incorporating all the image/camera and robot/physical constraints into the visual servo control loop is, if at all practical, quite challenging. Clearly some sort of path planning on top of the visual servo control loop is needed for incorporating all the aforementioned constraints, especially in complex visual servoing scenarios.

2.3 Path Planning for Visual Servoing

The main idea of path planning for visual servoing [69] is to plan and generate *feasible* image trajectories while accounting for the constraints mentioned in the previous section, and then to servo the robot along the planned trajectories. So, the initial large error is discretized and the error to regulate at each cycle of the servoing loop remains small. Overall, this results in a more robust servoing process with respect to the aforementioned image/camera and robot/physical constraints.

Over the past decade research has been devoted to incorporate aspects of path planning in visual servoing. Although there is no formal classification of the existing path planning techniques for visual servoing, considering the underlying path planning approach and the assumptions made in each technique, we divide them into four groups: (1) Image space path planning, (2) Optimization-based path planning, (3) Potential Field-based path planning, and (4) Global path planning. In this survey we discuss the major works done in each group to describe the main idea and the underlying problems.

2.3.1 Image Space Path Planning

Image space path planning techniques aim at interpolating a path in the image space between the initial and desired images without using any knowledge of camera calibration or target model. One of the difficulty of such approaches is that the planned image space path may not correspond to any single path for the camera. So, efforts have been devoted to planning image paths which correspond to feasible (yet unknown) camera paths in an uncalibrated domain. Various results from projective geometry have been applied in this context including: epipolar geometry, projective homography, and projective invariance.

Epipolar Geometry Given multiple views of the same scene, epipolar geometry [35] has been employed by a number of researchers for calibration-free visual servoing. In an early work [38], a trajectory generator for visual servoing was proposed directly in the image space based on epipolar constraints defined between the images obtained from a stereo camera mounted on a robotic arm (eye-to-hand configuration). The task was to accomplish obstacle avoidance (only for robot's end-effector) in an unknown environment. An uncalibrated visual servo controller based on a Jacobian estimator was used to track the planned image trajectories without using any knowledge of the system or camera calibration.

Park and Chung proposed an image space path planning approach for an eye-to-hand system using uncalibrated stereo cameras in a vision-based grasping scenario [75]. They generate a number of intermediate views of the robot's gripper along a straight line between the initial image and the final desired image in the projective space with the help of epipolar geometry and without using any 3D information regarding either the gripper or the target object. These intermediate views constitute the desired image trajectories. The robot is then controlled along the image trajectories using the IBVS technique presented in [30]. When followed by the robot, the planned trajectories allow the robot's gripper to track a straight line in the 3D workspace and through out its motion a selected set of features on the gripper are kept in the camera's field of view.

Projective Homography To avoid explicit computation of feasible camera paths which relies on the knowledge of the camera calibration and target model, a number of approaches have been developed using the projective geometry [31] relationship established between the initial and desired images. Working in projective space allows one to partially parameterize the Euclidean displacement of the camera without explicit reconstruction of the Euclidean components.

Projective homography matrix has been employed in the context of path planning for visual servoing. Projective homography captures the relationship between the images taken from different views of the same scene. Given the projective homogeneous coordinates $\mathbf{p} = (u, v, 1)^T$ and $\mathbf{p}^* = (u^*, v^*, 1)^T$ of a 3D point $\mathbf{P}$ in the current and desired images, respectively, the projective homography matrix $\mathbf{G}$, also called *collineation* matrix, is defined (up to an scale α_g) as

$$\alpha_g \mathbf{p} = \mathbf{G}\mathbf{p}^* \tag{2.2}$$

The projective homography matrix can be estimated from the knowledge of several features such as points, lines, and contours matched between two images [17], [35], and [63].

In [70] a calibration-free path planning approach is proposed which consists of interpolating for the *collineation* matrix $\mathbf{G}$ between the initial and desired images to obtain closed-form analytical collineation paths. The image feature trajectories are then derived and followed using an IBVS technique. The proposed approach guarantees convergence to the desired location, however, the convergence does not hold in presence of visibility constraints such as field of view limits. This approach has been extended in [82] to take visibility constraints into account by guiding the image of an arbitrary selected reference point on the target along a straight line in the image which guarantees that the reference point remains in the camera's field of view. However, the camera will not follow a straight line anymore and the other features may still leave camera's field of view. A depth modulation approach has been proposed to keep the visibility of other features by controlling the camera backwards along an optical ray whenever a feature reaches the borders of camera's field of view.

If the camera calibration is known, one could derive further information regarding the camera transformation. In particular, one can compute the Euclidean homography matrix $\mathbf{H}$ (up to a scale α_h) as

$$\alpha_h \mathbf{H} = \mathbf{K}^{-1} \mathbf{G} \mathbf{K} \tag{2.3}$$

where $\mathbf{K}$ is a non-singular matrix and contains the camera intrinsic parameters. The Euclidean homography (from a set of planar features) can be decomposed to obtain the corresponding (Euclidean) camera transformation parameters as

$$\mathbf{H} \Rightarrow \left\{ \mathbf{R}, \frac{\mathbf{t}}{d^*}, \mathbf{n}^* \right\} \tag{2.4}$$

where $\mathbf{R}$ and $\mathbf{t}$ denote the translation and rotation from the desired to the current camera frame, and d^* is the distance of the plane containing the features from the desired camera frame and $\mathbf{n}^*$ is the normal to the plane expressed with respect to the desired frame.

Decomposition of Euclidean homography has been employed by some researchers to plan for image paths corresponding to feasible (yet unknown) camera paths without explicit reconstruction of the camera paths in the Cartesian space.

A shortest path approach has been proposed in [51] which avoids the use of 3D reconstruction by using homography-based partial pose estimation. The proposed approach moves the in-hand camera directly along the direction (obtained through the homography decomposition) towards the desired pose in the 3D workspace while maintaining the visibility of (only) a virtual point located at the origin of the target object. The virtual point is used to control two degrees of rotation of the camera (around $x-$ and $y-$axes) and the third rotation axis (around camera optical axis) is controlled using the rotation matrix retrieved from homography. This technique yields a straight line trajectory for the virtual point and, hence, keeps the virtual point always in the camera's field of view. However, the camera can get too close to the target so that some features may get lost. Switching between visual servoing strategies or using repulsive potentials can be employed to avoid such situations, however, without ensuring straight line trajectories.

In [1] a similar approach has been proposed based on homography decomposition in which helicoidal shape paths (instead of straight path) are chosen as the reference path to represent camera translation from the initial position to the desired position. One should note that since the homography is known only up to an *unknown* scale, the actual camera path is not completely known and one can only determine its shape. However, regardless of the value of unknown scale factor, the entire image path will remain the same and since the control is defined directly in the image, the positioning task can be successfully accomplished given a feasible image path. In [8] a particular decomposition of homography is used to interpolate a path for a planar object with known model from the initial image to the desired final image. Given the known object model, the interpolated desired path is then transformed to a camera path by using 3D reconstruction. The camera path can then be checked for workspace boundary singularities.

Projective Invariance Malis [62] proposed an image-based path planning approach in an invariant space defined through a projective transformation. The basic idea of using projective invariance is to create a task function which is invariant to camera intrinsic parameters and only depends on the position of the camera with respect to the observed object and on its 3D structure. This allows one to generate a path for a feature vector in the invariant space (independent of camera's intrinsic parameters) which, when followed, results in a straight line path for the camera in the workspace. The visibility of the features is (partially) achieved using a motorized zooming mechanism available on the vision system.

The main advantage of direct path planning in image space is the independence of such approaches from camera calibration and/or object model. On the other hand, since the planning is done directly in the image space, robot/physical constraints cannot be handled through such approaches and these techniques are shown to be ineffective in complex visual servoing scenarios.

2.3.2 Optimization-based Path Planning

Planning optimal paths has absorbed a great amount of interest in robotics community. In a visual servoing task, there might be many different paths, which when followed, will result in successful accomplishment of the same task. This motivates optimization techniques aimed at finding the optimal path with respect to various costs such as distance from the image boundary, length of the path traversed by the robot, energy expenditure, etc.

In an early work [86], a path planning framework is proposed based on the concept of Perceptual Control Manifold (PCM) defined on the product of the robot's joint space and the space of all image features related to a target object. PCM can be considered as a mapping which relates a robot configuration to the vector of image features visible at that configuration. Given the model of the camera, the object, and the robot kinematic model, the PCM needs to be computed only once (in an eye-to-hand configuration) and is then applicable to any manipulation task. Constraints such as the camera's field of view and the robot joint limits and/or physical obstacles are mapped into the PCM to yield a subset of PCM as the feasible solution space. This mapping could be quite time consuming considering the number of constraints and the robot's degrees of freedom. Various optimization criteria such as minimum velocity, minimum interception time, and minimum robot movement have been considered to plan optimal paths in the feasible subset of the PCM. The proposed approach has been considered for the task of intercepting a moving target (with a known trajectory) using the visual feedbacks obtained from a fixed camera which simultaneously views both the robot's end-effector and the moving target.

In [70] closed-form collineation paths corresponding to minimum energy and minimum acceleration camera paths are planned in the image space. The proposed strategy is then generalized to the case where a number of relay (intermediate) images are available in addition of the initial and desired images. The proposed approach guarantees convergence, however, it does not take visibility constraints into account and image features might leave the camera's field of view.

In [91] a motion generation approach called visual motion planning has been proposed to plan optimal image paths for mobile robots under motion and visibility constraints. The constraints on the motion of the robot along with the field of view limits are described in form of a number of equalities and inequalities. An optimization problem is then solved numerically using Lagrange Multipliers to obtain optimal image paths minimizing a given weighted sum cost function (here kinetic energy). The proposed approach has been applied only to mobile robots moving in 2D and 3D environments.

To pose the problem of path planning for visual servoing as an optimization problem some researchers have introduced various parameterizations of camera trajectories. A polynomial parametrization of the scaled camera paths has been proposed in [18] where the translational path is linearly interpolated and Cayley's rotation representation is employed to rationally parameterize the rotation paths. This allows the distance of the image trajectories from the boundary of image for a single path to be easily calculated as the root of some polynomials. Hence, an optimization problem is then formulated to maximize the distance to the boundary of the image with respect to all parameterized paths. By following the planned image path, the camera follows a straight line in the workspace in the absence of calibration errors. In presence of calibration errors, the camera does not follow a straight line but moves along a different curve whose distance from the planned line grows as the calibration errors increase.

In [16] an optimal path planning approach is proposed which allows one to consider constraints on the camera's field of view, workspace and joint limits, in the form of inequalities, together with the objective of minimizing trajectory costs including spanned image area, trajectory length, and curvature. A polynomial parametrization is devised to represent all the camera paths connecting the initial and desired locations (up to a scale factor) through an object reconstruction from image measurements and, if available, the target model. Occlusion constraints and collision avoidance for the whole robot's body cannot be represented (in the form of inequality constraints) in their formulation. Moreover, the devised optimization is nonconvex which may lead to multiple feasible regions and multiple locally optimal solutions within each region and, hence, it makes it very difficult to find the global optimal solution across all feasible regions.

In a similar work [14], a general parameterizations of trajectories from the initial to the desired location is proposed via homogeneous forms and a parameter-dependent version of the Rodrigues formula. The constraints are modeled using positivity conditions on suitable homogeneous forms. The solution trajectory is obtained by solving a Linear Matrix Inequality (LMI) test which is a convex optimization. The proposed approach allows one to maximize some desired performances such as distance of features from the boundary of the image, camera's distance from obstacles, and similarity between the planned trajectory and a straight line.

Ideas from optimal control theory have been employed to devise image trajectories for visual servoing under visibility constraints. Planning shortest path for a Differential Drive Robot (DDR) maintaining the visibility of a landmark using a camera with limited field of view has been considered in [7]. It is shown that the set of shortest (optimal) paths for this system consist of curve segments that are either straight-line segments or that saturate the camera's field of view. The latter correspond to exponential spirals known as T-curves. In [59] these shortest paths are followed using a switched homography-based visual servo controller. The controls that move the robot along these paths are devised based on the convergence of the elements of the homography matrix relating the current image to the final desired image. In a recent work [37], a complete motion planner for a DDR is proposed in which optimal curve segments obtained in [7] are used as motion primitives to devise locally optimal paths in an environment cluttered with obstacles. The necessary and sufficient conditions for the feasibility of a path for the DDR in the presence of obstacles and with visibility constraints (i.e., sensing range and field of view limits) are also provided. In their proposed planner, occlusions due to workspace obstacles are not considered and the obstacles are assumed to be transparent.

In [78] the set of optimal curves obtained in [7] are extended and also described in the image space, so as to enable their execution using an IBVS controller directly in the image space. Feedback control along these optimal paths in the image is achieved through a set of Lyapunov controllers, each of which is in charge of a specific kind of maneuver. Nonetheless, the complete characterization of all the shortest paths and their analytic descriptions remain unsolved for a DDR.

Although the above optimization-based path planning techniques provide a better insight into the complexity of the problem and feasible optimal paths, they are more or less limited to simple scenarios and systems. Introducing general robot/physical constraints greatly adds to the complexity of the optimization problem and, hence, accounting for such constraints in the above frameworks would greatly increase the time complexity of such techniques.

2.3.3 Potential Field-based Path Planning

In the field of robot path planning, Potential Field method has been proposed as a promising local and fast obstacle avoidance strategy to plan safe and real-time motions for a robot in a constrained environment [46]. The main idea is to construct an artificial potential field defined as the sum of attractive potentials, pulling the robot towards the desired location, and repulsive potentials, pushing the robot away from various constraints such as the obstacles or robot's joint limits. A driving force computed along the negated gradient of the potential field moves the robot towards the goal location.

Mezouar and Chaumette [69] introduced robust image-based control based on the Potential Field method for a robotic arm with eye-in-hand configuration. In their proposed approach, two types of constraints are considered: field of view and robot's joint limits. To obtain valid robot trajectories, the motion of the robot is first planned in the workspace and then projected into the image space. The attractive potentials are defined in the workspace to pull the robot towards the final desired configuration. To account for field of view limits, repulsive potentials are defined in the image space pushing the image trajectories away from the image boundary. Joint limits are avoided by imposing repulsive potentials in the joint space of the robot. So, the total force applied to the robot is a weighted sum of the individual forces computed as the negated gradient of the above potentials. The image trajectories are obtained in an iterative scheme by moving along the direction of the total force applied to the robot. The discrete image trajectories are then time scaled and tracked using an IBVS technique. The above strategy has been applied to targets with known as well as unknown models. In the latter case, a scaled Euclidean reconstruction is employed to obtain scaled camera paths in the workspace. Image local minima are automatically avoided by updating the image Jacobian using the values of the current desired image features along the time scaled feature trajectories.

As an inherent deficiency of Potential Field-based path planning method, the above strategy might lead to local minima and the robot gets stuck. Although the authors reported no encounter of such local minima in their experiments, imposing physical constraints such as collisions with obstacles and occlusions highly increase the chance of having local minima in the overall potential field.

In [27] a potential field-based strategy is employed to account for workspace obstacles, field of view limits, and robot's joint limits in a global planning framework. To escape local minima generated by addition of the attractive and repulsive forces, *Simulated Annealing* [47] is employed in which proper tuning of the initial temperature and the cooling rates are required to probabilistically ensure the method to escape from local minima and converge to the global minimum. In the proposed planning framework two different trajectory generation strategies are employed: method A, in which a trajectory for the end-effector is planned with respect to the stationary target frame, and method B, in which a trajectory for the target is planned with respect to the current end-effector frame. The former results in a camera path close to a straight line in the workspace, while in the latter the image trajectory of the target's origin is constrained to move as close as possible to a straight line in the image which lessens the chance of image features leaving the camera's field of view. A local switching strategy is devised to switch from image-based control to position-based control when closeness to image local minima and image singularities are detected along the planned trajectories. This is done only once to avoid instability caused due to repetitive switching, however there is no complete guarantee that the field of view and joint limits are always ensured after the system is switched to position-based control.

One of the main advantage of Potential Field-based approaches is the fast computation of driving force which makes these approaches suitable for real-time applications such as visual servoing. For example, the above strategy can be employed when tracking image trajectories to account for possible deviations from the planned trajectory due to uncertainties in modeling and/or calibration (e.g. [16]).

2.3.4 Global Path Planning

The convergence problems and deficiencies of the above path planning techniques in accounting for all the constraints in visual servoing tasks motivates the need for general and global path planning approaches. A great deal of research has been carried out on global path planning for various robotic systems within the path planning community, see e.g. [55] and [56]. Here we report on some of these techniques which have been successfully incorporated into the visual servoing framework.

A global stabilizing strategy using navigation functions is presented in [22] which guarantees convergence to a visible goal from almost every initial visible configuration while maintaining viability of all features along the way without following a predefined reference image trajectory. One should note that constructing such navigation functions is limited to very simple scenarios only.

In [4] a probabilistic roadmap approach has been utilized to plan minimal-occlusion paths for an in-hand camera with respect to a target object. They employ the technique proposed in [88] to compute the boundary separating the visible regions (from where the target is visible) from the occluded regions (from where the target is not visible due to occlusion by workspace obstacles). Their proposed algorithm then assigns penalties to camera's trajectories within a given probabilistic roadmap (for camera translation) proportional to the distance the camera travels while outside the visible region. One should note that camera's orientation or field of view limits are not taken into account in their proposed approach.

Inspired by the work in [90] on global path planning with general end-effector constraints, we incorporated sampling-based global path planning with visual servoing for a robotic arm equipped with an in-hand camera [41]. The proposed planner explores the camera task space (i.e., space of camera poses) for camera paths satisfying field of view limits and occlusion constraints, and utilizes a local planner to track these paths in the robot's joint space to ensure feasible motions of the robot while accounting for robot's joint limits and collision with obstacles. The result is a search tree as in [50] which alternatively explores the camera and joint spaces (see Fig.1.2). The solution camera trajectories are then projected into the image space to obtained the corresponding image feature trajectories which require proper time-scaling and smoothing to be executed using an image-based visual servoing technique.

Proper time scaling of feature trajectories directly depends on the dynamic characteristics of the underlying system and highly affects the performance at the execution stage. Usually, the planning and execution stages are decoupled by first, solving the path planning problem and then either synthesizing a controller with the desired dynamic characteristics of the underlying system to execute the planned trajectories (e.g., [73]), or performing optimization over the kinematically planned paths to satisfy the dynamic constraints (e.g., [53, 6]). However, the robot may not be able to execute the trajectories produced by the kinematic planner due to its limits on actuator forces/torques and/or dynamic effects of machine vision system (see [20]). We extended the work in [41] and proposed a kinodynamic planning approach [42] which attempts to address this issue at the planning stage by performing kinodynamic planning for the camera in its state space while taking its dynamic constraints into account. Unlike the kinematic path planning approach in [41] which requires an additional (cubic) spline interpolation step to properly time scale the feature trajectories and obtain smooth trajectories (e.g., see [69]), the solution trajectories obtained using the proposed kinodynamic planning framework are (by construction) C^1-smooth with bounded acceleration and require no further post-processing to be executed smoothly using an IBVS technique preventing a stop-and-go motion, an effect which is inevitable in kinematic planning as observed in our previous work [41].

To be successful, global path planning approaches require a complete and (relatively) precise knowledge of the environment, camera calibration, and object model. These requirements can be limiting in many visual servoing scenarios. The need for such exact knowledge can be relaxed by accounting for modeling and calibration uncertainties at planning stage. In the following section we discuss the effects of uncertainties in visual servoing, especially in tracking planned trajectories at the execution time, and report on a few recent works on path planning under uncertainties for visual servoing.

2.4 Path Planning under Uncertainty for Visual Servoing

Planned paths need to be executed and the robot may not exactly follow the planned path due to the uncertainties and in fact, in some cases, the followed paths in the workspace and in the image space can be quite different from the planned ones thereby resulting in violation of some of the constraints even if they have been fulfilled at the planning stage. The influence of errors in intrinsic and extrinsic camera parameters on the performance of visual servoing scheme has been examined in [29]. In [52] the propagation of image error through pose estimation and visual servoing control law has been analyzed.

A number of researchers have proposed local and real-time techniques to account for likely deviations from the planned path at the execution (tracking) stage. For example, these could be locally taken care of using Potential Field type techniques, however as mentioned before, such techniques are prone to failure due to local minima [69]. An alternative framework to deal with unmodelled uncertainties is to retreat the robot/camera and/or re-plan quickly when encountering violation in a constraint. A variable zooming technique was suggested by [54] to bring the target back within the visibility range if occluded by an obstacle. This zooming effect can also drastically improve the performance of the underlying image-based visual servoing technique by reducing the measurement noise in fixed-size objects viewed by a camera from distance.

Comport et al. [19] proposed an augmented reality approach for visual servoing. Although their approach mainly focuses on camera pose estimation by means of a virtual visual servoing method, but this can be extended to scenarios in which some feature points on the target may go out of sight temporarily, e.g. due to unmodelled uncertainties. An augmented reality approach can then be utilized to virtually position the missing feature points in the image based on rudimentary information obtained from other objects in a scene cluttered with known features, i.e. straight edges, etc. In this case, the target is used as the primary object for visual servoing while other image features can contribute to the pose estimation, and eventually to the servoing task, when a finite number of feature points fall off the cameras field of view.

Non-linear model predictive control strategies have been proposed to account for uncertainties in planned trajectories in visual servo control loop as well, e.g. [81]. Systems' parameters would be corrected beyond a temporal receding horizon (i.e., the time span during which the optimal control action is computed and executed) after each iteration. The discrepancy between the predicted system's behavior based on the computed control action and that in real implementation is then used to further correct the estimates of the system's parameters. The time required to estimate these parameters via a non-linear optimization technique must be way shorter than the receding horizon in which this optimization is carried out. Otherwise, the applicability of this technique for real-time scenarios would be questionable. Developing a guideline for selecting the optimal size for the receding horizon for robust visual servoing in real time remains an open research area.

Robustness with respect to calibration errors in terms of the tracking error boundness along the planned trajectories has been considered in [73]. Given a user defined bound on the tracking error, they propose a control strategy to modulate control gains and/or the desired tracking velocity to guarantee error boundness. Through the proposed velocity modulation technique, one could use low control gains while keeping the tracking error bounded. While this technique and those mentioned above, to some extent, are expected to take care of the deviations from the planned trajectories in the image space, the deviations from the physical space trajectories can cause robot/physical constraints violations. The above mentioned local strategies for accounting deviations from planned path are not general enough to account for all types of constraints (and the related uncertainties), in particular robot/physical constraints. Hence, there is need for taking the uncertainties into account in a global as well as general manner at the planning stage.

The planned paths obtained based on only a nominal model of the camera and/or robot may not be fully traversable by the robot without violating certain constraints. In a recent work [13], Chesi proposed a planning approach to design a robust image trajectory that satisfies the required constraints not only for the nominal model but rather for a family of admissible models. In the proposed approach an uncertain model has been considered for image correspondence between the initial and desired images, and the camera's intrinsic parameters are assumed to be affected by some unknown random errors with known bounds. Given the above uncertain models, there are different admissible camera poses and consequently different camera trajectories rather than a common and robust one. A polynomial parametrization is proposed through which each camera trajectory is parameterized by a possible camera pose and by a design variable which is common to all admissible trajectories. So, the robust trajectory is computed through an optimization problem determining the common design variable that satisfies field of view limits and maximizes the distance of the image features from the boundary of image on all parameterized trajectories.

Although the results obtained through the above approaches in taking calibration uncertainty and measurement errors into account seem promising, more research needs to be done. Physical constraints, especially collisions and occlusions, are highly affected by the uncertainties in the modeling of the environment. Robot path planning considering uncertainties in modeling, localization, and sensing has been studied for decades within path planning community [56] yielded a number of promising approaches, e.g. [9], [39], [60], [68], [76]. Incorporating the research results achieved through these approaches into the visual servoing framework would be a promising future direction. Moreover, planning robust trajectories for visual servoing tasks in unknown or partially known environments remains an open research problem.

2.5 Summary

We provided a comprehensive review of existing path planning for visual servoing techniques aimed at making the visual servoing more robust in complex scenarios, especially in applications where the initial and desired views are distant. Considering the underlying path planning approach, the existing techniques have been divided into four categories: (1) Image space path planning, (2) Optimization-based path planning, (3) Potential Field-based path planning, and (4) Global path planning. We reported on the previous works pertinent to each category and for each technique we discussed the set of assumptions along with its benefits and drawbacks and its integration with the reactive visual servo controllers.

Recent works (discussed in Section 2.4) demonstrated the effectiveness of accounting for modeling/calibration uncertainties and measurement errors at the planning stage in generating robust trajectories for visual servoing scenarios where the available data are affected by uncertainties. Towards that aim, incorporating the results achieved on robot path planning under uncertainty within the path planning community is a promising direction to follow.

Chapter 3

Randomized Path Planning for Visual Servoing

In this chapter, we present our proposed randomized path planning framework for visual servoing tasks. We discuss the strategies through which the image and physical constraints are taken into account within the proposed framework. We also explain the implementation of two planning strategies: kinematic planning in camera task space (i.e., space of camera poses), and kinodynamic planning in camera state space (i.e., space of camera poses and velocities).

We first begin by explaining the preliminary concepts and giving an overview of the proposed solution methodology as follows.

3.1 Preliminaries and overview

3.1.1 Notations

Consider a frontal pin-hole perspective camera model as shown in Figure 3.1. Let $\mathcal{P}_j$ be a 3D point with homogeneous coordinates $[X_j \ Y_j \ Z_j \ 1]^T$ in the camera frame $\mathcal{F}_c$. Assuming focal length f to be unit, the projection of $\mathcal{P}_j$ into the image plane of $\mathcal{F}_c$ is given as a point with homogeneous coordinates $m_j = [x_j \ y_j \ 1]^T$ where

$$x_j = \frac{1}{Z_j} X_j \quad \text{and} \quad y_j = \frac{1}{Z_j} Y_j. \tag{3.1}$$

The corresponding pixel coordinates can be calculated as $p_j = [u_j, v_j, 1]^T = Am_j$, where

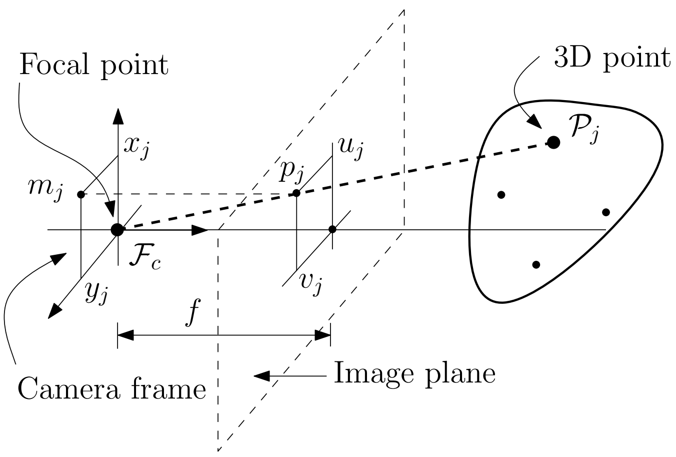


Figure 3.1: A frontal pin-hole perspective camera model

matrix A contains the intrinsic parameters of camera as

$$A = \begin{bmatrix} fp_u & 0 & u_0 \\ 0 & fp_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

In (3.2), f denotes the focal length, u_0 and v_0 are the pixel coordinates of camera principal point, p_u and p_v are the number of pixels per unit distance in image coordinates.

We use the 2-D images of a number of 3-D points $\mathcal{P}_j$ for $j = 1, \ldots, n$ as image features to represent the solution trajectories in the image. Hence, a point $\mathbf{s}$ along a feature trajectory is then represented as a vector $\mathbf{s} = [u_1 \; v_1 \ldots u_n \; v_n]^T$ where u_j and v_j are the image pixel coordinates of 3-D points $\mathcal{P}_j$ for $j = 1, \ldots, n$.

Considering Fig.3.2, let $\mathcal{F}_o$ be the frame attached to the target, $\mathcal{F}_i$ and $\mathcal{F}_d$ denote the camera frames at the initial and desired camera/robot location, and $\mathcal{F}_k$ denotes the camera frame at an intermediate location. Given the coordinates of four or more point features in the object frame $\mathcal{F}_o$ and knowing the correspondences between their initial and desired image features, using techniques such as that in [26] and [66], one could compute the transformation between the initial and desired camera frames, $\mathcal{F}_i$ and $\mathcal{F}_d$ respectively, and the target object frame $\mathcal{F}_o$, i.e. ${}^iR_o$, ${}^it_o$, ${}^dR_o$, and ${}^dt_o$ (see Fig.3.2).

3.1.2 Problem formulation and solution methodology

The aim of path planning for visual servoing is to plan feasible feature trajectories as a sequence of image features $\mathbf{s}(t)$ for $t \in [0, t_f]$ in the image space between initial and desired

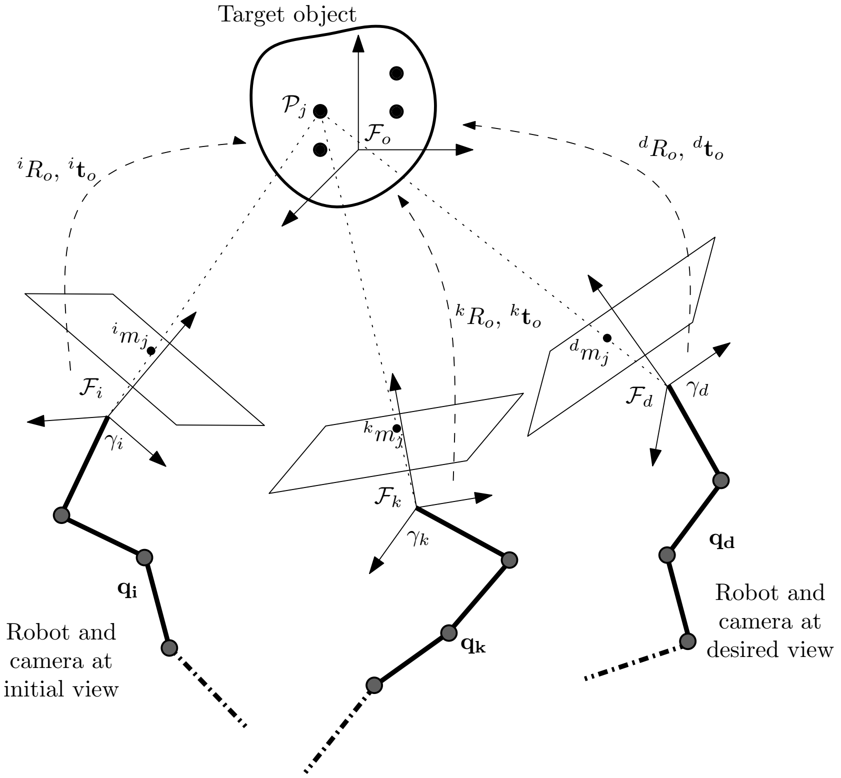


Figure 3.2: Robot/camera at the initial, intermediate, and desired configurations w.r.t. the target object.

image features, $\mathbf{s}(0) = \mathbf{s}_i$ and $\mathbf{s}(t_f) = \mathbf{s}_d$, extracted from the images taken at the initial and final desired camera poses, respectively. The time parameterization, i.e. $t \in [0, t_f]$, is usually determined as the result of underlying planning technique. Our proposed planning approach is summarized in the following two steps:

Step1. Planning feasible robot/camera trajectories: first we plan a feasible camera trajectory $\Gamma(t)$ for $t \in [0, t_f]$, in the camera planning space between the initial and desired camera planning vectors, $\Gamma(0) = \mathbf{x}_i$ and $\Gamma(t_f) = \mathbf{x}_d$, respectively. The definition of the camera planning vector $\mathbf{x}(t)$ depends on the topology of the space in which the camera trajectories are planned. The framework presented in this work is independent of the topology of the camera planning space. We consider two alternatives (see Section 3.2.2): camera configuration space, i.e., space of camera poses only , and camera state space, i.e., space of camera poses and velocities.

The camera trajectory is obtained by alternate exploration of the camera planning space and robot joint space which corresponds to a feasible robot trajectory $\mathbf{q}(t)$ for $t \in [0, t_f]$ in the robot configuration space between the start and goal configurations, $\mathbf{q}(0) = \mathbf{q}_i$ and

$\mathbf{q}(t_f) = \mathbf{q}_d$, respectively. One should note that the goal configuration $\mathbf{q}_d$ is not initially known and will be determined as the result of the planning. The proposed planner explores the camera planning space for permissible trajectories by iteratively extending a search tree in this space and simultaneously tracking these trajectories in the robot configuration space. The proposed planning framework is explained in more details in Section 3.2.

Step2. Computing image feature trajectories: the camera trajectory $\Gamma(t)$ planned in Step 1 is then utilized to project the 3D target feature points into the image space and obtain the feature trajectories $\mathbf{s}(t)$ between the initial and desired image features in the image space.

For a given camera frame denoted by a homogeneous rotation/translation transformation $[R(t_k)|\mathbf{p}(t_k)]$ along the planned camera trajectory $\Gamma(t_k)$, the 2D image feature $\mathbf{s}_j(t_k) = [u_j(t_k)\ v_j(t_k)]^T$ of the 3D target feature point $\mathcal{P}_j$ with homogeneous coordinates ${}^oP_j$ in the object frame is computed as

$$Z_j(u_j, v_j, 1)^T = A\Big[R^T\Big| - R^T\mathbf{p}\Big]{}^oP_j, \tag{3.3}$$

where Z_j is the depth of the target feature $\mathcal{P}_j$ in the camera frame. Similarly, all object features $\mathcal{P}_j$ are projected to their pixel coordinates $\mathbf{s}_j$ for $j = 1, \ldots, n$. The above projection is repeated for all camera poses along the planned camera trajectory to obtain the corresponding feature trajectories in the image.

3.2 Alternate exploration of camera planning space and robot configuration space

Our proposed planning approach explores the camera planning space by extending an exploring tree as in Rapidly-exploring Random Tree (RRT) approach [50] in the camera planning space and simultaneously tracking the tree local paths in the robot configuration space. So, through this strategy the search in the camera planning space is used to effectively guide the search in the robot configuration space. The proposed strategy is summarized in Algorithm 1. First, we provide an overview of Algorithm 1 below, and we will discuss its implementation details next.

Algorithm 1: Alternate camera planning space and robot C-space exploration

```
input  : Camera initial and desired vectors, $\mathbf{x}_i$ and $\mathbf{x}_d$
output: Camera trajectory $\Gamma$, or null in case of failure
begin
    $\mathcal{T} \leftarrow$ InitializeTree($\mathbf{q}_i$, $\mathbf{x}_i$);
    repeat
        $\mathbf{x}_{rand} \leftarrow$ GenerateRandomCameraVector();
        $\mathbf{x}_{near} \leftarrow$ FindNearestInTree($\mathcal{T}$, $\mathbf{x}_{rand}$);
        ($\mathbf{x}_{new}$, $\mathcal{E}_x$) $\leftarrow$ ExtendWithImageConstraints($\mathbf{x}_{near}$, $\mathbf{x}_{rand}$);
        if $\mathcal{E}_x \neq$ null then
            ($\mathbf{q}_{new}$, $\mathcal{E}_q$) $\leftarrow$ TrackCameraPathInCSpace($\mathcal{E}_x$);
            if $\mathcal{E}_q \neq$ null then
                Add $\mathbf{x}_{new}$ and $\mathcal{E}_x$ to tree $\mathcal{T}$;
                if rand() $\leq p_{greedy}$ then
                    ($\mathbf{x}_d$, $\mathcal{E}_x$) $\leftarrow$ ExtendGreedyToGoal($\mathbf{x}_{new}$, $\mathbf{x}_d$);
                    if  $\mathcal{E}_x \neq$ null then
                        Add $\mathbf{x}_d$ and $\mathcal{E}_x$ to tree $\mathcal{T}$;
                        $\Gamma \leftarrow$ RetrieveCameraTrajectory($\mathcal{T}$, $\mathbf{x}_d$);
                        return $\Gamma$;
    until timeout;
    return null;
```

3.2.1 Overview of the proposed approach

As shown in Algorithm 1, the exploring tree $\mathcal{T}$ is iteratively extended in the camera planning space until either it reaches the goal or the planning time is up. Consider Fig. 3.3. At each iteration a node with a random camera vector $\mathbf{x}_{rand}$ is generated and the node with the nearest vector $\mathbf{x}_{near}$ to $\mathbf{x}_{rand}$ in the tree is found. Then, $\mathbf{x}_{near}$ is extended toward $\mathbf{x}_{rand}$. The local path $\mathcal{E}_x$ obtained as the result of this extension is projected into the image space to check for image constraints (i.e. field of view limits and occlusions of target object by other obstacles or itself) by sampling synthetic images along the local path. As mentioned earlier, the extension of the camera paths depends on the topology of the camera planning space which will be discussed in more details later.

Given that the local camera path violates no image constraints, it is then tracked in the robot configuration space using a local planner to check for physical constraints, i.e. collision with obstacles and joint limits. Tracking camera paths in the robot configuration space depends on the kinematic structure of the robot and one may adopt any desired local planner as long as the constraint at the camera (or robot's end-effector) is maintained.

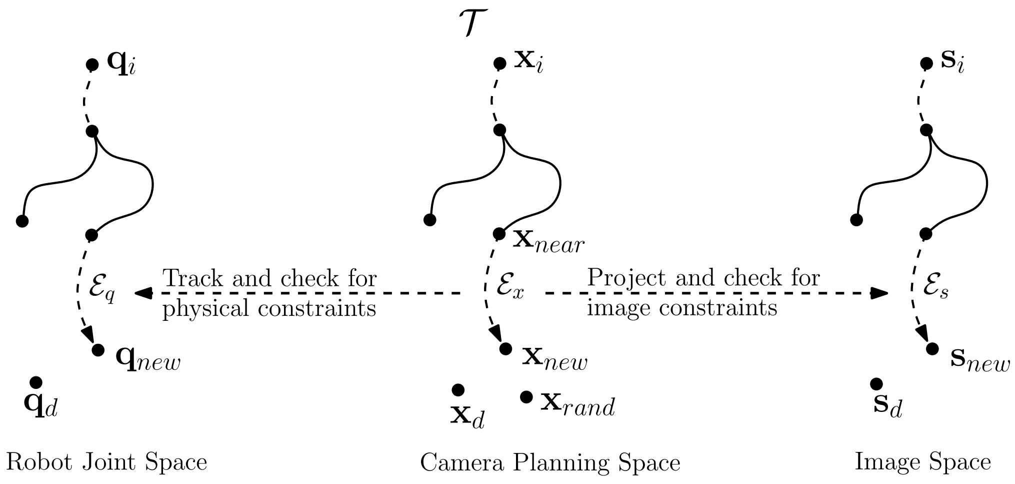


Figure 3.3: Alternate camera planning space and robot C-space exploration

We adopt an Inverse Kinematic controller where the robot is considered as a kinematic structure for which the Jacobian that relates the robot generalized velocities (i.e., joint space velocities) to its end-effector task space velocities is available. The details of the inverse kinematic controller are provided in Section 3.2.6.

In case of successful tracking of the camera local path in robot configuration space, the newly generated vector $\mathbf{x}_{new}$ (along with edge $\mathcal{E}_x$) is added as a new node to the camera tree. After each successful addition of a new node to the camera tree, with a probability p_{greedy}, a greedy extension from the newly added node is attempted to reach the goal vector $\mathbf{x}_d$. The greedy extension depends on the topology of the camera planning space and will be explained accordingly.

3.2.2 Camera planning space

The algorithm starts off by initializing a tree structure $\mathcal{T}$ and generating the root of the tree as its only node (Line 2). Each node of the tree embeds the following information:

- $\mathbf{x}$, camera planning vector
- $\mathbf{q}$, robot configuration
- $\mathcal{E}_x$, camera sub-path from the parent node
- $\mathcal{E}_q$, robot sub-path from the parent node
- a pointer to the parent node

The root node is initialized with the camera initial planning vector $\mathbf{x}_i$ and the corresponding robot initial configuration $\mathbf{q}_i$ with a null parent pointer.

The definition of camera planning vector $\mathbf{x}$ depends on the topology of the space in which the camera trajectory is planned. Within the proposed framework, we consider two alternatives:

- Camera *configuration space* (i.e., space of camera poses), where a vector $\mathbf{x}$ (also called camera *configuration*) in this space is represented as

$$\mathbf{x} = \begin{bmatrix} \mathbf{p} \\ \mathbf{h} \end{bmatrix} \tag{3.4}$$

  where $\mathbf{p} = [p_x \; p_y \; p_z]^T$ is the position of the camera frame and $\mathbf{h} = [h \; \overrightarrow{\mathbf{h}}^T]^T$ denotes the unit quaternion (with scalar part h and vector part $\overrightarrow{\mathbf{h}}$) representation of the camera orientation. This makes the $\mathbf{x}$ a 7-D vector, but the camera configuration space is 6-D because of the constraint that the quaternion must be of unit norm.

- Camera *state space* (i.e., space of camera poses and velocities) where a vector $\mathbf{x}$ (also called camera *state*) in this space is represented as

$$\mathbf{x} = \begin{bmatrix} \mathbf{p} \\ \mathbf{h} \\ \mathbf{v} \\ \boldsymbol{\omega} \end{bmatrix} \tag{3.5}$$

  where $\mathbf{p}$ and $\mathbf{h}$ are defined as in (3.4), and $\mathbf{v} = [v_x \; v_y \; v_z]^T$ and $\boldsymbol{\omega} = [\omega_x \; \omega_y \; \omega_z]^T$ denote the camera linear and angular velocities, respectively. This makes the state $\mathbf{x}$ a 13-D vector, but the state space is 12-D because of the constraint that the quaternion must be of unit norm.

Extending the trajectories in the camera configuration space results in a kinematic planning scheme which yields C^0-smooth trajectories with discontinuous velocities and infinite acceleration, hence require smoothing and proper time-scaling to be executed on the robot. Planning in camera state space (as opposed to camera configuration space) allows for taking the camera dynamics as well as its kinematic constraints into account at the planning stage leading to a *kinodynamic* planning scheme which yields C^1-smooth with continuous velocities and bounded acceleration change and properly time-scaled trajectories. The extension of the camera trajectories in the camera configuration and state spaces are discussed in Section 3.2.5.

3.2.3 Generating random camera vectors

At each iteration of Algorithm 1, a random camera planning vector $\mathbf{x}_{rand}$ is generated. In the case of planning in camera configuration space, a random camera configuration is built by generating a uniform random camera position $\mathbf{p}_{rand}$ and a uniform random unit quaternion $\mathbf{h}_{rand}$ within the camera workspace. The random camera position is generated as

$$\mathbf{p}_{rand} = \begin{bmatrix} x_{min} + (x_{max} - x_{min})r_x \\ y_{min} + (y_{max} - y_{min})r_y \\ z_{min} + (z_{max} - z_{min})r_z \end{bmatrix} \tag{3.6}$$

where r_x, r_y, r_z are scalar values generated uniformly in the range $[0, 1]$, and $[x_{min}, x_{max}]$, $[y_{min}, y_{max}]$, $[z_{min}, z_{max}]$ denote the limits of the camera workspace along XYZ axes. A unit quaternion is generated uniformly as [49]

$$\mathbf{h}_{rand} = \begin{bmatrix} \cos(2\pi r_y)\sqrt{r_z} \\ \sin(2\pi r_x)\sqrt{1 - r_z} \\ \cos(2\pi r_x)\sqrt{1 - r_z} \\ \sin(2\pi r_y)\sqrt{r_z} \end{bmatrix} \tag{3.7}$$

In case of planning in camera state space, a random camera state is built by extending a random configuration as generated above and adding random uniform linear and angular velocities as

$$\mathbf{v}_{rand} = \begin{bmatrix} v_{x_{min}} + (v_{x_{max}} - v_{x_{min}})r_x \\ v_{y_{min}} + (v_{y_{max}} - v_{y_{min}})r_y \\ v_{z_{min}} + (v_{z_{max}} - v_{z_{min}})r_z \end{bmatrix} \quad \text{and} \quad \boldsymbol{w}_{rand} = \begin{bmatrix} \omega_{x_{min}} + (\omega_{x_{max}} - \omega_{x_{min}})r_x \\ \omega_{y_{min}} + (\omega_{y_{max}} - \omega_{y_{min}})r_y \\ \omega_{z_{min}} + (\omega_{z_{max}} - \omega_{z_{min}})r_z \end{bmatrix} \tag{3.8}$$

where $[v_{x_{max}}, v_{x_{min}}]$, $[v_{y_{max}}, v_{y_{min}}]$, and $[v_{z_{max}}, v_{z_{min}}]$ denote the limits of camera linear velocities, and $[\omega_{x_{max}}, \omega_{x_{min}}]$, $[\omega_{y_{max}}, \omega_{y_{min}}]$, and $[\omega_{z_{max}}, \omega_{z_{min}}]$ are the limits of camera angular velocities.

3.2.4 Finding nearest node in the tree

At each extension of the camera tree the tree node $\mathbf{x}_{near}$ with the closest distance to $\mathbf{x}_{rand}$ is found based on a metric $\rho(\mathbf{x}_1, \mathbf{x}_2)$ defined as a measure of relative closeness between two camera vectors, i.e., $\mathbf{x}_1$ and $\mathbf{x}_2$, in camera planning space.

The metric $\rho(\mathbf{x}_1, \mathbf{x}_2)$ in camera configuration space is defined as [58]

$$\rho(\mathbf{x}_1, \mathbf{x}_2) = w_p\|\mathbf{p}_1 - \mathbf{p}_2\| + w_h(1 - |\mathbf{h}_1.\mathbf{h}_2|), \tag{3.9}$$

where $\mathbf{p}_1$ and $\mathbf{p}_2$ are the camera positions, $\mathbf{h}_1.\mathbf{h}_2$ denotes the inner product of unit quaternions corresponding to camera orientations pertinent to two poses, $\mathbf{x}_1$ and $\mathbf{x}_2$, respectively. The weight coefficients w_p and w_h are user defined (in our implementation, we set $w_p = 0.6$ and $w_h = 0.4$).

When $\mathbf{x}_1$ and $\mathbf{x}_2$ represent camera states (as in kinodynamic planning scheme), the closeness metric $\rho(\mathbf{x}_1, \mathbf{x}_2)$ is defined as [58]

$$\rho(\mathbf{x}_1, \mathbf{x}_2) = w_p\|\mathbf{p}_1 - \mathbf{p}_2\| + w_h(1 - |\mathbf{h}_1.\mathbf{h}_2|) + w_v\|\mathbf{v}_1 - \mathbf{v}_2\| + w_\omega\|\boldsymbol{\omega}_1 - \boldsymbol{\omega}_2\|, \tag{3.10}$$

where $\mathbf{v}_1$ and $\mathbf{v}_2$ are the camera linear velocities, and $\boldsymbol{\omega}_1$ and $\boldsymbol{\omega}_2$ denote the camera angular velocities pertinent to two states, $\mathbf{x}_1$ and $\mathbf{x}_2$, respectively. The weight coefficients w_p, w_h, w_v, and w_ω are user defined (in our implementation we set $w_p = 0.4$, $w_h = 0.3$, $w_v = 0.2$, and $w_\omega = 0.1$).

3.2.5 Extending camera paths with image constraints

Once the tree node with the nearest camera vector, i.e. $\mathbf{x}_{near}$, to $\mathbf{x}_{rand}$ was found, the camera vector $\mathbf{x}_{near}$ is extended toward $\mathbf{x}_{rand}$ using a local controller. One may choose any arbitrary controller as long as it extends $\mathbf{x}_{near}$ towards and close to $\mathbf{x}_{rand}$. In RRT-like schemes designing a proper extension controller is a challenging task. Basically, the problem is to find an appropriate control input $\mathbf{u}(t)$ which is applied based on the equation of motion f to advance the state of the system, i.e.

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)) \tag{3.11}$$

over a pre-defined time interval so that the new state reaches a desired state. This can be formulated as a boundary value problem. The solution to such problem, depending on the complexity of the system and the underlying constraints, can be as hard as the original planning problem, if at all possible. Usually, in RRT literature (e.g., [58]), the input $\mathbf{u}$ is chosen at random or by trying all the inputs from a discretized set of possible inputs and choosing the one that yields a new state as close as possible to the randomly selected vector $\mathbf{x}_{rand}$. Please note that this does not sacrifice the performance of the RRT extension since extension is still achieved toward the randomly selected node and hence the tree will be expanded into the Voronoi region of the random node, thereby guaranteeing the extension of the tree into less explored regions of the space [58].

The camera tree is extended from $\mathbf{x}_{near}$ by applying the control input $\mathbf{u}$ and integrating (3.11) over a user-defined time step Δt. The length of each extension has a direct relationship with Δt which can be chosen arbitrarily (but non-zero) with the knowledge that higher values of Δt (e.g., 2.0 seconds) yields longer tree edges (with higher probability of violating a constraint along the extension in cluttered space) but faster extension of the tree in wide open regions, while lower values of Δt (e.g., 0.5 seconds) result in shorter edges, and slow but dense extension of the tree into the space. Usually, in a cluttered environment lower time steps are recommended. The above integration requires a numerical approximation of (3.11): given the current state $x(t)$ and input $\mathbf{u}(t)$ applied over a time interval Δt, the task is to compute the next state $\mathbf{x}(t + \Delta t)$. For the extension strategies in this thesis, we employ a simple fixed-step Euler method for numerical integration, i.e. $\mathbf{x}(t + \delta_t) = \mathbf{x}(t) + \delta_t f(\mathbf{x}(t), \mathbf{u}(t))$, where δ_t is the integration time step (we used $\delta_t = 0.04$ sec. in our experiments).

We discuss the local control strategies for extending the tree in camera configuration space as well as camera state space, as two alternative planning spaces for camera paths.

Kinematic extension in camera configuration space

The tree in camera configuration space is extended from $\mathbf{x}_{near}$ toward $\mathbf{x}_{rand}$ by applying appropriate linear and angular velocity control inputs, i.e., $\mathbf{v}(t)$ and $\boldsymbol{\omega}(t)$. The equation of motion for the camera under the influence of control input $\mathbf{u}(t) = (\mathbf{v}(t), \boldsymbol{\omega}(t))$ is given as

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)) = \begin{bmatrix} \dot{\mathbf{p}}(t) \\ \dot{\mathbf{h}}(t) \end{bmatrix} = \begin{bmatrix} \mathbf{v}(t) \\ \frac{1}{2}\boldsymbol{\omega}_{\times}(t).\mathbf{h}(t) \end{bmatrix} \tag{3.12}$$

where $\boldsymbol{\omega}_{\times}(t).\mathbf{h}(t)$ denotes the quaternion dot product between $[0\ \omega_x\ \omega_y\ \omega_z]^T$ and $\mathbf{h}(t)$.

The control $\mathbf{u}$ must be chosen in a way to ensure extension of the $\mathbf{x}_{near}$ toward $\mathbf{x}_{rand}$. We choose the directions of the linear and angular velocity inputs along the relative direction of the position and orientation of the randomly selected configuration $\mathbf{x}_{rand}$ with respect to the current state, i.e. $\overrightarrow{\mathbf{v}} = \overrightarrow{\mathbf{p}_{rand} - \mathbf{p}_{near}}$ and $\overrightarrow{\boldsymbol{\omega}}$ is chosen as the axis of the quaternion representing the relative orientation between $\mathbf{x}_{near}$ and $\mathbf{x}_{rand}$ computed as $\mathbf{h}_{near}^{-1}\mathbf{h}_{rand}$. The magnitude of the velocity inputs are chosen randomly from pre-defined intervals, i.e., $\|\boldsymbol{\omega}\| \in [0, \omega_{max}]$ and $\|\mathbf{v}\| \in [0, v_{max}]$ (in our implementation, we set $v_{max} = 0.1$ meter/second and $\omega_{max} = 5$ degree/second). The proposed extension scheme results in C^0-smooth camera trajectories with discontinuous velocities at each node, hence virtually assuming infinite acceleration is achievable. Of course, such assumption cannot be guaranteed by the underlying physical robot.

Kinodynamic extension in camera state space

For the case of planning in camera state space, the camera is modeled as an unconstrained rigid body of mass M and inertia of I as required by the motion equation. The control inputs are the force and torque which can be exerted to the camera as a rigid body to change the course of its motion. The equation of motion for the camera under the influence of control input $\mathbf{u}(t) = (\mathbf{f}(t), \boldsymbol{\tau}(t))$, i.e. the pair of force-torque input, is given as [2]

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)) = \begin{bmatrix} \dot{\mathbf{p}}(t) \\ \dot{\mathbf{h}}(t) \\ \dot{\mathbf{v}}(t) \\ \dot{\boldsymbol{\omega}}(t) \end{bmatrix} = \begin{bmatrix} \mathbf{v}(t) \\ \frac{1}{2}\boldsymbol{\omega}_{\times}(t).\mathbf{h}(t) \\ \mathbf{f}(t)/M \\ R(t)I^{-1}R(t)^T\boldsymbol{\tau}(t) \end{bmatrix} \tag{3.13}$$

where $\boldsymbol{\omega}_{\times}(t).\mathbf{h}(t)$ denotes the quaternion dot product between $[0\ \omega_x\ \omega_y\ \omega_z]^T$ and $\mathbf{h}(t)$. The rotation matrix $R(t)$ and its transpose $R(t)^T$ are computed by converting the quaternion $\mathbf{h}(t)$ to its matrix representation.

We have defined nominal intervals for the amount of force and torque from which the magnitudes of force and torque inputs are randomly selected for each extension, i.e. $\|\mathbf{f}(t)\| \in [0, f_{max}]$ and $\|\boldsymbol{\tau}(t)\| \in [0, \tau_{max}]$; and, similarly to the kinematic extension scheme explained above, the directions of the force and torque inputs are computed along the relative direction of the position and orientation of the randomly selected state $\mathbf{x}_{rand}$ with respect to the current state, i.e. $\overrightarrow{\mathbf{f}} = \overrightarrow{\mathbf{p}_{rand} - \mathbf{p}_{near}}$ and $\overrightarrow{\boldsymbol{\tau}}$ is chosen as the axis of the quaternion representing the relative orientation between $\mathbf{x}_{near}$ and $\mathbf{x}_{rand}$ computed as $\mathbf{h}_{near}^{-1}\mathbf{h}_{rand}$.

Independent of the underlying local controller, the result of the tree extension (if successful) is a camera sub-path $\mathcal{E}_x$ starting from $\mathbf{x}_{near}$ and ending at a new camera vector $\mathbf{x}_{new}$ (see Figure 3.3). The camera sub-path $\mathcal{E}_x$ must then be checked for image constraints, i.e. camera field of view and occlusion due to obstacles, as explained below.

Checking for camera field of view limits

The camera paths (as edges of the camera tree) should be examined for the camera field of view constraints to ensure that the target features remain within the field of view through out the path. For this, we first use perspective projection to project object 3-D features into the image space using (3.3). For a 3-D point $\mathcal{P}_j$ with coordinates $[X_j \; Y_j \; Z_j]^T$ in the camera and image plane coordinates $\mathbf{s}_j = [u_j \; v_j]^T$, following simple inequalities check weather $\mathcal{P}_j$ remains in the camera field of view

$$\left.\begin{array}{rcl} u_{min} < & u_j & < u_{max} \\ v_{min} < & v_j & < v_{max} \\ & Z_j & > 0 \end{array}\right\} \quad \text{for} \quad j = 1, \ldots, n \tag{3.14}$$

where $u_{min}, u_{max}, v_{min}$, and v_{max} define the field of view limits and n denotes the number of target features. The last constraint in (3.14) merely guarantees that the target features will remain in front of the camera.

Checking for occlusion due to physical obstacles

Target features' occlusions (due to workspace obstacles, robot body, or the target itself) for each camera pose along the camera paths are checked by ray-tracing from the camera optical center towards each target feature. This is done by checking for collision between the ray radiated from the target feature toward the camera optical center with the physical obstacles including workspace obstacles, robot body, and target itself. One could also employ the dynamic occlusion checking technique proposed in [3] to check for occlusion along the camera sub-path. However, the technique in [3] requires explicit computation of the boundary of occluded and visible regions.

3.2.6 Tracking camera paths in robot configuration space

The camera sub-path $\mathcal{E}_x$ must be tracked in the joint space to check its feasibility and obtain the corresponding robot sub-path $\mathcal{E}_q$ which maintains the camera motion along $\mathcal{E}_x$.

We adopt an inverse kinematic controller as in [89] to track the camera sub-paths in the robot configuration space. Given a desired camera pose $[\mathbf{p}_d^T \; \mathbf{h}_d^T]^T$ and velocity screw $[\mathbf{v}_d^T \; \boldsymbol{\omega}_d^T]^T$ along a given camera path, the robot current configuration $\mathbf{q}(t)$ is advanced as

$$\mathbf{q}(t + d_t) = \mathbf{q}(t) + \dot{\mathbf{q}} d_t \tag{3.15}$$

with joint velocities computed as

$$\dot{\mathbf{q}} = K_p \widehat{\mathbf{J}}^+ \begin{bmatrix} \mathbf{e}_p \\ \mathbf{e}_o \end{bmatrix} + \widehat{\mathbf{J}}^+ \begin{bmatrix} \mathbf{v}_d \\ \boldsymbol{\omega}_d \end{bmatrix} \tag{3.16}$$

and where $\widehat{\mathbf{J}}^{+}$ is the pseudo-inverse of a nominal model of the robot Jacobian $\mathbf{J} = \partial \mathbf{r}/\partial \mathbf{q}$, K_p is a positive proportional gain, d_t is the integration time step, the position error $\mathbf{e}_p$ is calculated as

$$\mathbf{e}_p = \mathbf{p}_d - \mathbf{p}, \tag{3.17}$$

and the orientation error $\mathbf{e}_o$ is given as [83]

$$\mathbf{e}_o = h\overrightarrow{\mathbf{h}_d} - h_d\overrightarrow{\mathbf{h}} - \mathbf{S}(\overrightarrow{\mathbf{h}_d})\overrightarrow{\mathbf{h}} \tag{3.18}$$

where h, h_d and $\overrightarrow{\mathbf{h}}$, $\overrightarrow{\mathbf{h}_d}$ are the scalar and vector parts of the corresponding quaternions, i.e., $\mathbf{h} = [h\ \overrightarrow{\mathbf{h}}^T]^T$ and $\mathbf{h}_d = [h_d\ \overrightarrow{\mathbf{h}_d}^T]^T$, and $\mathbf{S}(.)$ is the skew-symmetric operator.

The new configuration $\mathbf{q}(t + d_t)$ is then checked for collision with obstacles and joint limits as well as closeness to singularities in robot Jacobian and joint velocity limits. After successful tracking of the entire camera sub-path $\mathcal{E}_x$ in joint space and obtaining the corresponding joint space path $\mathcal{E}_q$, a new edge is added to the tree along with a new node containing $\mathcal{E}_x$, $\mathcal{E}_q$, and their end points.

3.2.7 Greedy extension of camera tree to the goal

Following each successful extension of the camera tree and with a probability of p_{greedy} a greedy extension is attempted toward the desired camera goal vector $\mathbf{x}_d$ defined by the camera desired position and orientation, i.e. $\mathbf{p}_d$ and $\mathbf{h}_d$, and in the case of kinodynamic planning, zero linear and angular velocities, i.e., $\mathbf{v}_d = \mathbf{0}$ and $\omega_d = \mathbf{0}$ (the camera comes to a stop at the goal).

In the case of kinematic extension in camera configuration space, we adopt a simple linear interpolation (Lerp) between the newly generated configuration $\mathbf{x}_{new}$ and the goal configuration of the camera, $\mathbf{x}_d$. The greedy trajectories for the camera position $\mathbf{p}(t)$ and orientation $\mathbf{h}(t)$ paths are given as

$$\mathbf{p}(t) = \alpha\mathbf{p}_d + (1-\alpha)\mathbf{p}_{new} \tag{3.19}$$

$$\mathbf{h}(t) = \alpha\mathbf{h}_d + (1-\alpha)\mathbf{h}_{new} \tag{3.20}$$

for $\alpha \in [0, 1]$. The linear interpolation in (3.20) results in a straight line in quaternion space with non-unit quaternions along the path. Hence, the quaternions along $\mathbf{h}(t)$ need to be normalized.

In case of kinodynamic planning in the camera state space, the above linear interpolation is not sufficient since it may not satisfy the camera velocity constraints at the newly generated state $\mathbf{x}_{new}$ and/or at the goal state $\mathbf{x}_{goal}$. The greedy extension of the camera tree can be formulated in form of a boundary value problem: i.e., we are interested in computing a camera trajectory $\mathbf{x}(t)$ for $[t_0, t_1]$ from a given camera state $\mathbf{x}(t_0) = \mathbf{x}_{new}$ to the desired camera state $\mathbf{x}(t_1) = \mathbf{x}_d$ subjected to (3.13). The interval $t_g = t_1 - t_0$ is a predefined greedy extension time. To solve the above boundary value problem one needs to design proper force and torque input profiles for the entire greedy extension interval. However, we devise a trajectory planning approach [23] by using cubic polynomials to generate camera position and orientation trajectories along the greedy extension. Therefore, the trajectories of camera position $\mathbf{p}(t)$ and orientation $\mathbf{h}(t)$ (represented as unit quaternion) are given as

$$\mathbf{p}(t) = A_3 t^3 + A_2 t^2 + A_1 t + A_0 \tag{3.21}$$

$$\mathbf{h}(t) = B_3 t^3 + B_2 t^2 + B_1 t + B_0 \tag{3.22}$$

for $t \in [t_0, t_1]$ where A_i's and B_i's are coefficient matrices which can be computed given the initial and end pose and velocity conditions imposed by $\mathbf{x}_0$ and $\mathbf{x}_d$, and also the termination time.

3.3 Discusions

3.3.1 Kinodynamic vs. Kinematic Planning

In the above section we explained two alternative kinematic and kinodynamic extension strategies. Although the proposed planning framework is independent of the chosen extension strategy, the performance of tracking solution trajectories at the execution stage is highly affected by the quality (i.e., smoothness) of the planned trajectories.

The kinematic extension strategy results in C^0-smooth trajectories with discontinuous velocities (and infinite acceleration) at extension points, i.e., nodes of the tree. Hence, the solution trajectories are not suitable to be executed directly on the robotic system and require to be first smoothened and properly time-scaled. However, the smoothing comes with the price of deviating from the solution trajectories close to the extension points (cutting the corners) which may not be plausible without violating the image/physical constraints. Hence, the robot needs to come to a full stop at extension points to be able to reach and pass through the sharp corners, which is not desired when performing closed-loop control. In [41] we show the planning results obtained using the kinematic planning scheme, but we did not try executing the planned trajectories on a real system.

On the other hand, the kinodynamic planning scheme, by construction, results in C^1-smooth trajectories with continuous velocities and bounded acceleration change, hence require no further smoothing/time-scaling to be executed on the robot. All the real world experiments in this thesis use the kinodynamic planning scheme.

3.3.2 Remarks on Complexity

In practice, sampling-based planning algorithms (including RRT-based planners as we adopted in this work) have been quite practically effective for high degree of freedom spaces. RRT is probabilistically complete, i.e., the probability that a solution is found tends to 1 as the number of sampling/extension iterations approach infinity (see [58] for the proof). Unfortunately, as pointed out in [58], it remains a challenge to express the RRT convergence rate for a particular problem in terms of parameters which can be measured easily. As a common practice and also to give the reader a better understanding of problem complexity, we have reported the planning time and the number of successfully explored nodes for each of the experiments provided in this thesis.

3.4 Summary

In this chapter, we presented our proposed planning framework for visual servoing. The proposed planning framework, built on randomized tree-based planning strategies, explores the camera planning space for feasible camera trajectories and simultaneously tracks the planned trajectories in the robot configuration space. The proposed planning scheme, considers the robot as a kinematic mechanism with a Jacobian relating its task space velocities to the generalized joint space velocities. This facilitates the application of the proposed framework to both (serial chain) robotic arms (as shown in Chapter 4), and (non-holonomic) wheeled mobile manipulators (as shown in Chapter 5).

We discussed the implementation details of planning algorithm, and proposed two alternative kinematic and kinodynamic planning schemes. We explained the flexibility of the proposed framework and showed how various image and physical constraints can be taken into account. Table 3.1 lists the parameters used in our proposed planning scheme and their typical values we used in our experiments.

		Typical value	
Parameter	Definition	Kinematic	Kinodynamic
ω_p	Position weight coefficient in closeness metric	0.6	0.4
ω_h	Orientation weight coefficient in closeness metric	0.4	0.3
ω_v	Linear velocity weight coefficient in closeness metric	N/A	0.2
ω_ω	Angular velocity coefficient in closeness metric	N/A	0.1
Δt	Camera tree extension time step (sec.)	2.0	1.0
δ_t	Camera tree integration time step (sec.)	0.04	0.04
v_{max}	Maximum linear velocity extension input (m/sec.)	0.1	N/A
ω_{max}	Maximum angular velocity extension input (deg./sec.)	5	N/A
f_{max}	Maximum force extension input (N)	N/A	0.2
τ_{max}	Maximum torque extension input (N.m)	N/A	0.2
t_g	Greedy extension time (sec.)	5	5

Table 3.1: List of parameters used in our proposed planning framework along with their typical values we used in our experiments.

Chapter 4

Case Study I: 6-DOF Robotic Arm

In the previous chapter, we presented our proposed planning strategy to plan feasible camera trajectories which are then projected into image space to obtain the desired feature trajectories. In this chapter, we consider a 6-DOF robotic arm and explain the execution of the planned feature trajectories using an image-based visual servoing technique. We present real-world experimental results to show the merit of the proposed planning framework and its effectiveness in accounting for various image and physical constraints in complex visual servoing scenarios. We also provide an empirical comparison between the proposed image-based trajectory tracking controller and a joint space trajectory tracking controller and show the effect of modeling and calibration uncertainties on the performance of each controller.

4.1 Implementation

The software/system architecture of our proposed framework is shown in Fig. 4.1. The proposed randomized kinodynamic planning approach has been implemented using our in-house motion planning kernel (MPK) [34]. The obstacles are imported as known CAD models into our planning environment. The MPK uses V/I-Collide library to perform collision checking given the objects CAD models. Please note that the type of environment representation does not affect the planning scheme given that proper constraint checker (e.g., collision and occlusion) are in place for the specific representation of the environment. For instance, in a future extension of our work, we can easily imagine that the environment model can be acquired using perception techniques based on stereo/depth sensors. We used the camera factory settings (i.e., intrinsic parameters) with no further calibration.

The initial and desired camera poses are computed using the Dementhon's technique

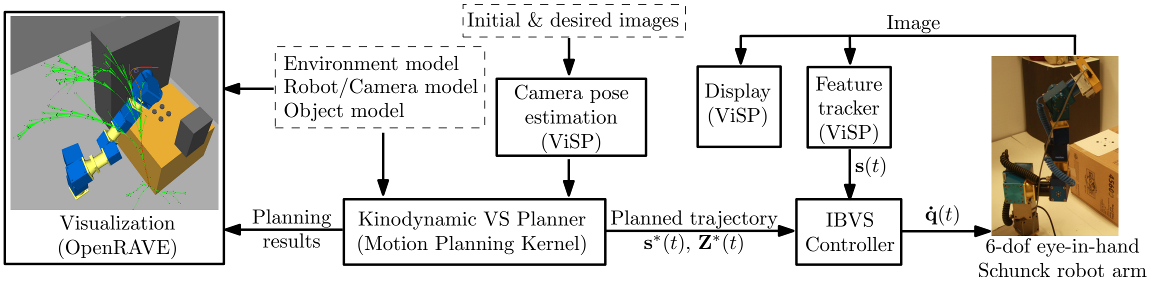


Figure 4.1: Software/system architecture of our proposed framework

[26] and the virtual visual servoing approach provided through the visual servoing platform (ViSP) [67]. For visualization purposes the planning results including the planned robot/camera trajectory and the camera tree are sent to an OpenRAVE [28] viewer plug-in where the robot trajectory is visualized/animated. The planned feature trajectories are sent to an IBVS controller which generates appropriate robot/camera velocity inputs to our 6-DOF *Schunck* robotic arm with an in-hand camera. We employed the dot feature tracker in ViSP to keep tracking of multiple black markers on the target object. The IBVS controller is explained in more details as follows.

4.1.1 Tracking Feature Trajectories using IBVS

We adopt an image-based trajectory tracking controller as in [69] to track the desired feature trajectories, $\mathbf{s}^*(t)$; the following error function is defined in the image space

$$\mathbf{e}_s = \mathbf{s}(\mathbf{r}(\mathbf{q}(t))) - \mathbf{s}^*(t), \tag{4.1}$$

where $\mathbf{s}(\mathbf{r}(\mathbf{q}(t)))$ is the vector of current image features at the current camera pose $\mathbf{r}$ specified by the robot configuration $\mathbf{q}(t)$. Taking the derivative of (4.1) with respect to time, we have

$$\begin{aligned} \dot{\mathbf{e}}_s &= \frac{\partial \mathbf{s}}{\partial \mathbf{r}}\frac{\partial \mathbf{r}}{\partial \mathbf{q}}\dot{\mathbf{q}} - \dot{\mathbf{s}}^* \\ &= \mathbf{L}\mathbf{J}\dot{\mathbf{q}} - \dot{\mathbf{s}}^*, \end{aligned} \tag{4.2}$$

where $\mathbf{L} = \frac{\partial \mathbf{s}}{\partial \mathbf{r}}$ denotes the image Jacobian and $\mathbf{J} = \frac{\partial \mathbf{r}}{\partial \mathbf{q}}$ is the robot Jacobian. To impose exponential decay of $\mathbf{e}_s(t)$ to zero, we let $\dot{\mathbf{e}}_s(t) = -\lambda_s \mathbf{e}_s$, where λ_s is a positive gain. Hence, we have

$$\dot{\mathbf{q}} = \mathbf{J}^+\mathbf{L}^+(-\lambda_s \mathbf{e}_s + \dot{\mathbf{s}}^*), \tag{4.3}$$

where $\dot{\mathbf{q}}$ denotes the joints velocity input, and $\mathbf{J}^+$ and $\mathbf{L}^+$ are the pseudo-inverses of $\mathbf{J}$ and $\mathbf{L}$, respectively. For a 3-D point $\mathcal{P}_j$ with coordinates $[X_j\ Y_j\ Z_j]^T$ in the camera coordinate frame and with image pixel coordinates $\mathbf{s}_j = [u_j \ v_j]^T$ where $[u_j \ v_j \ 1]^T = A[x_j \ y_j \ 1]^T$, the interaction matrix related to $\mathbf{s}_j$ is given as

$$L(\mathbf{s}_j, Z_j) = \begin{bmatrix} fp_u & 0 \\ 0 & fp_v \end{bmatrix} \begin{bmatrix} -\frac{1}{Z_j} & 0 & \frac{x_j}{Z_j} & x_j y_j & -1 + x_j^2 & y_j \\ 0 & -\frac{1}{Z_j} & \frac{y_j}{Z_j} & 1 + y_j^2 & -x_j y_j & -x_j \end{bmatrix}$$

where $x_j = (u_j - u_0)/fp_u$ and $y_j = (v_j - v_0)/fp_v$. The interaction matrix for n feature points $\mathbf{s} = [\mathbf{s}_1^T \dots \mathbf{s}_n^T]^T$ with corresponding depths $\mathbf{Z} = [Z_1 \dots Z_n]^T$ can then be created by stacking up the interaction matrices of all features as

$$\mathbf{L}(\mathbf{s}, \mathbf{Z}) = \Big[L^T(\mathbf{s}_1, Z_1) \dots L^T(\mathbf{s}_n, Z_n) \Big]^T \tag{4.4}$$

As it is seen the interaction matrix depends on the depth information which is not usually available at the execution time. Hence, a nominal value is used instead [11]. Similarly to [69] we use $\widehat{\mathbf{L}} = \mathbf{L}(\mathbf{s}^*(t), \mathbf{Z}^*(t))$ as a nominal model of the interaction matrix which is created by stacking up the interaction matrices related to all the current desired image features along the planned trajectory. Moreover, the precise model of the robot kinematic may not be achievable in practice. Hence, a nominal robot Jacobian $\widehat{\mathbf{J}}$ is used in the formulation of the control law.

Finally, the control law can be written as

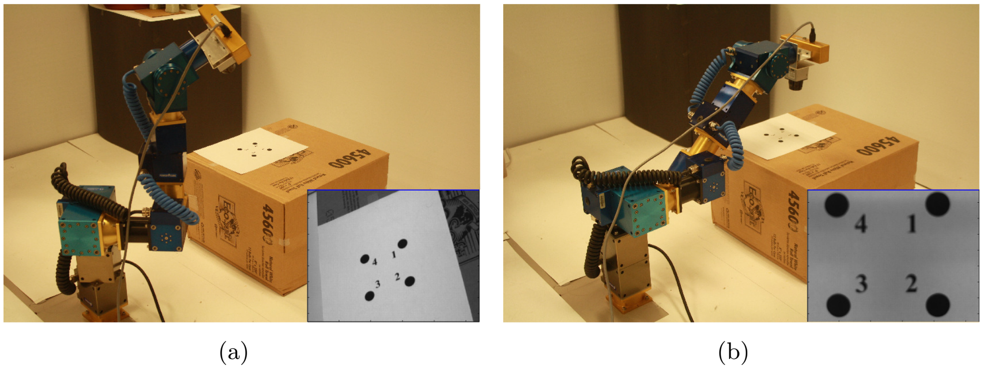
$$\dot{\mathbf{q}} = \widehat{\mathbf{J}}^{+} \widehat{\mathbf{L}}^{+} (-\lambda_s \mathbf{e}_s + \dot{\mathbf{s}}^*). \tag{4.5}$$

In the following section, first we present results of our experiments demonstrating the effectiveness of the proposed planning framework in accounting for image and physical constraints in visual servoing tasks.

4.2 Experimental Results

We validated the effectiveness of the proposed planning framework on a 6-DOF eye-in-hand robotic arm. The target object is composed of black dots as shown in Fig. 4.2. For ease of image processing, black dots on white background were used as features; but one could easily substitute this step with more general features such as lines and contours as long as real-time tracking is available for features of interest.

The focus of these experiments is to demonstrate the capabilities of the proposed approach in taking physical and image constraints into account. To better visualize and illustrate the effectiveness of our approach, we first show how it is approachable to take care



(a) (b)

Figure 4.2: Experimental setup: field of view and joint limits avoidance experiment

of field of view and joint limit constraints, and then show how it takes care of occlusion and obstacle avoidance constraints in a cluttered workspace. We contrast our results with an IBVS method in which the control law is given as

$$\dot{\mathbf{q}} = -\lambda_s \widehat{\mathbf{J}}^{+} \widehat{\mathbf{L}}^{+} (\mathbf{s}(t) - \mathbf{s}^{*}), \tag{4.6}$$

where $\mathbf{s}^{*} = \mathbf{s}^{*}(t_f)$ contains the fixed final desired features and the interaction matrix $\widehat{\mathbf{L}}$ is computed at $(\mathbf{s}^{*}, \mathbf{Z}^{*})$.

In all experiments we used the camera intrinsic parameters provided by the manufacture with no further calibration. The video clips of these experiments are available on-line at `https://sites.google.com/site/moslemk/research/kinodynamic-planning-for-vs`

4.2.1 Field of view and joint limits avoidance

The robot at the initial and desired configurations is shown in Fig. 4.2(a) and Fig. 4.2(b) along with the corresponding initial and desired images on their right-bottom corners. The corresponding camera displacement between the initial and desired camera frames is $[20\ 35\ 21]^T$ centimeters in x-y-z translation and $[34\ 26\ 8]^T$ degrees in roll-pitch-yaw rotation. Figure 4.3 shows the results obtained using the IBVS technique based on the control law in (4.6). As is seen in Fig. 4.3(a), some of the feature trajectories (features 1 and 4 specifically) leave the camera's field of view. Figure 4.3(b) shows the corresponding joint trajectories. All joint values have been normalized to $[-1, 1]$ where 1 and -1 determine the upper and lower limits for each joint, respectively. As shown in Fig. 4.3(b), the trajectories for joints 2 and 3 violate their limits. Either of the above violations results in failure of visual servoing task.

We applied our proposed approach to the same task and the results are presented in Fig. 4.4. Using our proposed approach we planned feasible feature trajectories in the image space as shown in Fig. 4.4(a). These trajectories do not violate the camera field of view and robot joint limits which have been taken into account at the planning stage. The robot has been servo controlled along the planned feature trajectories using the tracking control law given in (4.3). As shown in Fig. 4.4(b), the followed feature trajectories respect the camera field of view and remain within the image limits. Figure 4.4(e) shows the corresponding joint trajectories which are within their respective limits. The planned camera tree (in green) along with the camera trajectory (in red) are shown in Fig. 4.4(h). The solution trajectory was found in about 80 seconds after about 1500 states were successfully explored. Please note that the transient tracking errors in image space (4.4(c)) are initially large (up to 30 pixels) primarily due to image measurement errors in segmenting the centers of 2D features and slow tracking control loop (12Hz) due to the limited frame rate (15Hz) and delays in processing the image. Supported by our empirical results presented in Section 4.3, and also existing literature [69] we believe that a faster vision system ($\approx$ 50Hz, as studied in [20]) will highly improve the tracking performance in the image space.

4.2.2 Obstacle collision and occlusion avoidance

In the following experiments the robot workspace is cluttered with obstacles (i.e., boxes) which make some parts of the workspace unreachable by the robot (see Fig.4.5). The obstacles also create occluded regions in the workspace from which the target object is occluded and not visible in the camera image. The robot at its initial and desired configurations along with the corresponding images taken by the camera are shown in Fig.4.5.

First, we employed the pure IBVS technique to carry out the task of servoing the robot towards its desired configuration (see Fig. 4.6). The robot trajectory induced by the IBVS technique results in both occlusion of the target object (see Fig.4.6 left-bottom) and robot collision with workspace obstacles (see Fig. 4.6). Hence, the IBVS technique fails to accomplish this task and was stopped. Most of the previous approaches on path planning for visual servoing (as reviewed in Chapter 2 and [43]) either fail to incorporate general occlusion and collision avoidance constraints or are susceptible to local minima in such complex environments.

Our proposed randomized planning framework can, however, effectively take care of these constraints and, hence, ensures robust execution of the servoing task with respect to the violation of image and physical constraints. The planned camera tree along with the camera trajectory have been visualized in Fig. 4.7(h). The solution trajectory was found in about 200 seconds after about 900 states were successfully explored. The corresponding feature trajectories, obtained by projecting the planned camera trajectory into the image space, are shown in Fig. 4.7. The robot, when servo controlled along the desired feature trajectories, reaches its desired configuration avoiding occlusion of any of the object features while avoiding collision with workspace obstacles. The feature trajectories followed by the servoing control law (4.3) are shown in Fig. 4.7(b). The corresponding planned and followed robot joint trajectories are shown in Fig. 4.7(d) and Fig. 4.7(e), respectively.

As is apparent from Fig.4.7(f) the joint tracking error of up to about 9 degrees is somewhat more than desired and a systematic way to do this would be via a new controller that regulates errors in both image and joint spaces, indeed a challenging task and future work.

4.3 Discussion

As mentioned earlier, in the current work the planning is performed without accounting for uncertainties in modeling and calibration. Although we assume that the system is nominally calibrated and we have access to robot model and camera parameters provided by the manufacture, the effect of uncertainty may not be fully removed and the tracking performance is highly affected as noticed through the results presented in previous section. We derive the error dynamics in image and joint space, both for an IBVS controller (controller I) and for a joint space controller (controller II). However, a definitive general statement thorough analytical study of parameters' uncertainty and their effects on errors in image and joint spaces is not apparent from these expressions. Therefore, we also provide empirical results on the effects of uncertainty on tracking errors in image and joint spaces for both controllers and draw some empirical conclusions discussing the relative merits of the two controllers.

4.3.1 Image Space Trajectory Tracking (Controller I)

Error analysis in image space: replacing (4.5) in (4.2) we obtain the image error dynamics:

$$\dot{\mathbf{e}}_s = -\lambda_s \mathbf{L}\mathbf{J}\widehat{\mathbf{J}}^{+}\widehat{\mathbf{L}}^{+}\mathbf{e}_s + (\mathbf{L}\mathbf{J}\widehat{\mathbf{J}}^{+}\widehat{\mathbf{L}}^{+} - \mathbf{I})\dot{\mathbf{s}}^{*}. \tag{4.7}$$

The error $\mathbf{e}_s$ will have an exponential decay if the term $\mathbf{L}\mathbf{J}\widehat{\mathbf{J}}^{+}\widehat{\mathbf{L}}^{+}$ remains positive definite along the trajectory which can be ensured locally as long as the camera is controlled closely along the desired feature trajectories. However, the error will converge to zero only if either $\mathbf{L}\mathbf{J}\widehat{\mathbf{J}}^{+}\widehat{\mathbf{L}}^{+} = \mathbf{I}$ or $\dot{\mathbf{s}}^{*} = 0$ which nullify the last term in (4.7). As also studied by Malis [62], setting a high gain λ_s or reducing the desired velocity $\dot{\mathbf{s}}^*$ will reduce the tracking error. We point out that, ensured by our kinodynamic planning approach, the desired velocity $\dot{\mathbf{s}}^*$ approaches zero as the camera reaches its final desired pose $\mathbf{x}_f$, reducing the tracking error towards the end of the trajectory. Malis and Rives [65] have also studied the stability of Controller I and showed that it is fairly stable in presence of camera calibration errors, but with a smaller stability domain when coarse features depth distribution is used.

Error analysis in joint space: the tracking error in joint space is defined as

$$\mathbf{e}_q(t) = \mathbf{q}(t) - \mathbf{q}^*(t) \tag{4.8}$$

and we have

$$\dot{\mathbf{e}}_q(t) = \dot{\mathbf{q}}(t) - \dot{\mathbf{q}}^*(t) \tag{4.9}$$

Inserting the commanded velocities (4.5) in the above, we obtain

$$\dot{\mathbf{e}}_q(t) = \widehat{\mathbf{J}}^+\widehat{\mathbf{L}}^+(-\lambda_s\mathbf{e}_s + \dot{\mathbf{s}}^*) - \dot{\mathbf{q}}^*(t) \tag{4.10}$$

, and knowing $\dot{\mathbf{q}}^*(t) = \widehat{\mathbf{J}}^+\widehat{\mathbf{L}}^+\dot{\mathbf{s}}^*$ we have

$$\dot{\mathbf{e}}_q(t) = -\lambda_s\widehat{\mathbf{J}}^+\widehat{\mathbf{L}}^+\mathbf{e}_s \tag{4.11}$$

From (4.7) we obtain

$$-\lambda_s\widehat{\mathbf{J}}^+\widehat{\mathbf{L}}^+\mathbf{e}_s = (\mathbf{L}\mathbf{J})^+\left[\dot{\mathbf{e}}_s - (\mathbf{L}\mathbf{J}\widehat{\mathbf{J}}^+\widehat{\mathbf{L}}^+ - \mathbf{I})\dot{\mathbf{s}}^*\right] \tag{4.12}$$

which away from singularities in $\mathbf{J}$ and $\mathbf{L}$ can be simplified as

$$-\lambda_s\widehat{\mathbf{J}}^+\widehat{\mathbf{L}}^+\mathbf{e}_s = \mathbf{J}^+\mathbf{L}^+\dot{\mathbf{e}}_s - \mathbf{J}^+\mathbf{L}^+(\mathbf{L}\mathbf{J}\widehat{\mathbf{J}}^+\widehat{\mathbf{L}}^+ - \mathbf{I})\dot{\mathbf{s}}^* \tag{4.13}$$

Replacing the result in (4.11) we obtain the relationship between error dynamics in joint and image spaces as

$$\dot{\mathbf{e}}_q = \mathbf{J}^+\mathbf{L}^+\dot{\mathbf{e}}_s + \mathbf{J}^+\mathbf{L}^+(\mathbf{L}\mathbf{J}\widehat{\mathbf{J}}^+\widehat{\mathbf{L}}^+ - \mathbf{I})\dot{\mathbf{s}}^* \tag{4.14}$$

The first term in (4.14) relates the dynamics between the image and joint spaces in ideal conditions. But, in practice the second term is added and introduces disturbance to joint space error dynamics, which in turn may reduce the tracking performance in joint space; Although, as imposed by our proposed kinodynamic planning approach, the desired feature trajectories velocities approach zero towards the end of the trajectory, $\dot{\mathbf{s}}^*(t) \to 0$ as $t \to t_f$, tracking errors in joint space will be observed along the trajectory because of second term in (4.14). Moreover, the rate of change of the tracking error is proportional to the desired tracking velocity $\dot{\mathbf{s}}^*$ and is also affected by the singularities in robot Jacobian and image interaction matrices. Our proposed planning framework yields trajectories away from singularities in both robot Jacobian and image interaction matrices. Obviously, reducing the tracking velocity helps to reduce the tracking error in joint space. Unfortunately, a full analysis of uncertainties in the robot Jacobian and image Interaction matrices and their effects on the performance of the proposed tracking algorithm remains to be out of reach and requires further study.

4.3.2 Joint Space Trajectory Tracking (Controller II)

We start by deriving the joint trajectory tracking control law with the objective of tracking a desired (planned) trajectory $\mathbf{q}^*(t)$. The error function in joint space is defined as

$$\mathbf{e}_q(t) = \mathbf{q}(t) - \mathbf{q}^*(t) \tag{4.15}$$

and we obtain

$$\dot{\mathbf{e}}_q = \dot{\mathbf{q}}(t) - \dot{\mathbf{q}}^*(t) \tag{4.16}$$

To ensure exponential decay of the error to zero, we impose $\dot{\mathbf{e}}_q = -\lambda_q \mathbf{e}_q$ in the above equation which yields the control law

$$\dot{\mathbf{q}} = -\lambda_q \mathbf{e}_q + \dot{\mathbf{q}}^* \tag{4.17}$$

Error analysis in joint space: by closing the loop, i.e. replacing (4.17) in (4.16), the control law in (4.17) guarantees exponential convergence of error to zero in joint space, i.e.

$$\dot{\mathbf{e}}_q = -\lambda_q \mathbf{e}_q \tag{4.18}$$

Error analysis in image space: now, let us look at the error dynamics in the image space under the control law given in (4.17). The error in the image space is defined as in (4.1) with the dynamics given in (4.2). Solving (4.16) for $\dot{\mathbf{q}}$ and replacing the result in (4.2), we obtain a relationship between the error dynamics in image and joint spaces as

$$\dot{\mathbf{e}}_s = \mathbf{LJ}(\dot{\mathbf{e}}_q + \dot{\mathbf{q}}^*) - \dot{\mathbf{s}}^* \tag{4.19}$$

Knowing $\dot{\mathbf{q}}^* = \widehat{\mathbf{J}}^+\widehat{\mathbf{L}}^+\dot{\mathbf{s}}^*$, we obtain a relationship between the error dynamics in image and joint spaces as

$$\dot{\mathbf{e}}_s = \mathbf{LJ}\dot{\mathbf{e}}_q + (\mathbf{LJ}\widehat{\mathbf{J}}^+\widehat{\mathbf{L}}^+ - \mathbf{I})\dot{\mathbf{s}}^* \tag{4.20}$$

Given the above relationship, the error dynamics in joint space would translate to the error dynamics in image space through the term $\mathbf{LJ}$, i.e. the actual robot-image Jacobian. Hence, we would expect similar trends for error dynamics in both spaces if $\mathbf{LJ}$ remains positive definite, i.e., decays (or growth) of the error in image space leads to decays (or growth) of the error in joint space. However, the second term in (4.20) introduces an additional error to the error dynamics in joint space. This additional term comes from the discrepancy between the actual and nominal values for the image-robot Jacobian. Clearly, under ideal conditions there is no such error.

Table 4.1 summarizes the image and joint trajectory tracking controllers given in (4.5) and (4.17), respectively, along with their corresponding error dynamics in each space.

	Controller I: image trajectory tracking $\dot{\mathbf{q}} = \widehat{\mathbf{J}}^{+}\widehat{\mathbf{L}}^{+}(-\lambda_s \mathbf{e}_s + \dot{\mathbf{s}}^*)$	Controller II: joint trajectory tracking $\dot{\mathbf{q}} = -\lambda_q \mathbf{e}_q + \dot{\mathbf{q}}^*$
Image err dyn	$\dot{\mathbf{e}}_s = -\lambda_s \mathbf{LJ}\widehat{\mathbf{J}}^{+}\widehat{\mathbf{L}}^{+}\mathbf{e}_s + (\mathbf{LJ}\widehat{\mathbf{J}}^{+}\widehat{\mathbf{L}}^{+} - \mathbf{I})\dot{\mathbf{s}}^*$	$\dot{\mathbf{e}}_s = \mathbf{LJ}\dot{\mathbf{e}}_q + (\mathbf{LJ}\widehat{\mathbf{J}}^{+}\widehat{\mathbf{L}}^{+} - \mathbf{I})\dot{\mathbf{s}}^*$
Joint err dyn	$\dot{\mathbf{e}}_q = \mathbf{J}^{+}\mathbf{L}^{+}\dot{\mathbf{e}}_s + \mathbf{J}^{+}\mathbf{L}^{+}(\mathbf{LJ}\widehat{\mathbf{J}}^{+}\widehat{\mathbf{L}}^{+} - \mathbf{I})\dot{\mathbf{s}}^*$	$\dot{\mathbf{e}}_q = -\lambda_q \mathbf{e}_q$

Table 4.1: Error dynamics in image and joint spaces for two controllers: image and joint space trajectory tracking

4.3.3 Empirical study of calibration and modeling uncertainties

We provide an empirical comparison between the performance of the two controllers in Table 4.1 under various conditions: uncertainties in camera focal length and robot kinematics, and image measurement errors. We consider a simple planning example for sake of simulations, where the camera trajectory is defined as a 90 degrees rotation around its optical axis. The target consists the vertices of a square with the optical axis of the camera passing through its center and orthogonal to its plane. The camera is assumed to be mounted at the end-effector of a 6-DOF robotic arm (the one we used for our experiments in previous section) in a way that the 90 degree rotation of the camera can be achieved by rotating joint 6 only. The desired feature and joint trajectories are shown in Fig.4.8. The total trajectory running time is 30 seconds. We use the controllers in Table 4.1 to execute these trajectories and compare their performance. Both controllers are simulated at 50Hz and have access to joint encoder data and image measurements at the same rate.

Ideal conditions: First, as a reference, we simulate an ideal condition, i.e., no calibration/modeling uncertainty and away from singularities in robot-image Jacobian. Under such ideal conditions both controllers demonstrate fairly similar performances as shown through the tracking errors in both image and joint spaces in Fig.4.9. These results serve as a reference to compare the performance of the controllers in Table 4.1 under the following conditions simulated individually:

- 5% error in camera focal length (Fig.4.10)
- up to 5 pixels (random) image measurement error (Fig.4.11)
- 1 cm position error and 1 degree orientation error in robot forward kinematic model (Fig.4.12)

Summary of results: comparing the tracking errors obtained as the result of using Controller I (i.e., image trajectory tracking) with those yielded through Controller II (i.e., joint trajectory tracking) a somewhat expected trend is observed for all uncertainties. These results do confirm our intuitive expectation that neither the image trajectory tracking nor the joint trajectory tracking can achieve low errors in both image and joint spaces. However, relatively speaking, the image trajectory tracking performs well (i.e., yielding an exponential decay of tracking error to zero) in image space (see subfigures (a) in Fig.4.10-4.12) while the joint trajectories converge to within about 5 degree (i.e., not too large) steady state error (see subfigures (b) in Fig.4.10-4.12). Joint trajectory tracking yields an exponential decay of tracking error to zero in joint space (see subfigures (d) in Fig.4.10-4.12) but with poor performance in image space (see subfigures (c) in Fig.4.10-4.12), particularly in presence of forward kinematic errors (very large steady state image errors of up to about 50 pixels). This lends some support to our decision to adopt an image space trajectory tracking controller.

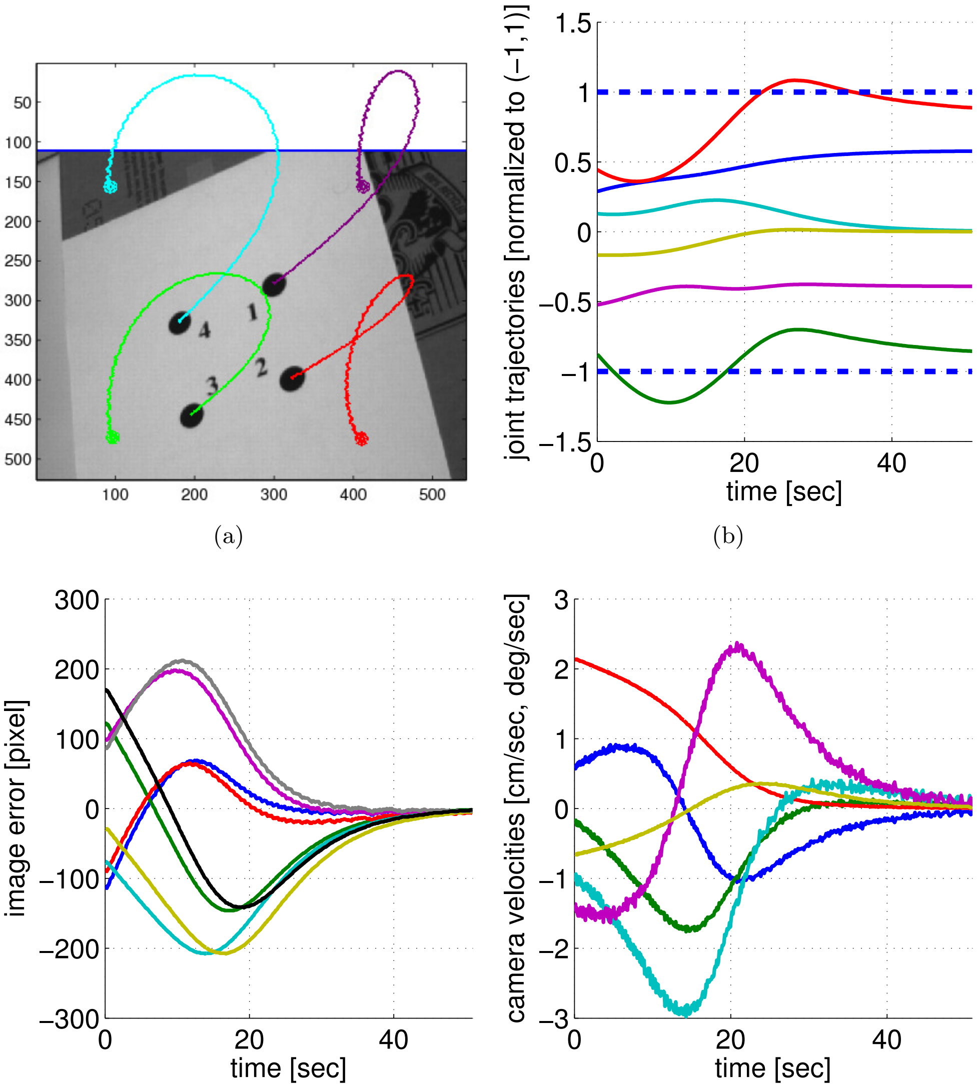
We also note that the answer to the question - which controller should be used to execute the planned trajectories - also depends on the task. That is, for example, if the task objective requires a close tracking of feature trajectories within the image while the error in joint space can be tolerated then the image trajectory tracking controller should be adopted. For example, in MEMS micro-manipulation applications close tracking of micro-parts trajectories is very crucial to avoid occlusion of micro-parts by the gripper or loosing them from the field of view. Hence, an image-based trajectory tracking through out the whole trajectory is advisable (e.g., as adopted in [87]).

Our current work does not account for calibration, modeling errors as well as sensing and control errors that may result in deviations from planned (joint space) paths during execution - such as the observed joint space tracking errors in our simulations and in experiments. Such deviations may result in violation of, for example, physical constraints (joint limits and collisions). However, we argue that it is indeed possible to extend the planning framework to incorporate such errors. We outline such an approach.

In order to account for such joint errors at planning stage, we need to know the error bounds as a function of uncertainty parameters. In the absence of any analytical results on these bounds so far, the key observation is that the joint space tracking errors can be estimated via a simulated image-based trajectory tracking step. More precisely, after each successful extension of the camera tree, the robot is simulated to move along the (locally) extended image trajectory using the same controller as will be used in the actual execution (say, Controller I, as explained above) with appropriate uncertainties on kinematic model and/or camera parameters and thus determining the maximum deviation of joints from the planned joint trajectory. Once this error bound is known, we can easily determine the volume swept by the robot corresponding to the error bound (see the orientation slicing method, Section 5.1, page 283 in [55]) and augment the robot size by this swept volume. The RRT extension is then re-checked for collisions with the augmented robot and discarded if there is collision, and kept otherwise.

However, one should note that the problem, in general, may not be convex, i.e., extreme levels of uncertainty in parameters may not correspond to extreme joint tracking errors. This makes determining joint error bounds while accounting for all uncertainties computationally intensive. For instance, if there were n uncertainty parameters, each discretized to k quantization levels, one would need to run the simulated control k^n times, and hence determine the global bounds for the joint errors. An approximate and faster alternative would be to consider only a subset of most influential parameters, e.g. camera intrinsic parameters such as camera focal length.

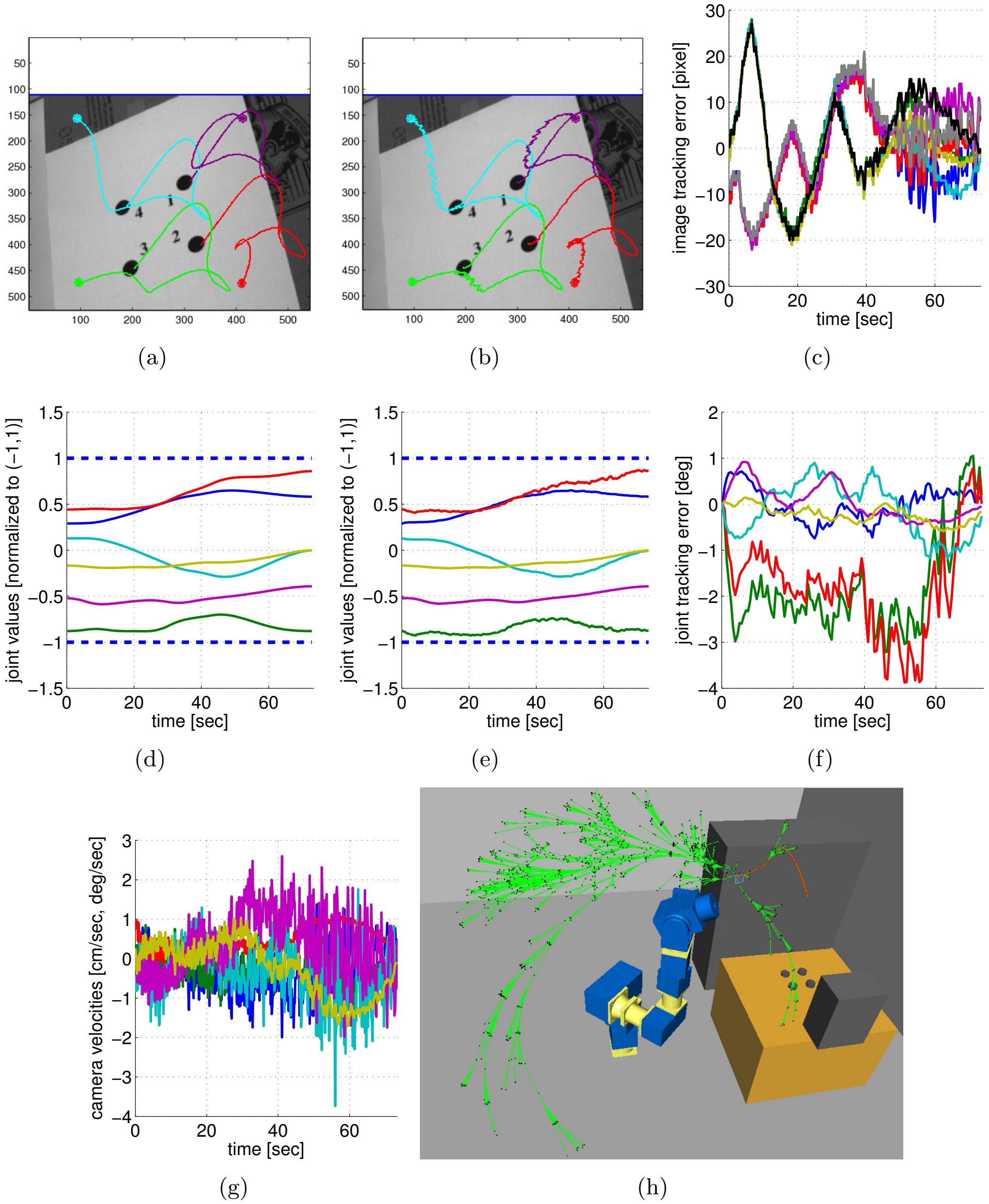
We carried out some simulations allowing uncertainty in the camera focal length only. Figure 4.13 shows the joint space tracking errors obtained for servoing the robot along the image trajectories in Fig.4.8(a) under different amounts of uncertainty in focal length. As shown, the joint tracking errors monotonically increase (or decrease) as the focal length error increases (or decreases), i.e., extremes of the joint errors occur at extreme values of uncertainty. This convexity property can be utilized towards providing general guidelines for selecting the joint error bounds at the planning stage. Nonetheless, we point out that this property may not always hold especially when including more than one source of uncertainty, hence, one needs to calculate more conservative bounds by simulating over the entire parameters uncertainty space.



(a) (b)

(c) (d)

Figure 4.3: Results obtained using pure IBVS: as shown in (a) the feature trajectories induced by the IBVS violate the virtual limits imposed on the camera field of view, and as seen in (b) the joint trajectories violate the limits imposed on two robot joints.

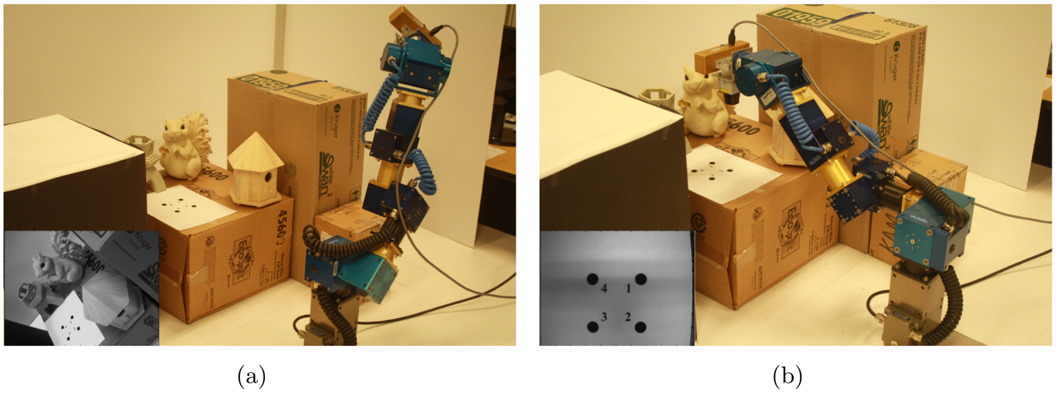


(a) (b) (c)

(d) (e) (f)

(g) (h)

Figure 4.4: Results obtained using our proposed kinodynamic planning approach: as shown in (b) the followed feature trajectories respect the field of view limits and as seen in (e) the corresponding joint trajectories respect the joint limits. The planned camera tree (in green) along the planned trajectory (in red) are shown in (h).



(a)

(b)

Figure 4.5: Experimental setup for the collision and occlusion avoidance experiment: in this experiment the robot has to be servo controlled towards its desired configuration located in narrow empty space between the obstacles which cause occlusion of the target and/or collision with the robot.

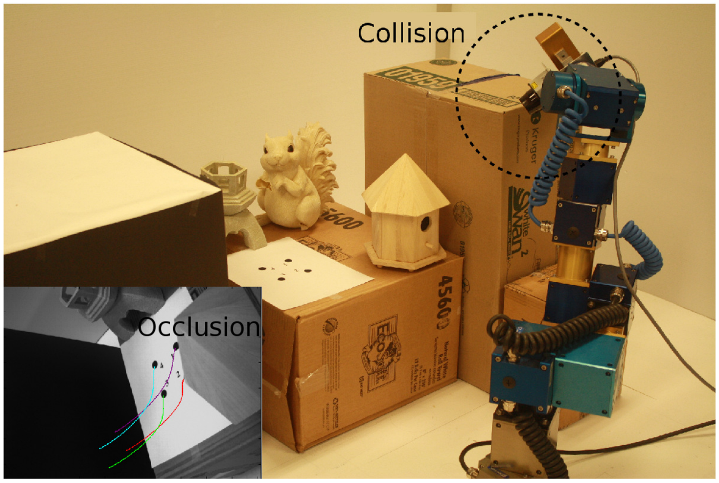
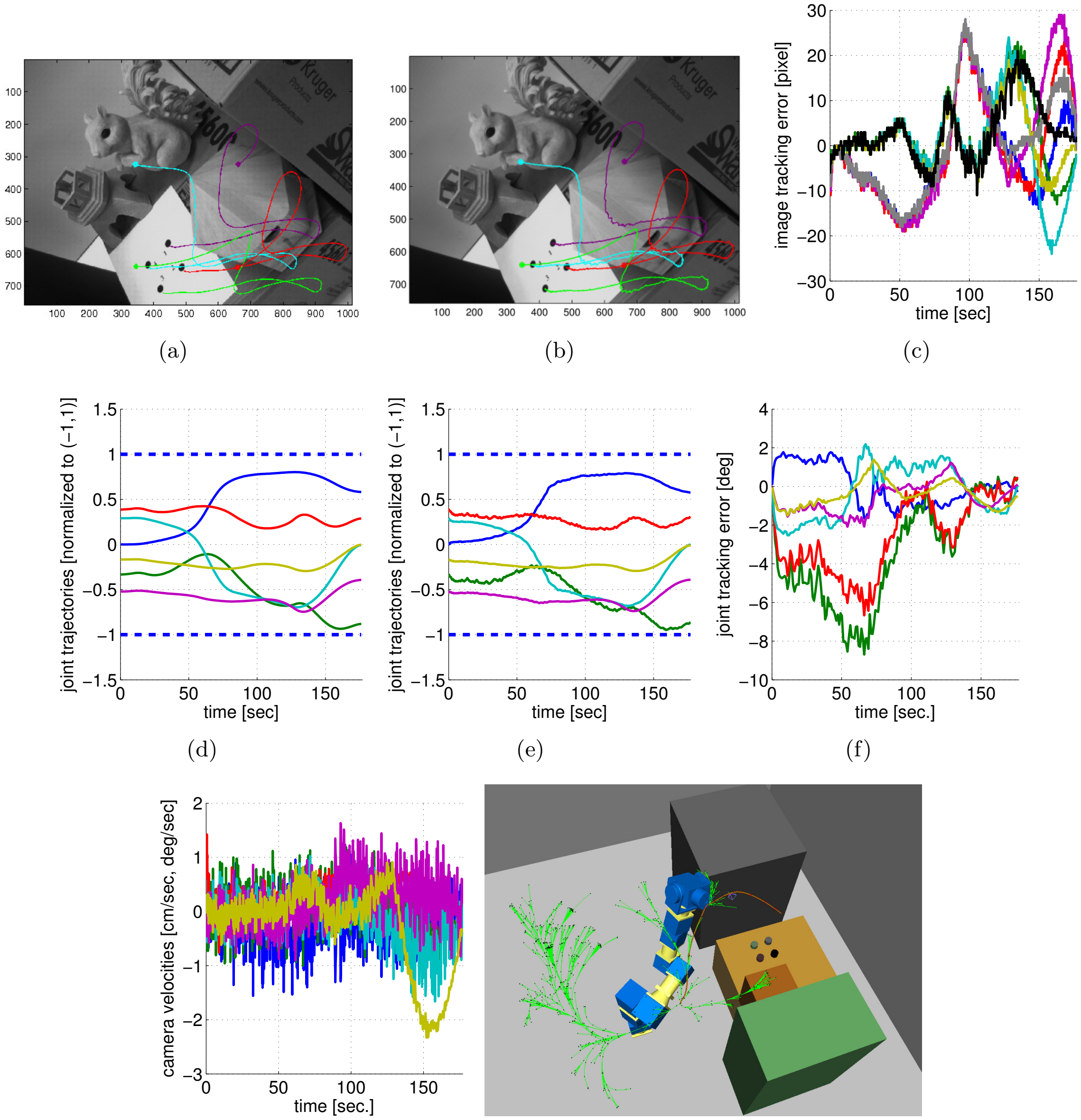


Figure 4.6: Results obtained using pure IBVS: the trajectories induced by the IBVS technique results in occlusion of some of the target features by the workspace obstacles (see the image at the bottom-left corner), and collision of the robot with workspace obstacles as shown at the top-right corner. Hence, the servoing task fails.

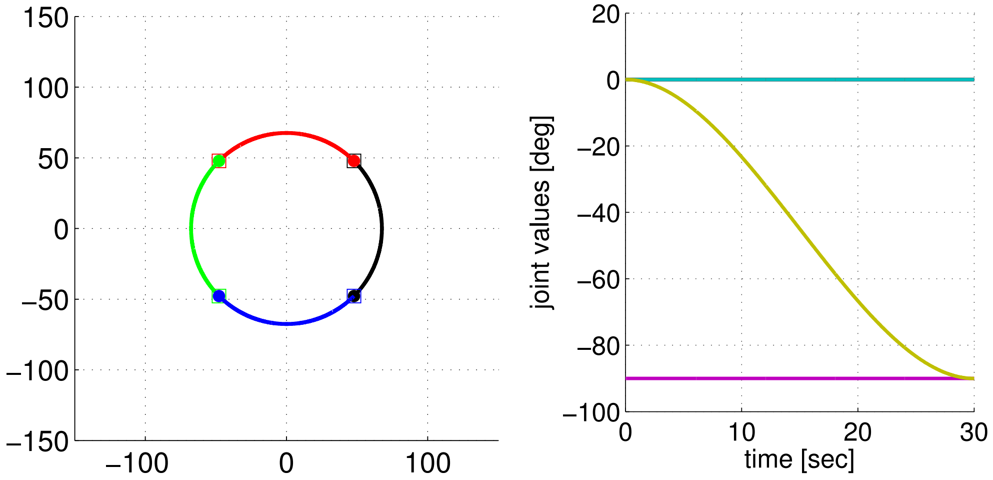


(a) (b) (c)

(d) (e) (f)

(g) (h)

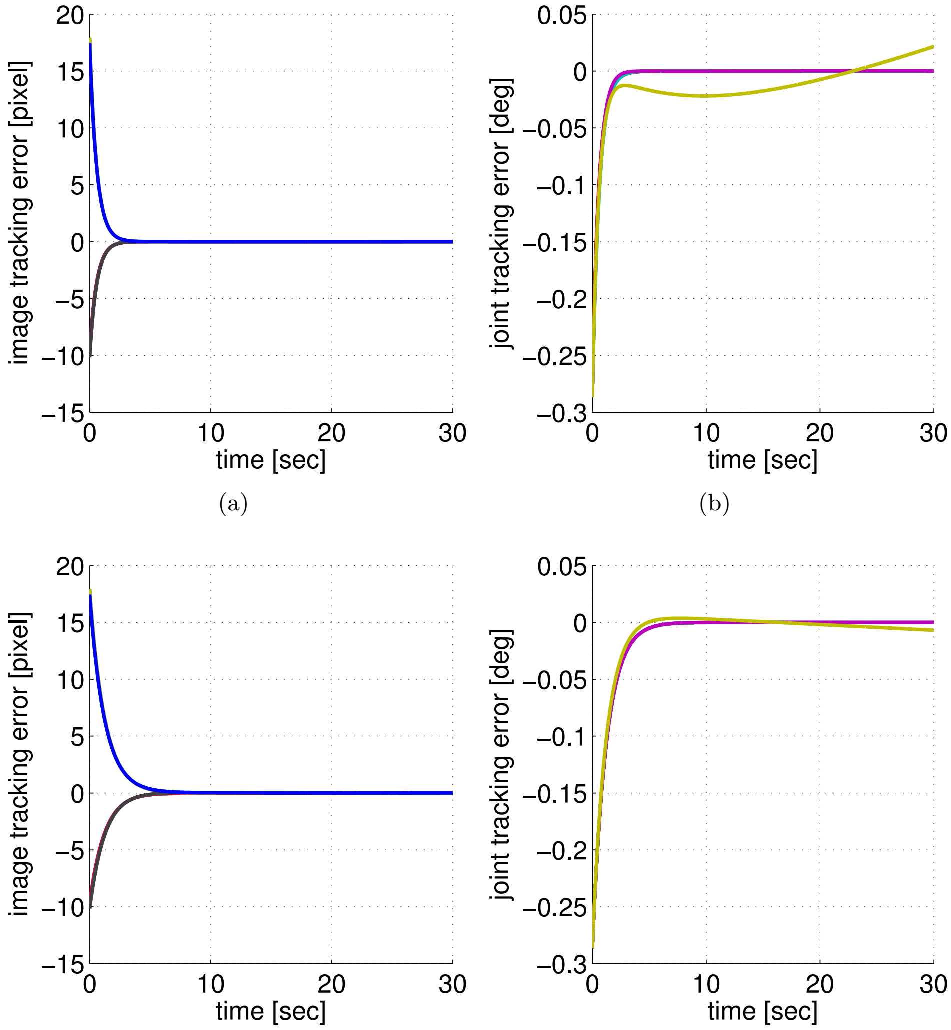
Figure 4.7: Results obtained using the proposed kinodynamic planning approach: the planned camera tree (in green) along the planned trajectory (in red) are shown in (a). The robot (servo controlled along the planned feature trajectories) moves towards its desired configuration located in the narrow empty space between the obstacles while avoiding collision with obstacles and keeping target in the field of view. So, the servoing task has been performed successfully.



(a)

(b)

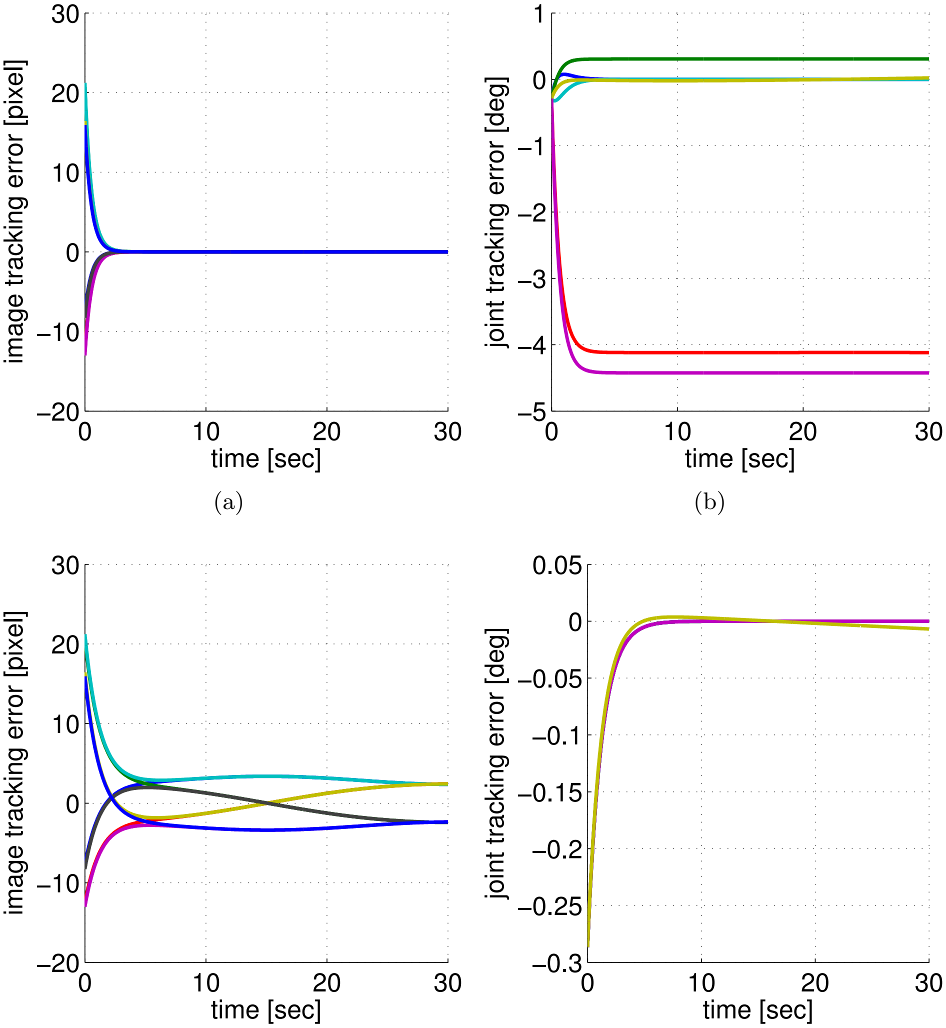
Figure 4.8: (a) The desired trajectories pertinent to four point features: each feature moves on a circular arc about 90 degrees around the camera optical center (b) the desired joint trajectories related to the feature trajectories in (a)



(a) (b)

(c) (d)

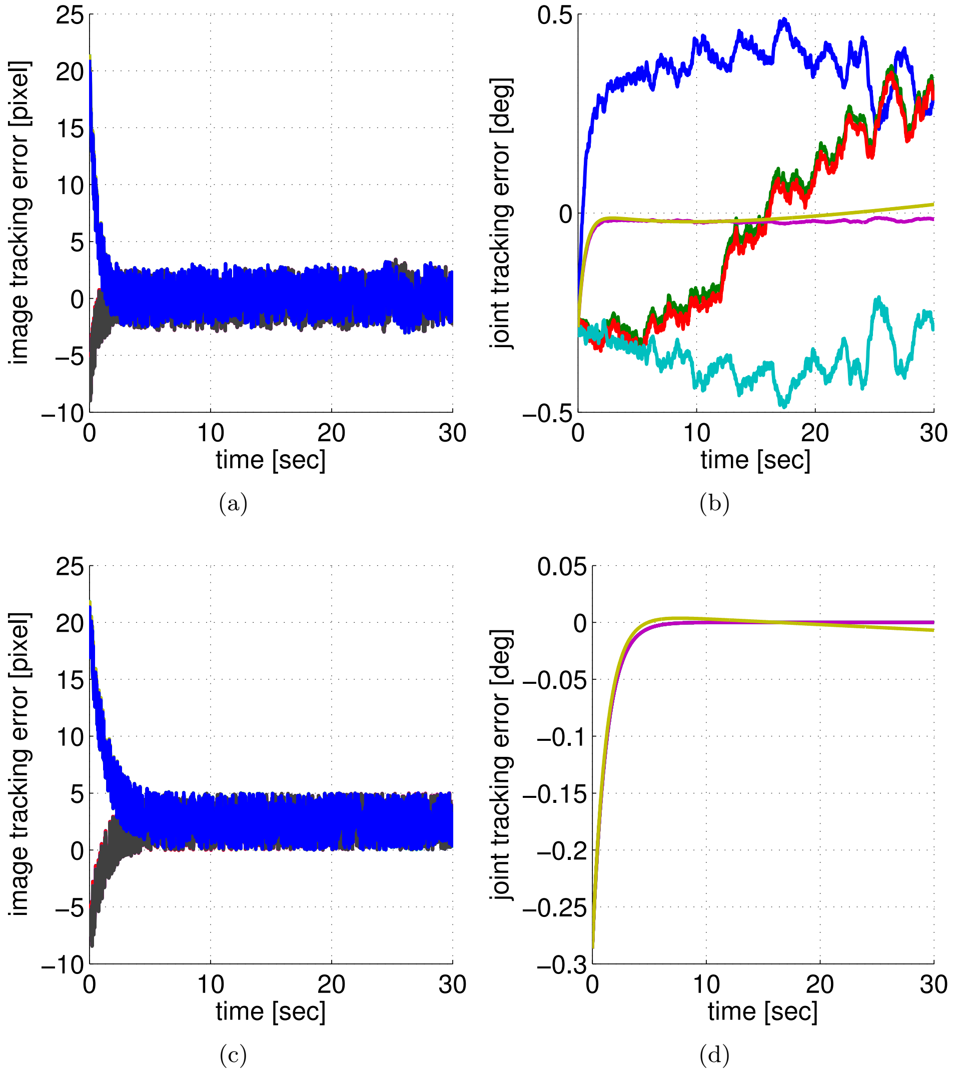
Figure 4.9: Tracking errors in image and joint spaces under ideal conditions, i.e., no calibration/modeling uncertainties and away from singularities in robot-image Jacobian: (a)(b) image trajectory tracking, and (c)(d) joint trajectory tracking



(a) (b)

(c) (d)

Figure 4.10: Tracking errors in image and joint spaces in presence of 5% error in camera focal length: (a)(b) image trajectory tracking, and (c)(d) joint trajectory tracking



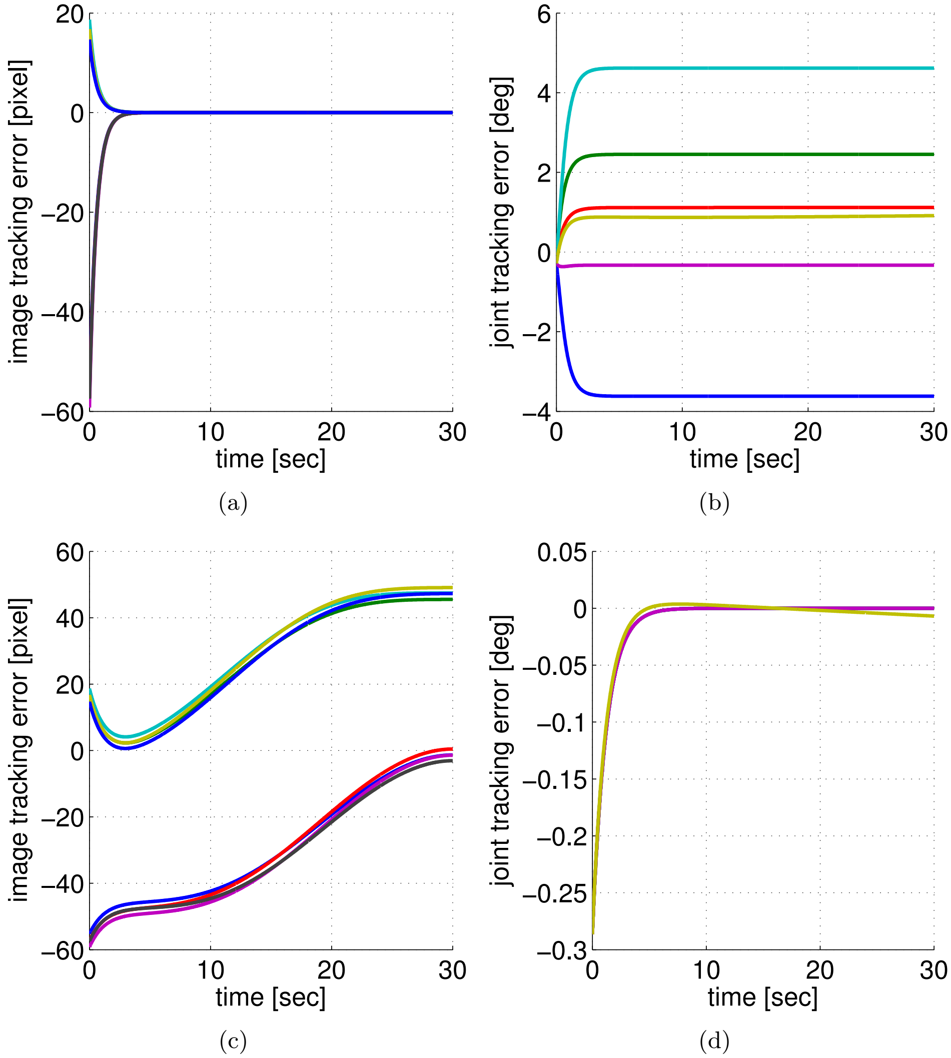
(a)

(b)

(c)

(d)

Figure 4.11: Tracking errors in image and joint spaces in presence of up to 5 pixels (random) image measurement errors: (a)(b) image trajectory tracking, and (c)(d) joint trajectory tracking



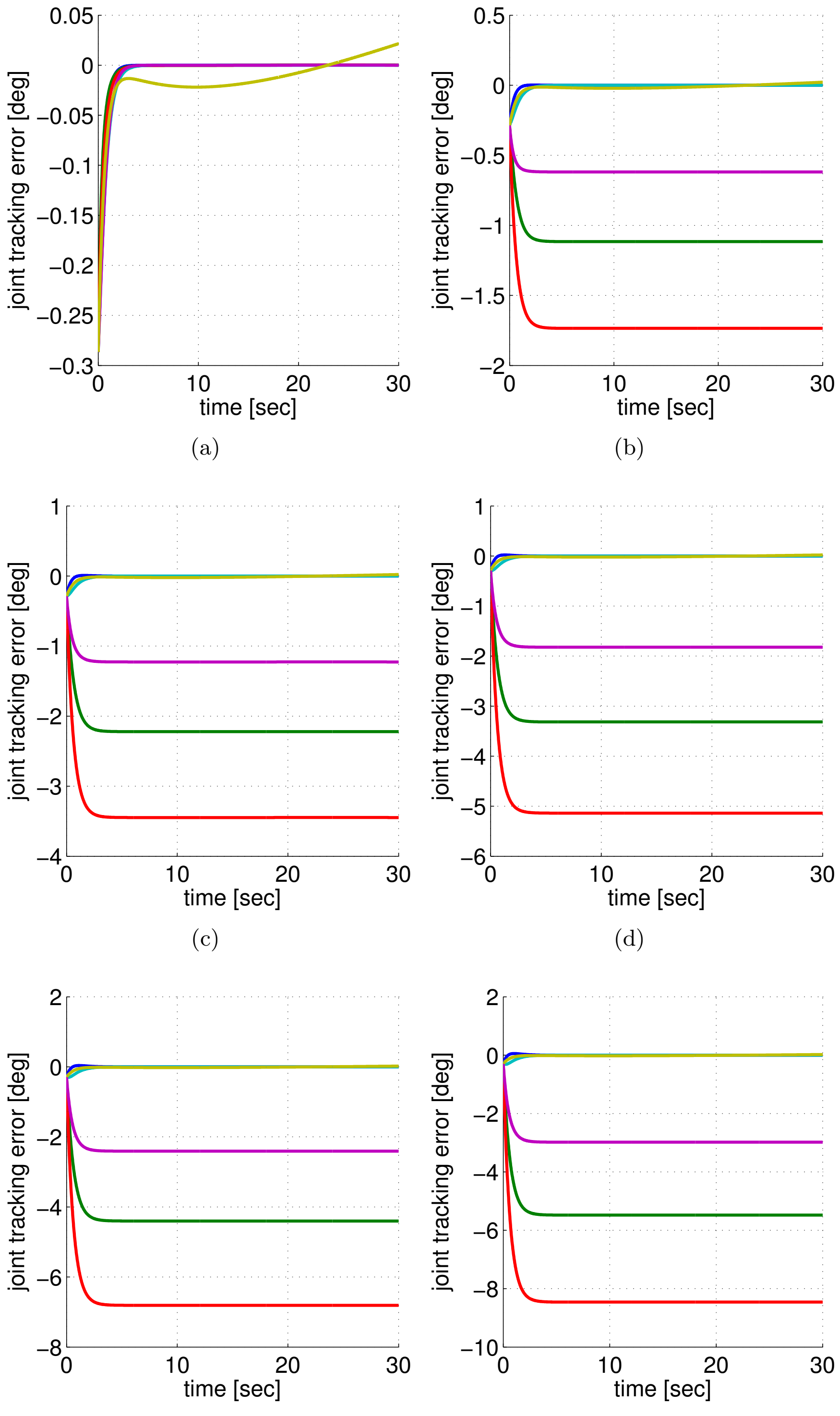
(a)

(b)

(c)

(d)

Figure 4.12: Tracking errors in image and joint spaces in presence of 1 cm/deg error in robot forward kinematic model: (a)(b) image trajectory tracking, and (c)(d) joint trajectory tracking



(a) (b)

(c) (d)

(e) (f)

Figure 4.13: Joint tracking errors obtained by applying an image-based trajectory tracking along the image trajectories in Fig.4.8(a) under (a) 0%, (b) 1%, (c) 2%, (d) 3%, (e) 4%, and (f) 5% error on focal length. As shown, the joint tracking errors monotonically increase (or decrease) as the focal length error increases (or decreases).

Chapter 5

Case Study II: 9-DOF Wheeled Mobile Manipulator

Wheeled Mobile Manipulators (WMM) are major efforts to bring both mobility and manipulation capabilities to human environments (Figure 5.1). To move autonomously and accomplish tasks robustly in complex environments high-level global motion planning techniques should be closely integrated with sensor-based control of such systems. Many efforts have been devoted to both motion planning and sensor-based control with promising advances in each individual area over the past decades. However, the integration of planning and control, in particular for complex systems such as a WMM, remains a challenging topic, and also crucial towards fully autonomous and robust solutions.

Despite many efforts on image-based control of robotic arms and mobile robots, there are very few works extending classical image-based control techniques to mobile manipulators. Wheeled mobile manipulators introduce two main challenges: non-holonomic kinematic constraints at the mobile base, and task redundancy. In [77] an inverse kinematic control is applied to servo control a mobile hand-eye system by deriving a Jacobian for the whole system as a kinematic chain. The work in [24] benefited from the task redundancy through a task sequencing strategy to avoid kinematic singularities during the servo control of a WMM. Due to the complexity of WMM systems accounting for image and physical constraints becomes even more challenging compared to robotic arms.

In this chapter, we extend our efforts on incorporating randomized path planning techniques with image-based control of robotic arms ([41][42]) to wheeled mobile manipulators. To the best of our knowledge, this is the first effort toward integrating path planning with image-based control of wheeled mobile manipulators.

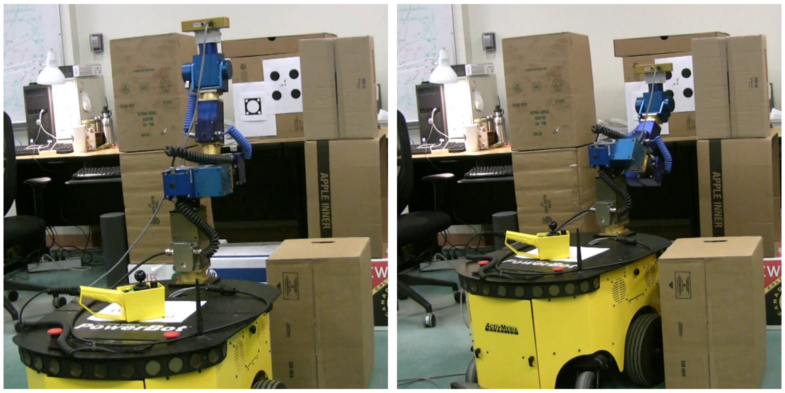


Figure 5.1: SFU wheeled mobile manipulator system (a Powerbot mobile platform with an on-board 6-DOF Schunck robotic arm) reaches a desired location by tracking a target object.

We consider a visually-guided task [48], i.e., to move the robot from an initial to a desired location with respect to a stationary target while keeping the target in the field of view of the in-hand camera and avoiding occlusions/collisions (due to obstacles) without violating the kinematic constraints of the robot (Figure 5.1).

5.1 Path Planning for Image-based Control of Wheeled Mobile Manipulators

The two-step plan-then-execute methodology presented in Chapter 3 is also applied here: first, we plan a feasible trajectory for the whole WMM system. The planned trajectory is then projected into the image space to obtain the desired feature trajectories which are then executed using a closed-loop feedback strategy.

5.1.1 Path planning with image and physical constraints

We employ the planning framework presented in Chapter 3 to plan feasible trajectories for the whole WMM (as a kinematic chain) under image and physical constraints to accomplish the above visually-guided task. The camera tree extension follows the kinodynamic extension in camera state space (as explained in Chapter 3). The local paths obtained as the result of this extension are projected into the image space to check for image constraints, i.e. field of view limits and occlusions of target object by other obstacles or itself. Given that the local camera paths violate no image constraints, they are then tracked in the WMM configuration space (C-space) using a local planner to check for physical constraints, i.e. collision with obstacles and joint limits. All the above steps follow the implementation explained in Chapter 3. However, to exploit the redundancy of the WMM system we employ weighted pseudo-inverse Jacobian solutions combined with a null space optimization technique to effectively coordinate the motion of the mobile platform and the arm for tracking the camera local paths in C-space.

The above planning stage yields a feasible trajectory for the whole WMM system from which one may extract and generate individual trajectories for the mobile platform, onboard robotic arm, in-hand camera, and image features. The planned trajectories are then executed using a decoupled feedback control scheme as explained below.

5.1.2 Decoupled feedback control scheme

The planned trajectories could then be executed in different ways. An ideal and challenging way would be to design a controller that reconciles errors in both C-space and image space and is a future research topic. More simply, one could execute the WMM trajectory directly in the WMM C-space. However, deviation may occur due to uncertainties in the motion of the mobile platform. Using a feedback control strategy would help to reduce the mobile platform tracking error. Nonetheless, even small tracking errors of the mobile platform may result in large deviation of the camera at the end-effector, hence loosing the target in camera field of view. This suggests the need for closing the loop within the image space to account for potential tracking errors of the mobile platform.

Here we propose a decoupled feedback control strategy to move the WMM along the above desired trajectories; we utilize an image-based control technique to servo control the robotic arm along the desired feature trajectories, while a state feedback control is employed to move the mobile platform along its desired trajectory in Cartesian space. These two controllers run simultaneously in two separate threads to move the whole WMM along the planned trajectories.

This chapter presents the preliminary results we obtained by planning feasible trajectories for the SFU 9-DOF WMM system with an in-hand camera to accomplish a visually-guided task (Figure 5.1). We also show the results of executing the planned trajectories on the real system using the above decoupled feedback control scheme.

First, we present the WMM kinematic modeling which is used to derive and implement the planning and control schemes.

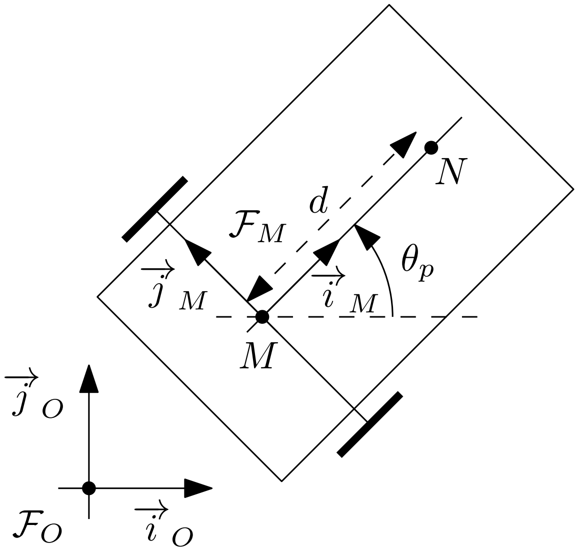


Figure 5.2: Differential-drive mobile platform

5.2 Kinematic Modeling

5.2.1 Mobile Platform Kinematics

We assume that the mobile platform moves on a planar surface (Fig. 5.2); and it has two actuated wheels which can roll independently with fixed headings. We make the rolling-without-slipping assumption for the wheels, i.e., the contact point of each wheel with the ground has zero velocity which introduces a non-holonomic constraint in the kinematic equations of the motion of the platform.

We follow the representation proposed by Samson and Ait-Abderrahim [80] in which the mobile platform position (x_p and y_p) is represented as the components of the vector $\overrightarrow{NO}$ in the basis of mobile frame $\mathcal{F}_M$ attached to the platform, i.e. $\overrightarrow{NO} = x_p \overrightarrow{i}_M + y_p \overrightarrow{j}_M$. The point N is located at a distance d from the wheels axis such that $\overrightarrow{MN} = d\,\overrightarrow{i}_M$ and point O is the origin of the fixed frame $\mathcal{F}_O$. The platform orientation θ_p is chosen as the angle between $\overrightarrow{i}_O$ and $\overrightarrow{i}_M$. Hence, the platform configuration is given as $\mathbf{q}_p = [x_p, y_p, \theta_p]^T$. The above representation is well adapted to the strategy we adopted from [80] to track the mobile platform along its planned trajectory in Cartesian space (see Section 5.4.2).

Considering the mobile platform as a kinematic system, we assume that the pseudo velocity $\mathbf{u}_p = [v_p \ \omega_p]^T$ where v_p is the linear velocity along $\overrightarrow{i}_M$ and $\omega_p = \dot{\theta}_p$, can be taken as the mobile platform control input.

The state equation of the mobile platform is then given as [80]

$$\dot{\mathbf{q}}_p = G_p(\mathbf{q}_p)\mathbf{u}_p \tag{5.1}$$

with

$$G_p(\mathbf{q}_p) = \begin{bmatrix} -1 & y_p \\ 0 & -(d + x_p) \\ 0 & 1 \end{bmatrix} \tag{5.2}$$

5.2.2 Robotic Arm Kinematics

The C-space of the n-DOF robotic arm can be parameterized as $\mathbf{q}_a = [q_{a_1}\ q_{a_2}\ \dots\ q_{a_n}]^T$ where $q_{a_1}, q_{a_2}, \dots, q_{a_n}$ are the individual joint values. The arm is a kinematically unconstrained, i.e. holonomic, system. Hence, the joint velocities $\dot{\mathbf{q}}_a$ can be independently specified at any configuration. So, the arm control input vector $\mathbf{u}_a$ can be set as

$$\mathbf{u}_a = \dot{\mathbf{q}}_a \tag{5.3}$$

The pose of the arm's (or equivalently, WMM's) end-effector is represented as a 7-D vector

$$\boldsymbol{\xi}_a = [\mathbf{p}_\xi^T\ \mathbf{h}_\xi^T]^T \tag{5.4}$$

where $\mathbf{p}_\xi$ is a 3-D vector representing the end-effector position and $\mathbf{h}_\xi$ is a 4-D unit quaternion representing the end-effector orientation.

5.2.3 Mobile Manipulator Kinematics

The whole mobile manipulator C-space can be parameterized in terms of the configurations of both the mobile platform and on-board robotic arm, i.e.

$$\mathbf{q} = \begin{bmatrix} \mathbf{q}_p \\ \mathbf{q}_a \end{bmatrix} \tag{5.5}$$

The kinematic model of the mobile manipulator sets the location of its end-effector (equivalently, the end-effector of the on-board robotic arm) as a function of its configuration, i.e.

$$\boldsymbol{\xi} = \boldsymbol{\xi}_a = \mathbf{F}(\mathbf{q}_p, \mathbf{q}_a) \tag{5.6}$$

By differentiating (5.6) with respect to time and using (5.2) we have

$$\begin{aligned}\dot{\boldsymbol{\xi}} &= \begin{bmatrix} \frac{\partial \mathbf{F}(\mathbf{q}_p,\mathbf{q}_a)}{\partial \mathbf{q}_p} G(\mathbf{q}_p) & \frac{\partial \mathbf{F}(\mathbf{q}_p,\mathbf{q}_a)}{\partial \mathbf{q}_a} \end{bmatrix} \begin{bmatrix} \mathbf{u}_p \\ \mathbf{u}_a \end{bmatrix} \\ &= \mathbf{J}(\mathbf{q}_p, \mathbf{q}_a)\mathbf{u} \qquad (5.7)\end{aligned}$$

where $\mathbf{u} = [\mathbf{u}_p^T \ \mathbf{u}_a^T]^T$ denotes the mobile manipulator control input which is related to C-space (generalized) velocities as

$$\dot{\mathbf{q}} = \begin{bmatrix} G_p(\mathbf{q}_p) & \mathbf{0}_{3\times n} \\ \mathbf{0}_{n\times 2} & \mathbf{I}_{n\times n} \end{bmatrix} \begin{bmatrix} \mathbf{u}_p \\ \mathbf{u}_a \end{bmatrix} = G(\mathbf{q}_p)\mathbf{u} \qquad (5.8)$$

and to the end-effector velocity screw through the mobile manipulator Jacobian $\mathbf{J}(\mathbf{q}_p, \mathbf{q}_a)$.

5.3 Camera Trajectory Tracking in C-space

As explained before, we employ the same planning approach explained in Chapter 3 to plan feasible trajectories for the whole WMM system. However, to exploit the redundancy of the system and coordinate the motion of the mobile platform and the on-board arm, we extend the local controller used to track the camera trajectories in the WMM C-space. We employ an *inverse kinematic control* in which kinematic tracking control laws are designed using the differential kinematic model in (5.7). This step extends our work in [42] on robotic arms by coordinating the motion between the mobile platform and the arm through a weighted pseudo-inverse Jacobian strategy, and performing null-space optimization to utilize the task redundancy of the WMM.

Without loosing generality, we assume that the camera frame coincides with the robotic arm end-effector which is effectively the WMM end-effector. Now, given an end-effector trajectory $\boldsymbol{\xi}^*(t)$ (extracted from a camera local trajectory), the problem is to find control input $\mathbf{u}(t)$, such that

$$\dot{\boldsymbol{\xi}}^*(t) = \mathbf{J}\mathbf{u}(t) \qquad (5.9)$$

as in (5.7), which asymptotically stabilizes the task error $\mathbf{e}(t) = \boldsymbol{\xi}(t) - \boldsymbol{\xi}^*(t)$. All the exact (least-squares) solutions which satisfy the end-effector task constraint (5.9) are given as

$$\mathbf{u}(t) = \mathbf{J}^+\dot{\boldsymbol{\xi}}^*(t) + (\mathbf{I} - \mathbf{J}^+\mathbf{J})\mathbf{z}(t) \qquad (5.10)$$

in which $\mathbf{J}^+$ is the pseudo-inverse of $\mathbf{J}$, the term $\mathbf{I}-\mathbf{J}^+\mathbf{J}$ represents the orthogonal projection matrix in the null space of $\mathbf{J}$, and $\mathbf{z}(t)$ is an arbitrary input velocity vector. The second term of the solution is therefore a null space velocity which does not affect the task at the end-effector (in Section 5.3.2 we will explain how the task redundancy is used to increase the manipulability of the robotic arm along the planned trajectories). This solution minimizes the Euclidean norm $\|\dot{\xi}^* - \mathbf{J}\mathbf{u}\|$.

Recalling $\dot{\mathbf{q}}(t) = G(\mathbf{q}_p)\mathbf{u}(t)$ from (5.1) we have

$$\dot{\mathbf{q}}(t) = G\mathbf{J}^+\dot{\boldsymbol{\xi}}^*(t) + G(\mathbf{I} - \mathbf{J}^+\mathbf{J})\mathbf{z}(t) \tag{5.11}$$

The corresponding C-space trajectory $\mathbf{q}^*(t)$ can then be reconstructed using a discrete time integration technique, e.g., an Euler forward integration as

$$\mathbf{q}^*(t + \Delta t) = \mathbf{q}^*(t) + \dot{\mathbf{q}}(t)\Delta t \tag{5.12}$$

where Δt is the time step. To avoid drifting due to the numerical integration errors, and hence to asymptotically stabilize the error $\mathbf{e}(t)$, a closed-loop inverse kinematic solution is chosen instead, i.e.

$$\dot{\mathbf{q}} = G\mathbf{J}^+\left[\dot{\boldsymbol{\xi}}^* - K_\xi(\boldsymbol{\xi} - \boldsymbol{\xi}^*)\right] + G(\mathbf{I} - \mathbf{J}^+\mathbf{J})\mathbf{z} \tag{5.13}$$

where K_ξ is a constant positive-definite gain matrix.

The C-space trajectory $\mathbf{q}^*(t)$ is then checked for collision between the WMM body and the obstacles, as well as joint limits of the arm. Given that $\mathbf{q}^*(t)$ is collision free and does not violate any joint limits, the camera trajectory $\boldsymbol{\xi}(t)$ is added to the tree along with its end-point as a new node.

5.3.1 Motion Coordination via Weighted Pseudo-Inverse Jacobian:

The components of $\mathbf{u}(t)$ in (5.9) have different physical dimensions, i.e., it is composed of two linear velocity and one angular velocity components for the mobile platform and six angular velocity components of the arm joints. So, it would be necessary to evaluate the magnitude of the solutions obtained using (5.13) based on appropriate weighting of the components. For example, we prefer a weighting strategy in which the mobile platform has a relatively bigger contribution in translational motions comparing to the robotic arm. This can be done by obtaining weighted-norm solutions of (5.9) using a weighted pseudo-inverse Jacobian formulation [74]. Let

$$\mathbf{J}^\# = \mathbf{J}Q_0^{-1}$$

where Q_0 is the square root of a symmetric and positive definite weighting matrix Q, i.e., $Q = Q_0^T Q_0$. Then, all the Q-weighted-norm solutions of (5.9) are given as

$$\mathbf{u}(t) = Q_0^{-1}\mathbf{J}^{\#^+}\dot{\boldsymbol{\xi}}^*(t) + Q_0^{-1}(\mathbf{I} - \mathbf{J}^{\#^+}\mathbf{J}^\#)\mathbf{z}(t) \tag{5.14}$$

Correspondingly, (5.13) can be written as

$$\dot{\mathbf{q}} = GQ_0^{-1}\mathbf{J}^{\#^+}\left[\dot{\boldsymbol{\xi}}^* - K_\xi(\boldsymbol{\xi} - \boldsymbol{\xi}^*)\right] + GQ_0^{-1}(\mathbf{I} - \mathbf{J}^{\#^+}\mathbf{J}^{\#})\mathbf{z} \tag{5.15}$$

The weights in Q can be either modified dynamically (on-line) based on some proper measures, or set fixed. We have chosen the latter approach in which we choose relatively smaller weights for the mobile platform linear velocities, which in turn favors the motion of the mobile platform over the arm's in translation movements.

5.3.2 Improving Manipulability via Null Space Optimization:

The redundancy of the WMM with respect to the task at the end-effector (i.e., end-effector trajectory tracking) can be used to improve the value of (configuration-dependent) scalar performance criteria $H(\mathbf{q})$, such as distance from singular configurations, joint limits, and/or obstacles. This can be achieved by a gradient descent method in which the potential function $H(\mathbf{q})$ has its minimum value corresponding to the objective requirements. We can write the time variation of the $H(\mathbf{q}(t))$ as

$$\dot{H} = \nabla^T H(\mathbf{q})\dot{\mathbf{q}}, \tag{5.16}$$

and considering only the motion due to null space velocity term in (5.15), we have

$$\dot{H} = \nabla^T H(\mathbf{q}) GQ_0^{-1}(\mathbf{I} - \mathbf{J}^{\#^+}\mathbf{J}^{\#})\mathbf{z} \tag{5.17}$$

In order to decrease H, we need to ensure $\dot{H} \leq 0$. One choice for $\mathbf{z}$ can then be

$$\mathbf{z} = -k_H\left[\nabla^T H(\mathbf{q}) GQ_0^{-1}(\mathbf{I} - \mathbf{J}^{\#^+}\mathbf{J}^{\#})\right]^T \tag{5.18}$$

where k_H is a positive scalar.

In our proposed planning framework, we use the problem redundancy to increase the *manipulability* of the robotic arm only (not that of the whole WMM) through the above local optimization strategy. The rational behind choosing this objective (and not others mentioned above) goes back to our proposed decoupled control strategy in which the robotic arm is servo controlled along the feature trajectories using a closed-loop (image-based) inverse kinematic control; clearly, maintaining good manipulability for the robotic arm will improve the performance of such controller which strongly relies on the Jacobian inverse. It is also noteworthy that through some trajectories the robotic arm manipulability may be poor whereas the whole WMM system keeps a good measure of manipulability, i.e., improving the latter does not necessary improves the former (see [5]).

In its general form the manipulability measure is defined as $\mu = \sqrt{\left|\mathbf{J}\mathbf{J}^T\right|}$. For a square Jacobian matrix, e.g., the case of 6-DOF robotic arm here, it is equivalent to the arm Jacobian determinant $\mu_a = \left|\mathbf{J}_a\right|$. Letting $H = -\mu_a(\mathbf{q}_a)$ in (5.18) will favor those null space motions leading to trajectories (or configurations) with good manipulability for the robotic arm. However, it must be noted that due to the local optimization nature of the above scheme, one cannot guarantee its singularity avoidance and its performance may degrade over long trajectories. Hence, the trajectory obtained through (5.15) needs to be checked for singularities.

5.4 Decoupled Trajectory Tracking Scheme

The proposed path planning explained above yields a WMM trajectory $\mathbf{q}^*(t)$, which in turn can be decomposed into the corresponding mobile platform trajectory $\mathbf{q}_p^*(t)$, robotic arm trajectory $\mathbf{q}_a^*(t)$, camera trajectory $\boldsymbol{\xi}^*(t)$, and finally image features trajectories $\mathbf{s}^*(t)$ pertinent to the object features, i.e.,

$$\mathbf{q}^*(t) \rightarrow \mathbf{q}_p^*(t),\ \mathbf{q}_a^*(t),\ \boldsymbol{\xi}^*(t),\ \mathbf{s}^*(t) \tag{5.19}$$

We propose a preliminary version of a decoupled feedback control strategy to move the WMM along the above desired trajectories; we utilize an image-based control technique to servo control the robotic arm along the desired feature trajectories $\mathbf{s}^*(t)$, while a state feedback control is employed to move the mobile platform along its desired trajectory $\mathbf{q}_p^*(t)$ in Cartesian space. These two controllers run simultaneously in two separate threads to move the whole WMM along the planned trajectories.

Remark: It must be noted that successful execution of the planned WMM trajectory using the above decoupled control strategy relies on two assumptions: first, the robotic arm and the mobile platform both are precisely controlled along their respective trajectories, and second, the trajectory tracking threads are synchronized. The preliminary experiments presented in this work were obtained without a synchronized tracking strategy which is the subject of our future research.

5.4.1 Image-based control of the arm

In the proposed decoupled control scheme, the on-board arm is controlled along the planned feature trajectories $\mathbf{s}^*(t)$ using the control scheme employed for the stationary arm in the previous chapter. However, the the mobile platform motion contribution in the image must be taken into account. More precisely, the control law to servo control the arm along the planned features trajectories is given as

$$\dot{\mathbf{q}} = \widehat{\mathbf{J}_a}^{-1}\Big(\widehat{\mathbf{L}}^{+}(-\lambda_s\mathbf{e}_s + \dot{\mathbf{s}}^*) - \widehat{\mathbf{J}}\begin{bmatrix}\mathbf{u}_p^* \\ \mathbf{0}\end{bmatrix}\Big) \tag{5.20}$$

where $\mathbf{u}_p^* = [v_p^* \ \omega_p^*]^T$ is the vector of linear and angular velocities of the mobile platform along the planned trajectory, and $\widehat{\mathbf{J}}$ is the mobile manipulator Jacobian.

5.4.2 State feedback control of the mobile platform

We adapt the non-linear state feedback control law devised by [80] for stabilizing the non-holonomic mobile platform to its planned trajectory $\mathbf{q}_p^*(t) = [x_p^*(t) \ y_p^*(t) \ \theta_p^*(t)]^T$ in the Cartesian space. The linear and angular input velocities $(\omega_p(t),\, v_p(t))$ are calculated as [80]

$$\begin{aligned}\omega_p(t) &= \frac{k_3}{k_2}\tilde{\theta} - \frac{k_1}{k_2}h_2(\tilde{y} + d\tilde{\theta}) + \frac{k_6}{k_2}\tilde{x} \\ v_p(t) &= k_3k_5\tilde{x} + (2k_3k_4 + h_1 + k_6)\tilde{\theta} + [(1-k_1)\tilde{y} - k_1d\tilde{\theta}](\omega_p(t) + \dot{\theta}_p^*(t))\end{aligned} \tag{5.21}$$

where

- $\tilde{\theta}_p(t) = \theta_p(t) - \theta_p^*(t)$, $\tilde{x}_p(t) = x_p(t) - x_p^*(t)$, and $\tilde{y}_p(t) = y_p(t) - y_p^*(t)$
- $h_1(\tilde{\theta}, t) = v_p^*(t)\frac{\cos\tilde{\theta}-1}{\tilde{\theta}} + d\theta_p^*(t)\frac{\sin\tilde{\theta}}{\tilde{\theta}}$
- $h_2(\tilde{\theta}, t) = d\theta_p^*(t)\frac{\cos\tilde{\theta}-1}{\tilde{\theta}} - v_p^*(t)\frac{\sin\tilde{\theta}}{\tilde{\theta}}$
- $v_p^*(t)$ is the desired linear velocity along the planned trajectory $\mathbf{q}_p^*(t)$
- $k_1, \ldots, k_6$ are constant gains

Mobile platform state estimation: As a property common to all such control laws, they require a fairly precise estimate of the mobile robot state (or pose) to ensure local stabilization to the desired trajectory. One can use existing localization techniques such as Monte Carlo localization to localize the base. This might require other types of sensors such as laser scanners to be incorporated, and hence increase the complexity of the overall system. Moreover such approaches are relatively slow, and require multiple scans of the scene to be able to provide a fairly precise estimate of the platform pose.

We implemented a state estimation strategy based on the image feedback from the known target object: we first localize the camera with respect to the target object using the technique in [66] where a virtual visual servoing techniques is used to estimate the camera pose. The estimated camera pose is then translated back to the arm base pose, i.e., the mobile platform pose, using the known arm kinematic model and its joint encoder data. The estimated pose is then used in the state feedback control law in (5.21). The above localization process runs in real-time on a separate thread while both the arm and the platform are controlled along their respective trajectories.

The decoupled feedback control scheme proposed above is our first effort to execute the planned trajectories on the mobile manipulator. Other strategies are worth to be compared. For example, one may control the robotic arm directly in the joint space (using, e.g., a simple resolved rate motion controller) along its desired trajectory without closing the loop in the image. However, through this approach the robotic arm is not constrained to track the desired feature trajectories in the image, and even small errors in positioning of the mobile platform will result in deviation of the robotic arm from the desired feature trajectories, and hence loosing the target object in the camera field of view and not being able to localize the platform through the image feedback. Closing the control loop in the image is an interesting advantage of the proposed control strategy which compensates for the possible errors in positioning of the mobile platform, and hence avoids deviation from the desired feature trajectories.

5.5 Preliminary Experiments and Results

We implemented our proposed kinodynamic planning approach and feedback control schemes on a wheeled mobile manipulator system which consists a Powerbot mobile platform with an on-board 6-DOF Schunck robotic arm with a Bumblebee2 camera mounted at its end-effector (see Fig. 5.5). Both the arm and the mobile platform are equipped with low-level velocity controllers which facilitates the implementation of our proposed image and state feedback trajectory tracking controllers.

We used the ViSP visual servoing platform [67] to implement the proposed image-based trajectory tracking for the arm. We used the camera pose estimation using virtual visual servoing [66] provided by ViSP for the real-time pose estimation of the mobile platform as we explained in previous section.

Here we present the results of our first experiments on the WMM system to perform a visually guided task: moving from an initial to a desired location while avoiding image and physical constraints. The initial pose of the mobile platform is estimated using the initial view, and the final configuration of the WMM is implicitly defined by the desired view at the

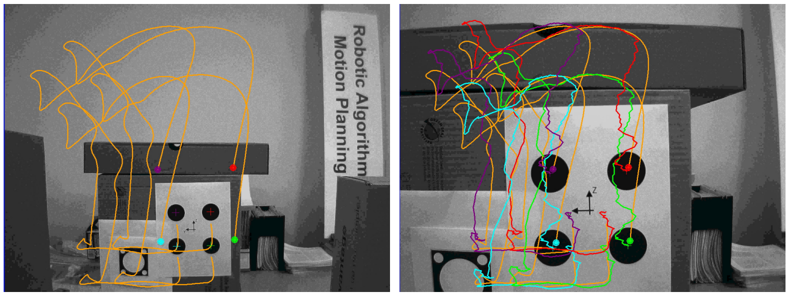


Figure 5.3: Left: initial view and the desired feature trajectories, right: desired final view, and the followed feature trajectories

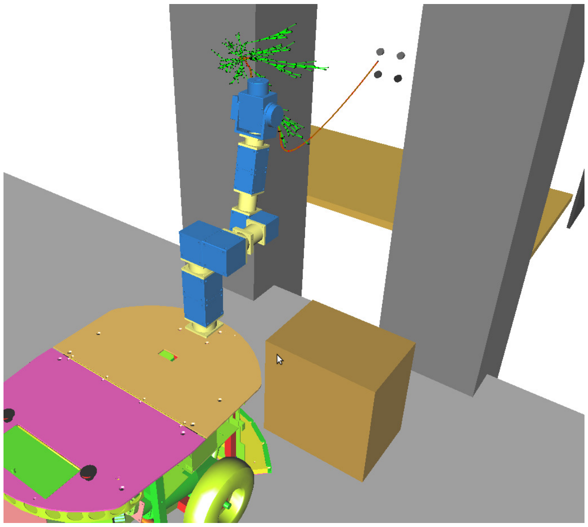


Figure 5.4: Planning environment visualized in OpenRAVE: camera tree (in green) and the planned trajectory (in red)

four coplanar dots (Fig. 5.3) on the target object. Please note that the final configuration of the WMM falls out as the result of the planning which may vary on different runs but will yield the same desired view which is the objective of the task.

As shown in Fig. 5.4, the desired location of the camera is located in between the long obstacles on the sides of the target dots. To reach the desired location, the WMM system needs to move around the obstacle in front of it while keeping the feature in view and making sure that the long obstacles do not occlude its field of view. Planning in such a constrained space is very challenging. Our proposed kinodynamic planner succeeds to find a trajectory in about 200 seconds. The planned camera tree and its final trajectory are shown in Fig.5.4. The planned camera trajectory was then projected into the image space to obtain the feature trajectories as shown in Fig. 5.3(left).

Figure 5.5 shows the snapshots of the WMM tracking the planned trajectories using the proposed decoupled control scheme. As shown the WMM manages to move around the obstacle at the front while keeping the features in the field of view and avoiding occlusion/collision due to obstacles. The followed feature trajectories are shown in Fig. 5.3(right). The video of this experiment is available on-line at `http://youtu.be/m6HCwEctxj0`

5.6 Discussions

The experiment presented above is our first effort on running the planned trajectories using our proposed decoupled feedback control scheme. Through this experiment we had a few observations which lead to our future research and developments:

- Developing a synchronization strategy between the motion of the mobile platform and the arm is very crucial: we observed that due to the lag of the mobile platform in responding to velocity control inputs (up to 5 cm/degrees in position/orientation) the arm deviates from its desired trajectory (up to 20 degrees for some joints) to compensate for the platform tracking error while achieving its feature trajectory tracking task in image space.

- The performance of the mobile platform state estimation strategy degrades due to shaky movements of the platform in particular at the beginning of the trajectory and during the turns. Moreover, for a better estimation accuracy the image features should be spread in the image which requires the camera to be close to the target. In the above experiment the camera was initially located about 150 cm away from the target. For distances above 100cm, or when the platform shakes during the tracking we had to use the on-board odometry to maintain an estimate of the platform state. One interesting direction is to combine the odometry with the image-based state estimation for a better performance.

- Finally, although the error dynamics and stability of each individual controller used in our decoupled control scheme has been studied previously ([80][30]), the overall stability and performance of the proposed decoupled control scheme requires further studies.

5.7 Summary

In this chapter we extended our efforts on incorporating randomized path planning techniques with image-based control of robotic arms to wheeled mobile manipulators. We employed a two-step plan-then-execute strategy as proposed in Chapter 3. We used the planning framework proposed in Chapter 3 to plan feasible trajectories for the whole mobile manipulator system to accomplish a visually-guided task. We extended the inverse kinematic controller used to track the camera trajectories in the WMM configuration space, by using weighted pseudo-inverse Jacobian solutions combined with a null space optimization technique to effectively coordinate the motion of the mobile platform and the arm for tracking the camera local paths in C-space.

We proposed a decoupled feedback control scheme to execute the planned trajectories: the on-board arm is servo controlled along the planned feature trajectories using an image-based visual servoing scheme (as adapted for the 6-DOF robotic arm in Chapter 4). Simultaneously, the mobile platform is controlled along its planned trajectory in Cartesian space using a state feedback control scheme.

We presented our first results obtained by executing the planned trajectories on our 9-DOF WMM system using the above decoupled scheme to accomplish a visually-guided task. This is the first step towards integrating path planning with image-based control of wheeled mobile manipulators and, as discussed in previous section, some issues still remain to be addressed before we can achieve robust execution of the planned trajectories.

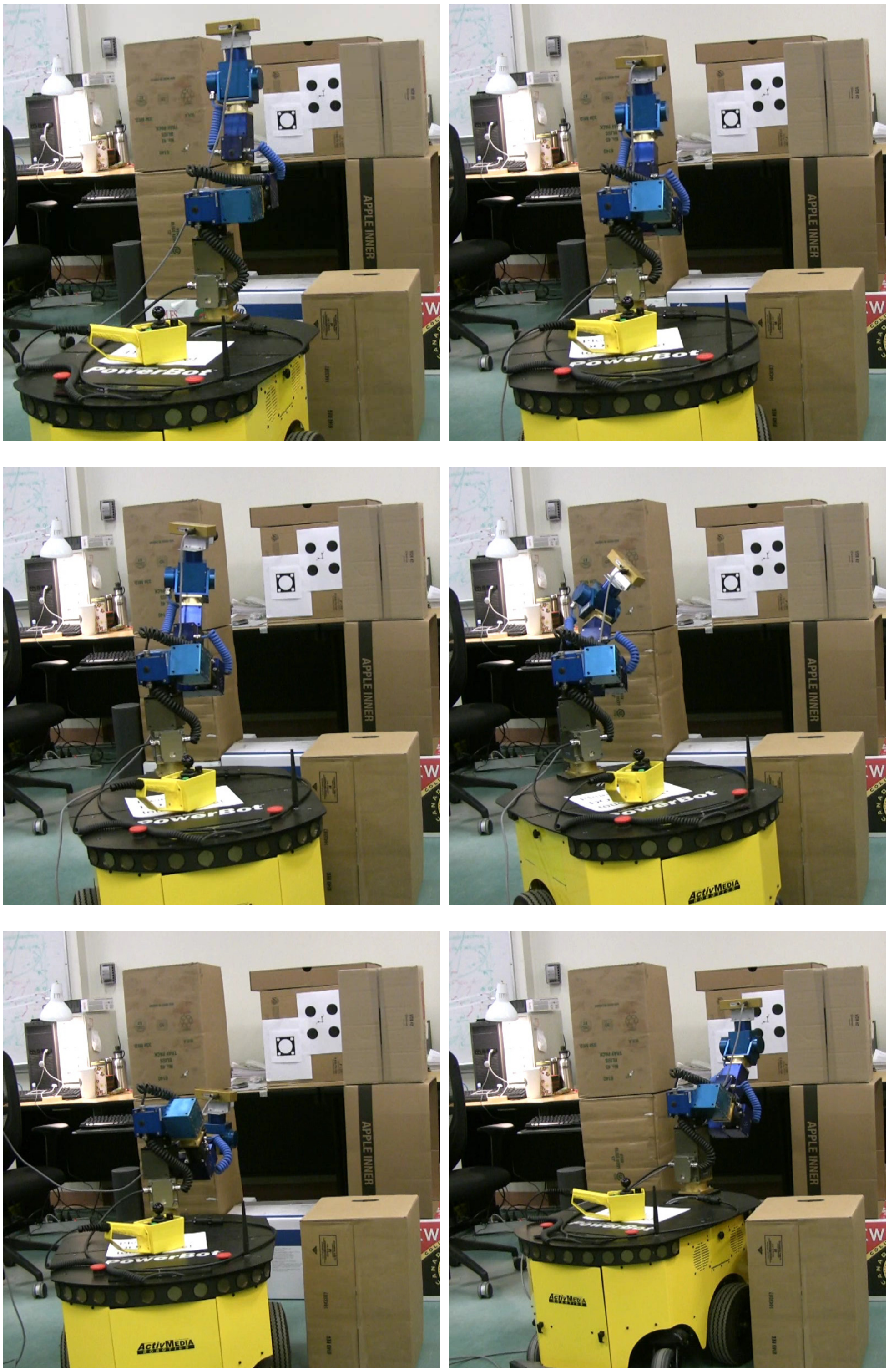


Figure 5.5: Snapshots of the wheeled mobile manipulator system following the planned trajectories to reach the desired location while avoiding collision/occlusion due to obstacles and keeping the target in the field of view.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

We presented our efforts on incorporating randomized path planning technique with image-based control of robotic arms and wheeled mobile manipulators. We proposed and implemented a general planning framework which effectively accounts for image and physical constraints encountered in visual servoing tasks. The proposed planner explores the camera planning space (using an RRT-like scheme) for feasible camera trajectories satisfying field of view limits and occlusion constraints, and simultaneously the camera trajectories in the robot configuration space and checks for robot kinematic constraints and collision with obstacles. The result is a search tree which explores both camera planning space and robot configuration space for feasible trajectories for the whole robot body. The solution camera trajectories are then projected into the image space of the camera to be executed on the robot using an image-based visual servoing scheme.

We developed two local planning scheme within the proposed framework: a kinematic planning scheme in camera configuration space, i.e., space of camera poses, and a kinodynamic planning scheme in camera state space, i.e., space of camera poses and velocities. In contrast to the former, the latter approach results in smooth trajectories by construction which require no further smoothing/time-scaling to be executed on the real robot.

We implemented the proposed framework on a 6-DOF robotic arm with an in-hand camera. We implemented an image-based visual servoing scheme to execute the solution feature trajectories on the 6-DOF robotic arm. We performed a number of real-world experiments on the 6-DOF robotic arm to show the merit of the proposed planning scheme and its effectiveness in accounting for a variety of image and physical constraints. Pertinent to the execution of solution trajectories, we also contrasted the image-based trajectory tracking controller with a joint space trajectory tracking controller through a number of simulation results. We provided an empirical study comparing the performance of the two controllers under modeling and calibration uncertainties.

We also extended the proposed framework to path planning for image-based control of wheeled mobile manipulators. We employed weighted pseudo-inverse Jacobian solutions combined with a null space optimization technique to effectively coordinate the motion of the mobile platform and the arm for tracking the camera local paths in the whole mobile manipulator configuration space. We executed the solution trajectories on a 9-DOF wheeled mobile manipulator system, i.e., a non-holonomic mobile platform with an on-board 6-DOF robotic arm with an in-hand camera. We adopted a decoupled feedback control strategy to execute the solution trajectories: the on-board robotic arm is controlled along the feature trajectories using an image-based visual servoing scheme while the mobile platform is simultaneously controlled along its planned trajectory in Cartesian space. We presented our first results obtained using the proposed scheme to accomplish the visually-guided task of moving the mobile manipulator system from initial location to a desired location while maintaining image and physical constraints.

To the best of our knowledge, this work is the first effort toward integrating randomized path planning techniques with image-based control of robotic arms and wheeled mobile manipulators. The flexibility of the proposed framework in accounting for image and physical constraints facilitates further extensions and its application to complex scenarios. We discuss some possible future directions as follows.

6.2 Future Work

Based on our observations through the experiments presented in Chapters 4 and 5, and also considering the flexibility of the proposed framework, we suggest a few directions to enhance this work, and also extend it further:

6.2.1 Planning under modeling and calibration uncertainties

As demonstrated through the results presented in Section 4.3, the tracking performance in image and joint spaces is affected by inevitable modeling and calibration uncertainties. Incorporating such uncertainties at the planning step will ensure more robust execution of planned trajectories. The flexibility of framework presented here allows for such extension.

For example, in [68] each extension of the tree is treated as a stochastic process and is simulated multiple times to account for uncertainties. Similar strategy can be adapted to extend the camera tree in the proposed framework. More recently, followed by our work, Shademan and Jagersand [84] incorporated an uncalibrated visual servoing scheme with a RRT-like planning approach. Their technique achieves robustness with respect to outliers using a statistically robust Jacobian estimation technique in the so called visual-motor space. Such schemes can be readily incorporated within our proposed framework to account for uncertainties.

6.2.2 Real-time constraint avoidance

Incorporating uncertainties at the planning stage, as suggested above, will increase the robustness with respect to image and physical constraints at the execution stage, however, this does not fully guarantee constraint avoidance due to inevitable deviations from planned trajectories. Hence, to ensure successful execution of the visual servoing task, one may apply real-time constraint avoidance strategies such as potential field [69] during the execution. Please note that this does not eliminate the need for the global path planning stage since a global plan is still necessary to guarantee reaching the goal. Real-time constraint avoidance will allow for possible deviations (within a threshold) from planned trajectories while pushing the robot away from constraints.

6.2.3 Motion coordination of wheeled mobile manipulator

Our preliminary experiments on the wheeled mobile manipulator showed us that a proper synchronization between the motion of the on-board arm and the mobile platform is crucial to ensure close tracking of feature trajectories. This may require dynamic time-scaling of feature trajectories to compensate for the mobile platform inertia in responding to its velocity command inputs along its trajectory in Cartesian space.

Bibliography

[1] B. Allotta and D. Fioravanti. 3D motion planning for image-based visual servoing tasks. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 2173–2178, 2005.

[2] D. Baraff. An introduction to physically based modeling: Rigid body simulation I - unconstrained rigid body dynamics. In *SIGGRAPH '97 Course Notes*, page 97, 1997. Avialable at `http://www.cs.cmu.edu/~baraff/pbm/pbm.html`.

[3] M. Baumann, S. Leonard, E.A. Croft, and J.J. Little. Path planning for improved visibility using a probabilistic road map. *IEEE Trans. Robot.*, 26(1):195–200, 2010.

[4] M.A. Baumann, D.C. Dupuis, S. Leonard, E.A. Croft, and J.J. Little. Occlusion-free path planning with a probabilistic roadmap. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 2151–2156, 2008.

[5] B. Bayle, J.-Y. Fourquet, and M. Renaud. Manipulability of wheeled mobile manipulators: application to motion generation. *Int. J. Robot. Res.*, 22(7-8):565–581, 2003.

[6] I. Belousov, C. Esteves, J.-P. Laumond, and E. Ferre. Motion planning for the large space manipulators with complicated dynamics. In *IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems*, pages 2160–2166, 2005.

[7] S. Bhattacharya, R. Murrieta-Cid, and S. Hutchinson. Optimal paths for landmark-based navigation by differential-drive vehicles with field-of-view constraints. *IEEE Trans. Robot.*, 23(1):47–59, 2007.

[8] J.A. Borgstadt and N.J. Ferrier. Visual servoing: path interpolation by homography decomposition. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 723–730, 2001.

[9] A. Censi, D. Calisi, A.D. Luca, and G. Oriolo. A Bayesian framework for optimal motion planning with uncertainty. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 1798–1805, 2008.

[10] F. Chaumette. Potential problems of stability and convergence in image-based and position-based visual servoing. In D.J. Kriegman, G.D. Hager, and A.S. Morse, editors, *Confluence of Vision and Control*, pages 66–78. Springer Berlin / Heidelberg, 1998.

[11] F. Chaumette and S. Hutchinson. Visual servo control Part I: Basic approaches. *IEEE Robot. Autom. Mag.*, 13(4):82–90, 2006.

[12] F. Chaumette and S. Hutchinson. Visual servo control Part II: Advanced approaches. *IEEE Robot. Autom. Mag.*, 14(1):109–118, 2007.

[13] G. Chesi. Desigining image trajectoris in the presence of uncertain data for robust visual servoing path planning. In *Proc. IEEE Int. Conf. Robot. Autom*, pages 1492–1497, 2009.

[14] G. Chesi. Visual servoing path planning via homogeneous forms and LMI optimizations. *IEEE Trans. Robot.*, 25(2):281–291, 2009.

[15] G. Chesi, K. Hashimoto, D. Prattichizzo, and A. Vicino. Keeping features in the field of view in eye-in-hand visual servoing: a switching approach. *IEEE Trans. Robot.*, 20(5):908–914, 2004.

[16] G. Chesi and Y.S. Hung. Global path-planning for constrained and optimal visual servoing. *IEEE Trans. Robot.*, 23(5):1050–1060, 2007.

[17] G. Chesi, E. Malis, and R. Cipolla. Automatic segmentation and matching of planar contours for visual servoing. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 2753–2758, 2000.

[18] G. Chesi, D. Prattichizzo, and A. Vicino. Straight line path-planning in visual servoing. *Trans. ASME, J. Dyn. Syst. Meas. Control*, 129(4):541–3, 2007.

[19] A.I. Comport, E. Marchand, M. Pressigout, and F. Chaumette. Real-time markerless tracking for augmented reality: the virtual visual servoing framework. *IEEE Trans. Vis. Comput. Graphics*, 12(4):615–628, 2006.

[20] P. Corke and M.C. Good. Dynamic effects in visual closed-loop systems. *IEEE Trans. Robot.*, 12(5):671–683, October 1996.

[21] P.I. Corke and S.A. Hutchinson. A new partitioned approach to image-based visual servo control. *IEEE Trans. Robot. Autom.*, 17(4):507–515, 2001.

[22] N.J. Cowan, J.D. Weingarten, and D.E. Koditschek. Visual servoing via navigation functions. *IEEE Trans. Robot. Autom.*, 18(4):521–533, 2002.

[23] J. J. Craig. *Introduction to Robotics: Mechanics and Control.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[24] A. De luca, G. Oriolo, and P. R. Giordano. Image-based visual servoing schemes for nonholonomic mobile manipulators. *Robotica*, 25:131–145, March 2007.

[25] K. Deguchi. Optimal motion control for image-based visual servoing by decoupling translation and rotation. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 705–11, 1998.

[26] D.F. Dementhon and L.S. Davis. Model-based object pose in 25 lines of code. *Int'l J. of Computer Vision*, 15(1-2):123–141, 1995.

[27] L. Deng, F. Janabi-Sharifi, and W.J. Wilson. Hybrid motion control and planning strategies for visual servoing. *IEEE Trans. Ind. Electron.*, 52(4):1024–1040, 2005.

[28] R. Diankov. *Automated Construction of Robotic Manipulation Programs.* PhD thesis, Carnegie Mellon University, Robotics Institute, August 2010.

[29] B. Espiau. Effect of camera calibration errors on visual servoing in robotics. In *Proc. 3rd Int. Symp. Exp. Robot.*, pages 182–192, 1994.

[30] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Trans. Robot. Autom.*, 8(3):313–326, 1992.

[31] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint.* MIT Press, Cambridge, MA, 1993.

[32] N.R. Gans and S.A. Hutchinson. Stable visual servoing through hybrid switched-system control. *IEEE Trans. Robot.*, 23(3):530–540, 2007.

[33] N.R. Gans, S.A. Hutchinson, and P.I. Corke. Performance tests for visual servo control systems, with application to partitioned approaches to visual servo control. *Int. J. Robot. Res.*, 22(10-11):955–981, 2003.

[34] I. Gipson, K. Gupta, and M. Greenspan. MPK: an open extensible motion planning kernel. *J. Robot. Syst.*, 18(8):433–443, 2001.

[35] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, Cambridge, UK, second edition, 2003.

[36] K. Hashimoto and T. Noritsugu. Potential problems and switching control for visual servoing. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 423–428, 2000.

[37] J. Hayet, C. Esteves, and R. Murrieta-Cid. A motion planner for maintaining landmark visibility with a differential drive robot. In *Algorithmic Foundations of Robotics VIII*, pages 333–347, 2009.

[38] K. Hosoda, K. Sakamoto, and M. Asada. Trajectory generation for obstacle avoidance of uncalibrated stereo visual servoing without 3d reconstruction. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 29–34, 1995.

[39] Y. Huang and K. Gupta. RRT-SLAM for motion planning with motion and map uncertainty for robot exploration. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 1077–1082, 2008.

[40] S. Hutchinson, G.D. Hager, and P.I. Corke. A tutorial on visual servo control. *IEEE Trans. Robot. Autom.*, 12(5):651–670, 1996.

[41] M. Kazemi, K. Gupta, and M. Mehrandezh. Global path planning for robust visual servoing in complex environments. In *IEEE Int. Conf. on Robot. and Automat.*, pages 326–332, May 2009.

[42] M. Kazemi, K. Gupta, and M. Mehrandezh. Kinodynamic planning for visual servoing. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, pages 2478–2484, May 2011.

[43] M. Kazemi, K. Gupta, and M. Mehrandezh. Path planning for visual servoing: A review and issues. In G. Chesi and K. Hashimoto, editors, *Visual Servoing via Advanced Numerical Methods*, chapter 11, pages 189–208. Springer, Mrach 2010.

[44] M. Kazemi, K. Gupta, and M. Mehrandezh. Path planning for image-based control of wheeled mobile manipulators. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 5306–5312, October 2011.

[45] M. Kazemi, K. Gupta, and M. Mehrandezh. Randomized kindynamic planning for robust image-based visual servoing. *IEEE Trans. Robot.*, under review.

[46] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *Int. J. Robot. Res.*, 5(1):90–98, 1986.

[47] S. Kirkpatrick, C.D. Gelatt Jr., and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[48] D. Kragic, L. Petersson, and H.I. Christensen. Visually guided manipulation tasks. *Robot. Auton. Syst. (Netherlands)*, 40(2-3):193–203, 2002.

[49] J.J. Kuffner. Effective sampling and distance metrics for 3D rigid body path planning. In *IEEE Int. Conf. on Robot. and Automat.*, pages 3993–3998, 2004.

[50] J.J. Kuffner and S.M. LaValle. RRT-connect: an efficient approach to single-query path planning. In *Proc. IEEE Int. Conf. Robot. Autom*, pages 995–1001, 2000.

[51] V. Kyrki, D. Kragic, and H.I. Christensen. New shortest-path approaches to visual servoing. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 349–354, 2004.

[52] V. Kyrki, D. Kragic, and H.I. Christensen. Measurement errors in visual servoing. *Robot. Auton. Syst.*, 54(10):815–827, 2006.

[53] F. Lamiraux, E. Ferre, and E. Vallee. Kinodynamic motion planning: connecting exploration trees using trajectory optimization methods. In *Proc. IEEE Int'l Conf. on Robot. and Automat.*, pages 3987–3992, 2004.

[54] O. Lang and A. Graser. Visual control of 6 DOF robots with constant object size in the image by means of zoom camera. In *Proc. Conf. IEEE Ind. Electron. Society*, pages 1342–1347, 1999.

[55] J-C. Latombe. *Robot Motion Planning.* Kluwer Academic Publishers, Norwell, MA, USA, 1991.

[56] S.M. LaValle. *Planning Algorithms.* Cambridge University Press, Cambridge, U.K., 2006.

[57] S.M. LaValle, H.H. Gonzalez-Banos, C. Becker, and J-C. Latombe. Motion strategies for maintaining visibility of a moving target. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 731–736, 1997.

[58] S.M. LaValle and J.J. Kuffner. Randomized kinodynamic planning. *Int. J. Robot. Res.*, 20(5):378–400, 2001.

[59] G. Lopez-Nicolas, S. Bhattacharya, J.J. Guerrero, C. Sagues, and S. Hutchinson. Switched homography-based visual control of differential drive vehicles with field-of-view constraints. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 4238–4244, 2007.

[60] T. Lozano-Perez, M.T. Mason, and R.H. Taylor. Automatic synthesis of fine-motion strategies for robots. *Int. J. Robot. Res.*, 3(1):3–24, 1984.

[61] R. Mahony, T. Hamel, and F. Chaumette. A decoupled image space approach to visual servo control of a robotic manipulator. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 3781–3786, 2002.

[62] E. Malis. Visual servoing invariant to changes in camera-intrinsic parameters. *IEEE Trans. Robot. Autom.*, 20(1):72–81, 2004.

[63] E. Malis and F. Chaumette. 2 1/2 D visual servoing with respect to unknown objects through a new estimation scheme of camera displacement. *Int. J. Comput. Vis.*, 37(1):79–97, 2000.

[64] E. Malis, F. Chaumette, and S. Boudet. 2-1/2-D Visual servoing. *IEEE Trans. Robot. Autom.*, 15(2):238–250, 1999.

[65] E. Malis and P. Rives. Robustness of image-based visual servoing with respect to depth distribution errors. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, pages 1056–61, 2003.

[66] E. Marchand and F. Chaumette. Virtual visual servoing: a framework for real-time augmented reality. In *Proc. EUROGRAPHICS Conf.*, volume 21, pages 289–298, 2002.

[67] E. Marchand, F. Spindler, and F. Chaumette. ViSP for visual servoing: a generic software platform with a wide class of robot control skills. *IEEE Robot. Autom. Mag.*, 12(4):40–52, Dec. 2005.

[68] N.A. Melchior and R. Simmons. Particle rrt for path planning with uncertainty. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, pages 1617–1624, 2007.

[69] Y. Mezouar and F. Chaumette. Path planning for robust image-based control. *IEEE Trans. Robot. Autom.*, 18(4):534–549, 2002.

[70] Y. Mezouar and F. Chaumette. Optimal camera trajectory with image-based control. *Int. J. Robot. Res.*, 22(10-11):781–803, 2003.

[71] P. Michel, C. Scheurer, J. Kuffner, N. Vahrenkamp, and R. Dillmann. Planning for robust execution of humanoid motions using future perceptive capability. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 3223–3228, 2007.

[72] G. Morel, T. Liebezeit, J. Szewczyk, S. Boudet, and J. Pot. Explicit incorporation of 2D constraints in vision based control of robot manipulators. In *Proc. 6th Int. Symp. Exp. Robot.*, pages 99–108, 2000.

[73] G. Morel, P. Zanne, and F. Plestan. Robust visual servoing: bounding the task function tracking errors. *IEEE Trans. Control Syst. Technol.*, 13(6):998–1009, 2005.

[74] Yoshihiko Nakamura. *Advanced Robotics: Redundancy and Optimization.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1990.

[75] J.S. Park and M.J. Chung. Path planning with uncalibrated stereo rig for image-based visual servoing under large pose discrepancy. *IEEE Trans. Robot. Autom.*, 19(2):250–258, 2003.

[76] R. Pepy and A. Lambert. Safe path planning in an uncertain-configuration space using RRT. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 5376–5381, 2006.

[77] R. Pissard-Gibollet and P. Rives. Applying visual servoing techniques to control a mobile hand-eye system. In *IEEE Int'l Conf. on Robotics and Automation*, pages 166–171, 1995.

[78] P. Salaris, F. Belo, D. Fontanelli, L. Greco, and A. Bicchi. Optimal paths in a constrained image plane for purely image-based parking. In *Proc. IEEE/RSJ Int. Conf. on Intell. Robot. Syst.*, pages 1673–1680, 2008.

[79] P. Salaris, D. Fontanelli, L. Pallottino, and A. Bicchi. Shortest paths for a robot with nonholonomic and field-of-view constraints. *IEEE Trans. Robot.*, 26(2):269–281, 2010.

[80] C. Samson and K. Ait-Abderrahim. Feedback control of a nonholonomic wheeled cart in cartesian space. In *IEEE Int'l Conf. on Robotics and Automation*, pages 1136–1141, 1991.

[81] M. Sauvee, P. Poignet, E. Dombre, and E. Courtial. Image based visual servoing through nonlinear model predictive control. In *Proc. IEEE Conf. Decis. Control*, pages 1776–1781, 2006.

[82] F. Schramm and G. Morel. Ensuring visibility in calibration-free path planning for image-based visual servoing. *IEEE Trans. Robot.*, 22(4):848–854, 2006.

[83] L. Sciavicco and B. Siciliano. *Modeling and Control of Robot Manipulators.* McGraw-Hill, 1996.

[84] A. Shademan and M. Jagersand. Robust sampling-based planning for uncalibrated visual servoing. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, page to appear, 2012.

[85] R. Sharma and S. Hutchinson. Motion perceptibility and its application to active vision-based servo control. *IEEE Trans. Robot. Autom.*, 13(4):607–617, 1997.

[86] R. Sharma and H. Sutanto. A framework for robot motion planning with sensor constraints. *IEEE Trans. Robot. Autom.*, 13(1):61–73, 1997.

[87] B. Tamadazte, N.L. Piat, and S. Dembele. Robust trajectory tracking and visual servoing schemes for MEMS manipulation. In *Proc. IEEE/ASME Int'l Conf. on Advanced Intelligent Mechatronics*, pages 860–865, 2010.

[88] K. Tarabanis, R.Y. Tsai, and A. Kaul. Computing occlusion-free viewpoints. *IEEE Trans. Pattern Anal. Mach. Intell.*, 18(3):279–292, 1996.

[89] D.E. Whitney. Resolved motion rate control of manipulators and human prostheses. *IEEE Trans. Man-Mach. Syst.*, 10(2):47–53, 1969.

[90] Z. Yao and K. Gupta. Path planning with general end-effector constraints. *Robot. Auton. Syst.*, 55(4):316–327, 2007.

[91] H. Zhang and J.P. Ostrowski. Visual motion planning for mobile robots. *IEEE Trans. Robot. Autom.*, 18(2):199–208, 2002.