Vision-based Robot Navigation using an Online Visual Experience

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

Master of Science (by Research)
in
Computer Science & Engineering

by

D. Santosh Kumar
200607024
http://students.iiit.ac.in/~santosh
santosh@research.iiit.ac.in

Center for Visual Information Technology
International Institute of Information Technology
Hyderabad - 500 032, INDIA
June 2007

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled “Vision-based Robot Navigation using an Online Visual Experience” by D. Santosh Kumar, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Adviser: Prof. C. V. Jawahar

To the field of Robotics

Acknowledgments

I would like to thank Dr. C. V. Jawahar for his guidance and support during my research. He provided me with the opportunities to pursue my personal research interests and I am thankful for the same. I would also like to thank Prof. P. J. Narayanan for advising me and motivating me at critical junctures. They have been great mentors and have guided me through the ups and downs of my research career until now.

I have immensely benefited from the members (both past & present) of the Center for Visual Information Technology (CVIT), specifically Abdul Hafez, Visesh Chari and Paresh Jain in terms of their help and stimulating company. Thanks to Supreeth Achar for building the robot and helping me conduct the experiments. Financial Support from CVIT and IIIT Hyderabad during the period of my Masters is acknowledged.

I especially wish to acknowledge my father D. Venkatadri, my mother D. Sujatha and my sister A. Tina.

Abstract

Vision-based robot navigation has long been a fundamental goal in both robotics and computer vision research. While the problem is largely solved for robots equipped with active range-finding devices, for a variety of reasons, the task still remains challenging for robots equipped only with vision sensors. Vision is an attractive sensor as it helps in the design of economically viable systems with simpler sensor limitations. It facilitates passive sensing of the environment and provides valuable semantic information about the scene that is unavailable to other sensors. Two popular paradigms have emerged to analyze this problem, namely Model-based and Model-free algorithms. Model-based approaches demand *apriori* model information to be made available in advance. In case of the latter, required $3D$ information is computed online. Model-free navigation paradigms have gained popularity over model-based approaches due to their simpler assumptions and wider applicability. This thesis discusses a new paradigm to vision-based navigation, namely Image-based navigation. The basic concept is that model-free paradigms involve an unnecessary intermediate depth computation, which is redundant for the purpose of navigation. Rather the motion instruction required to control the robot can be inferred directly from the acquired images. This approach is more attractive as the modeling of objects is now simply substituted by the memorization of views, which is far easier than 3D modeling.

In this thesis, a new image-based navigation architecture is developed, which facilitates online learning about the world by a robot (Chap. 2). The framework capacitates a robot to autonomously explore and navigate a variety of unknown environments, in a way that facilitates path planning and goal-oriented tasks, using visual maps that are contextually built in the process. It also facilitates the incorporation of feedback received from performing specific goal oriented tasks to update the visual representation. Based on this architecture, the design of the individual algorithms required for performing the navigation task (exploration, servoing and learning) is discussed.

In Chap. 3, a novel image-based exploration algorithm based on the idea of frontier-based exploration is proposed. The algorithm infers the frontier boundaries directly from monocular images and uses them to efficiently explore the environment. The frontiers are detected by using a modified segmentation scheme, that separates floor regions from non-floor regions. This method exploits the advantage of the natural structure in the world without involving any 3D reconstruction of the scene. The proposed algorithm can systematically discover an unknown environment and build a visual representation that is most suited for navigation.

Chap. 4 discusses the three major algorithms required for performing the navigation task, namely localization, planning and servoing. First, the idea of qualitative localization is introduced, wherein the localization only identifies the most similar image in the database to the current image. A planning algorithm is then used to infer the right set of intermediate images (from the visual representation) that would lead the robot from its initial position to the required destination. Finally a feed-back based control algorithm that exploits the projective transformations existing between the intermediate views is employed to servo the robot to the goal configuration. The overall algorithm only operates using images and converges to the destination reliably.

An online learning algorithm is described in Chap. 5. The algorithm utilizes additional scene information, gathered over time by the robot, to improve its visual representation. This is done by the way of updating the scene feature descriptors using an incremental learning algorithm. Learning is also employed to exploit the feedback received from previous experiences of the robot for improving the navigation performance. Specifically the task of path-planning for trajectory generation is discussed. An improved reinforcement learning scheme utilizing the potential field method to generate optimal motion trajectories is presented.

The basic advantage of the overall proposed framework is that the robot is no more-limited to a restrictive teach-and-replay scenario. It can now build an internal representation of its workspace autonomously and improve it automatically. Further, the navigation task can be accomplished more efficiently by simply exploiting the constraints existing between the images. The proposed approach enables mobile robots to progress in the direction of increased applicability and robustness. Experimental results on a laboratory set-up confirm the efficacy of the proposed algorithms.

Similar principles had been applied for solving the navigation/servoing problem in non-rigid environments during the earlier phase of this thesis and are presented in the Appendix (Chap. A and B).

Contents

List of Figures

Chapter 1

Introduction

The field of robotics has engendered great interest among researchers across various fields in the recent past. The idea of employing a robot to perform a specific task instead of a human has been fascinating. Robotics encompasses a broad spectrum of technologies in which computational intelligence is embedded into physical machines, creating systems with capabilities far exceeding the individual basic components. Such robotic systems can carry out tasks that are unachievable by conventional machines, or even by humans working with conventional tools.

Robotic systems can be employed for a variety of tasks ranging from performing medical surgery [9] to the task of assembling a car or to the task of traversing in an urban environment [17]. One principle ability that one aspires of such systems to perform the above tasks is to move by themselves, that is, 'autonomously'. Mobile robots are machines that move autonomously, either on the ground or in the air/space or underwater. Such vehicles are generally unmanned, in the sense that no humans are on board. The machines move by themselves, with sensors and computational resources on-board to guide their motion.

The primary application of such robotic vehicles is their capability of traveling where people cannot go, or where the hazards of human presence are great. For instance, to reach the surface of Mars, a spacecraft must travel more than a year, and on arrival the surface has no air, water, or resources to support human life. Hence robotic exploration is a fundamental step that provides enormous scientific and technological rewards enhancing the knowledge of other planets. The Mars rover is a specific example of a robotic vehicle capable of local autonomous operation for segments of motion and defined scientific tasks [4, 43]. Another example of a hostile and hazardous environment where robotic vehicles are essential tools of work and exploration is the undersea world. Human divers may dive to a hundred meters or more, but pressure, light, currents and other factors limit such human exploration of the vast volume of the earth's oceans [81]. Apart from the above, these vehicles are also employed in routine tasks that occur over spaces and environments where machine mobility can effectively replace direct human presence. For example, in large scale cultivation of crops, underground mining etc [26]. Finally applications of robotic vehicles also includes the support of personal assistance (rehabilitation),

in household tasks, and in entertainment. For example, a wheelchair that utilizes emerging robotic technologies for providing mobility to the handicapped [22].

Mobile Robot Navigation is known as the ability of a robot to act based on its knowledge and sensor values in order to reach its goal positions as efficiently and as reliably as possible [5, 72]. At the first instance, it may seem a seemingly trivial task to navigate a robot as compared to the task of brain surgery or automobile manufacturing. However the latter tasks are carefully cut out and formed such that they are largely a high-precision positioning application for a very specialized tool. Whereas in the former case, the problem is that there is no high precision around, no available databases about what are the objects in the world and the floor plan. Further, the environment may be unknown (with obstacles), there may be people moving around, apart from presence of deformable objects such as plants, toys etc. Dealing with such a variable environment poses a plethora of challenges to a mobile system.

1.1 Mobile Robot Navigation Using Vision Sensors

One of the main obstacles that have hindered the penetration of mobile robots into wide consumer markets is the unavailability of powerful, versatile and cheap sensing. Vision technology is potentially a clear winner as far as the ratio of information provided versus cost is considered. Cameras of acceptable accuracy are currently sold at a price which is one to two orders of magnitude less than laser and sonar scanners. Vision is an attractive sensor as it helps in the design of economically viable systems with simpler sensor limitations. Vision potentially offers more portable and cost effective solutions, as new mass market for camera technology has resulted in significantly reduced price and increased performance. Moreover it can provide information unavailable to other sensors: for instance, it provides semantic information of a scene through the understanding of its visual appearance and not just the geometrical information about it.

The current trend in robot navigation is to try and use vision instead of more traditional range sensors. Vision based robotic systems have gained popularity recently, and several approaches have been proposed in the recent past. These systems analyze the images of the scene taken by the camera attached to the robot and use the visual cues to plan their action [8, 11, 22, 33]. The systems employ either regular cameras (single or multiple) or omnidirectional vision sensors for viewing the environment. The major distinguishing factor amongst the approaches is the method in perceiving the scene and the way of extracting the features from the scene. Much attention is being devoted to solve the non-trivial problems implied by using visual information for navigating an agent through the environment [5, 23, 30].

Vision-based robot navigation [22] has now become a fundamental goal in both robotics and computer vision research (Structure from Motion [30], Visual SLAM [18], Visual Servoing [62] etc). Progress has been made in the last two decades on two separate fronts: indoor and outdoor robot navigation. The strides made in both these areas have been significant. For example, earlier it would have been impossible for an autonomous indoor mobile robot to find its way in a cluttered hallway but now it is not much of a challenge. Equally impressive progress has been achieved in computer vision for outdoor robot-

ics, as represented by the NAVLAB [72] and the Prometheus [26] system. Unstructured and dynamic environments pose a crucial challenge to many real-world applications. With non-vision sensors, it is impossible to predict and model every possibility. As a result the parameters of the robot system were earlier tuned in order to work properly in the new environment. However with the emergence of cameras, such environments can be tackled more efficiently. For a summary on the progress of vision-based navigation, the reader may refer to [5, 22].

Traditionally, it has been assumed that the position of the target and/or the robot was known (or at least partially known). However, the direct outputs of vision sensors are generally not position information, but image features, which may be distorted due to projection, and restricted by the field of view. In order to obtain the global position and orientation of one object or even just to determine their relative pose, various algorithms of calibration and transformation are required. Hence, all of the proposed approaches formulate the vision-based navigation problem as a two-step process: first, to transfer the sensor features back to pose information, and then make a motion plan in the pose space. However, the transfer from sensor space to pose space is redundant and introduces unnecessary uncertainty into the loop. It would be more beneficial to directly use the sensory information and navigate the mobile robot. It is this aspect, which is the focus of the thesis. More specifically, this thesis deals with the problem of using off-the-shelf cameras fixed on inexpensive mobile platforms, to enable navigation and control to given goal configurations directly in the sensor space.

1.2 Image-based Navigation: An Overview

A mobile robot that navigates in a large scale environment needs to know its position in the world in order to successfully plan its path and its movements. This requires establishing a close relation between the perceived environment and the commands sent to the low-level controller, which necessitates complex spatial reasoning relying on some kind of internal environment representation. The general approach to this problem is to provide the robot with a detailed description of the environment (usually a geometrical map) obtained using a stereo/monocular vision sensor mounted on the robot. Unfortunately, extracting geometric information of the environment from the camera is time-consuming and intolerant of noise. Few authors have successfully addressed this solution using very robust uncertainty management systems [20, 42, 76], while few have circumvented it by efficient management of the environment [7, 31, 66, 73]. Unfortunately, either of the above paradigms are not always feasible. There are situations in which an exact map of the environment is either unavailable or useless: for example, in old or unexplored buildings or in environments in which the configuration of objects in the space changes frequently. Therefore, it would be beneficial for the robot to build its own representations of the world.

The philosophy of memory-based reasoning offers an interesting perspective. In the field of artificial intelligence research, memory-based reasoning has been studied for a long time, which has been originally motivated from the human reasoning process [8]. In addition to the capability of reasoning about the environment topology and geometry, humans show a capability for recalling memorized scenes that

help themselves to navigate. This implies that humans have a sort of visual memory that can help them locate themselves in a large environment. From these considerations, a new approach to the navigation and localization problem has been developed, namely image-based navigation [16, 52, 67].

This alternative approach employs a sensor-centered representation of the environment, which is usually a multidimensional array of sensor readings. In this case, the robotic agent is provided with a set of views of the environment taken at various locations. These locations are called reference locations because the robot will refer to them to locate itself in the environment. The corresponding images are called reference images. In the context of computer vision, the representation usually contains a set of key-images which are acquired during a training stage and organized within a graph. Nodes of the graph correspond to key-images, while the arcs link the images containing a distinctive set of common landmarks. When the robot moves in the environment, it can compare the current view with the reference images stored in its visual memory. When the robot finds which one of the reference images is more similar to the current view, it can infer its position in the environment. (If the reference positions are organized in a metrical map, an approximate geometrical localization can also be derived.) With this technique, the problem of finding the position of the robot in the environment is reduced to the problem of finding the best match for the current image among the reference images. A path to follow is then described by a set of images extracted from the database. This image path is designed so as to provide enough information to control the robotic system [59].

This research area has attracted recent interest. In [17], neural networks were employed to learn the relation between input view image and steering angle, to drive systems for both indoor and outdoor use. Crespi [10] applied memory-based approach to indoor navigation to learn the relation between the input view image and the lateral position in the corridor. The data learnt in the training phases are models of the road, but they were only used for following and not for identifying the robot's position along a route. In [52], a new model of the route following, called the View-Sequenced Route Representation was proposed, which was especially useful for corridor environments. In [12], a new algorithm for navigation was advocated. Using a teach-replay approach, the robot is manually led along a desired path in a teaching phase, then the robot autonomously follows that path in a replay phase. In [7], an algorithm for robot navigation using visual servoing was proposed. In this method, visual paths are topologically organized and the robot navigation mission is defined as a concatenation of visual path subsets, referred as 'visual routes'. To address the issue of huge memory requirements and computational costs for modeling and matching, Nayar et.al. [58] proposed a object recognition method where 3D objects were represented as manifolds in eigenspace, with parameters of their pose and lighting condition. This was a significant step as it had made image-based navigation a feasible approach in many application areas.

Current research on image-based navigation faces several issues, of which robustness and adaptability are the most challenging. A navigation system should be robust to many types of variations such as changes in illumination conditions, people wandering around, or objects being used and moved etc. In addition, the visual appearance of its environment changes continuously in time. These issues pose serious problems for recognition algorithms that are trained off-line on data acquired once and for all during

a fixed time span. The current effort is mainly focused in equipping the existing navigation algorithms with the above desirable though challenging characteristics.

1.3 Problem Statement and Contributions

The goal of this research is to develop autonomous robots that can not only explore their environment indigenously but also navigate in their workspace intelligently. To achieve this, the recent advances in the field of computer vision and machine learning research are utilized. The research in these areas have advanced to such a level that it is now not only possible to make robust and accurate inferences about the geometry of the world by only using (single or multiple) images in fully natural scenes [32, 63, 70] but also efficiently learn their models [14, 35, 51]. All these developments have led to an increase in their application to solve robotic problems, for example see [18, 62, 66]. Specifically, the techniques developed in these fields provide ample opportunity to perform better in the current context and thus they are utilized to enhance the image-based navigation paradigm.

In particular, the basic image-based navigation framework has several limitations. The proposed algorithms are restricted by the limited amount of information that is acquired by them during their off-line training stage. This dependence limits the knowledge of the robot. However the images that a robot acquires during navigation provide additional information about its world that could be utilized to update its memory and thus account for variations in the scene. Secondly, the performance of the robot over similar tasks remains the same over time. The feedback received by the robot in its previous experiences is not exploited to improve its performance in later tasks. Lastly the robot workspace is limited to the explored regions that it visualizes during the off-line training stage. It would be interesting if the existing approaches could be enabled to automatically adapt to new and changing environments. Rather than limiting image-based navigation paradigms to a simple teach-and-replay scheme, they can be extended so as to autonomously learn and navigate in unknown environments, which extend their capabilities and applications. In this thesis, the above issues are analyzed and a novel framework for image-based navigation is proposed, which achieves the following characteristics.

- Automatic exploration of new environments to gradually expand the robot workspace and mapping directly using images (rather than 3D mapping)
- Autonomous navigation only using information inferred from the robot visual memory
- Incorporation of additional information acquired over time into the robot memory incrementally, allowing long-term memory building
- Performance improvement via the process of online learning from its current and previous experiences

1.4 Navigation Environments

Different environmental scenarios have been considered for evaluating the applicability potential of the proposed algorithms. Two major environments considered are described below.

1.4.1 Indoor and Outdoor Environment for a Mobile Robot

The experiments are conducted in two different real-world environments. The environments considered include usual office and lab space apart from hallways and corridors. Such environments contain chairs, desks, tables etc. The goal is to position a robot anywhere in these environments and it should be able to navigate across different regions.

Outdoor environments pose a more challenging case to the algorithms due to their highly varying conditions (illumination, texture, dynamic objects etc). However, the methods should be able to adapt to such variations.

1.4.2 Navigation and Control in Deformable Environments

The problem of navigating a robot end-effector in presence of deformable targets has also been considered. In robotic vision research, motion analysis has been largely restricted to rigid objects due to their simplicity, elegance and immediate industrial applicability. However, in real world situations, motion of physical objects is often non-rigid [1] in nature. Common examples include motion of human body, flying birds, ocean waves etc.

Dealing with non-rigid motion poses several challenges in the design of optimal servoing strategies. Non-rigid objects undergo a persistent change in their pose which forbids any single image to characterize their state. This is because motion instruction planned based on the features extracted at current time instant might not be relevant in the next instant as the object undergoes a change in its pose. Further, the desired configuration of the end-effector cannot be described by using only a single image or a single pose as it will lead to oscillations of the manipulator even after the goal position is reached. Note that the unavailability of static features (in case of whole body deformations) and background features (in case of moving targets) makes it imperative to engender new representation schemes for visual servoing using only the pose-varying features on the object surface. This necessitates a time-based representation, rather than a purely spatial one, due to the temporal nature of the object deformations. It must be emphasized that non-rigid motion encompasses wide range of possible motions ranging from simple translatory motion such as a waving hand to highly complex motion like that of a beating heart. A general representation for all kinds of motions is preferable, but appears to be inconceivable at this stage. Establishing correspondence between image features is usually the primary step in visual servoing. However, finding accurate correspondences is often difficult in practical situations; especially, while matching points in two views separated by large displacement. This is a highly formidable require-

ment in case of deformable objects as this demands frame-to-frame matching of the object deformations which is complicated even for simple motions.

Existing servoing schemes are not designed to tackle non-rigidity. Cartesian-based algorithms require complete $3D$ information of the object which is a strong assumption for deformable targets. Image-based servoing schemes cannot be directly used as these schemes use information only from a single image to guide the robot, which results in an oscillatory camera trajectory [61]. Also, these methods are not completely model-free, since depths of the observed features are needed in the control law [33]. Further, they demand the exact frame-to-frame correspondences between image features. Moreover, new representations conceived for modeling the non-rigid motion cannot be directly utilized in these schemes as the corresponding interaction matrix relating the feature motion in the image space to the camera motion in the Cartesian space has to be derived. Until now, Jacobian has been derived only for simple primitives (points, lines) and designing new Jacobian for higher order primitives is a tedious task.

A different approach to visual servoing is proposed here, in which the motion characteristics of active non-rigid objects are used to perform the servoing task, without the requirement of $3D$ structure information. The approach is based on the bi-dimensional appearance of the objects in the environment and explicitly takes into account independent object motions. In most cases, where an object has a repetitive motion, the space-time trajectories of representative points on it will serve to uniquely represent the object. These trajectories are invariant to object deformations and can be utilized to obtain a stable estimate of the projective transformation relating the initial and desired views. The estimated transformation is then used in a feedback-based hybrid control to perform the servoing task.

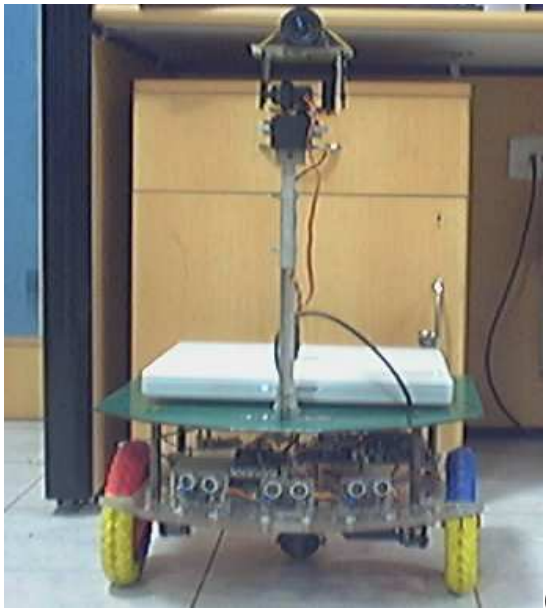
In the first part of this work, issues in working with a deformable environment were addressed and results of this study are reported in Appendix (See Chap. A). The rest of this thesis limits its scope to the specific explanations concerning a mobile robot, though most of the arguments are also valid for the navigation issues of a manipulator arm in a deformable environment.

1.5 Experimental Set-up

A low-cost apparatus was employed to highlight the applicability of the proposed algorithms. The experimental setup is comprised of an indigenously designed and built differential drive robotic platform[1] (with kinematics similar to a unicycle). The vehicle has two symmetric rows of three wheels on its sides each actuated by a single low-resolution stepper-motor actuator. It is also comprised of a simple controller, able to avoid feature occlusions and obstacles (See Fig. 1.1). The vehicle is equipped with an encoder feedback, ultrasonic range finder and a pan tilt head for a camera.

The proposed algorithms are analyzed using commercially available cameras. Specifically, an IEEE1394 Firewire camera 'Flea2' (manufactured by PointGrey) was used. The camera operates at 640×480 pixel resolution and affords a horizontal field of view of approximately 60°. The camera was fitted with a

[1]The robot was built by Supreeth Achar, B.Tech 2003 Student, IIITH


(a)


(b)

Figure 1.1 Experimental Set-up (a) A differential drive robot with a pan-tilt head and an on-board laptop (b) camera (Flea2)

$2.1mm$ wide-angle lens and achieves a peak frame rate of 30 frames per second (fps). The algorithms were also analyzed using another off-the-shelf USB camera (Logitech QuickCam Pro 4000). A standard calibration technique was employed to ascertain the internal parameters of the cameras.

The camera was placed on the front part of the robot platform as shown in Fig. 1.1. A 1130 MHz Dell Inspiron 700m laptop was mounted on-board the vehicle. The camera was connected to the laptop using a RS-232 serial cable. The robot operates in its x-z plane and rotates around its y-axis. All the proposed algorithms were implemented in Matlab/C++.

The experiments were conducted in two different real-world lab environments. The environments contained chairs, desks, tables etc. Fig. 1.2(a) shows the environment and the robot in its workspace. Fig. 1.2(b) displays the robot's view of the workspace.

1.6 Organization of the Thesis

The rest of the thesis is organized as follows. Chapter 2 explains the proposed image-based navigation architecture. In Chapter. 3, the concept of image-based exploration is explicated and the proposed algorithm is presented. The details of localization, planning and navigation algorithms are elucidated in Chapter. 4. Chapter. 5 explains the need for life-long learning in mobile robotic systems and gives

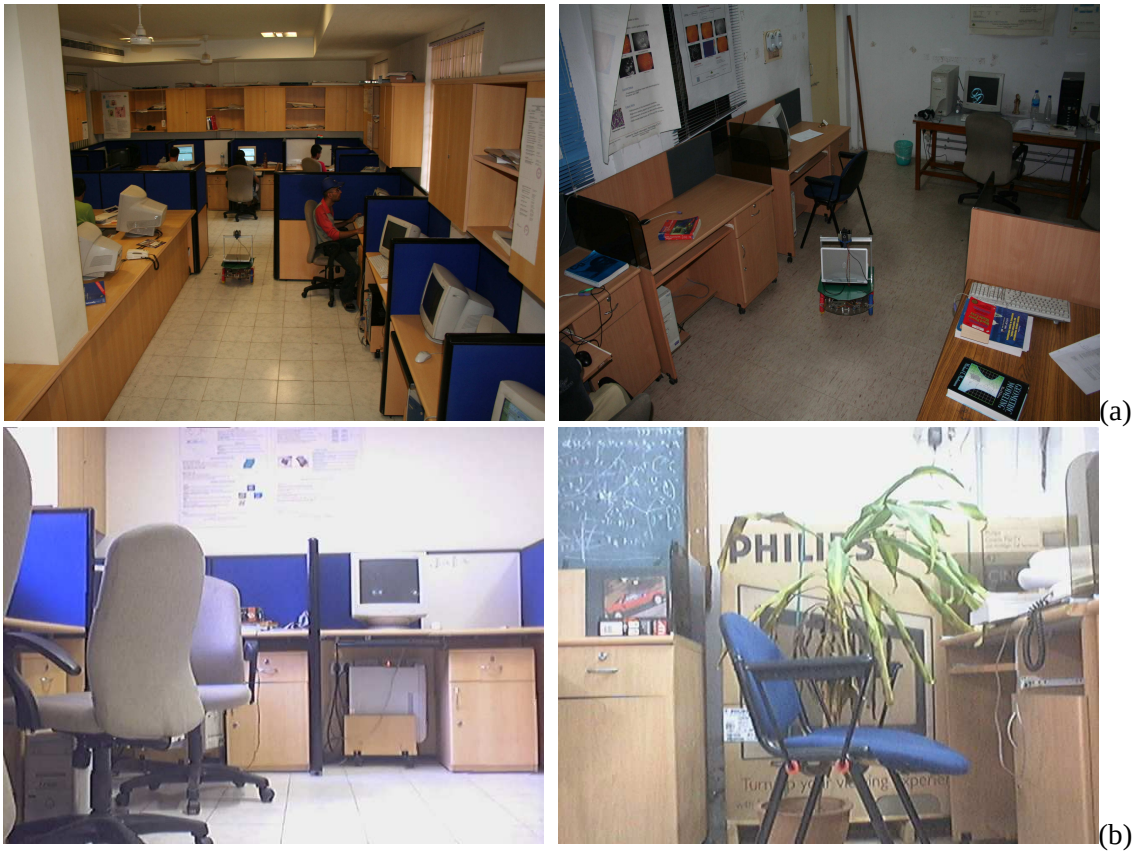


Figure 1.2 (a) Robot in its workspace (b) Robot's view of its workspace

details of the devised online learning method. Finally, Chapter. 6 gives the concluding remarks and provides pointers for future research.

Chapter 2

Visual Experience Architecture

2.1 Motivation

Vision-based navigation has been mostly analyzed as a localization problem in the literature. The robot is provided with a set of images obtained during a training stage to describe its environment. Localization is then performed by comparing the current image with the set of images. However, there has been no single method developed until now that addresses the issues of exploration, mapping, localization, planning, servoing and learning in a single comprehensive framework. Such a framework is more interesting rather than limiting image-based navigation paradigms to a simple teach-and-replay scheme. It allows the robot to autonomously learn and navigate in a wide variety of unknown environments extending their capabilities and applications.

The ability to automatically learn from its past experiences and simultaneously build a dynamic map while autonomously exploring an unknown environment opens the door for robotic systems to be widely deployed. Several industrial applications can benefit from this framework, for instance mobile robots providing services in a small-scale outdoor environment, performing path planning and navigation to arbitrary destinations, development of robotic navigation guides etc. The recent advances in the field of computer vision and machine learning (for instance, camera pose estimation under various challenging conditions [70], real-time visual tracking [66] etc) make this task possible. The techniques developed in these fields provide ample opportunity to perform better in the current context and thus they can be adapted to enhance the existing paradigms. The motivation is to adopt the advances in these fields to enhance the image-based navigation algorithms in the following manner.

1. It is now possible to explore only using simple vision sensors and map the environment simply as images (View-based Exploration).

2. Servoing can be performed by exploiting the constraints and relationships that exists between images (robust correspondences and accurate relative pose estimation).

3. Online data acquired by a robot can be utilized to enrich its visual memory (Incremental Updates).

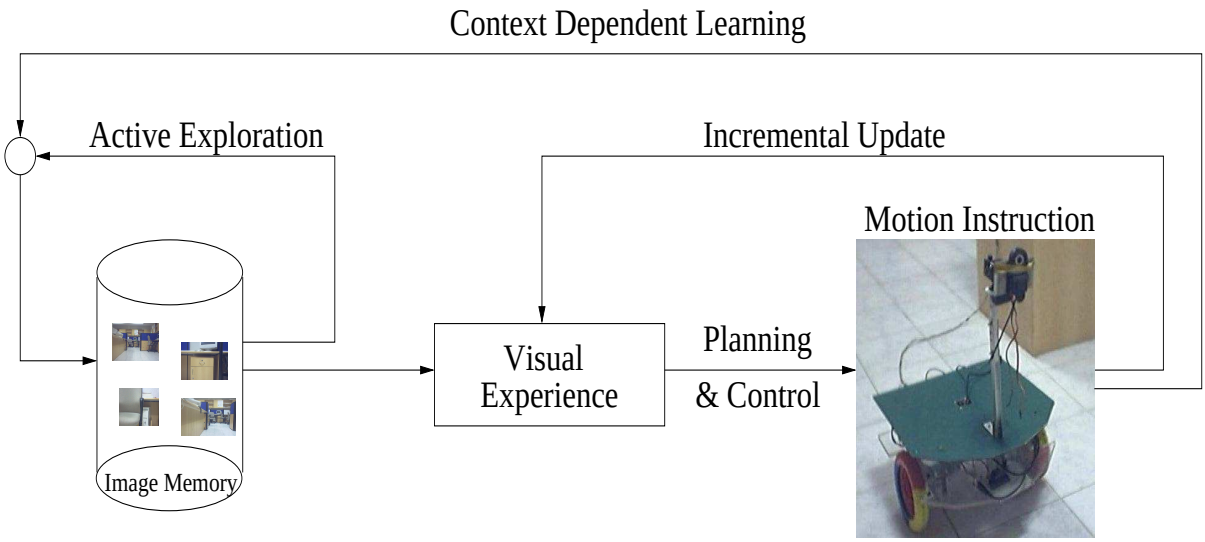


Figure 2.1 Proposed architecture for image-based navigation: Image memory built using an exploration algorithm is employed to construct a visual experience, which is utilized by the localization and servoing algorithms. The feedback received during navigation is used to improve its performance.

4. Knowledge gained during past experiences of the robot can be exploited to improve its performance in later navigation tasks (Reinforcement Learning).

With the above motivations, this thesis proposes the concept of *Online Visual Experience* for the incremental construction and updation of the visual memory of the robot. Further, it also develops strategies for systematic exploration of previously unknown environments and incorporation of the feedback from previous experiences. It must be emphasized that the proposed approach does not focus on the construction or improvement of the 3D map of the environment (unlike earlier approaches) but rather concentrates on improving the visual representation for optimized navigation.

2.2 Proposed Framework

The proposed architecture is illustrated in Fig. 2.1. A database of images describing the environment is first built by a robot. This is done during the exploration phase. These images are utilized to build a compact and statistical representation of the environment, which is referred to as visual experience. Information inferred from the visual experience is used by the localization and servoing algorithms to navigate the robot. Finally the feedback received during the navigation phase is incorporated back into the visual experience using the learning scheme. The following subsections elaborate the characteristics of each component in greater detail.

2.2.1 Image Database

Environmental models can be known beforehand, but gradual changes deteriorate their usability. A better approach is to maintain the model directly by using the images taken by the camera. Therefore an image base is used to efficiently describe the robot workspace. The database consists of a set of images $\mathcal{I} = \{I_1, \ldots, I_n\}$ of the environment taken at various poses $\mathcal{P} = \{P_1, \ldots, P_n\}$ where $P_i = (x_i, y_i, \theta_i)$ in a relative coordinate reference frame $\mathcal{F}$ with respect to an initial camera pose. These locations are the initial reference locations which the robot refers to for locating itself in the environment. The corresponding images are called reference images. These images should sufficiently sample the entire workspace.

2.2.2 Exploration

An exploration algorithm is required for systematically discovering a previously unknown environment or to gather additional information about a partially explored region. It is imperative to design a strategy to automatically obtain the images that sample and describe the entire workspace. The devised algorithm should intelligently infer unexplored regions only from the images captured by a camera and navigate to these locations to increase its knowledge of the environment. Such a strategy is more preferable than limiting the image set to a collection of views obtained during offline training step. The explored regions should be mapped using images rather than 3D mapping.

2.2.3 Visual Experience

Raw sensor information can overwhelm a planner and prevent it from finding a path because of the amount of noise and uncertainty that it contains. Therefore using the images $\mathcal{I}$, a statistically compact representation of the world $\mathcal{V}$ (visual experience) is built

$$\mathcal{V} = f(\mathcal{I}). \tag{2.1}$$

This not only ensures robustness to the effects of noise in the sensor system (due to sensor vibrations, calibration errors, illumination effects etc) but also invariance against different transformations on the scenes such as translation and scale. It produces a compressed form of the original scenes so as to speed up the computation of the comparisons yet maintain distinguishing representations of the scenes. Further, it aids in fusing information extracted from images acquired at different robot poses. We refer to such a representation as *visual experience* as it abstracts visual information (describing the robots experience) acquired during navigation. It can in fact be considered analogous to primary memory for the robotic system (while the image database act as the secondary memory) that makes the localization and servoing algorithms efficient. An important characteristic of such a representation is a facility to incrementally build it using the available images and further update itself upon the availability of alternate images to replace existing images in the database *i.e.*,

$$\mathcal{V}^{'} = g(\mathcal{V}, \mathcal{I}_{new}). \tag{2.2}$$

Such a representation scheme is as powerful as using a complete 3-D model.

2.2.4 Planning & Control

This module constitutes the algorithms required for localization and control of the robot. As the robot navigates in the environment, it acquires an image I^* from its current position and compares it with its visual experience $\mathcal{V}$ to infer the pose of the robot. When the robot finds which one of the reference images is more similar to the current view, it can infer its position P in the environment. With this technique the problem of finding the pose of the robot in the environment is reduced to the problem of finding the best match for the current image in the representation. Once the robot *localizes* itself, a planning algorithm is employed to infer the sequence of images connecting its initial scene view to the desired scene view. These set of intermediate images have some common overlap amongst them which is utilized by the servoing algorithm to perform specified goal-oriented tasks. It must be noted that the servoing algorithm should only utilize information available viz. the visual experience without any requirement of additional *a priori* knowledge of the world. Recent advances in computer vision enable us to design such a control by way of exploiting the constraints and relationships that exist between images [18, 70].

2.2.5 Feedback and Learning

In real world scenarios, all the information necessary to learn is rarely available a priori; rather new pieces of information become available over time using which the knowledge base can be constantly revised. Thus it would be preferable to have an incremental learning system. Specifically, as the robot maneuvers in the environment, it acquires a new set of images $\mathcal{I}^{'}$ describing its workspace. These views provide additional hypothesis about the scene that could be used to update the visual experience (See (2.2)). The improvement could either be in the form of addition of new views into it or replacement of existing images in the database with better images. Further, the incorporation of multiple views (hypothesis) of a feature help in capturing the variability in the geometry of the features. It adapts the representation to reflect the environmental modification that may occur in between the exploration and the navigation stages. The geometric estimates computed using an online visual experience will be more accurate leading to the improvement in the performance of the localization and servoing algorithms.

Additionally while performing assigned tasks, the robot might possess imperfect information about an explored region. One possible solution to circumvent this problem could be to employ geometric techniques to vary the level of detail of the representation (*i.e.*, either in the form of interpolation by employing geometric techniques [35]) or by re-invoking the exploration algorithm. However, this may not be effective all the time. For instance, in case of planning a path from its current position to a destination, it may originally lack the right set of images that would lead to an optimal trajectory. In this case, the system should be made intelligent enough to learn the optimal robot trajectory using the experiences that it gains over time. This facilitates the robot to improve its performance if a similar task

is performed again. It must be emphasized that in this work, geometric information directly available from the images is employed to facilitate learning unlike previous learning-based approaches [51, 80].

The proposed architecture presents visual experience as an extension of the visual memory for the purpose of map representation. As it can be incrementally built and updated, it is referred to as *online visual experience*. The feedback received from the current (and previous) experiences of the robot can be incorporated into the visual experience. Further the exploration technique facilitates expansion of the robot workspace into newer avenues.

2.3 Appearance Modeling as Visual Experience

It must be emphasized that a representation of the world is not something from which the world should be reconstructible. Rather a representation of the world is a statement of facts deducible from observations, and ideally includes enough facts that anything deducible from past observations is also deducible from the representation. A representation is not an analogous structure to the world; it is a collection of facts about the world.

A visually guided mobile robot inhabits a mental world different from the real world. Its observations do not exactly match the real world. Its physical actions do not occur exactly as intended. The task of navigating around the world can be eased by having the world map/representation based on primitives suitable for navigation. The map should also be constructable from visual observations. Conventional approaches have modeled the world as a projection in a two dimensional plane. However a robot may not perform many useful tasks in such a case. It must be noted that the world a robot must operate in is inherently three dimensional even if it can locally move in only a two dimensional plane. Moreover most approaches have an underlying assumption that it is necessary to produce a world model in an absolute coordinate system. This is achieved by calibration techniques, which are often time consuming and are confounded on mobile robots by the fact that the robot itself is not fixed to any coordinate system. Moreover, the sensors and the control systems involved in the calibration process have both systematic and random errors.

In the proposed approach, appearance statistics of the images are utilized to compute an efficient, compact and lower dimensional representation. Appearance may either refer to global image statistics or the local features extracted from the images. Conventional appearance-based methods model the underlying low-dimensional structure of the scene by projecting a training set of images with known camera positions onto a low-dimensional subspace spanned by a few of their principal components. The choice of the representation of the appearances is fundamental for the matching process (i.e., the calculation of the similarity between two images). In the following, a brief survey of the methods available in the literature is provided and the proposed method is then explained.

Eigen-space methods, more popularly known as Principal Component Analysis (PCA), have demonstrated their success in the field of robot localization as well as in face recognition, data compression,

and many other applications. In [58], a parametric eigenspace method was adopted. Scene images taken at various locations were represented as the points in the eigenspace. These points, connected by splines, composed a manifold in the eigenspace. An image taken during actual navigation is projected onto the eigenspace to obtain the current location (as the location for the closest point on the manifold). Pourraz and Crowley [16] compressed a large set of images using the Karhunen-Love transform and used the first few dimensions of the new representation space to capture the significant variations in scene appearance. In [42], active vision is combined with robot localization using PCA. In [3], the study of the problem of batch learning and the use of incremental PCA is presented. Their idea is to deal with on-line learning of the robot landmarks without recomputing the PCA for the whole samples each time. The work done in [37] presents the effect of illumination on PCA. It presents illumination invariant features by filtering the eigenimages rather than filtering the original samples. In [67], a comparison among different appearance-based representation schemes is made. Although their results show that PCA is more robust and accurate than other methods such as edge-density based etc, it remarks that PCA requires more computation power. Most of the devised method until now can be classified as either local or global, based on the feature extraction applied. In the global based approaches, the whole image is considered as a sample and applied to the PCA as a vector. An example of global features is the work done in [6], where PCA is globally applied to panoramic images. It introduces robust PCA using an expectation maximization approach where outliers can be resolved. On the other hand, in the local based approach, a set of landmarks (small patches) are first selected from the image and transformed into vectors to be further handled by PCA. In [36], solutions to the problems related to robustness against occlusions and invariance to the rotation of the sensor were proposed. It defined an eigenspace of spinning-images in which a model of the environment successfully exploits properties of panoramic images to efficiently calculate the optimal subspace in terms of principal components analysis of a set of training snapshots without actually decomposing the covariance matrix.

It must be emphasized that PCA is an appropriate model for data generated by a Gaussian distribution, or data best described by only a second order correlation. It encodes the data based on second order dependencies (pixel-wise covariance among the pixels), and ignores higher-order statistics including nonlinear relations among the pixel intensity values, such as the relationships among three or more pixels in an edge or a curve. However, it is well known, that the distribution of natural images is highly non-linear. To cope with the nonlinearities in the data, in [28], a nonlinear method for learning the low-dimensional pose of a robot from high-dimensional panoramic images was proposed. The local geometry of a point and its nearest neighbors on the manifold were used to project the point onto a low-dimensional coordinate space. An algorithm for Locally Linear Projection (LLP) approximated the mapping from images to camera positions using a locally weighted neighborhood. The image-based position measurements were integrated with odometry information in a Bayesian framework to yield an online estimate of a robots position. However, the method assumes the rotational orientation of the robot is known and considers only translational degrees of freedom. When rotational angle is included

in the pose space, the appearance manifold possesses the topology of a solid torus. In this case, it is impossible to estimate non-Euclidean manifolds.

In the proposed approach, the appearance of a scene is modeled using Kernel PCA (KPCA), which has gained recent popularity amongst the subspace techniques [14, 41]. Kernel PCA, originally proposed by Scholkopf et. al., [64] was investigated as a generalization of PCA. While PCA aims to find a second order correlation of patterns, KPCA takes into account higher order correlations amongst image patterns thus allowing it to model data generated by non-Gaussian distributions. The success of KPCA is demonstrated in the area of image processing, such as face recognition, image de-noising, texture classification and other applications in other different fields. In the following sub-section, the KPCA technique is briefly reviewed and then the proposed method is explained.

2.3.1 Kernel Principal Component Analysis

In PCA, the covariance matrix is computed as $C = \frac{1}{l}\sum_{i=1}^{l}(x_i - \mu)(x_i - \mu)^t$ and the principal components are obtained as $\mathbf{v} \ni C\mathbf{v} = \lambda\mathbf{v}$, where the vectors $\mathbf{v_1}, \mathbf{v_2} \cdots, \mathbf{v_k}$ corresponding to the k largest eigen values are the directions along which the data has maximum variance. Kernel PCA can be derived using the known fact that PCA can be carried out on the dot product matrix instead of the covariance matrix [64]. Let $\{x_i \in R^M\}_{i=1}^N$ denote a set of data. Kernel PCA first maps the data into an higher-dimensional feature space F by a function $\phi : R^M \rightarrow F$, and then performs standard PCA on the mapped data. Defining the data matrix X as $[\phi(x_1)\phi(x_2)\ldots\phi(x_N)]$, the covariance matrix C in F becomes

$$C = \frac{1}{N}\sum_{i=1}^{N}\phi(x_i)\phi(x_i)^T = \frac{1}{N}XX^T. \tag{2.3}$$

It is assumed that the mapped data is centered i.e., $\frac{1}{N}\sum_{i=1}^{N}\phi(x_i) = 0$. The eigenvalues and eigenvectors of C can be obtained via solving the eigenvalue problem

$$\lambda u = Ku, \tag{2.4}$$

where the $N \times N$ matrix K is the dot product matrix defined by $K = \frac{1}{N}X^TX$ with

$$K_{ij} = \frac{1}{N}\phi(x_i)\phi(x_j) = \frac{1}{N}k(x_i, x_j). \tag{2.5}$$

Let $\lambda \geq \ldots \geq \lambda_P$ be the nonzero eigenvalues of $K(P \leq N, P \leq M)$ and $u^1, \ldots, u^P$ the corresponding eigenvectors. Then C has the same eigenvalues as K and there is a one-to-one correspondence between the nonzero eigenvectors $\{u^h\}$ of K and the nonzero eigenvectors $\{v^h\}$ of C i.e., $v^h = \alpha_h X u^h$ where α_h is a constant for normalization. If both of the eigenvectors have unit length, $\alpha_h = \frac{1}{\sqrt{\lambda_h N}}$. It is assumed $||u^h|| = \frac{1}{\sqrt{\lambda_h N}}$ so that $\alpha_h = 1$.

Another perspective to obtain the eigenvectors in the kernel space is to define them in terms of the linearly independent samples. Without loss of generality, assume that a set of m linearly independent

samples $\{\phi(x_1,),\ldots,\phi(x_m)\}(m \leq N)$ span the space where N training samples are distributed. Then the ith eigenvector v_i is represented by

$$v_i = [\phi(x_1),\ldots,\phi(x_m)] \begin{bmatrix} \alpha_{1i} \\ \vdots \\ \alpha_{mi} \end{bmatrix} = \Phi_m \alpha_i \tag{2.6}$$

where $\alpha_i = [\alpha_{1i},\ldots,\alpha_{mi}]^T$ $(i = 1,\ldots,m)$ is a coefficient vector. For a test data x, its hth principal component y_h can be computed using kernel functions as

$$y_h = v^h \phi(x) = \sum_{i=1}^{N} u_i^h k(x_i, x) \tag{2.7}$$

Then the ϕ image of x can be reconstructed from its projections onto the first $H(\geq P)$ principal components in F by using a projection operator P_H

$$P_H \phi(x) = \sum_{h=1}^{H} y_h v^h \tag{2.8}$$

Commonly used kernels include:

- Gaussian Kernel: $k(x,y) = exp(-\frac{\|x-y\|}{2\sigma^2})$
- Sigmoid Kernel: $k(x,y) = tanh(k(x,y) + \Theta)$
- Polynomial Kernel: $k(x,y) = (x,y)^d$

The polynomial kernel has three famous degrees: (d = 1) is the linear classical PCA, $(d \geq 1)$ the polynomial kernel taking into account integer values, and $(0 \leq d < 1)$ is the fractional power polynomial.

In summary, the KPCA algorithm can be summarized as follows.

1. Center the data in F *i.e.,* $\sum \phi(x_i) = 0$
2. Compute the sample covariance for the l vectors in F as $C = \frac{1}{l}\sum_{i=1}^{l} \phi(x_i)\phi(x_i)^T = \frac{1}{l} X X^T$
3. Obtain a basis of kernel principal components (KPCs) $\mathbf{V}$ by diagonalizing C as $C\mathbf{V} = \lambda \mathbf{V}$
4. Using result from linear algebra, all solutions to the above problem (i.e., eigen-vectors) should lie in the linear span of the data vectors i.e., $\mathbf{V} = \sum_{i=1}^{l} \alpha_i \phi(x_i) = \mathbf{X}\alpha$
5. Using the results in Step 2 and 4 in Step 3, we have $\frac{1}{l} X X^T (X\alpha) = \lambda X \alpha$ which yields $l\lambda_i \alpha_i = \mathbf{K}\alpha_i$, where $\mathbf{K}$ is $\mathbf{X^T X}$.

Figure 2.2 Extraction of landmarks

Thus the KPCs are implicitly represented in terms of the inputs (image patches) $\mathbf{x_i}$'s, the kernel $\mathbf{K}$ and a set of linear coefficients α. By using an appropriate kernel function $\mathbf{K}$, the inner product computation in the high-dimensional space is avoided (popularly referred as the *kernel trick* [41]). Note that though references to the feature space vectors are made, they are never explicitly computed. To choose an appropriate kernel function, one can either estimate it from the data or select it *a priori* (in the implementations, a Gaussian kernel was chosen based on an empirical study).

2.3.2 Feature Extraction and Graph Representation

In the training or the exploration phase, the robot initially collects a large number of images of the scene from several different distributed positions. It then extracts sufficient landmarks from the collected images. Landmarks are the parts of an image which hold sufficient characteristic information about the image. Usually a small set of landmarks per image are required. Fig. 2.2 displays some sample images and the landmarks extracted from them. These landmarks are then vectorized. Using these vectors, the KPCA features are extracted as explained in Sect. 2.3.1 above.

Given these features, the visual experience is modeled as a hierarchical topological map. The map resembles a graph structure indicating the connected regions of the environment. The nodes in the graph correspond to images in the environment and edges correspond to paths connecting pairs of images that

can be traversed. Note that this graph is bidirectional as both front and rear views are stored at each pose. The relative pose between two adjacent views can be estimated by using the feature correspondences between them and these estimates can be stored along with the graph. It must be emphasized that methods that provide robust feature correspondences have been recently developed which thereby help in computation of accurate relative pose estimates (see [70] and the references within). Any ambiguity in scale of the pose estimates can be resolved using the coarse odometric estimates. (Odometry between two intermediate views is sufficiently reliable as only relative pose information is being used, and involves no accumulative errors).

2.4 Contribution and Summary

In summary, this chapter has analyzed different aspects involved in successfully conducting the navigation task and has presented a novel image-based navigation architecture based on the concept of visual experience. The framework allows online learning about the world by a robot and capacitates it to autonomously explore and navigate a variety of unknown environments. This is done in a way that facilitates path planning and goal-oriented tasks, using visual maps that are contextually built in the process. It also facilitates the incorporation of feedback received from performing specific goal oriented tasks to update the visual representation. Based on this architecture, the design of the individual algorithms required for performing the navigation task (namely, exploration, servoing and learning) were discussed. Finally, a topological-map based visual representation has also been presented.

Chapter 3

Image-based Exploration

3.1 Introduction

A mobile robot needs knowledge about the environment to plan its actions and to fulfill its mission goals. While many robots can navigate using maps, few can build their own maps. Usually a human must map the territory in advance, providing either the exact locations of obstacles or a graph representing the connectivity between open regions. However, gradual changes in the environment deteriorate the usability of such models for sensor data interpretation. As a result, most mobile robots become unable to navigate efficiently when placed in unknown environments. A better strategy is to explore the environment and maintain the geometric models by using the sensor system. The process of exploration has the potential to free robots from the above limitations.

Exploration is the task of guiding a vehicle through an unknown environment such a way that it covers the entire environment with its sensors while building a map that can be used for subsequent navigation. An exploration algorithm is required for systematically discovering a previously unknown environment or to gather additional information about a partially explored region. A good exploration strategy is one that generates a complete or nearly complete map in a reasonable amount of time. Such a strategy is more preferable than limiting the robot memory to *apriori* knowledge obtained during offline training step. If a robot is able to build and maintain a map by itself, its functionality within the environment will be much improved. To achieve such performance only a minimal set of assumptions about the environment should be made. The algorithm must be reliable and more sophisticated in such a scenario.

In this chapter, some problems faced in performing the exploration task exclusively using a monocular vision sensor are examined and an algorithm to build a reliable map of unknown natural environments in real time is presented. The basic aim is to develop online exploration strategies for inexpensive mobile robot systems working in non-structured domains such as the home, office space etc. The main contribution of this work is a fully autonomous mapping system that operates without the use of active ranger sensors and consistently produces reliable maps of large-scale environments suitable for robotic navigation. A key component of the presented work is the visual map representation. Unlike the vast

majority of mapping paradigms, which employ range measurements derived from sonar, laser or stereo cameras, visual maps make no attempt to infer scene geometry, but rather encode visual landmarks implicitly in the image domain.

3.2 Related Work

Generating maps is one of the fundamental tasks of mobile robots and many researchers have focused on the problem of representing the environment as well as acquiring models using the representation [29, 75]. However, much of the research has been conducted in simulation or with robots that passively observe the world as they are moved by a human controller. The simulations often view the world as a set of floor-plans. For instance, a blueprint view of a typical office building presents a structure that seems simple with straightforward rectangular offices, square rooms, straight hallways. Deng and Papadimitriou [21] investigated the problem of exploring an unknown polygonal room with a bounded number of polygonal obstacles. The length of the path taken by a robot that learns the environment for the first time is compared to the length of the shortest night watchman's tour [15]. Kalyanasundaram and Pruhs [38] considered the problem of conducting a systematic exploration of an unknown environment containing a number of convex polygonal obstacles. Iyengar and Rao [34] developed exploration algorithms inspired by the visibility graph approach to path planning. The problem is modeled in terms of a point robot moving through a 2-D configuration space populated with polygonal obstacles. In this case, perfect sensing is assumed, and the robot learns the visibility graph online.

A real mobile robot may have to navigate through rooms cluttered with furniture, where walls may be hidden behind desks and bookshelves. Though a few systems for autonomous exploration have been implemented on real robots, these robots have performed well only within environments that satisfy certain restrictive assumptions. For example, some systems are limited to environments that can be explored using wall-following [6], while others require that all walls intersect at right angles and that these walls be unobstructed and visible to the robot [9]. Some indoor environments meet these requirements, but many do not.

Another stream of researchers have been analyzing a very similar problem under the domain of Simultaneous Localization and Mapping (SLAM) [76], which has become a very active research topic in the last decade. The problem of SLAM or Concurrent Mapping and Localization (CML) has received considerable attention in the robotics community. The basic problem that is addressed here is to build environmental models or maps from sensor data collected from a moving robot. SLAM is considered to be one of the cornerstones of autonomous mobile robot navigation and is technically challenging because the robot position and the world features must be estimated simultaneously from noisy sensor data. The state of the art algorithms in SLAM can be broadly subdivided into one of the following two approaches (and various hybrids) [76]. One family of methods collects measurements and incrementally builds the map while the robot moves (i.e. in an on-line fashion). Usually the map is represented as a set of landmarks derived from a range sensor, and a Kalman filter is employed to minimize the

total uncertainty of the robot pose and the individual landmark positions that accumulate during robot motion. The second category involves first collecting measurements and then post-processing them in a batch manner. The standard post-processing method is to employ Bayesian framework or Expectation Maximization (EM), again to minimize the total uncertainty of robot poses and landmark positions. This method relaxes the restrictive conditions imposed by Kalman filtering methods.

Several exploration models have also been analyzed in literature. Three basic models that have gained popularity are topological maps, feature-based maps and occupancy grids. Topological maps can be expressed as a graph, where the nodes represent places and the edge represent adjacency, or direct connectivity. Occupancy grids use a 2D array to represent the environment. There, each cell takes one of three values: free space, occupied space or unknown space. Grid-based algorithms have proved to be very simple and quite useful for obstacle avoidance and planning purposes. However, when the size of the environment is large, these models become difficult to handle. Feature-based maps may portray a 2-D or 3-D model. They are another way to represent the environment by using geometric primitives. In general, all these approaches have, in common, the concept of information gain *i.e.*, moving to the destinations in the world that are most informative for mapping and which increase its confidence about its location.

While most of the prior work on mobile robot mapping exploits the use of range data to construct an explicit geometric map, few researchers have also considered the use of vision sensors [18, 68, 77] and visual data. Nayar et al. [58] were among the first to consider the use of a purely appearance-based representation of the world for robot navigation. Several authors have also considered the use of vision-based sensing to extract a geometric map, which can then be used in a more traditional SLAM context. Se, et al [65] extract stereo-based landmarks using a scale-invariant filter, and Davison and Kita [18] considered the problem of actively servoing a stereo head for landmark acquisition as a robot traverses uneven terrain. Finally, Dellaert et al. [20] take advantage of environmental invariants, such as a planar ceiling, to construct a mosaic-like map by registering an ensemble of images.

The Achilles heel of the above algorithms is their assumption that the robot can construct an accurate metric map of its workspace in a global coordinate system. In practice, this is extremely problematic. Once again the main difficulties stem from the fact that it is not easy to determine the position of the robot with respect to an absolute coordinate frame of reference. Whenever the robot encounters new features in the environment, it uses its estimate for its current position to determine where these features should appear in the map. This implies that any errors in the positioning system will be reflected in the map that the robot constructs. Moreover these methods are dependent on human control or active range sensing for planning and obstacle avoidance.

3.3 Proposed Approach

Most image-based navigation techniques assume that a sequence of images are acquired during a human-guided training step, which allows to derive paths for driving the robot from its initial to the

goal locations. To overcome this limitation, for the first time, the problem of systematically exploring an unfamiliar environment by using a single limited-field of view camera is considered. The purpose of exploration is to discover and memorize the unknown regions of the environment so that the robot can navigate reliably throughout the environment. The proposed exploration strategy is different from earlier mapping approaches. Here the mapping is performed directly in terms of images and thus is particularly suited for image-based navigation paradigms.

The central question in exploration is: Given what is known about the world, where should the robot be moved to gain as much new information as possible? Initially, it knows nothing except what it can see from where it is positioned. Its goal is to build a map that describes as much of the world as possible, and as quickly as possible. The basic idea of the image-based exploration algorithm is to acquire the right set of images that are pertinent for performing the navigation task. In this context, the requirement is of a image set $\mathcal{I}^*$ that facilitates optimal motion from one position to the other position in the robot's workspace. Rather than acquiring a large number of images and then arriving at $\mathcal{I}^*$, it would be preferable to have a limited set of views initially and then procure only the required additional images to build $\mathcal{I}^*$. This is the main motivation behind our approach.

Our approach is similar to the popular frontier-based exploration strategy [82]. The central idea behind frontier-based exploration is to gain new information about the world by moving to the boundary between open space and uncharted territory. Frontiers are the regions on the boundary between open space and unexplored space. When a robot moves to a frontier, it can see into unexplored space and add the new information to its map. As a result, the mapped territory expands, pushing back the boundary between the known and the unknown. Once frontiers have been detected, the robot attempts to navigate to the nearest accessible, unvisited frontier. By moving to successive frontiers, the robot can constantly increase its knowledge of the world and extend its map into new territories until the entire environment has been explored.

In the proposed approach, the frontiers are directly inferred from the images. The basic idea is to estimate the obstacle-free regions from the images and navigate the robot towards these regions so as to augment its knowledge about its workspace. In this context, the frontiers are more appropriately referred as 'horizons'. Recent research in computer vision and machine learning facilitates better understanding of images and allows inferring obstacles and obstacle-free regions [32, 63]. These results can be utilized in the current framework to devise an efficient image-based exploration strategy. The robot initially takes an image from its current position. From the images acquired, it infers the horizons (see Sect. 3.3.1). All the detected horizons are maintained in a list. Using this information, it navigates to the next position. A visited horizon is removed from the list of unexplored ones. When the robot reaches a particular horizon, that location is added to the list of previously visited horizon. The robot then acquires new images of the world. It relatively localizes the new view with respect to the previous view and adds the new information to its visual map. Then the robot again detects horizons present in the images and attempts to navigate to the nearest accessible, unvisited one. If the robot is unable to make progress toward it, then it determines that the destination in inaccessible. The robot will then attempt to navigate

to the closest remaining accessible, unvisited one. If the robot encounters an obstacle while in motion, the robot backs up a pre-specified amount along the path it was following, and captures new views of the environment at that location. It continues until no valid horizons are available in the environment.

Note that during the exploration process, at a given pose, the robot may have more than one area to explore. Thus, some unexplored regions are postponed to be explored later. To come back to the unexplored areas, the robot stores both front and rear views observed at each pose. Thus it can use the road-map built during exploration to return to a previous sensing location. The number of intermediate images needed between a path depends on various factors. If the path is straight-forward (e.g., along a corridor or facing a wall), then fewer images suffice, whereas for a complicated one (such as turnings etc), more images may be needed.

If a robot with a perfect map could navigate to a particular point in space, that point is considered accessible. All accessible space is contiguous, since a path must exist from the robots initial position to every accessible point. Every such path will be at least partially in mapped territory, since the space around the robots initial location is mapped at the start. Every path that is partially in unknown territory will cross a frontier. When the robot navigates to that frontier, it will incorporate more of the space covered by the path into mapped territory. If the robot does not incorporate the entire path at one time, then a new frontier will always exist further along the path, separating the known and unknown segments and providing a new destination for exploration. In this way, a robot using the above strategy will eventually explore all of the accessible space in the world. Such an approach balances the desire to see as much of the as-yet-unseen environment as possible, while at the same time having enough overlap and landmark information with the already scanned part of the indoor environment to guarantee good map registration and robot localization.

It must be emphasized that in the proposed approach, no *a priori* scene information is assumed to be available and the environment may consist of obstacles of unknown geometry. In summary, the overall algorithm can be summarized is as follows.

- Obtain Images from the Pan-tilt Camera
- Infer the horizons from the images
- Displace the robot towards the nearest horizon
- Relatively localize its new position (with respect to its previous pose)
- Repeat the above steps until no valid horizons remain

The obtained images, as a result of exploration, are stored in a topological graph as explained in Sect. 2.3. In this graph, each node is associated with its own local coordinate system that is used for navigation within that path. In the case that more than one path exists between a pair of nodes, an edge is stored in the graph for each path. An example of a topological map for a portion of a workspace is graphically illustrated in Fig. 3.1. It displays the map of a rectangular lab environment. The opaque

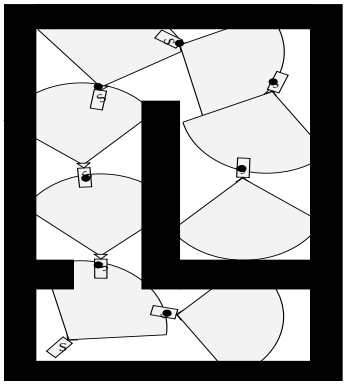


Figure 3.1 Example scenario illustrating the exploration algorithm: Frontiers chosen at each pose and their coverage

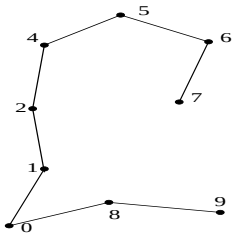


Figure 3.2 Graph built using the captured images

black regions indicate obstacles. The positions that the robot traverses during the exploration are shown in the graph (Fig. 3.2). The frontiers that the robot estimates are indicated by dots.

The above approach has three main advantages. First, it can explore environments containing both open and cluttered spaces. Second, it can explore environments where walls and obstacles are in arbitrary orientations. Third, it can explore efficiently by moving to the locations that are most likely to add new information to the map. In the following subsection, methods for detecting horizons for guiding the exploration process are described.

3.3.1 Inferring Horizons

Two basic method have been explored to infer horizons from the image. The first method is based on the idea of automatic photopopup proposed by Hoeim et al. [32], while the second uses the concept of plane segmentation.

Photo Pop-up In [32], the ability to infer scene geometry from a single outdoor image was demonstrated. Most scenes can be characterized simply by the ground plane, vertical objects i.e., the things that stick out of the ground (usually at right angles due to gravity), and the sky or the roof region. The approach attempts to learn the structure of the world through appearance-based models of geometry. Initially, the models are learnt from a diverse set of images. It then learns to segment the image into

geometric classes (ground, vertical, and sky) and trains classifiers to recognize each of these classes given a segmented image. After performing an over-segmentation into super-pixels, the super-pixels are grouped according to the probability that pairs of super-pixels belong to the same geometric class. The likelihoods are then estimated, which indicate whether each segment is 'good' (i.e., that it is entirely composed of one geometric label) and the likelihood of each possible geometric label, given that the segment is good is computed. The robustness of the algorithm is due to the use of large and varied set of image cues, including color, texture, location, perspective cues, and from combining estimates from multiple segmentations to get the final estimate of the scene geometry.

The above technique was employed to infer the horizons from the image captured by the camera. By making the assumption that the terrain local to the robot is planar, the recovered geometric information can be used to produce a coarse estimate of the geometry of the environment. We then find the geometric label in the image such that the cells are marked traversable if they belong to the ground class and obstacles are constructed at the boundary of the ground/vertical intersection. Each traversable region corresponds to a potential horizon that needs to be visited by the robot. Though the algorithm is robust enough to detect horizons, the major disadvantage is its computational complexity [32].

Plane Segmentation A simpler approach to detect horizons was also analyzed. The approach is based on the concept of identifying free spaces by segmenting the floor plane from the obstacle regions. This is done by partitioning each image frame into a grid of cells in which each cell is compared with a stored template of the floor. A representative grid of 15×15 pixels was used. This grid is correlated with the entire image using a simple sum of squared differences (SSD) method.

$$d(u,v) = \sum_{x,y} \left(I(x,y) - w(x-u, y-v)\right)^2, \tag{3.1}$$

where I is the image, w is the template window and the summation is over positions x,y under the template positioned at u,v.

This step differentiates the floor regions from non-floor regions by marking the former in dark colors (intensities close to zero) while the latter show up with brighter intensities (close to 255). This correlated image is now segmented into super-pixels using a graph-based segmentation algorithm [24] and the average intensity of each super-pixel is computed. As a result of this segmentation, the floor regions can be easily identified by those super-pixels (segments) with their mean intensity below a precomputed threshold. The overall algorithm is summarized below.

- In an offline step, set a 15×15 representative window/template of floor pixels
- Given a new image, correlate the template with the image
- Segment the resultant correlated image
- Calculate average intensity of each segment

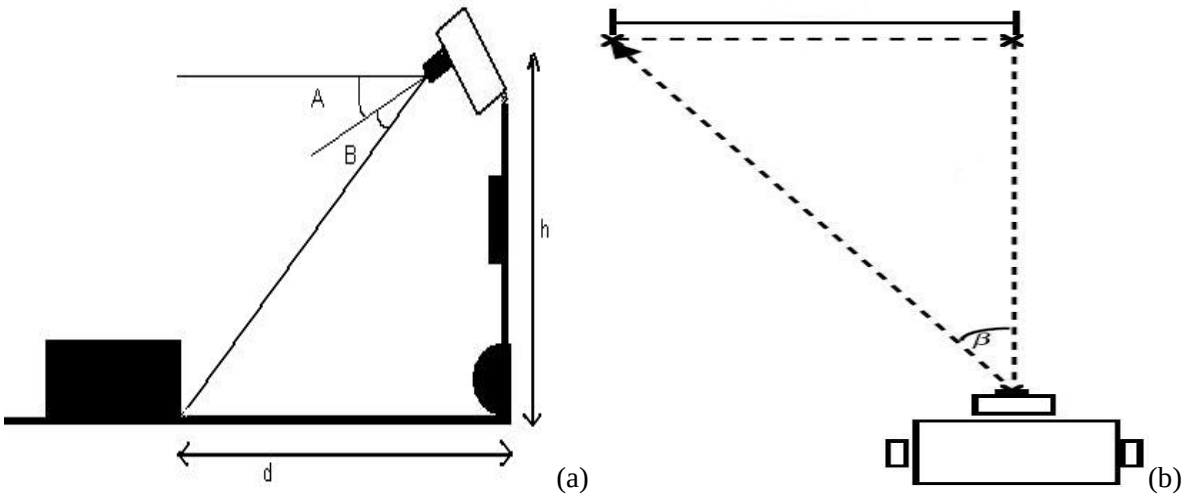


Figure 3.3 (a) Side view (b) Top view of the robot

- Threshold segments with mean intensity below a certain threshold

Though this method is environment specific (due to its assumption of a pre-specified template window), it is computationally efficient and produces reliable and accurate frontier estimates.

3.3.2 Horizon Boundary Computation

Given the pixel locations (x, y) in the image of the mid-points of the horizons, their actual location is computed using simple trigonometric relations. Fig. 3.3(a) shows the side view of the robot while Fig. 3.3(b) shows its top view. The height h of the camera center from the ground plane and its slope A from the horizontal are fixed and assumed to be known. Also the camera is assumed to be pre-calibrated i.e., internal parameters of the camera (the image center and the focal length) are already available. The transformation to the horizon location can be calculated as follows.

Using the concept of similar triangles, the angle B can be defined as

$$B = arctan(\frac{y - c_y}{f_y}), \tag{3.2}$$

where (c_x, c_y) is the camera center and (f_x, f_y) is the focal length. The distance d to the obstacle can be determined as $d = h\ cot(A + B)$. The orientation of the point can be computed as

$$\beta = arctan(\frac{x - c_x}{f_x}). \tag{3.3}$$

Thus the orientation and the distance to the horizon can be obtained.

Hence by using a single vision sensor, a frontier-based visual exploration algorithm has been devised. The method stores no explicit prescriptions for moving between the places but exploits the availability

of an active camera to achieve the task. Further it does not assume any *a priori* knowledge about the geometric structure of the environment nor the availability of any known landmarks. This exploration strategy enables the robot to obtain images of the environment autonomously rather than involving a human-guided teaching step.

3.4 Experimental Results and Analysis

The goal of the experiments presented is to illustrate that a robot can build reliable image maps of its workspace using the proposed method. The exploration algorithm was implemented and evaluated in two real world scenarios. The environment is currently restricted to indoors and the robot is allowed to wander anywhere within it. The robot is placed anywhere inside the lab and it begins exploring. As the robot moves across, it takes every visibility area into account.

In Fig. 3.4, the process of extraction of horizons using Photo Pop-up method is shown. The images consist of a ground floor plane and vertical obstacles. The algorithm identifies the ground plane region from the obstacle region. The intersection between the two indicates the horizon. In Fig. 3.5, the process of extraction of frontiers using the plane segmentation approach is displayed. In these images also, the scene consists of both obstacle and obstacle-free regions. As explained in the algorithm, the template of the ground is used to separate the ground region from the vertical regions. The horizon is set at the intersection of the two regions. Fig. 3.6 displays another example output obtained using the plane segmentation algorithm.

Using the horizons extracted from the image, the robot was navigated through the environment. Fig. 3.7 shows the different steps during the process of exploration. The robot initially starts in a corner of the lab, takes an image and detects the horizons. The robot then navigates to the closest horizon (See Fig. 3.7(b)). Once it arrives at its destination, it adds the new scene views from its new location to the graph and relatively localizes them (See Fig. 3.8). The robot then detects new horizons from these images and navigates to the closest one. Fig. 3.7(c) shows the robot after few iterations where it has explored more of its workspace. In Fig. 3.7(d), the robot has completed its exploration of the entire workspace. The total time required was less than half an hour. An improved version of this system can map the same region in about fifteen minutes. Fig. 3.8 shows the resultant graph built. The approach was also tested in another lab environment. Fig. 3.9 shows the image map built as a result of the exploration process. (The triangles indicate the positions at which the exploration images were captured.) It must be emphasized that the second lab area contained large open spaces as well as small boxes, chairs, tables etc, while the first one was narrow and cluttered with chairs, desks, and workstations. The algorithm was able to perform reliably in either of the environments.

In our experiments, sonars were used for backup safety. In all robotic applications, special attention must be devoted to the safety of the robot and other agents in the environment. Given that the visual map does not encode geometric information, obstacle inference and avoidance requires careful consideration. Sonar is an excellent sensor for reactive obstacle avoidance i.e., for monitoring local obstacles

missed by the visual sensor as the robot moves along. While the robot moves toward its destination, reactive obstacle avoidance behaviors prevent collisions with any obstacles not present while the map was constructed. In a dynamic environment, this is necessary to avoid collisions with, for example, people who are walking about etc. A sonar rather than a laser was used as the latter usually operates in a two-dimensional plane, while the sonar projects a three-dimensional cone. So any object that is above or below the laser plane will be invisible to the laser, but still detectable by the sonar.

3.5 Discussion

This chapter has examined some problems which must be solved by a mobile robot that explores an unknown environment. A new image-based exploration algorithm based on the concept of horizons has been introduced for automatically constructing a map from visual observations that is best suited for the task of navigation. The experiments confirm that by using such maps it is possible to build reliable maps of robot workspace. However one problem with this approach is that it only distinguishes between scanned and un-scanned areas and does not take into account the actual information gathered at each view-point. To overcome this limitation, a more formal notion of information gain (based on the idea of entropy) might be introduced.

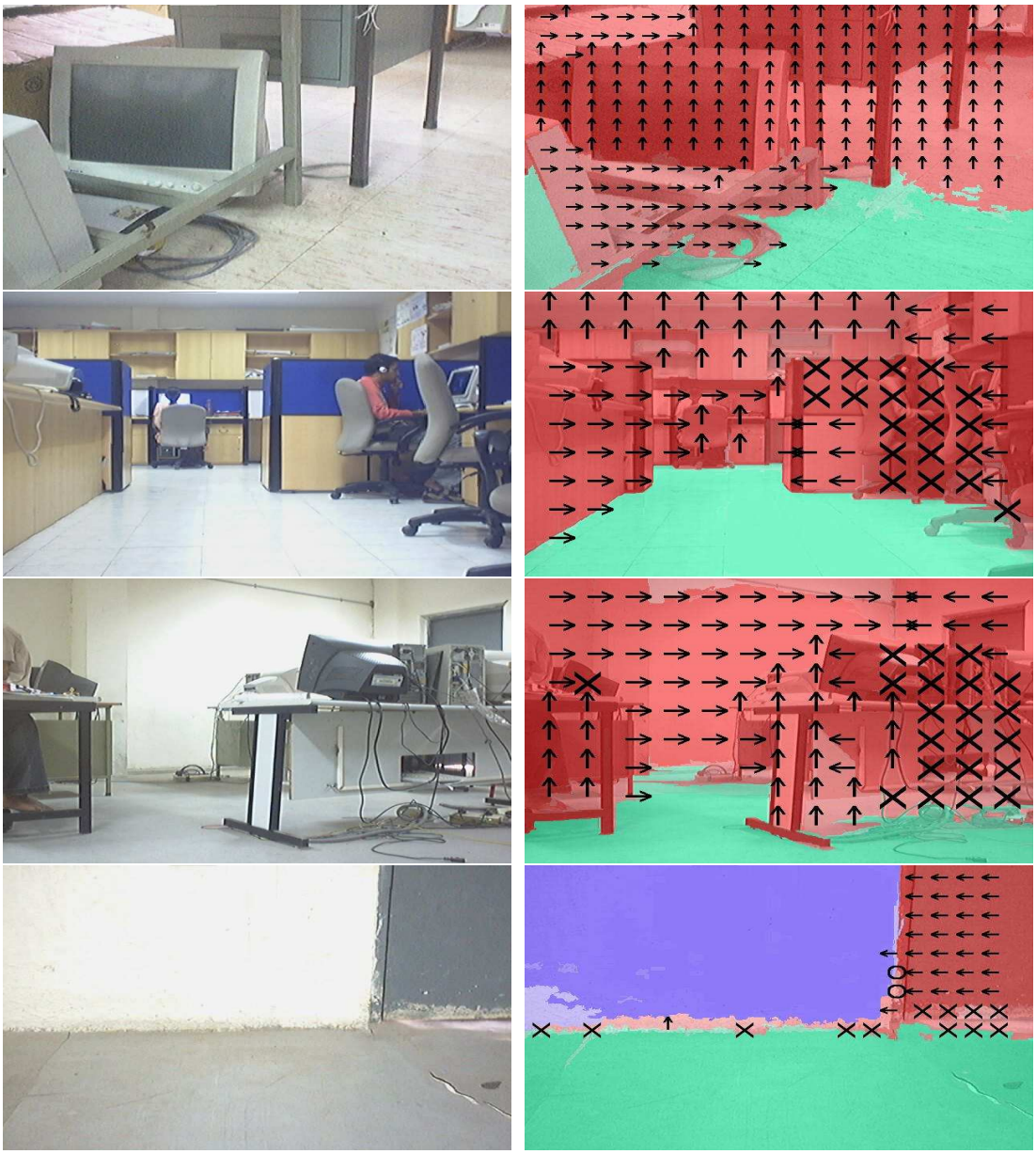


Figure 3.4 Inferring Horizons using geometric context from a single image (a) Original Images (b) Extraction of planes

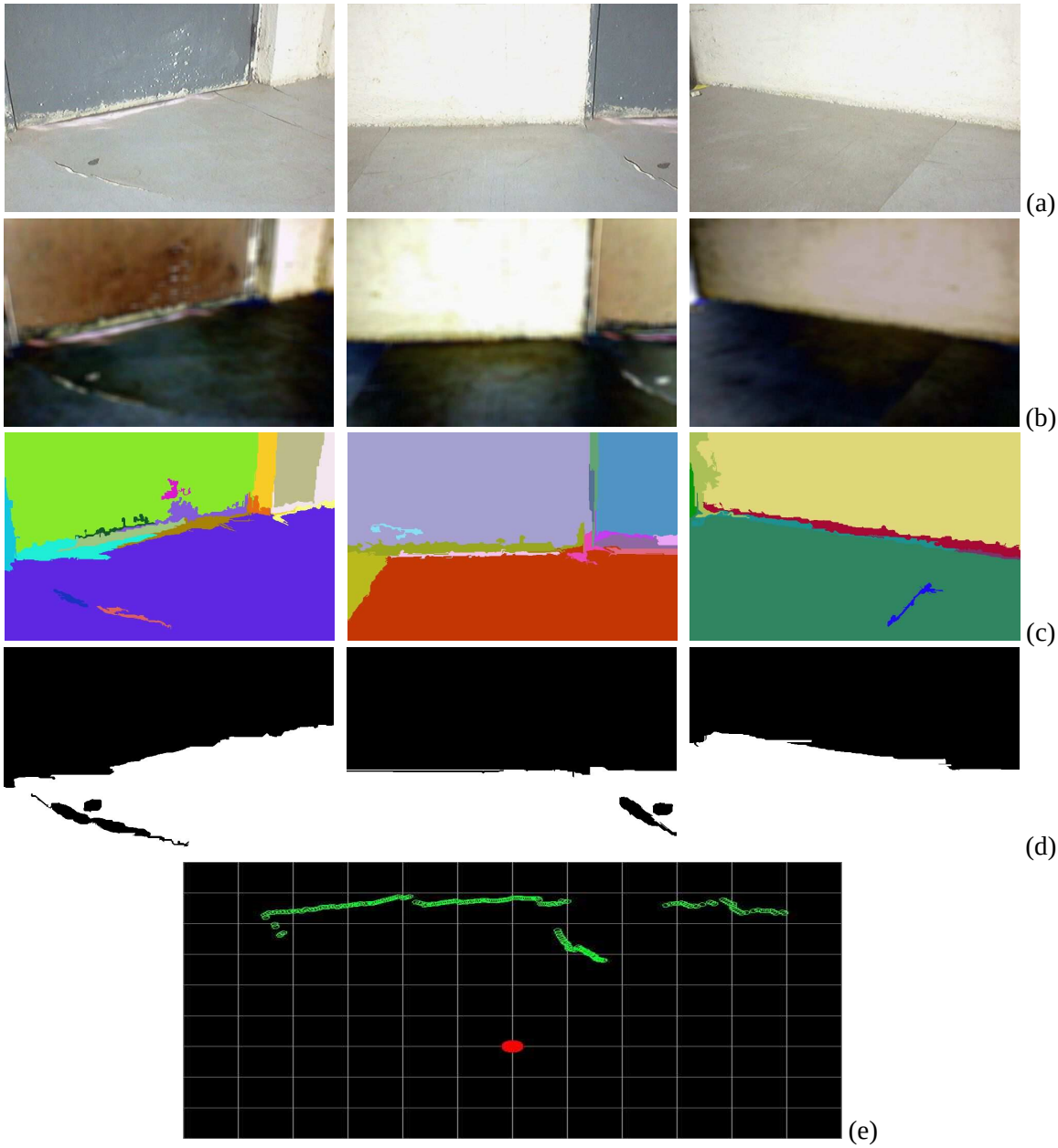


Figure 3.5 Inferring horizons using segmentation-based approach (a) Original Images (b) Correlated Images (c) Segmented Images (d) Horizons or the Frontier Regions (e) Frontier Map

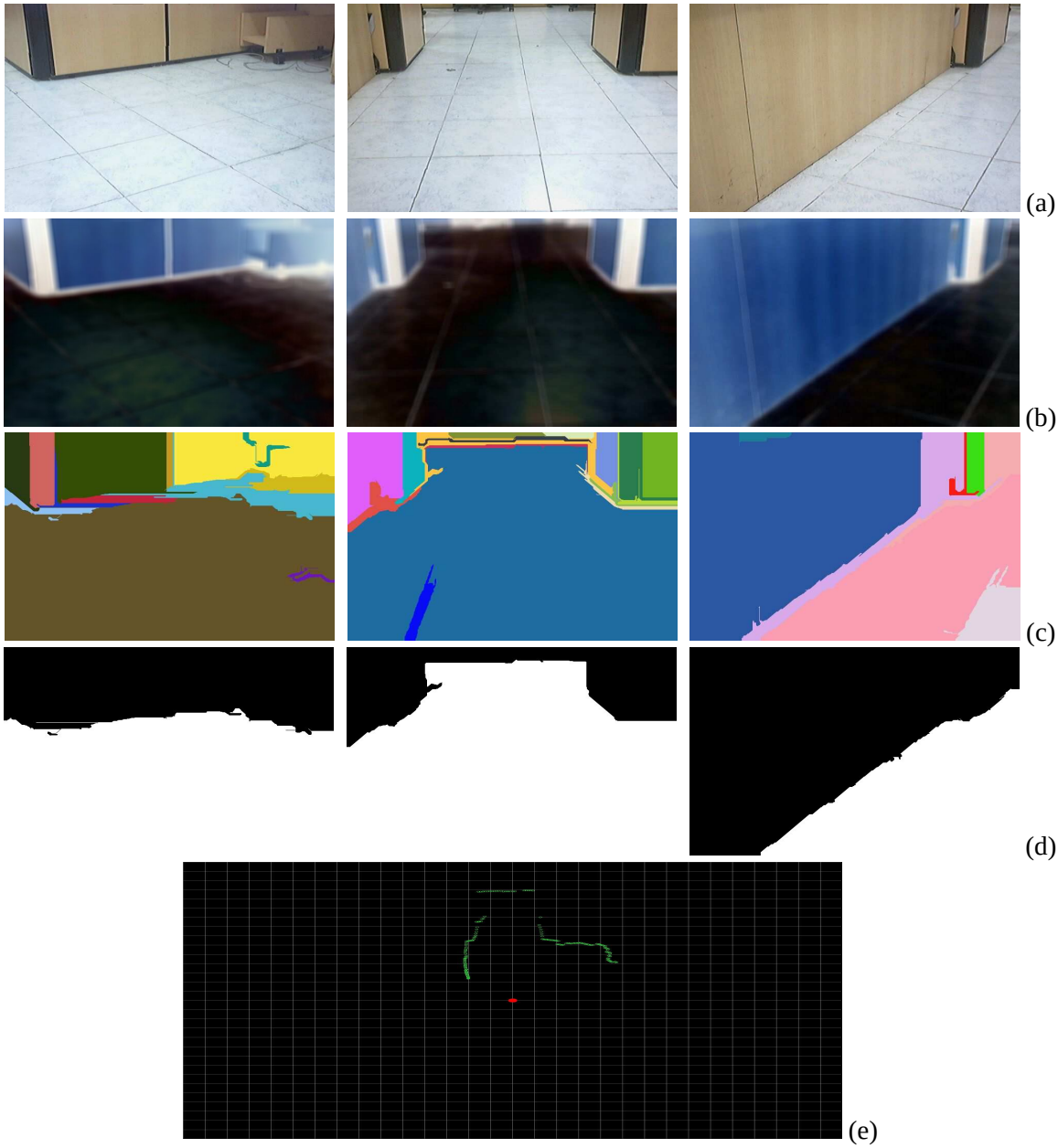


Figure 3.6 Another example illustrating the inference of horizons using segmentation-based approach (a) Original Images (b) Correlated Images (c) Segmented Images (d) Frontier Regions (e) Frontier Map

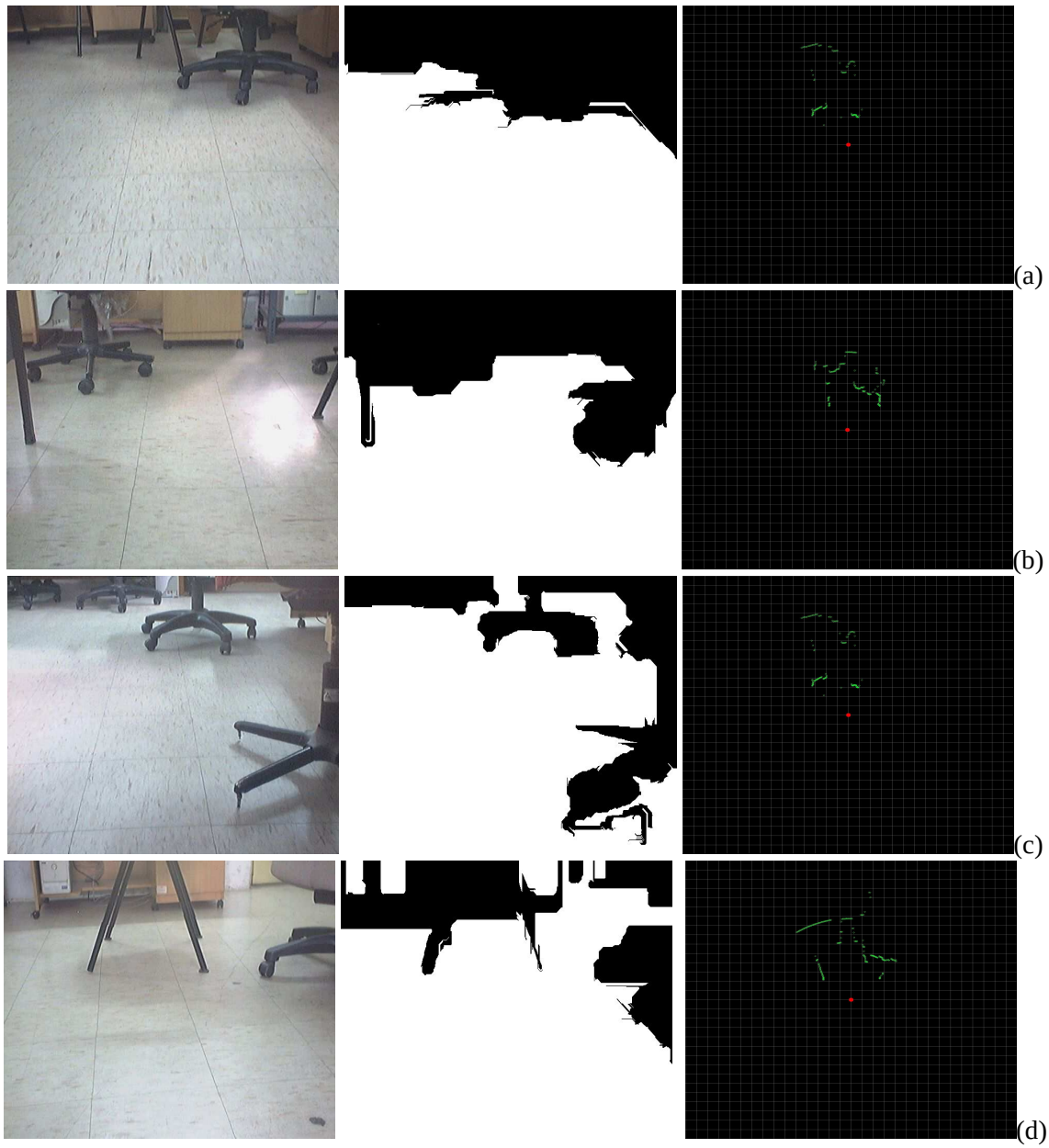


Figure 3.7 Image-based Exploration process (a) Robot Starts (b) Robot moves towards closest horizon (c) Robot after few iterations (d) After completing exploration

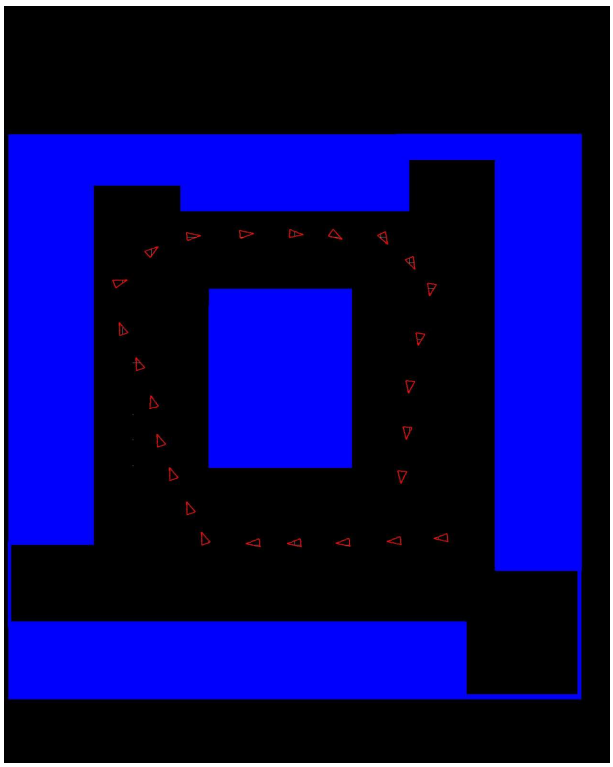


Figure 3.8 Graph built at the end of exploration: The positions indicate the robot poses taken

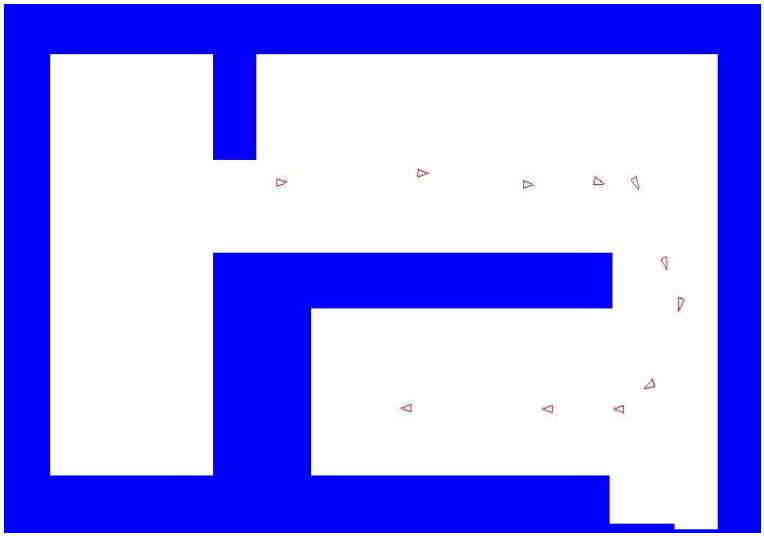


Figure 3.9 Result of the exploration process: The positions indicate the robot poses taken

Chapter 4

Robot Navigation using a Visual Memory

This chapter explains the algorithms required for localization, planning and control of a mobile robot. To find a solution to a problem, it is often easier to split the problem into two or more parts and construct the overall solution by combining the solutions to the subproblems. In robot navigation, this is commonly achieved by splitting the navigation problem into three parts. The first part concerns the localization of the robot which is essentially identifying the current position of the robot in the environment. The next part addresses the issue of planning a path for the robot from its current position to desired goal locations using an abstraction of the environment. The final step is the control part which handles the details of moving the robot using the path planned amidst obstacles.

The following section briefly reviews the literature in the area of navigation. The rest of the sections explain the proposed method for navigation.

4.1 Related Work

Based on the success of image understanding and advances in control theory, recent research in robot navigation has focused on the use of monocular camera-based systems. A significant issue with such systems is the lack of depth information. From a review of literature, various approaches have been developed to address the lack of depth information inherent in monocular vision systems. For example, using consecutive image frames and an object database, Kim et al. [40] proposed a mobile robot tracking controller based on a monocular visual feedback strategy. To achieve their result, they linearized the system equations using a Taylor series approximation, and then applied extended Kalman filtering (EKF) techniques to compensate for the lack of depth information. A drawback of using EKF techniques to estimate depth information is the requirement for linearizion. Song and Huang [71] use spatio-temporal apparent velocities obtained from an optical flow of successive images to estimate the depth information for a monocular guide robot. However, typical drawbacks of optical flow techniques include the need for temporal smoothing and excessive image processing to determine the image flow; resulting in an intensive computational burden for real-time robotic control.

Recently, a monocular visual servo control methodology was developed for unconstrained systems (e.g., robot manipulators) in a series of papers by Malis and Chaumette [47, 48, 49, 50]. The control method exploits a combination of reconstructed three-dimensional (3D) task-space information and two-dimensional (2D) image-space information. The 3D information is reconstructed by decoupling the interaction between translation and rotation components of a Euclidean homography. Advantages of this methodology include no requirement of accurate 3D models of the environment, exploitation of pixel information by the control which increases the potential to force the target to remain in the camera field-of-view, avoidance of local minimas during navigation and finally singularities only exist in the image-Jacobian in degenerate cases. Based on the observation that interaction between the translation and rotation of images can result in slower transient performance due to inefficient camera motions, Deguchi [19] proposed two algorithms for a robot manipulator application that decouple the rotation and translation components using a homography and an epipolar condition. More recently, Corke and Hutchinson [33] also developed a method for decoupling the rotation and translation components from the remaining degrees of freedom using a new hybrid image-based visual servoing scheme. Unfortunately, these approaches typically assume that a constant best guess estimate of a depth-related parameter can be used in lieu of the actual parameter, but the effects of the parameter mismatch are not included in the stability analysis. More recently, Chen et al. [12] developed a homography-based visual servo controller for robot manipulator systems that adaptively compensates for the unknown time-varying depth parameter for a monocular camera-in-hand system.

4.2 Image-based Qualitative Localization

In the localization phase, the robot should compare the features of the actual scene acquired by it using its camera with the stored features. The result of such a comparison would lead to the knowledge of its position in the world. More precisely, as the robot navigates in the environment, it acquires an image I^* from its current position and compares it with its visual experience $\mathcal{V}$ to infer the pose of the robot. In this context, localization is basically achieved by comparing or matching features of input image with stored image database. Note that in conventional model-based approach geometric features such as edges and corners are utilized in the matching process together with the 3D model of objects [67, 42]. In contrast with this, in image-based approach, 2D images are employed directly in the matching process.

The comparison is done by first extracting landmarks of the new images (See Fig. 2.2). Then, features are extracted from these landmarks using the KPCA approach as described in Sect. 2.3. It must be noted that these features are invariant to translation, rotation, scaling and illumination variations. Each new feature is compared with each of the feature stored in the map representation. The resulting comparison leads to the identification of minimum difference between the new features and the original ones. The most similar image to the new image is found as the one which has the largest weighted sum of the detected features, where the weights are according to the feature differences. The difference between a

new feature from the localization phase F_i and a labeled feature F_j is expressed by

$$d(F_i, F_j) = 1 - e^{-\epsilon \|F_i - F_j\|}, \tag{4.1}$$

where the difference is normalized between [0,1].

Once the robot detects which one of the reference images is most similar to its current view (i.e., the localized image), it then infers its relative pose in the environment with respect to this image. This is done by computing the homography [30] between the two views (using the feature correspondences) and then decomposing this homography to obtain the rotation and scaled translation [23]. For more details, the reader may refer to Appendix (Chap. B). It must be emphasized that localization is qualitative in nature as the current robot pose with respect to a reference frame is not searched; rather the retrieval process only informs that the robotic system is in the vicinity of one of the images in the database. Thus, with this technique the problem of finding the pose of the robot in the environment is reduced to the problem of finding the best match for the current image in the representation.

After the robot localizes itself, a servoing algorithm is employed to perform a goal-oriented task. It must be noted that the servoing algorithm should only utilize information available viz. the visual experience without any additional requirement of *a priori* knowledge of the world. Recent advances in computer vision enable us to design such a control by way of exploiting the constraints and relationships that exist between images [70, 18]. It must be emphasized that two image retrievals are successively done in this case: one for the initial image, and one for the desired one.

4.3 Planning Algorithm

Navigation involves both a planner and a controller. The primary goal of the planning process is to compute an optimal path for traversal, depending on the information gained from exploration and the current mission goal. Note that for maneuvering to the goal pose, the robot can only utilize information available in its visual experience. For achieving this, it needs to infer the right set of intermediate images that would lead it to its goal. The paths are planned by the robot using its topological graph of the environment.

This strategy allows to take advantage of the available representation to simplify the global planning task. It must be recalled that the topological graph is a set of nodes and edges where each node in the graph corresponds to a view in the environment and each edge corresponds to a path connecting a pair of views. Most edges in the graph are bidirectional, which means that the edge can be used in both directions (Sect. 2.3).

When a navigation task is defined, the first operation consists of linking the initial and desired images with the image base. Therefore, the nearest image to the initial image and the desired one are first searched in the database (localization). The algorithm then performs an image retrieval step to extract from the database a sequence of images. These images delimit the area of the whole environment that the robot is allowed to traverse to reach its goal. It must be noted that a straight line (or an optimal)

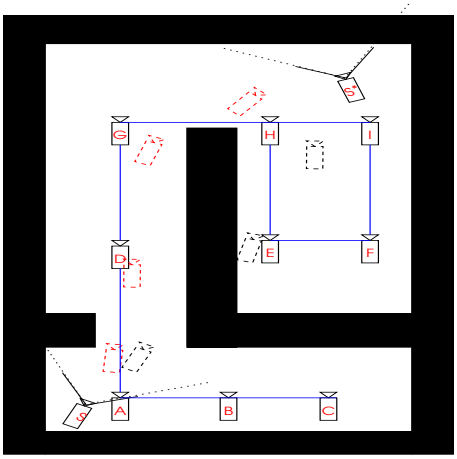


Figure 4.1 Example scenario illustrating path-planning: Optimal path (dotted camera poses) is not realizable due to presence of obstacles. Alternate path (red camera poses) taken by the robot

path may not be realizable due to the presence of obstacles (for instance as indicated by dotted camera positions in Fig. 4.1). Therefore a valid path is computed from the visual experience by searching for the shortest path between the two localized nodes using a standard dynamic programming technique (Dijkstra's shortest path algorithm) with distance (or time taken) between two nodes acting as the edge cost.

This methodology assures that consecutive images in the selected sequence contain enough common features (which is required for the control algorithm) and also that the selected path is the shortest one, with respect to the weighting system used.

4.4 Servoing Control

The goal of a control algorithm is to maneuver the robot along the planned path using hints from the sensor system. Given the current and desired images, the training images with the closest projection to them are found as the result of the localization step. The relative metric pose of the views is then computed with respect to the localized views from the visual experience using the concept of homography. The goal now is to regulate the error in pose so as to achieve the desired pose as illustrated in Fig. 4.2. Asymptotic regulation of the position/orientation of a mobile robot is achieved by exploiting visual servo control strategies inspired by the work given in [13, 49, 60]. By comparing the features of an object in the reference image to features of the object in the current image, image-based geometric relationships are exploited to construct a homography matrix (which relates the actual position and orientation of the mobile robot to a reference position and orientation) despite the fact that a geometric model of the object is not known. By decomposing the homography into separate translation and rotation components, measurable signals for the orientation and the scaled Euclidean position can be obtained. Full Euclidean reconstruction is not possible due to the lack of an object model and the

lack of depth information from the on-board camera to the target; hence, the resulting translation error system is unmeasurable.

It must be emphasized that intermediate images along the path only act as consecutive checkpoints to reach the goal in the sensor space. Therefore perfect convergence towards each intermediate view is not desirable. In such a scenario, an efficient alternative is to use a simple feed-forward control to displace the robot between successive intermediate views. Any error introduced while moving in between currently considered pair of intermediate views can be accounted while computing the relative pose for the next pair.

In the following, the homography-based visual servo control employed in the proposed approach is presented.

Homography-based Visual Control A visual servo control compares the current image of a target with the desired image and the difference (or 'error') is used to drive the camera towards the goal position. Often the task is not just to regulate the image error but also to ensure a realizable camera trajectory. In such scenarios, homography-based control acts as a convenient option as it regulates the error in camera pose by estimating the $3D$ motion parameters only using image information.

Homography is the intrinsic projective geometry between two views of a planar scene that is computed from point feature correspondences. If all the object points lie on a $3D$ plane, their coordinates in the current image I and the goal image I^* are related by a homography matrix or 'collineation' [30].

Assume that a point $\mathcal{P}$ lies on a plane π whose normal vector is n and the distance of the plane from the camera center is d in the camera frame $\mathcal{F}$ as shown in Fig. 4.3. The point expressed in current camera frame $\mathcal{F}$ is related to goal camera frame $\mathcal{F}^*$ by a rotation matrix R and translation vector t as

$$P^* = (\mathbf{R} + \mathbf{t}\frac{n^T}{d})P. \tag{4.2}$$

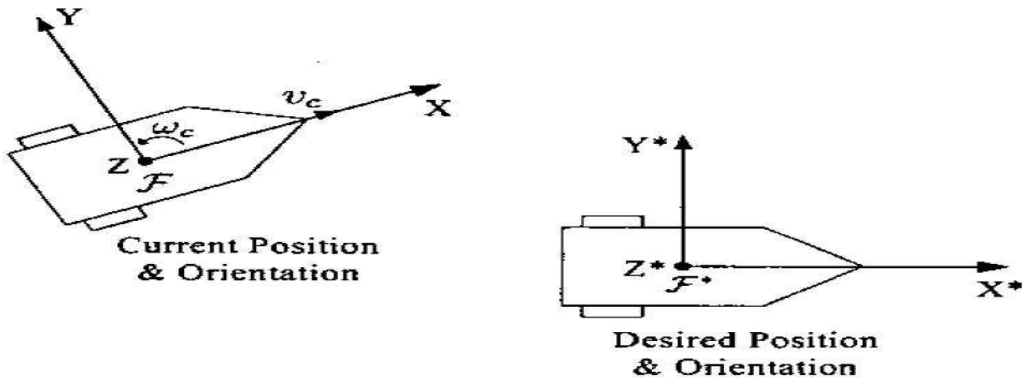


Figure 4.2 Illustration of the mobile robot servoing problem

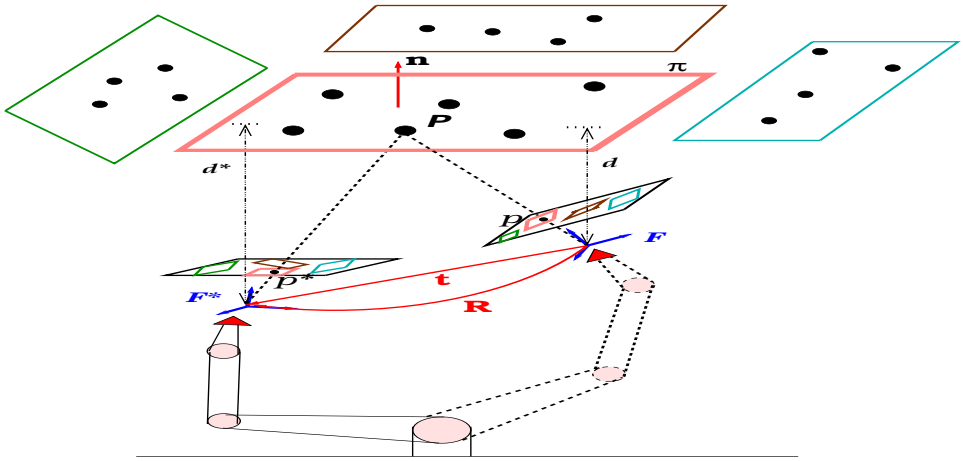


Figure 4.3 Homography-based Visual Servoing

Assuming the camera intrinsic parameters are known, the image coordinates of the points are given by $p = \frac{P}{Z}$ and $p^* = \frac{P^*}{Z^*}$ respectively. This transforms (4.2) to

$$\frac{Z^*}{Z}p^* = (R + t\frac{n^T}{d})p, \tag{4.3}$$

which can be rewritten as

$$\frac{Z^*}{Z}p^* = Hp, \tag{4.4}$$

where

$$H_{3\times 3} = R + t\frac{n^T}{d} \tag{4.5}$$

is called the homography matrix up to a scale factor α [30]. The recovered homography can be decomposed to obtain the rotation matrix R, the scaled translation vector $\frac{t}{d}$ and the plane normal n using the procedure described in [23].

The obtained R and t are applied as the control signals to the robot. The robot is instructed to move a pre-specified distance in the prescribed direction. It then again acquires a new image and computes the homography. The R and t obtained from the new H matrix is now used to control the robot. This process continues until the features in the current image match those in the desired image. More details about the control are elaborated in the appendix (Chap. B).

The goal image is assumed to be reached when the distance between the current feature coordinates and the desired ones fall below a fixed threshold. In case of non-planar environments, an approach based on epipolar geometry may be employed, wherein a essential matrix is computed instead of a homography matrix. However, the rest of the control strategy remains the same. In summary, the servoing algorithm can be described as follows.

- Localize the current and the goal image
- Infer the intermediate views between the two localized views

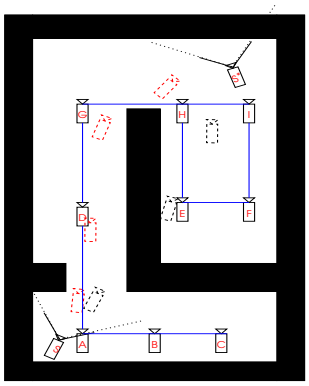


Figure 4.4 Example scenario illustrating the servoing algorithm

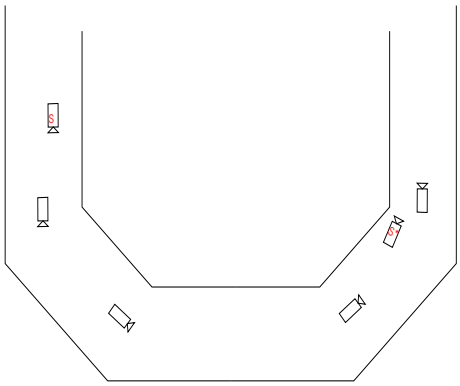


Figure 4.5 Another Example of the servoing strategy

- Compute homography matrix H using the correspondence between two adjacent views
- Decompose H to get R, T
- Use the estimated pose parameters to displace the robot
- Repeat until the error falls below a threshold (*i.e.*, the features in the current and the goal image match)

Fig. 4.4 shows an example of the navigation algorithm. The current robot position is denoted by S and the desired pose is S^*. Images A to I are the *a priori* available views. The localization algorithm would identify the location A as the closest to S and I closest to S^*. Images D and G are identified as intermediate images. Using these images, servoing to the goal pose is performed as described in the above algorithm. The dotted positions (in red) indicate the path taken by the robot. Fig. 4.5 illustrates another servoing example along a corridor workspace.

The basic advantage of this approach is that it does not require any *a priori* knowledge of the scene primitives and it does not force the robot to converge towards each intermediary position in the path.

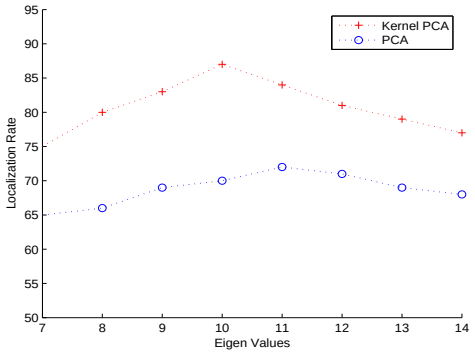


Figure 4.6 Graph showing the localization rate Vs number of eigenvalues using KPC features

In this aspect, it is similar to the recently advocated qualitative paradigms [59]. Further, it takes into account the presence of obstacles in the environment.

4.5 Experimental Results and Analysis

This section presents results of the localization and navigation algorithm to demonstrate their applicability.

To test the ability of localization, different set of images located in the area around the trained regions were acquired. The images exhibit different transformations when compared with the training images. It included images captured, first roughly along the path of the exploration, second in a path that deviates from the one of exploration (about 0.5 meter from the first exploration path), and third with different viewpoints and illumination conditions. The images were all 640×480 8-bit greyscale pixels. As explained in Sect. 2.3.1, the first step for localizing the images is to extract landmarks from the images. Each view has 20 candidate landmarks, taken from gray scale images. Each landmark is 15×15 pixels. Using these landmarks, the kernel principal components are extracted. The extracted features are then compared to the existing features as explained in Sect. 4.2 to infer the matching image.

The localization algorithm was accurate in almost all the considered cases. For the first environment scenario, the recognition rate was around 81% while in the second (lab) environment it was around 83%. In Fig. 4.7, some results are displayed. The first column shows the the current views acquired by the robot in its workspace while the second column displays the retrieved images from the image database that are most similar to the query. The degree of similarity of the retreived image with respect to the input image is also shown. It must be noted that only gray-scale versions of the displayed color images are employed in the algorithm.

We analyzed the performance of the algorithm by varying the number of eigenvalues. Fig. 4.6 displays an analysis of the result. Different number of eigenvalues were used and the localization rate for each given eigenvalue was calculated. Only part of study where the number of eigenvalues vary-

ing between 7 and 14 are illustrated. It can be observed that using around 9 to 11 eigenvalues yielded good performance. Higher eigenvalues lead to lower localization rate because of the increasing size of the feature vector which led to their embracement of noise. Feature vectors with too few eigenvalues performed poorly as they can hardly contain sufficient information required to localize an image. The figure also shows a comparison between PCA and kernel PCA. As displayed, kernel PCA leads to better performance than PCA. This is because of the inherent nature of KPCA to capture the non-linearities in the image features.

Several instances of the navigation algorithm were invoked and its performance was analyzed. As described earlier, the first stage involves the planning step which extracts the set of intermediate images from the database. Fig. 4.8 presents an example of an image sequence extracted from the base. The top row shows the initial and the destination images, while the bottom images are the intermediate images. It can be seen that each couple of images has a common region, which ensures that the environment between the initial and desired views is correctly defined.

Several instances of homography-based control method were also tested. The analysis was done both in simulation and through real results on the mobile robot. Simulations were performed in Matlab environment using a camera with a 512×512 pixel array and a sampling time of $T = 40ms$. An arbitrary configuration of points on a planar surface were considered as the scene features as shown in Fig. 4.9. The figure shows initial F and the final F^* camera coordinate frames along with the feature projections viewed at their respective poses. The image projection of the points at the initial view are displayed in blue, while the green points indicate the features at the desired position. The goal is to displace the features from their initial coordinates to the desired image coordinates while ensuring the features always remain in the camera field of view. Fig. 4.10 displays the image feature trajectories obtained by executing the homography-based control method in this scenario. The figure also demonstrates the exponential decrease in the error of feature coordinates. The application of the devised control yields a smooth velocity screw with exponential convergence. Fig. 4.11 shows the velocity screw obtained in case of the above scenario. It must be noted that both, translational and rotational, velocities achieve final convergence. The figure also displays the final camera trajectory in the Cartesian space.

Fig. 4.12 displays another result obtained using the algorithm. In this case, the features were chosen as contours extracted from an object. The rest of the approach remained the same. It must be emphasized that even in this scenario, the algorithm demonstrates good results and final velocity convergence is achieved.

The simulation experiments were supplemented with experiments performed on the real robot. Fig. 4.13 shows one particular instance of it with the initial view, desired view (that was largely different from the initial one) and a few of the intermediate images in its path. The average error of the servoing algorithm for considered test cases is shown in Table I. For each test case the robot was instructed to move to the same destination ten times from different starting locations that were $80cm$ to $100cm$ away.

Fig. 4.14 displays one instance of the servoing algorithm using the graph-based representation built for one of the lab-environments. The red square indicates the initial position of the robot while the

Table 4.1 Servoing Accuracy

test case	*translational error*	*rotational error*
1	102mm	5°
2	74mm	6°
3	100mm	4.5°
4	80mm	5°
5	95mm	6°
6	75mm	7°
7	98mm	4°
8	84mm	6.5°

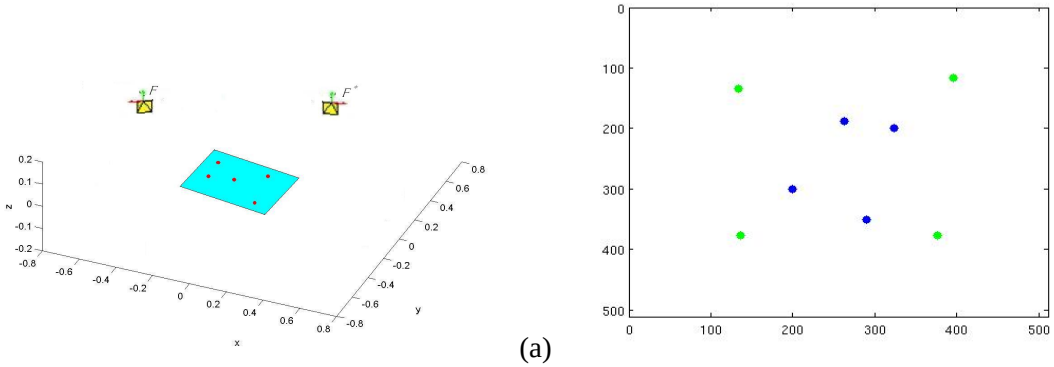
green square signifies the destination. The planning algorithm yielded two possible paths connecting the initial to the desired position (the yellow and the green paths). The green path was chosen as it is the shorter of the two. Using these intermediate images, the homography-based control algorithm was executed and the servoing was achieved. Fig. 4.15 shows the final robot trajectory. As the method only uses intermediate way-points to move towards the goal, high convergence is not required towards them.

4.6 Discussion

This chapter has examined some issues concerning mobile robot navigation and analyzed a homography-based visual servo control to navigate the robot. The control only operates using images and converges to the destination reliably. However one problem with the current strategy is that it is not confirmed to produce control signals that respect the non-holonomicity constraints of the robot. This is a critical problem as the rotation and translation obtained directly from the homography matrix may not be applicable to the robot. Another associated problem is the field of view constraint. It is currently a challenge to devise an efficient control strategy to navigate a non-holonomic mobile robot that would always ensure the object features to remain in the camera field of view.

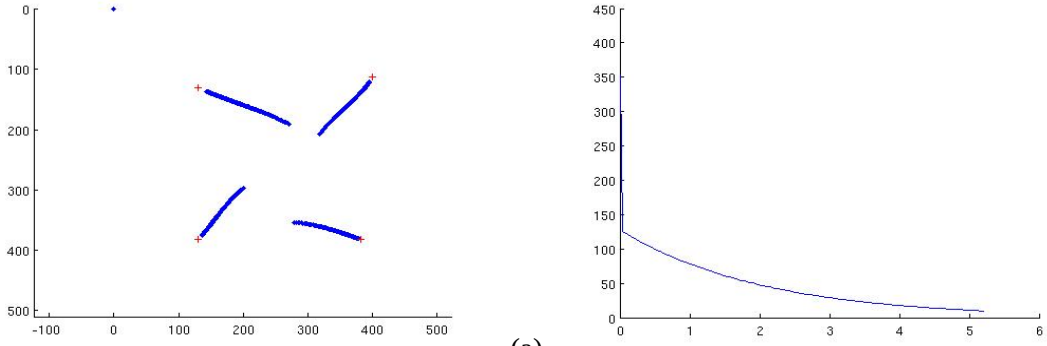
Figure 4.7 Results of the localization algorithm (Left Column: Query Images; Right column: Retrieved Images). The degree of similarity was 0.8, 0.78, 0.75, 0.8 respectively

Figure 4.8 Results of the Planning Step: Top row shows the initial and desired images. The rest of the images (in clockwise) are the selected intermediate images



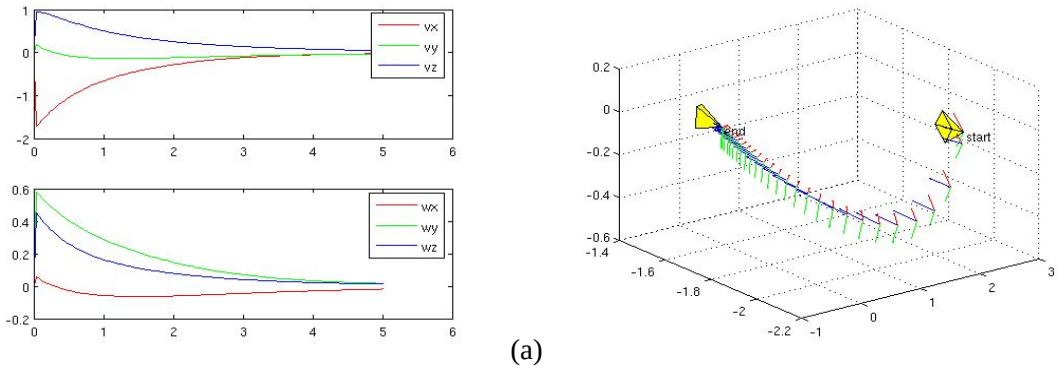
(a) (b)

Figure 4.9 Simulation Set-up (a) Camera Poses (b) Image Views



(a) (b)

Figure 4.10 (a) Image Trajectories: The path taken by the features from initial coordinates to the desired coordinates (b) Feature Error: Exponential decrease in the image coordinates error



(a) (b)

Figure 4.11 (a) Camera Screw Velocity: Convergence is achieved for both translational and rotational velocities (b) Final Cartesian Camera Trajectory

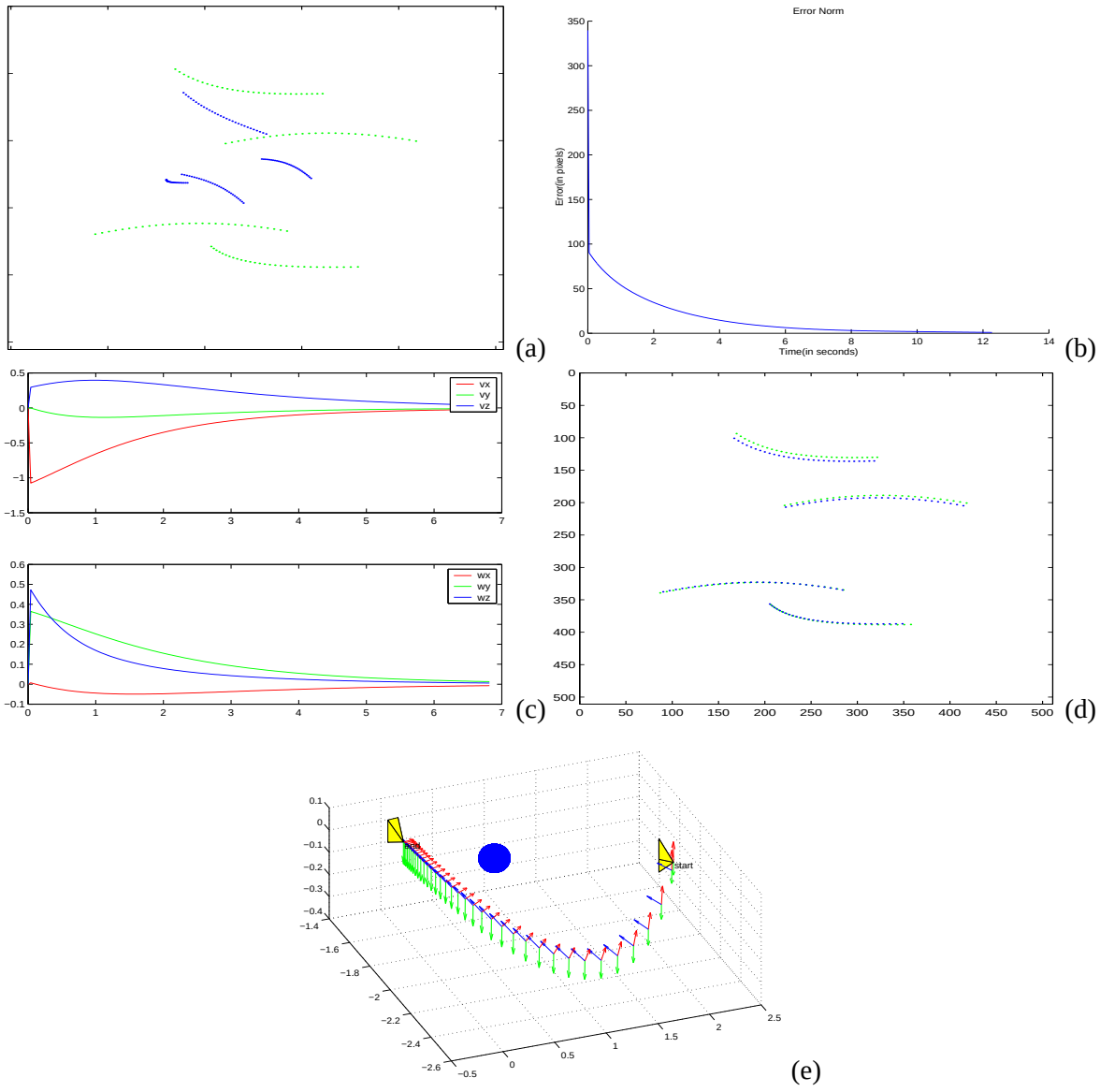


Figure 4.12 (a) Initial and Final Camera Views (b) Feature Error (c) Camera Screw Velocity (d) Final Camera Views (e) Cartesian Camera Trajectory

Figure 4.13 Result of the navigation: (a) initial view (b-k) intermediate views (l) desired view

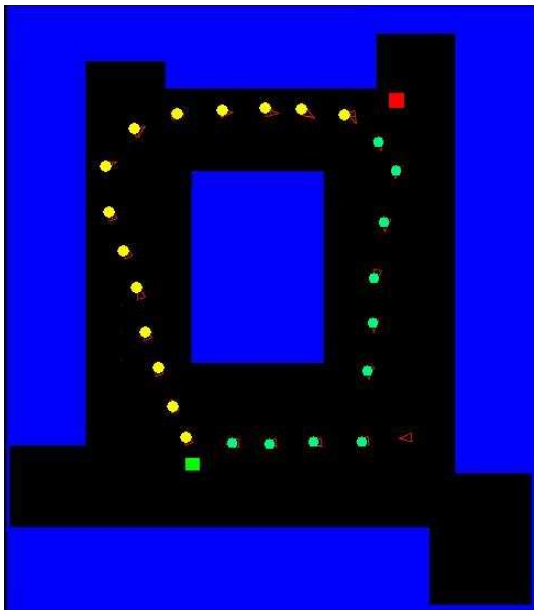


Figure 4.14 Analysis of the Servoing algorithm using the graph-based representation

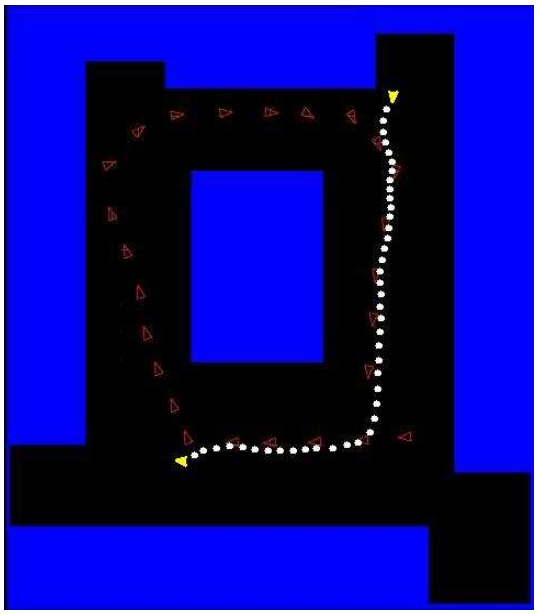


Figure 4.15 Servoing using homography-based Control: Actual Path executed by the mobile robot

Chapter 5

Online Learning for Improved Robot Navigation

5.1 Introduction

Autonomous navigation has long been a challenging task in the research of mobile robots. Throughout the last decades, this field has witnessed a large variety of approaches. Despite significant progress, most of them are specialized to perform a narrow set of tasks in a very restrictive kind of environment. Further, they employ specialized controls that are carefully designed by hand, using extensive knowledge of the robot, its environment and the task it shall perform. If one is interested in building autonomous multi-purpose robots, such approaches face some serious bottlenecks. Firstly, designing a controller requires prior knowledge about the robot, its environment and the tasks it is to perform. Some of the knowledge is usually easy to obtain, but other knowledge might be very hard to obtain. Moreover, certain knowledge (for instance the particular task one wants a mobile robot to do) might not be accessible at all at the design-time of the robot. Further, making domain knowledge available demands hand-coding explicit models of robot hardware, sensors and environments, which requires tremendous amounts of programming time. As robotic hardware becomes increasingly more complex, and robots are to become more reactive in more complex and less predictable environments, the task of hand-coding a robot controller will become more and more a cost-dominating factor in the design of robots. Finally even if the robot, its environment and its goals can be modeled in sufficient detail, generating control for a general-purpose robot is of enormous computational complexity.

The idea of having a robot learn to accomplish a task, rather than being programmed explicitly is an appealing one. It seems easier and much more intuitive to specify what the robot should be doing, and to let it learn the fine details of how to do it. Machine learning is essentially concerned with the design of such algorithms which, rather than encoding explicit instructions or programs for the solution of specific tasks, encode inductive mechanisms whereby solutions to broad classes of problems may be derived from examples.

The traditional formulation of the machine learning problem has been as a classification problem. Informally, some domain of individuals are provided for which a general classification is required. The classification is given by a function from this domain to a some small finite set corresponding to the

classes. Examples are available as a training set which provides the class of some typical individuals. In addition, some background knowledge relevant to the inductive task at hand may be available. From this, the general classification of the individuals must be induced.

Learning provides a useful tool for the automatic design of autonomous robots. A robot working in similar environments for long periods of time should improve its performance overtime. Initially its performance may be suboptimal but overtime it gains additional knowledge that should be utilized to gain optimal behavior. Learning paradigms enable a robot to collect the required knowledge on-the-fly, through real-world experimentation. If a robot is placed in an unknown environment, or faced with a novel task for which no a priori solution is available, a robot that learns shall collect new experiences, acquire new skills, and eventually perform new tasks all by itself.

In the following sections, a brief summary of the related work is presented followed by a quick review of two popular learning paradigms. The proposed learning algorithms are then explained.

5.2 Related Work

Learning techniques have frequently come to bear in situations where the physical world is extremely hard to model by hand. For example, Pomerleau describes a computer system that learns to steer a vehicle driving at high speed on public highways, based on sensor data from a video camera [17]. Learning techniques have also successfully been applied to speed-up robot control, by observing the statistical regularities of typical situations (like typical robot and environment configurations). For example, Mitchell [57] describes an approach in which a mobile robot becomes increasingly reactive, by using observations to compile fast rules out of a database of domain knowledge. In [78], a robot manipulator is described which learns to insert a peg into a hole without prior knowledge regarding the manipulator or the hole. Maes and Brooks [46] have successfully applied learning techniques to coordinate a leg motion for an insect-like robot. Their approach too, operates in the absence of a model of the dynamics of the system.

In principle a robot could learn any task from scratch given enough time. In practice, however, this time is too high for most complex tasks, and thus prior knowledge has to be incorporated into the learning process. In this context, it must be noted that approaches to machine learning can be divided into two broad categories: inductive learning and analytical learning. Inductive learning techniques generalize sets of training examples via a built-in, domain-independent inductive bias. They typically can learn functions from scratch, based purely on observation. Analytical approaches to learning generalize training examples based on domain-specific knowledge. They employ a built-in theory of the domain of the target function for analyzing and generalizing individual training examples. Both families of approaches are characterized by opposite strengths and weaknesses. Inductive learning mechanisms are more general in that they can learn in the absence of prior knowledge. In order to do so, however, they require large amounts of training data. Analytical learning techniques learn from much less training data, relying instead on the learners internal domain theory. They hence require the availability

of an appropriate domain theory. In mobile robot domains, large amounts of training data is typically hard to obtain due to the slowness of actual robot hardware. Therefore, analytical learning techniques seem to have a clear advantage. Their strong requirement for accurate domain knowledge, however, has found to be a severe obstacle in applying analytical learning to realistic robotics domains. Thrun [74] presented a explanation-based neural network learning algorithm which integrated both analytical and inductive learning by a smooth blending of both learning principles depending on the quality of the available domain knowledge. Neural network learning methods generalize from observed training data to new cases based on an inductive bias that is similar to a smooth interpolation between observed training points. Theoretical results on learnability, as well as practical experience, show that such purely inductive methods require dramatically increasing numbers of training examples to learn functions of increasing complexity. Explanation-based neural network learning is a method that generalizes from fewer training examples, relying instead on prior knowledge encoded in previously learned networks that encode domain knowledge.

Smart and Kaelbling addressed such practical issues on a real mobile robot [69]. The tasks investigated were wall following and obstacle avoidance. Learning was carried out in two phases: first, with the control policy being provided by a pre-programmed controller or a human with a joystick, and second, using the learned policy of the robot. Gaskett et al. [25] considered training of a mobile robot to wander (obstacle avoidance) and pursue a target using real-time vision. This was implemented in a subsumption architecture, such that target pursuit takes over from wandering when a valid target is detected.

5.3 Preliminaries

Learning paradigms can be broadly categorized into two major classes, namely, Incremental Learning and Reinforcement Learning. The following subsections briefly review each of the two paradigms.

5.3.1 Incremental Learning

An implicit assumption in conventional learning approaches is that the training set is available a priori and that learning ceases once this set has been duly processed. Henceforth, the induced classification is used exclusively to make predictions about new instances. Such an approach to learning is clearly not all-encompassing. There are a number of interesting situations where learning must take place over time, in a kind of continuous fashion rather than as a one-shot experience. For instance, in the past, most of the autonomous navigation systems relied on tracking specific features, such as lane markings of outdoor roads, floor or ceiling edges of hallway, while others detected road regions based on features such as color or texture. All of these systems had a strong reliance on the a priori model about the road's appearance, and hand-crafted rules were implemented as the detection algorithm. Unfortunately, these features may not always be reliable due to a change of environmental conditions. Some roads may not

have clear lane markings, or some none at all. The variation in illumination and the road conditions can often invalidate the underlying assumptions used in the vision algorithms. In order to deal with varying environmental conditions a mobile vehicle may experience, it is necessary for the system to employ adaptive or incremental mechanisms.

At the heart of the distinction between these forms of learning is the notion of incrementality. It is argued that incrementality is rather ubiquitous in learning and that the most natural and flexible way to tackle incremental learning tasks is with incremental learning algorithms. A learning task is incremental if the training examples used to solve it become available over time, usually one at a time. Note that, if one is prepared to wait long enough, any incremental learning task can, in principle, become a non-incremental one. Hence, for incremental learning tasks, there is an implicit assumption that waiting is undesirable and/or impractical. In particular, the nature of the application may render unfeasible the timely generation of a sufficiently large number of representative examples, either because the environment changes in time (and thus learning becomes situated or context-sensitive) or because the rate at which examples become available may be too slow. For example, a robot's environment is changing and often unpredictable. Hence, in order to survive and carry out its tasks successfully, a robot must be able to react and adapt incrementally to the environmental cues. With incremental learning, the training can be done on-line. This is very important for autonomous navigation, since the information in input images is huge and highly redundant. The system only needs information which is necessary for the navigation task [56].

5.3.2 Reinforcement Learning

Another popular learning paradigm that has has been extensively studied by researchers for autonomous robot navigation is Reinforcement Learning. Fig. 5.1 depicts a typical reinforcement learning system. The learner receives descriptions of the environment, which are called states, from peripheral sensors and chooses actions to perform. The effect of an action to the environment is evaluated by a critic or a teacher, which could either be a person or some special sensors, and fed-back to the learner in the form of positive or negative rewards. The mission of the learner is to find the action rules to optimally achieve a certain goal through its interaction with the environment. The learning algorithm handles continuously valued states and actions and can learn from both good and bad experiences.

A supervised learning approach would require a model of 'good behavior' from a teacher. Its performance would be limited by the ability of this teacher. Reinforcement learning requires only a critic, that gives scalar rewards (or punishments) based on behavior. Providing a critic only requires that we have some measure of whether a task is being achieved, we do not need to know how to achieve it. The reward signal need not be given immediately when an action is performed, as the true effect of an action can manifest itself after some time. Punishment is also given for use of energy and large changes in motor commands. Behavior improves based on knowledge of which actions led to rewards and punishments. In this way, both good and bad experiences are a valuable part of the learning process.

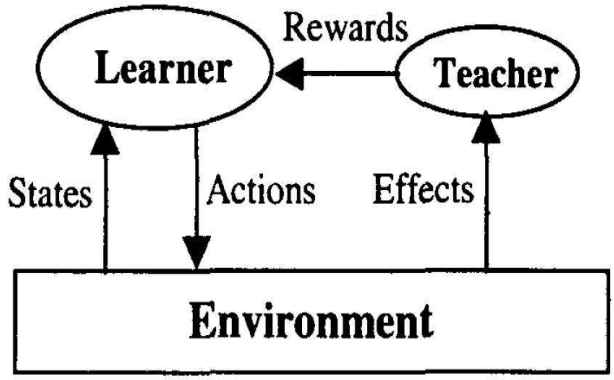


Figure 5.1 A typical structure of reinforcement learning

A simple and straightforward implementation idea of reinforcement learning is assigning each performed action an instant reward and adjusting the corresponding action policies right away. To overcome the drawbacks in the instant-rewarding method, the idea of *Delayed Rewarding* was introduced. Instead of evaluating actions separately, delayed-rewarding schemes analyze the effect of action sequences as a whole and the action policies are adjusted for maximizing the expected future discounted rewards. Provided that the system model, which includes the state transition matrix and the rewarding policy, is known, delayed-rewarding problems can be easily solved with traditional Dynamic Programming (DP) approaches. However, obtaining a precise model for each learning task is actually not easy, especially when the number of states is large. An elegant solution to model-free learning problems is offered by the *Temporal Difference* (TD) method [56]. Action policies are incrementally updated under the TD learning paradigm by consecutively exploring the outside world. Compared with the model-based approaches, TD methods learn more robustly, especially for real-time robot control tasks, since it does not require an accurate model of the world.

Q-Learning One of the most popular reinforcement learning method is the Q-learning paradigm. Q-learning is particularly interesting because it can learn from actions which it did not itself suggest, such as those from another controller, or historical data (which is often referred as being exploration-insensitive) [79]. Q-learning works by incrementally updating the expected values of actions in states. For every possible state, every possible action is assigned a value which is a function of both the immediate reward for taking that action and the expected reward in the future based on the new state that is the result of taking that action. This is expressed by the one-step Q-update equation:

$$Q(s,a) = r(s,a) + \gamma \ \arg\max_{a'}(Q(s_{t+1},a') - Q(s,a)), \tag{5.1}$$

where Q is the expected value of performing action a in state s, r is the reward and γ is the discount factor. The discount factor makes rewards earned earlier more valuable than those received later. The Q-values implicitly describe a controller that measures the state, then choose the action with the highest Q value.

Reinforcement learning, and in particular Q-learning, seems to be a natural choice for learning control policies on mobile robots. Instead of designing a low-level control policy, a much higher-level task description in the form of the reward function can be designed. Designing a sparse reward function is generally easier than designing the low-level mapping from observations to actions. Often, for robot tasks, rewards correspond to physical events in the world. This makes its easy to come up with simple reward functions for many tasks. For example, for an obstacle avoidance task, the robot might get a reward of $+1$ for reaching the goal, and -1 for hitting an obstacle. In theory, this is all that is necessary for the robot to learn the optimal policy.

5.4 Learning as Online Visual Experience

Because of the fast development of computer vision techniques, vision-based robot navigation has been intensively studied in the recent years, and model-based methods are often used in practical experiments. However, the establishment of environment models is really a time-consuming work. It is thus desired to design a process for automatic modeling of navigation environments. With this process, it is not necessary to measure the environment manually. Instead, the robot is just driven manually once along the desired path, and all jobs about initial model learning will be automatically accomplished without human involvement. However, certain problems arise when a fully automatic model establishing process is performed. The noise of image processing will reduce the accuracy of the obtained model. Since the noise coming from image processing will not appear at the same place in each navigation cycle, it is possible that more training using multiple navigation data will reduce the inaccuracy. This means that after initial model learning, we could utilize the information recorded in each navigation cycle to update the original coarse model, and hopefully a more reliable refined model could be obtained after several navigation trainings. This leads to our study of incremental learning for robot navigation in indoor environments.

5.4.1 Incremental Learning for Improved Robot Localization

In real world scenarios, all the information necessary to learn is rarely available a priori; rather new pieces of information become available over time using which the knowledge base can be constantly revised. Thus it would be preferable to have an incremental or online learning system. The primary goal of learning here is to equip the robot with the capabilities to explore the environment with its sensors, construct an appropriate model of the environment, and navigate efficiently in the learned environment. The proposed approach consists roughly of three stages. The first stage is initial training, in which the robot is driven to explore the environment while images captured by the camera and the control status data are recorded (Chap. 3). Then, a certain off-line procedure is performed to construct the initial model (Chap. 2). The second stage is to allow the robot to navigate automatically along the desired path (Chap. 4). And the final stage is to update the learned model with the information collected in the

previous navigation tasks. A more precise model is obtained after each navigation and update iteration. The second and the third stages, namely, navigation and model updating, may be repeated several times in order to obtain a more reliable model.

Specifically, as the robot maneuvers in the environment, it acquires a new set of images $\mathcal{I}^{'}$ describing its workspace. These views provide additional hypothesis about the scene that could be used to update the visual experience. The improvement could either be in the form of addition of new views into it or replacement of existing images in the database with better images. Further, the incorporation of multiple views (hypothesis) of a feature help in capturing the variability in the geometry of the features. It also adapts the representation to reflect the environmental modification that may occur in between the exploration and the navigation stages. The geometric estimates computed using an online visual experience will be more accurate leading to the improvement in the performance of the localization and servoing algorithms.

An online version of the KPCA algorithm was utilized to perform the incremental updates to the constructed visual experience. Incremental KPCA (IKPCA) [14] is a recent development in machine learning research. Here it has been adopted for the task of vision-based navigation. The basic idea is to extend a linear PCA updating algorithm [27] to the non-linear case by utilizing the kernel trick. Conventional Incremental Principal Component Analysis (IPCA) algorithm mainly consists of two operations: augmentation of an eigen-axis and the rotation of eigen-axes [2]. In addition to the above two operations, the IKPCA algorithm also includes the check on the linear independence of the new sample to the samples given so far. This is because, although the dimensions of KPCA feature space could be very large or possibly infinite depending on kernel functions, the feature space degenerates into a lower dimensional space whose dimensions correspond to the number of linearly independent mapped input samples. Thus these samples need to be kept in memory to represent eigenvectors during the incremental learning. The third operation ensures the linear independence of these samples.

Checking of Linear Independence Assume that N samples $\{x^1, \ldots, x^N\}$ have been given of which $\{x_1, \ldots, x_m\}$ are linearly independent. To determine the independence of a new sample x, we calculate the new kernel matrix K' in which the kernel functions of x and the independent samples are added to the last row and column of the current kernel matrix K.

$$K' = \left[\begin{array}{ccc|c} & & & k(x_1, x) \\ & K & & \vdots \\ & & & k(x_m, x) \\ \hline k(x, x_1) & \ldots & k(x, x_m) & k(x, x) \end{array}\right] \tag{5.2}$$

The dimensionality of the feature space spanned by the eigenvectors and x is equivalent to the rank of K' in (5.2). Therefore, the linear independence of x can be judged by the rank of K'. If K' is a full rank matrix, then x is linearly independent to $\{x_1, \ldots, x_m\}$; then $m \leftarrow m + 1$ and $K = K'$. Otherwise, x is judged to be dependent, and there is no change to the matrix K.

Augmentation of Eigen-axis The decision to augment the eigen-axis is based on the accumulation ratio. Accumulation ratio is defined as the ratio of d eigenspace components to the entire set of eigen components.

$$A(d) = \frac{\sum_{i=1}^{d} \lambda_i}{\sum_{i=1}^{m} \lambda_i} \tag{5.3}$$

The minimum number of components d is chosen such that $A(d)$ is greater than a threshold value θ. Given a new sample, the updated A value can be computed as follows.

The ith eigenvalue corresponds to the variance of the projection of $x^j\{j = 1, \ldots, N\}$ onto the ith eigenvector v_i (from the famous result $CV = \lambda V$). Given a new sample, the new variance $\sigma_i^{'2}$ is given as

$$\sigma_i^{'2} = \frac{1}{N+1}\left[\sum_{j=1}^{N}\{v_i^T(\phi(x^j) - c')\}^2 + \{v_i^T(\phi(x) - c')\}^2\right], \tag{5.4}$$

where c' is the new mean of the given samples defined as

$$c' = \frac{1}{N+1}\{Nc + \phi(x)\}. \tag{5.5}$$

The numerator and the denominator of the updated accumulation ratio $A'(d)$ are given by

$$\sum_{i=1}^{d} \lambda_i^{'} = \frac{N}{N+1}\sum_{i=1}^{d}\left(\lambda_i + \frac{1}{N+1} \parallel v_i^T(\phi(x) - c \parallel^2)\right) \tag{5.6}$$

$$\sum_{i=1}^{m} \lambda_i^{'} = \frac{N}{N+1}\left(\sum_{i=1}^{m}\lambda_i + \frac{1}{N+1} \parallel \phi(x) - c \parallel^2\right). \tag{5.7}$$

The above equations involve $\phi(x)$ and c. By applying the kernel and with a few simplifications, their second terms can be redefined as

$$\parallel v_i^T(\phi(x) - c) \parallel^2 = \{\alpha_i^T(K_m(x) - \tilde{c})\}^2 \tag{5.8}$$

$$\parallel \phi(x) - c \parallel^2 = k(x,x) - 2\beta^T\tilde{c} + \tilde{c}^T\tilde{c}, \tag{5.9}$$

where $K_m(x) = [k(x_1, x), \ldots, k(x_m, x)]^T$, $\beta = K^{-1}K_m(x)$ and the mean vector $\tilde{c}$ is defined as

$$\tilde{c'} = \frac{1}{N+1}\begin{bmatrix} N\tilde{c} + K_m(x) \\ N\beta^T\tilde{c} + k(x,x) \end{bmatrix} \tag{5.10}$$

Thus their explicit computation is avoided. If the updated accumulation ratio satisfies $A'(d) \leq \theta$, a new eigen-axis should be augmented (See Fig. 5.2). The new axis v_{d+1} is defined as the normalized residue vector $\frac{h}{\|h\|}$ where h is given by

$$h = \phi(x) - \sum_{i=1}^{d}\{\phi(x)^T v_i\}v_i. \tag{5.11}$$

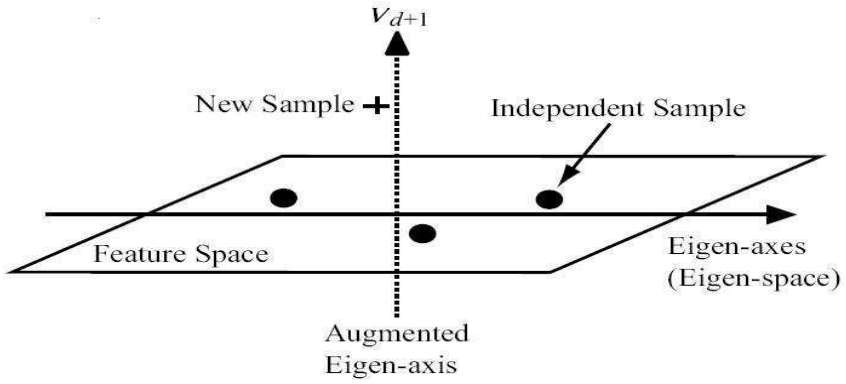


Figure 5.2 Augmentation of a new eigen-axis

Since the eigenvector v_{d+1} is a vector in the feature space, this is not explicitly calculated. As the eigenvectors are used only to obtain the features of a query pattern y, it is sufficient to obtain only the projection value to v_{d+1}, which is given as

$$v_{d+1}^T(\phi(y) - c') = \frac{1}{||h||}\alpha' \left(\begin{bmatrix} K_m(y) \\ k(x,y) \end{bmatrix} - \tilde{c}' \right) \tag{5.12}$$

where

$$\alpha' = \begin{bmatrix} -\sum_{i=1}^{d}(K_m(x)^T\alpha_i)\alpha_i \\ 1 \end{bmatrix}. \tag{5.13}$$

(5.14)

Rotation of Eigen-axis The final step of IKPCA is the rotation of eigen-axis to adapt to the variation of the sample distribution. When a new sample x is given, a new covariance matrix C' is given by

$$C' = \frac{N}{(N+1)^2}\{(N+1)C + \bar{\phi}(x)\bar{\phi}(x)^T\} \tag{5.15}$$

where $\bar{\phi}(x) = \phi(x) - c$. Then the new eigenvalue problem is given by

$$C'V' = V'\Lambda' \tag{5.16}$$

where V' is a new eigenvector matrix and Λ' is a diagonal matrix whose diagonal elements are the eigenvalues.

Assume that an eigen-axis v_{d+1} is augmented. Then the relation between old and new eigenvectors is given by

$$V' = [V, v_{d+1}]R \tag{5.17}$$

where R is a rotation matrix. Substituting (5.17) and (5.15) into (5.16) and multiplying by $[V, v_{d+1}]^T$, the new eigen value problem is rewritten as

$$\frac{N}{N+1}[V, v_{d+1}]^T \left\{ C + \frac{\bar{\phi}(x)\bar{\phi}(x)^T}{N+1} \right\} [V, v_{d+1}]R = R\Lambda'. \tag{5.18}$$

The two primary terms in the above equation are separately calculated as

$$[V, v_{d+1}]^T C[V, v_{d+1}] = \begin{bmatrix} V^T CV & V^T Cv_{d+1} \\ v_{d+1}^T CV & v_{d+1}^T Cv_{d+1} \end{bmatrix} \quad (5.19)$$

$$\approx \begin{bmatrix} \Lambda & \mathbf{0} \\ \mathbf{0^T} & 0 \end{bmatrix} \quad (5.20)$$

$$[V, v_{d+1}]^T \bar{\phi}(x)\bar{\phi}(x)^T [V, v_{d+1}] = \begin{bmatrix} V^T \bar{\phi}(x)\bar{\phi}(x)^T V & V^T \bar{\phi}(x)\bar{\phi}(x)^T v_{d+1} \\ v_{d+1}\bar{\phi}(x)\bar{\phi}(x)^T V & v_{d+1}^T \bar{\phi}(x)\bar{\phi}(x)^T v_{d+1} \end{bmatrix} \quad (5.21)$$

$$= \begin{bmatrix} gg^T & fg^T \\ fg^T & f^2 \end{bmatrix} \quad (5.22)$$

where

$$g = [\alpha_1, \ldots, \alpha_d](K_m(x) - \tilde{c}) \quad (5.23)$$

$$f = \alpha' \begin{bmatrix} K_m(x) \\ k(x,x) \end{bmatrix} - \begin{bmatrix} \tilde{c} \\ \beta^T \tilde{c} \end{bmatrix}. \quad (5.24)$$

In order to obtain a set of new eigen-axis, the following intermediate eigenvalue problem has to be solved

$$\frac{N}{N+1}\left((N+1)\begin{bmatrix} \Lambda & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix} + \begin{bmatrix} gg^T & fg^T \\ fg^T & f^2 \end{bmatrix}\right) R = R\Lambda'. \quad (5.25)$$

The solution of the above equation gives the rotation matrix R, using which the new eigen-axes can be obtained from (5.17). The overall incremental KPCA algorithm can be summarized as follows.

- Compute the KPCA algorithm on the landmarks extracted from the first few images
- Given a new image, check for the linear independence of the landmarks extracted from it
  - If yes, check if it affects the updated eigen value ratio *i.e.,* $\frac{\sum_1^n \lambda_i}{\sum_1^N \lambda_i}$ falls below the desired threshold
  - If so, add the new eigen-axis into the basis
  - Update the orientation of the eigen-axes
- Update the sample mean

The incremental method not only facilitates efficient updates to the representation but also eases the computational complexity of building the visual representation. It must be noted that in order to obtain accurate non-linear principal components from complex data distributions, large training datasets are required, especially for data embedded in a high-dimensional space [56]. This presents a difficulty for KPCA since it has to store and manipulate all data at once. Moreover the resulting KPCs have to be

defined implicitly by linear expansions of the training data, thus all data must be saved after training. For massive datasets, this means high costs for storage resources and computational load during utilization of KPCs. The IKPCA algorithm essentially avoids the computation of a full-scale KPCA at every time step by only applying the necessary computation to smaller data blocks for updating the basis incrementally.

It must be emphasized that the learning in this case is in terms of the increase in the accuracy of the appearance space, which in turn increases the accuracies in the feature correspondences. This indirectly improves the geometric estimates computed using the updated visual representation. As a result, this propels a significant improvement to the accuracy of the localization and servoing algorithms. Further, this method enables the robot to easily adapt to the variations in the scene.

5.4.2 Reinforcement Learning for Improved Path Planning

This section demonstrates the application of reinforcement learning to the problem of robot navigation. In particular, the problem of path planning of a robot is examined [44], which is the task of planning motions of a robot for performing a designated task while ensuring that collisions with objects in its workspace are avoided.

In the path-planning problem, one seeks to generate the course by which the robot can move towards its destination. The popular methods to solve this problem have been the graph-search and the potential field methods. In the former, a course is established by considering the connectivity in a network consisting of road maps connecting the initial to the destination; while in the latter, the path is established by way of applying attractive forces towards the goal and repulsive forces away from the obstacles. Though the above paradigms have gained popularity, they have certain limitations. In the former case, complete geometric information about the environment is needed. This is often a strong assumption. While in the latter, there is possibility of approaching a local minima (where the attractive and repulsive forces cancel each other) and thereby displaying oscillatory behavior. Another limitation is their assumption that the place topology is perfectly known, which implies that a reconstruction step must be done *apriori*. Another stream of research has recently been evaluating the possibility of applying a reinforcement learning (RL) strategy towards path planning. However, much learning time is needed to establish a path by RL because the dimension and size of the value functions used in this context are quite large.

Here a new path-planning scheme is proposed that involves a reinforcement learning scheme along with the potential field strategy. This scheme does not have the problem of deadlocks like the potential field method nor does it require huge amounts of learning time like conventional reinforcement learning methods. This approach exploits information available over multiple iterations to shorten the learning time.

In the following, the path planning problem using potential field method is briefly reviewed and then the proposed approach is explained.

Path Planning using Potential Fields The basic concept of the potential field method is to fill the robots workspace with an artificial potential field (V) in which the robot is attracted to its goal position and is repelled away from the obstacles [39]. It essentially involves a gravitation potential (V_a) that pulls a robot towards the goal configuration (Υ_g) and a repulsive potential (V_r) that repels the robot from an obstacle. Here the $\Upsilon_{6\times1}$ represents a parametrization of robot workspace.

Motion planning is performed in an iterative fashion. Initially the robot is at Υ_i with $\mathcal{F}_i$ being the initial camera frame and it has to be displaced to the desired camera frame $\mathcal{F}_g(\Upsilon = \Upsilon_g)$. We choose the parametrization Υ as $[^g t_c^T \ (u\theta)^T]$, where $^g R_c$,$^g t_c$ are the rotational and translational matrix between current camera frame $\mathcal{F}_c$ and $\mathcal{F}_g$ and u and θ are the rotation axis, angle obtained from $^g R_c$. We thus have $\Upsilon_i = [^g t_i^T \ (u\theta)_i^T]$ and $\Upsilon_g = 0_{6\times1}$. Recall that the rotation matrix and the translation vector can be obtained from the decomposition of the homography matrix as in equation (4.5) (Refer Chap. B).

At each iteration, an artificial force $F(\Upsilon)$ is induced by the potential function. The force is defined as $F(\Upsilon) = -\nabla V$ where ∇V denotes the gradient vector of V at Υ. Using the above conventions, $F(\Upsilon)$ can be decomposed as the sum of two vectors, $F_a(\Upsilon) = -\nabla V_a$ and $F_r(\Upsilon) = -\nabla V_r$, which are referred as the attractive and repulsive forces respectively. Path planning proceeds along the direction of $F(\Upsilon)$ and the discrete-time trajectory is given by the transition equation:

$$\Upsilon_{k+1} = \Upsilon_k + \epsilon_k \frac{F(\Upsilon_k)}{\parallel F(\Upsilon_k) \parallel}, \tag{5.26}$$

where k is the incremental index and ϵ_k is a positive scaling factor denoting the length of the kth increment. By repetition of this procedure, a robot moves along a path of operation and finally arrives at the destination. It must be noted that this procedure does not ensure the optimality of the path. However, it has gained wide popularity because of its mathematical elegance and simplicity.

Our goal is to deploy a Q-learning framework over the trajectory generated by the potential field method so as to achieve the optimal robot motion. In [54, 55], an analytical method to obtain the optimal camera trajectory was presented. Here a learning based approach is proposed to achieve the same. We assume the robot is observing a unknown planar target. The idea is graphically illustrated in Fig. 5.3.

The concept of reinforcement learning and Q-learning have been elaborated earlier in Sect. 5.3.2. The state space, actions and the immediate reward function for the current problem are as follows. The state s corresponds to the camera pose i.e., $^g R_c$ and $^g t_c$. The actions a are the motion instructions commanded at each pose i.e., $\triangle R, \triangle t$ and the reward function $r(s, a)$ is defined as the gradient of the net potential at the current pose i.e., $-\nabla V_a - \nabla V_r$. The Q-value function can be perceived as the resultant force at each pose. In (5.1), although the value function $Q(s, a)$ is a function of a state and action, in certain scenarios, only a state function is used. This is done so as to decrease the dimension of the function. In the following, we also parametrize the value function only using the state variable. The potential functions employed in the current formulation are as defined below.

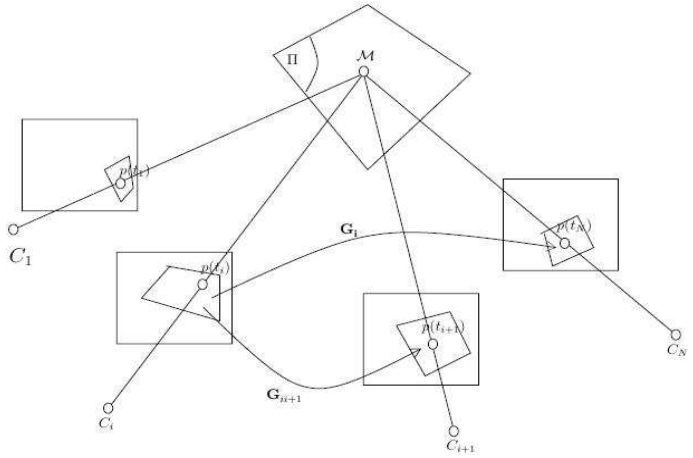


Figure 5.3 Computing the trajectory for optimal path planning

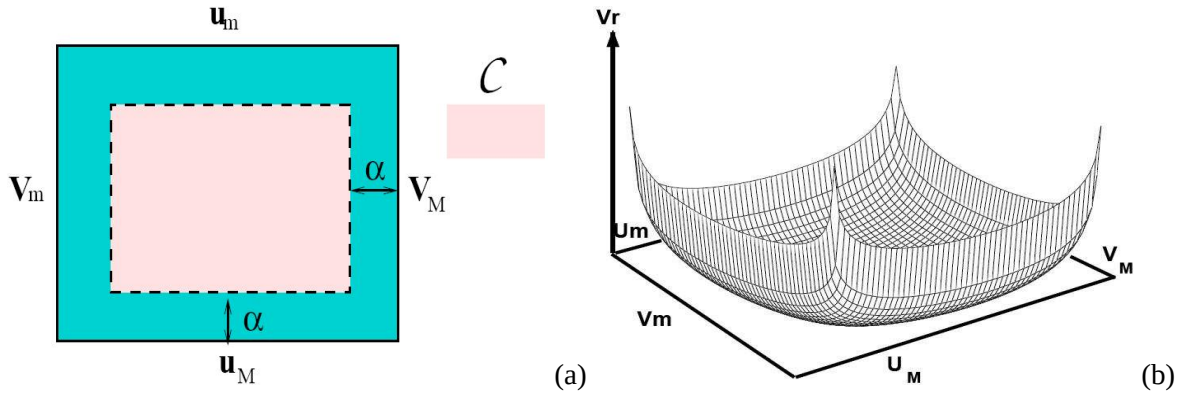


Figure 5.4 Repulsive Potential (a) Limits of Image (b) Potential Function incorporating the visibility constraint

The attractive potential field V_a is simply defined as a parabolic function in order to minimize the distance between the current position and the desired one:

$$V_a = \frac{1}{2} \parallel \Upsilon - \Upsilon_g \parallel^2 = \frac{1}{2} \parallel \Upsilon \parallel^2 . \tag{5.27}$$

Its gradient, also referred as the attractive force, is obtained as

$$F_a(\Upsilon) = -\nabla V_a = -\Upsilon. \tag{5.28}$$

The repulsive potential is employed so as to avoid displacing the camera to those poses that result in the features leaving the camera field of view. To ensure this, a potential barrier is created around the camera field of view, assuring that all the features are always observable and do not affect the camera motion when they are sufficiently far away from the image limits (See Fig.5.4). Thus, the potential V_r is defined as

$$V_r(s) = \begin{cases} -v_s^2 log\left(\prod_{j=1}^{n}(1-\frac{u_j}{u_M})(1-\frac{u_j}{u_m})(1-\frac{v_j}{v_M})(1-\frac{v_j}{v_m})\right) & \text{if s is in C} \\ 0 & \text{otherwise} \end{cases} \tag{5.29}$$

where

$$v_s(s) = \prod_{j=1}^{n}(u_j - u_M - \alpha)(u_j - u_m - \alpha)(v_j - v_M - \alpha)(v_j - v_m - \alpha). \tag{5.30}$$

Here s is the vector made up of the image coordinates of a point (u_j, v_j) for $j = 1, \ldots, n$ and C is the set $\{s|\exists j\ u_j \in [u_m\ \alpha] \cap [u_M - \alpha\ \ u_M]\ \cup\ v_j \in [v_m\ \alpha] \cap [v_M - \alpha\ \ v_M]\}$ with u_m, u_M, v_m, v_M being the limits of the image and α being a positive constant denoting the distance from the image edge. The gradient potential, referred as the artificial repulsive force, is obtained as [53]

$$F_r(\Upsilon) = -\left(\frac{\partial V_r(s)}{\partial \Upsilon}\right)^T = -M^+ L^+ \nabla_s^T V_r \tag{5.31}$$

where

$$M = \begin{bmatrix} {}^gR_c^T & \mathbf{0} \\ \mathbf{0} & L_w^+ \end{bmatrix} \tag{5.32}$$

$$L = \begin{bmatrix} 1 & 0 & u_j & u_j v_j & -1-u_j^2 & v_j \\ 0 & -1 & v_j & 1+v_j^2 & -u_j v_j & -u_j \end{bmatrix} \tag{5.33}$$

with L_w^+ defined as $I_{3\times3} + \frac{\theta}{2} sinc^2(\frac{\theta}{2})[u]_x + (1 - sinc(\theta))[u]_x^2$. Note that the symbol $+$ indicates the pseudo inverse. $\nabla_s^T V_r$ is defined as

$$\nabla_s^T V_r = \begin{cases} 2\begin{bmatrix} \nabla v_s^{2j} \\ \nabla v_s^{2j+1} \end{bmatrix} v_s \psi_s + v_s^2 \begin{bmatrix} -\frac{1}{u_M}(1-\frac{u_j}{u_M})^{-1} - \frac{1}{u_m}(1-\frac{u_j}{u_m})^{-1} \\ -\frac{1}{v_M}(1-\frac{v_j}{v_M})^{-1} - \frac{1}{v_m}(1-\frac{v_j}{v_m})^{-1} \end{bmatrix} & \text{if s is in C} \\ 0 & \text{otherwise} \end{cases} \tag{5.34}$$

where

$$\psi_s = log\left(\prod_{j=1}^{n}(1-\frac{u_j}{u_M})(1-\frac{u_j}{u_m})(1-\frac{v_j}{v_M})(1-\frac{v_j}{v_m})\right) \tag{5.35}$$

$$\begin{aligned} \nabla v_s^{2j} = {} & (2u_j - u_M - u_m - 2\alpha)(v_j - v_M - \alpha)(v_i - v_m - \alpha) \\ & \prod_{j=1, j\neq1}^{n}(u_j - u_M - \alpha)(u_j - u_m - \alpha)(v_j - v_M - \alpha)(v_j - v_m - \alpha) \end{aligned} \tag{5.36}$$

$$\begin{aligned} \nabla v_s^{2j+1} = {} & (2v_j - v_M - v_m - 2\alpha)(u_j - u_M - \alpha)(u_i - u_m - \alpha) \\ & \prod_{j=1, j\neq1}^{n}(u_j - u_M - \alpha)(u_j - u_m - \alpha)(v_j - v_M - \alpha)(v_j - v_m - \alpha) \end{aligned} \tag{5.37}$$

Initially, the Q-values at each state are set equal to the net potential forces at that pose. At each iteration, the Q-values are updated not only using the immediate potential function at that pose but also

using the difference in the adjacent potentials. This ensures that the Q-values are adjusted such the net potential force at every pose proceeds towards their minimum. The vector Υ_k at each instant is now computed using the current state of Q-values (that act as the new potential force functions at that pose) as

$$\Upsilon_{k+1} = \Upsilon_k + \epsilon_k \frac{Q}{\parallel Q \parallel} \tag{5.38}$$

Observe that this is similar to the manner in which the configuration is described according to the slope of a potential at a particular pose in the potential field method (See (5.26)). The rotation matrix ${}^{g}R_c$ and the translation vector ${}^{g}t_c$ can then be recovered from the parameterization Υ_k and are used to compute the homography matrix H_π as [30]

$$H_\pi = {}^{g}R_c - {}^{g}R_c^T \, {}^{g}t_c \, n^{*T}, \tag{5.39}$$

where n^* denotes the plane normal. According to (4.4), the image coordinates of points belonging to the plane at time k are given by

$$\mu p_k = [\mu u_k \;\; \mu v_k \;\; \mu]^T = H_\pi p^*, \tag{5.40}$$

where μ indicates a scalar constant. The image coordinates p_k can be easily obtained by dividing μp_k by its last component. Thus using the above equations, the intermediate images along the entire trajectory can be generated at the required resolution. These intermediate images now act as the desired image configurations that need to be reached at every iteration and in-turn guide the robot from its initial to the final goal position.

The proposed method ensures that the final generated image trajectory is the optimal one. It must be recalled that the optimal camera path is defined as the the set of poses $[R\,T](t)$ that are characterized by minimum acceleration or energy [53]. In terms of potential forces, the property of minimum acceleration or energy can be redefined as

$$\sum_0^1 F_i^T F_i \quad or \quad \sum_0^1 \Delta F_i^T \Delta F_i \quad (discrete) \tag{5.41}$$

$$\int_0^1 F^T F \quad or \quad \int_0^1 \dot{F}^T \dot{F} \quad (continuous) \tag{5.42}$$

where F is the net potential force at a given pose (5.26). This signifies that the optimal camera path corresponds to the set of force functions that result in minimum energy. In the above described Q-learning approach, at each iteration, the Q-values are adjusted such that the net potential force at every pose proceeds towards their minimum. This indicates that as the Q-learning episodes progress, the Q-values finally settle into the state of minimum energy, which suggests that the resultant set of force functions that satisfy the above conditions. Thus the Q-learning framework indeeds achieves the optimal trajectory.

In summary, the overall learning-based path planning algorithm can be described as follows -

- In an off-line step, run the conventional potential field based path planning algorithm. During this iteration, set $Q_k = F_k$ (i.e., net force at kth iteration)
- In each online iteration, given the current and the destination image, extract features from them and compute the rotation R and translation T parameters (obtained from the computation and decomposition of Homography between the images)
- Using the R and T parameters, compute the attractive and repulsive forces and obtain the net potential force F at that pose
- Set $Q_k = F + \gamma(Q_{k+1} - Q_k)$
- Reset $F_k = Q_k$
- Set $\Upsilon_{new} = \Upsilon + \epsilon \frac{F_k}{||F_k||}$
- Compute R_{new}, T_{new} from Υ_{new}
- Obtain the new desired image features
- Servo the robot so as to converge to the above image features
- Repeat until convergence

After the optimal robot trajectory is learnt at the end of Q-learning process, the new set of images replace the old set of intermediate images connecting the current and the desired views in the visual representation. As a result, the representation evolves over time. Thus the feedback received from performing specified goal oriented tasks is being utilized to update the visual experience.

It must be emphasized that the proposed approach is even valid in case of non overlapping image views i.e., when there is no overlap between the initial and the desired image. In this case, the final state Q values for each sub-goal are redefined as follows (rather than setting them simply as the absorbing states). They are now set as sum of potential field force at their current pose and the difference in potential with the initial Q-value of the next sub-goal. Note that in this case, the attractive potential force is computed towards the next sub-goal.

It must be noted that rather than a random (or a trial-and-error) search for solutions in the search space, potential field strategy is being employed in the Q-learning scheme to guide the search process for the optimal Q-values. This significantly reduces the learning time for generating the optimal path for a particular task. Moreover, in a trial-and-error method, it is not ensured that the robot stays within its workspace during the entire learning process. Thus by providing the knowledge gained in terms of potential field forces, unnecessary trial actions are reduced. Another advantage of employing the Q-learning scheme is that the knowledge of the potential functions need not be complete at every iteration i.e., the force at each robot pose need not be perfectly known. If there are any discrepancies, they can be accounted by the learning methodology.

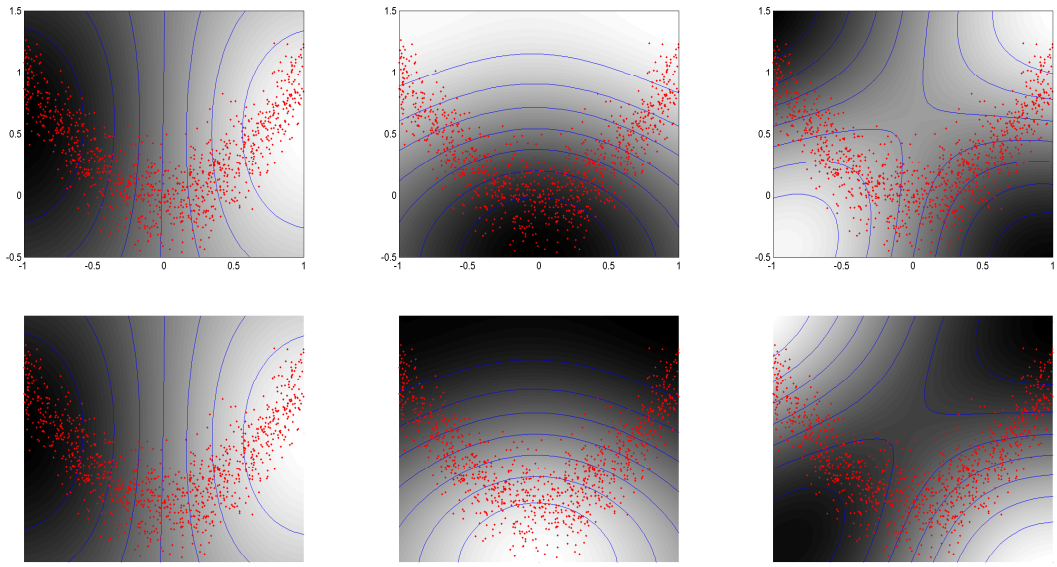


Figure 5.5 The first 3 principal components: The top row correspond to the KPCA while the bottom is the IKPCA

5.5 Experimental Results and Analysis

The goal of the experiments presented is to illustrate that a robot can improve its understanding of the scene overtime and also to analyze the improvement in its performance. All the proposed algorithms were implemented and evaluated in simulations as well as extensive real world scenarios. Real experiments were performed to analyze the benefit of using an incremental learning scheme. The environments considered were same as the ones used during the exploration and navigation stages. The robot wanders anywhere inside the lab. At each time instant, an image is grabbed from the camera and is used to query the database. If the difference between the current and the desired output is beyond a pre-specified tolerance, the current sample is learnt to update the visual experience using the incremental learning algorithm. Otherwise, it is rejected.

The results of the IKPCA algorithm are displayed below. An image database of features that represent several different distributed positions is initially constructed. As explained in Sect. 2.3.1, the first step is to extract landmarks from the images. Using the landmarks, the principal components are extracted as explained in Sect. 5.4.1. As the robot navigates in the environment and acquires new images, it uses the landmarks extracted from these new images to incrementally update the kernel space. Fig. 5.5 shows the comparison of the extracted components with those obtained using the regular KPCA technique for the initial set of images. It can be observed that the components are similar in both cases. As learning transpires over time, the features are updated and the representation evolves over time.

Fig. 5.6 displays some results of the improved localization. The first column shows the the current views acquired by the robot in its workspace while the second column displays the retrieved images from

Figure 5.6 Incremental Learning: In each row, the first image on the left is a query image, the second is the retrieved image. The degree of similarity was 0.85, 0.89, 0.78 respectively

the image database that are most similar to the query. It must be noted that the degree of similarity of the retrieved image with respect to the input image has improved as compared to the earlier KPCA based method. As mentioned earlier, only gray-scale versions of the displayed color images are employed in the algorithm. An overall improvement in the localization rate was also observed. For the first lab environment, the localization accuracy was $\sim 85\%$ and $\sim 88\%$ in the second lab environment using the incremental algorithm. This resulted in improved feature correspondences across different image views, which in turn improved the relative pose accuracies. An additional advantage of this method was that it avoided the need of saving all the image data at a time for computing the KPCs.

For evaluating the performance of the proposed approach, we compared it to other localization algorithms. In particular, the SIFT-based algorithm [45] which has gained recent popularity in object recognition was considered. On comparison, it was observed that the localization accuracies using the SIFT-based approach were greater. This is because of the inherent property of the SIFT representation. As the features are built from the scale-space, they are quite robust and invariant to regular image transformations. However, the proposed algorithm is more interesting as it facilitates incremental learning of

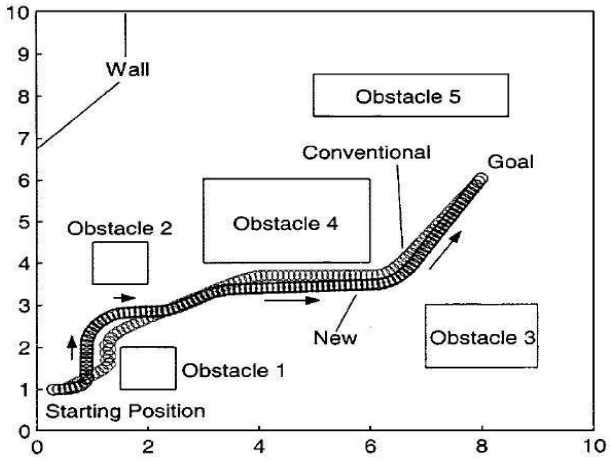


Figure 5.7 Illustration of the path planning method: Path obtained by using a potential field method along with a reinforcement learning scheme generates more optimal and smoother trajectories

the environment features which is not possible by other algorithms. As additional hypothesis of scene features are available from the new images, IKPCA algorithm provides better updates to the feature descriptors and thus the accuracy of the kernel basis improve over time. This provides robustness even in case of wide-baseline views. It must be noted that this selective-learning mechanism not only allows incremental updation of the representation but also effectively prevents redundant learning.

The proposed path planning algorithm was analyzed via practical experiments. Fig. 5.7 graphically illustrates the idea. The figure demonstrates the task of navigating a robot from its initial to the final destination position. Though a valid path is computed using the conventional potential field method, it is not optimal. When a Q-learning scheme is applied onto the initially generated robot trajectory, it converges to the optimal one.

Simulations were conducted to study the effect of the proposed method and confirm that introducing a learning step into the path-planning algorithm indeed improves the navigation performance. The simulations were conducted in MATLAB environment using a camera (attached to an end-effector) with a 512×512 pixel array and a sampling time of $T = 40ms$. The implemented algorithms can directly be applied to a real robot if matched points in the initial and desired images are available and can then be tracked. Since tracking is not a primary aspect of this work, the target here is composed of simple circular marks on a planar surface. The figure shows initial F and the final F^* camera coordinate frames. The image views corresponding to the desired and initial camera positions are displayed in Fig. 5.9. Results obtained using the potential field method are first demonstrated and then the results with the Q-learning method are presented.

The servoing is conducted by perceiving the point features on one of the planes in the scene. The image projection of the points at the initial view are the starting points of the image trajectory while the '+' indicate the features at the desired position. The camera displacement between the desired and the initial camera frames is very important $(t_x = 300mm, t_y = 550mm, t_z = 120mm, (u\theta)_x =$

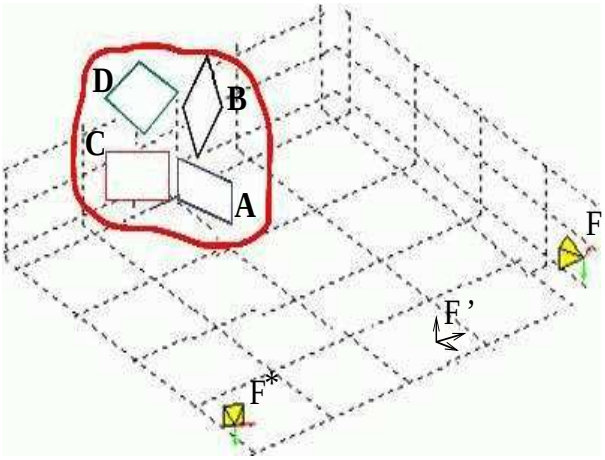


Figure 5.8 Simulation Set-up: The coordinate frames F,F* denote the initial and goal configurations. The features on one of the planes are considered during the servoing process.

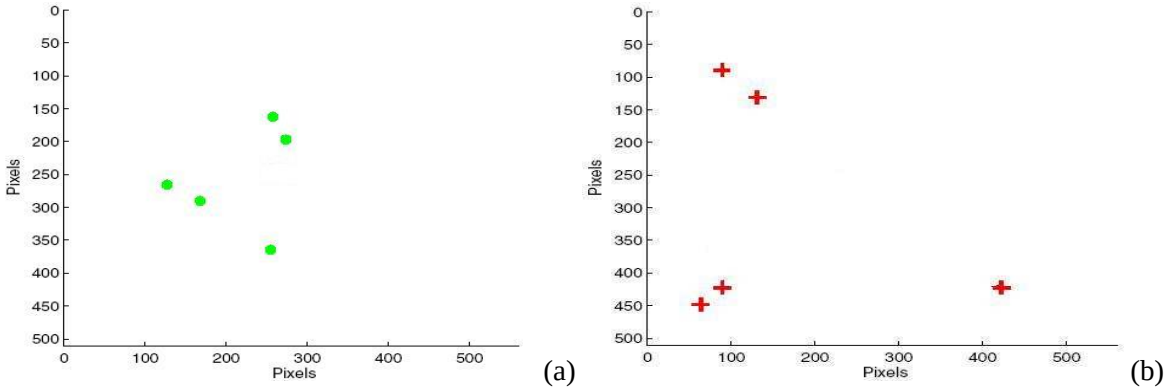


Figure 5.9 Image views at the current and final poses

$28°, (u\theta)_y = 78°, (u\theta)_z = 147°$). The goal was to displace the features from their initial coordinates to the desired image coordinates in an optimal path such that they always remain in the camera field of view. In order to emphasize the importance of the visibility constraint in the trajectories, a path-planning experiment without the repulsive potential was performed (See Fig. 5.10). It must be observed that the visual features get out of the camera field of view. In Fig. 5.11, the planned and the tracked trajectories using the simple potential field method are plotted. It must be noticed that the tracked trajectories and the planned trajectories are almost similar. The tracking error $(s(t) - s^*(t))$ is plotted in Fig. 5.11, and it confirms the previous comment, since the maximal error is less than ten pixels. The error on the coordinates of each target point between its current and its desired location in the image is also given. The convergence of the coordinates to their desired value demonstrates a valid realization of the task. The 3D camera trajectory is also plotted.

In the next experiment, the same task was repeated but by employing the learning method. In the experiment, γ was set to 0.7. The number of intermediate points i.e., k used was about 100. Twenty

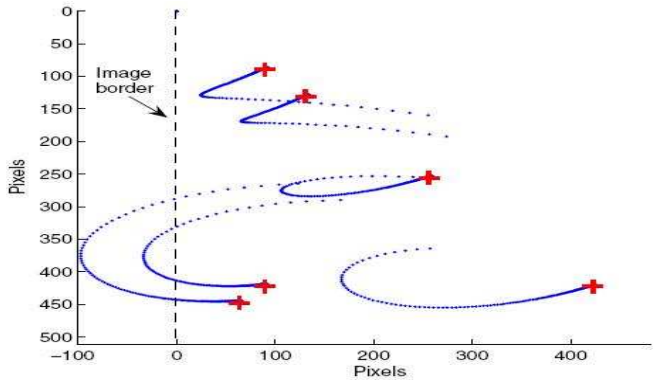


Figure 5.10 Path obtained without using repulsive potential

episodes are performed while attempting to servo to the target. The learning algorithm processes these experiences and incrementally improves its ability to servo to the target. As can be seen in Fig. 5.12, the expected trajectories are really obtained and perfect convergence of the coordinates to their desired value was achieved. The motion of each point in the image is, thus, perfectly predictable (observe the reduction in the tracking error). This experiment confirms that a trajectory following in the image space using the learning method indeed provides better results and demonstrates the correct realization of the task. Notice the successful achievement of optimal camera path (unlike the result obtained using the conventional method). Hence the task is correctly realized and confirms the efficiency of the proposed learning scheme.

Another result obtained using a different initialization of camera positions is displayed in Fig. 5.13. The figure displays the variation in the camera trajectories. Observe the improvement in the final camera trajectory as compared to the one obtained using the conventional potential field method.

5.6 Discussion

In summary, this chapter had presented two important learning-based schemes to improve the navigation performance of a robot. The incremental learning scheme facilitates updation of the visual representation built during the exploration stage. This not only achieves improved localization but also makes it robust to variations in the scene. The second contribution has been a reinforcement learning based approach to path-planning along with potential fields. The algorithm facilitates the learning of the optimal trajectory for navigating the robot from its current position to the desired position. It must be emphasized that the proposed approach does not focus on construction or improvement of the 3D map of the environment (unlike most other learning approaches); rather it improves the internal representation of the robot workspace so as to yield an optimal navigation performance.

However, certain aspects still need to be analyzed. First, a comprehensive analysis of the performance of the incremental learning algorithm is desirable. Second, a more-formal convergence proof

demonstrating that the proposed reinforcemental learning approach to path planning indeed achieves the optimal camera trajectory without getting affected by the local minima has to be derived. Another aspect includes the analysis of a continuous Q-learning framework rather than the proposed discrete method to achieve superior result. In addition, in order to apply the devised approach onto a mobile robot, the constraint of non-holonomicity has to be incorporated as a repulsive potential. Finally, the performance of learning approach should be analyzed by more comprehensive simulations and extensive real experimentation.

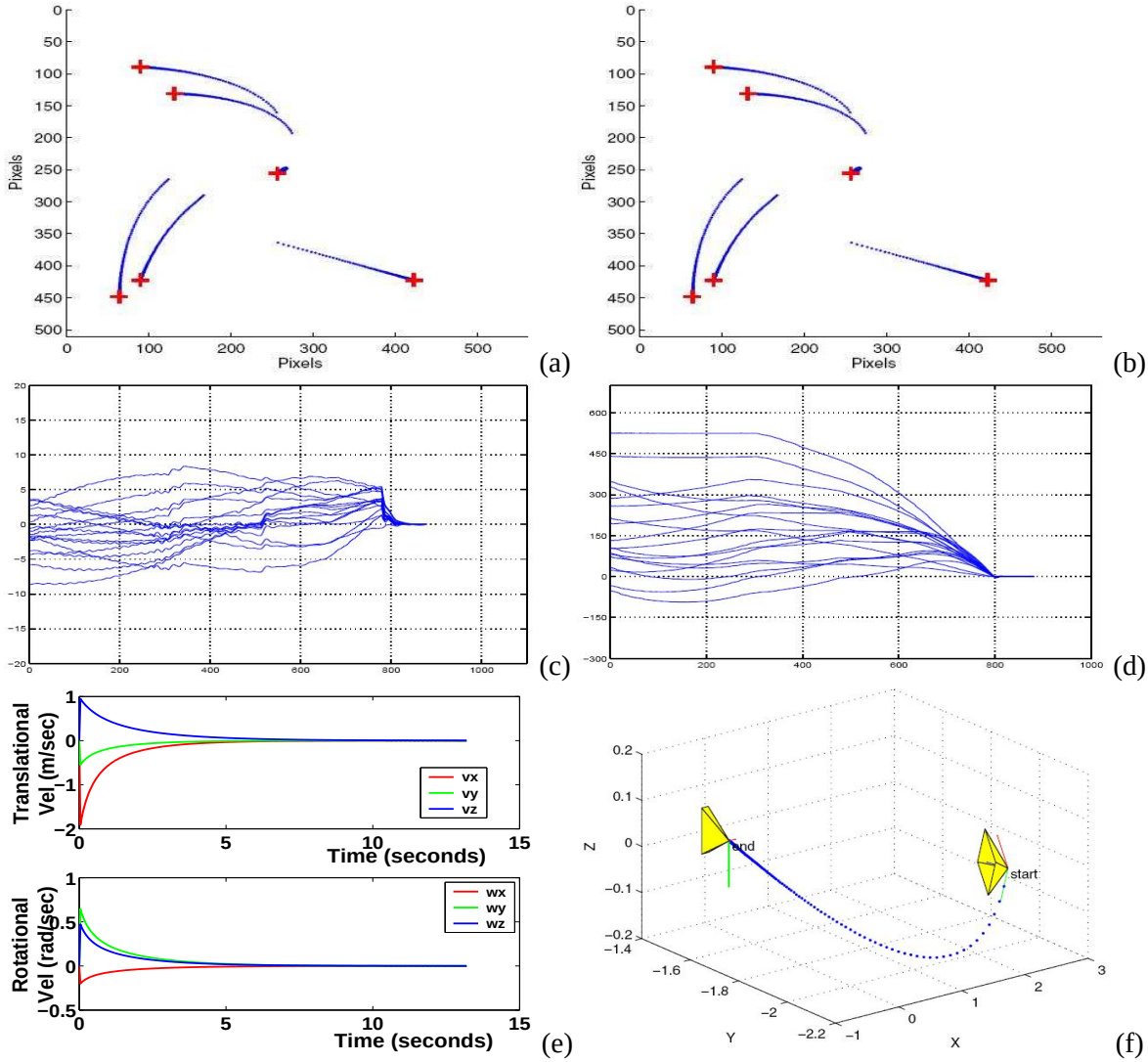


Figure 5.11 Conventional Potential Field Method: (a) Planned and (b) Realized trajectories (c) Tracking Error (d) Image Feature Error (e) Camera velocity screw (f) Camera Trajectory

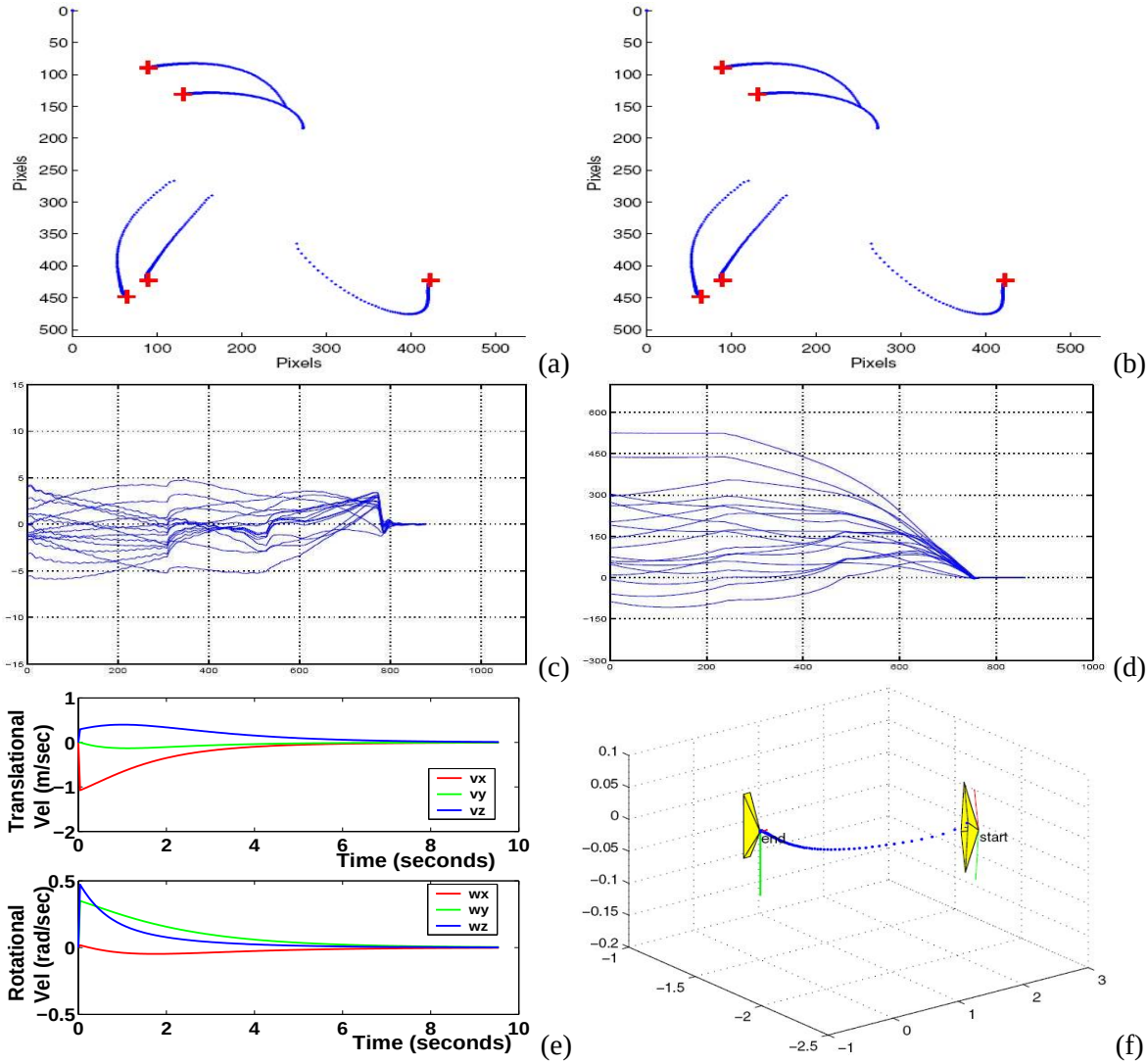


Figure 5.12 Learning Method: (a) Planned and (b) Realized trajectories (c) Tracking Error (d) Image Feature Error (e) Camera velocity screw (f) Camera Trajectory

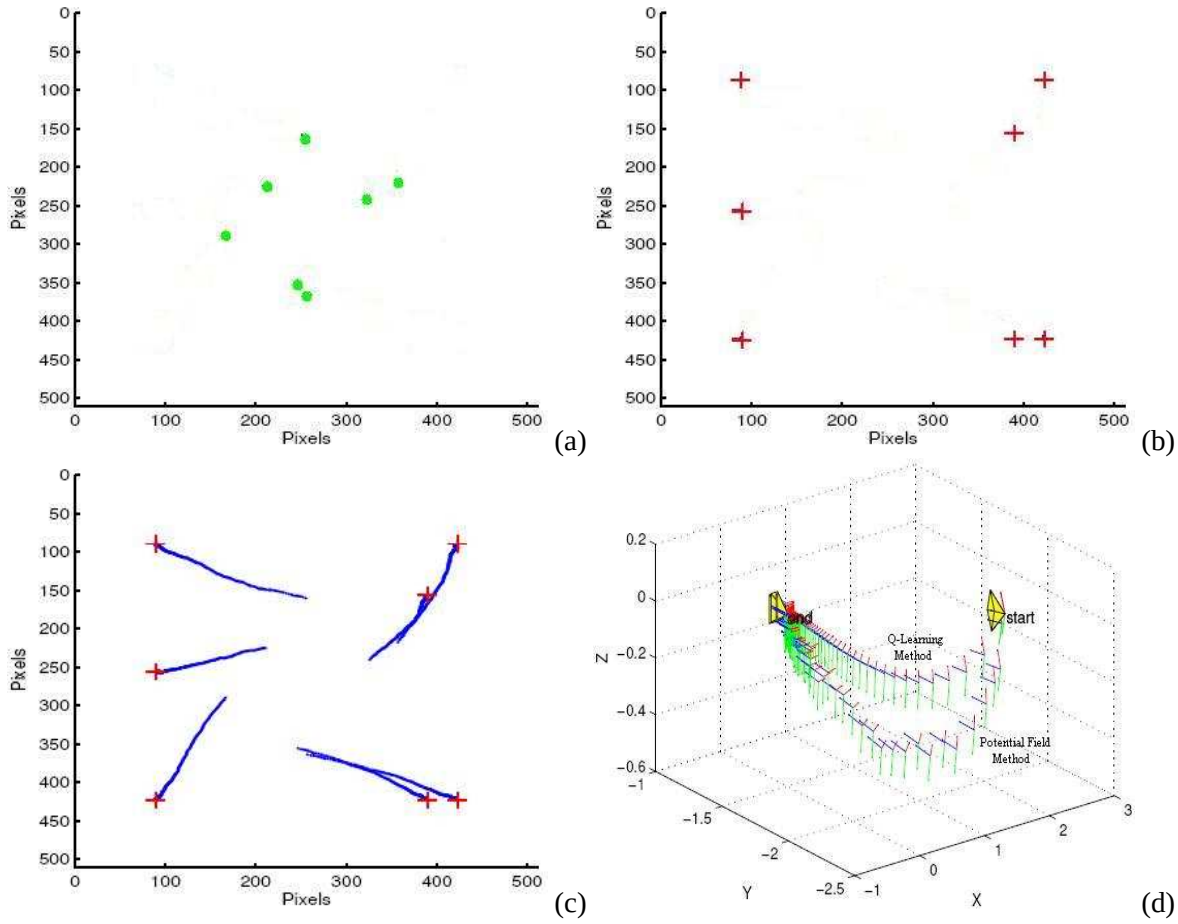


Figure 5.13 Learning Method: (a) Current and (b) Desired image view. (c) Image Trajectory (d) Camera Trajectory using potential field method and learning method

Chapter 6

Conclusions

In this thesis, various aspects involved in successfully conducting the robot navigation task were analyzed. Several ideas to augment the existing approaches so as to improve their performance were discussed. A novel framework based on visual experience for vision-based robot navigation has been proposed. The framework allows dynamic building of robot memory and facilitates performance improvement by the process of learning from past experiences. Based on this architecture, the design of the individual algorithms required for performing the navigation task were discussed.

A novel image-based exploration algorithm based on the idea of frontier-based exploration was proposed. The algorithm inferred the frontier boundaries (horizons) from images acquired by a limited field-of-view camera and used them to efficiently explore the environment. The horizons were detected by using a graph-based segmentation algorithm that identified obstacle-free regions from obstacles. The method exploited the natural scene structure in the world and did not involve any partial reconstruction. The proposed algorithm was able to systematically discover unknown environments and had built a reliable visual representation that was suitable for the purpose of navigation.

Different algorithms required for performing the navigation task were also presented. The first algorithm concerned the task of qualitative localization, wherein the most similar image in the database to the current image was retrieved. The set of intermediate images leading from the initial robot view to the desired view were extracted from the visual memory using a planning algorithm. A feed-back based algorithm was then employed to control the robot from its current position to the desired position. It was demonstrated that the navigation task can be accomplished efficiently, reliably and in real-time by only exploiting the constraints between the images.

Two important learning algorithm were described to facilitate online learning about the world by the robot. An incremental learning algorithm that was able to incorporate additional scene information, gathered over time by the robot, to improve its visual representation was presented. Another learning approach that exploited the feedback received from previous experiences of the robot to improve not only the navigation performance but also the visual representation was proposed. The reinforcement learning scheme utilizing the potential field method was applied to the task of path-planning. The approach generated optimal motion trajectories for guiding the robot to the desired positions in its assigned

tasks. The proposed algorithms do not focus on the construction or the improvement of geometric maps of the environment. They achieve performance improvement by improving the internal representation of the robot workspace.

However, there are certain limitations with the presented algorithms. Specifically, the devised exploration approach only distinguishes between scanned and un-scanned areas of the environment and does not take into account the actual information gathered at each view-point. To overcome this limitation, a notion of information gain might be introduced. With regards the servoing approach, a major problem is that the control strategy is not guaranteed to produce control signals that respect the non-holonomicity constraints of the robot. Another associated issue is the visibility constraint (i.e., to ensure the object always is within the camera field of view). The current challenge is to devise an efficient control strategy to navigate a non-holonomic mobile robot that would always ensure the object features to remain in the camera field of view. In the context of learning, a critical issue is the optimality of the path generated by the reinforcement learning scheme. Though the learning approach is guaranteed to generate better paths in comparison to a direct potential field method, it still needs to be theoretically proven whether the path generated is indeed the optimal.

In future, this line of inquiry can be continued to develop improved navigation techniques. The specific modules that could improve the online visual experience architecture include: An algorithm evaluating the 'utility' of an image in terms of whether it describes a new region, or improves the description of an known region to decide if it should be added to the visual map representation (using the notions of information gain); An image-based visual servoing algorithm respecting the robot's non holonomic constraints while ensuring feature points are in camera FOV (without making any assumption about the planarity or non-planarity of the scene); A learning-based algorithm that can tackle dynamic environments, which could be done by dynamically assigning 'weights' to features extracted from images (the weights measure the reliability and discriminating power of a feature).

In conclusion, this thesis is an introductory step in the direction of autonomous image-based navigation. We believe that the proposed approach will be essential for mobile robots to progress in the direction of increased applicability and robustness.

Visual Servoing in Non-Rigid Environments: A Space-Time Approach

D Santosh and C V Jawahar

***Abstract*— Most robotic vision algorithms are proposed by envisaging robots operating in structured environments where the world is assumed to be rigid. These algorithms fail to provide optimum behavior when the robot has to be controlled with respect to active non-rigid targets. This paper presents a new framework for visual servoing that accomplishes the robot positioning task even in non-rigid environments. We introduce a space-time representation scheme for modeling the deformations of a non-rigid object and propose a model-free hybrid approach that exploits the two-view geometry induced by the space-time features to perform the servoing task. Our formulation can address a variety of non-rigid motions and can tackle large camera displacements without being affected by the degeneracies in the task space. Experimental results validate our approach and demonstrate the robust and stable behavior.**

I. Visual Servoing in Presence of Non-Rigid Motion

The problem of controlling the movement of robotic systems using visual feedback has been a topic of substantial research in the field of Visual Servoing [10]. Several influential approaches in this area have been envisaged to perform the servoing task [4], [13]; however, much of the research until now presumes structured and rigid environments. In this paper, we propose an approach to visual servoing that can control a dynamic system even in an unknown active non-rigid environment.

In robotic vision research, motion analysis has been largely restricted to rigid objects due to their simplicity, elegance and immediate industrial applicability. However, in real world situations, motion of physical objects is often non-rigid [2] in nature. Common examples include motion of human body, flying birds, ocean waves etc. It has been a persistent desire to employ robots in such natural and unconventional environments. For this to be successful, it is desirable to develop servoing strategies and algorithms that can perform optimally even in such unstructured scenarios. Our motivation towards non-rigid motion analysis has been driven by applications in the areas of surgical robotics, underwater robotics, active vision systems etc.

Dealing with non-rigid motion poses several challenges in the design of optimal servoing strategies. Non-rigid objects undergo a persistent change in their pose which forbids any single image to characterize their state. This is because motion instruction planned based on the features extracted at current time instant might not be relevant in the next instant as the object undergoes a change in its pose. Further, the

D Santosh and C V Jawahar are with the Center for Visual Information Technology, International Institute of Information Technology, Hyderabad 500032, India {santosh@research.,jawahar@}iiit.ac.in

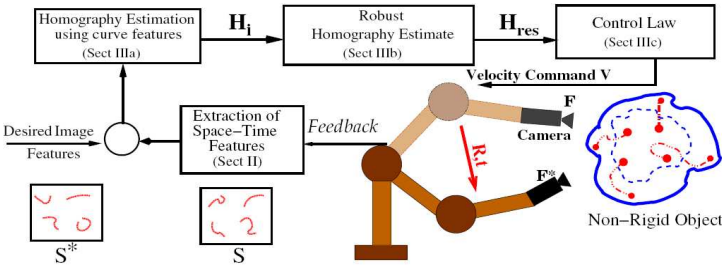


Fig. 1. Proposed scheme for visual servoing in non-rigid environments

desired configuration of the end-effector cannot be described by using only a single image or a single pose as it will lead to oscillations of the manipulator even after the goal position is reached. Note that the unavailability of static features (in case of whole body deformations) and background features (in case of moving targets) makes it imperative to engender new representation schemes for visual servoing using only the pose-varying features on the object surface. This necessitates a time-based representation, rather than a purely spatial one, due to the temporal nature of the object deformations. It must be emphasized that non-rigid motion encompasses wide range of possible motions ranging from simple translatory motion such as a waving hand to highly complex motion like that of a beating heart. A general representation for all kinds of motions is preferable, but appears to be inconceivable at this stage. Establishing correspondence between image features is usually the primary step in visual servoing. However, finding accurate correspondences is often difficult in practical situations; especially, while matching points in two views separated by large displacement. This is a highly formidable requirement in case of deformable objects as this demands frame-to-frame matching of the object deformations which is complicated even for simple motions.

Existing servoing schemes are not designed to tackle non-rigidity. Cartesian-based algorithms require complete $3D$ information of the object which is a strong assumption for deformable targets. Image-based servoing schemes cannot be directly used as these schemes use information only from a single image to guide the robot, which results in an oscillatory camera trajectory [15]. Also, these methods are not completely model-free, since depths of the observed features are needed in the control law [10]. Further, they demand the exact frame-to-frame correspondences between image features. Moreover, new representations conceived for modeling the non-rigid motion cannot be directly utilized in these schemes as the corresponding interaction matrix relating the feature motion in the image space to the camera

2007 IEEE International Conference on Robotics and Automation
Roma, Italy. 10-14 April 2007.

motion in the Cartesian space has to be derived.

In this paper, we propose a different approach to visual servoing in which the motion characteristics of active non-rigid objects are used to perform the servoing task, without the requirement of $3D$ structure information. Our approach is based on the bi-dimensional appearance of the objects in the environment and explicitly takes into account independent object motions. In most cases, where an object has a repetitive motion, the space-time trajectories of representative points on it will serve to uniquely represent the object (Sect. II). These trajectories are invariant to object deformations and can be utilized to obtain a stable estimate of the projective transformation relating the initial and desired views (Sect. III-A). The estimated transformation is then used in a feedback-based hybrid control to perform the servoing task (Sect. III-C). The overall algorithm is summarized in Fig. 1. In [15], a preliminary strategy to tackle non-rigidity was discussed. In that approach, gross features of the object deformations were extracted and used in the servoing algorithm. However, the method handles only simple non-rigid motions. Further, issues of optimal camera trajectory, degenerate configurations (such as local minima, singularity) etc. have not been analyzed. In the current formulation, we aim to not only generalize our approach to complex non-rigid motions but also achieve the desirable characteristics of the servoing algorithm (Sect. III).

II. Geometry of Non-rigidity

Active non-rigid motion can essentially be classified into three primary types, namely articulated motion, elastic motion and fluid motion [2]. This classification is based on the constraints on the degree of the smoothness and continuity in the motion. Among the different forms of non-rigidity, elastic motion constitutes the most general form of non-rigid motion [2]. Elastic or cyclic motion is ubiquitous in the natural world. For instance, the motion of heart and other body organs; motion of flying birds, swaying trees and moving aquatic animals; ambulatory motion of humans and animals etc. All such motions involve a regularly repeating sequence of motion events and thus demonstrate a cyclic pattern in their deformations. In this paper, we concentrate on accomplishing the servoing task in presence of such stationary elastic objects. It may be noted that global motion from a moving non-rigid object can be separated by performing rigid and non-rigid motion segmentation [3].

Different modeling strategies have been proposed in the field of computer vision to characterize non-rigidity. Most approaches employ methods like the linear subspace model (appearance manifold), kinematic chains, dynamic Markov models etc. to model the deformations as variations to the model parameters. In these methods, assumptions regarding the objects and their motion are made, which restrict the class of objects that can be handled. A standard modeling scheme for all kinds of motion is very much desirable for the design of a general servoing strategy. Our pursuit is to engender a stable representation for a generic non-rigid object.

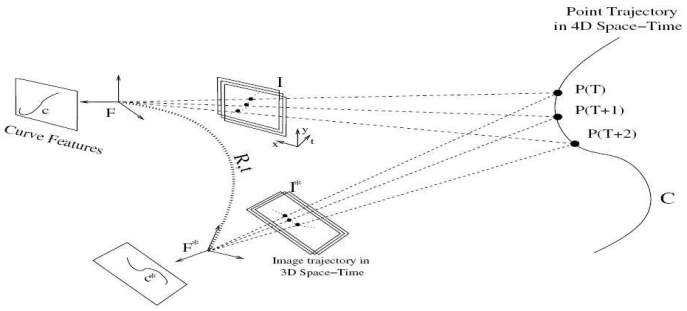


Fig. 2. Projective transformation in the Space-Time: Point trajectory in the $4D$ space-time projects onto the $3D$ space-time describing a curve in the image space

A. Non-Rigid Motion as Space-Time Curves

We represent a non-rigid object using a set of representative points moving with different velocities [12]. These interest points are locations where the object deforms in shape, and constitute to its surface appearance. A deformation of the object induces a change in the point locations. For a stationary target, its deformations can be described using the motion of these configuration of points. This is because repeated activity transforms the points such that they traverse a fixed trajectory in the three-dimensional space. For a static observer perceiving the object, these points always appear to traverse along the same $3D$ curves.

Projections from $4D$ to $3D$ Space-Time More formally, let $\mathcal{O}$ be the observed object and $P_1, \ldots, P_n$ be the interest points on the object surface in the $3D$ space. Let the $3D$ coordinate of the point P_i, $i = 1, 2, \ldots, n$ be $[X_i \;\; Y_i \;\; Z_i]^T$. The motion of this point in the Euclidean space can be considered as a set of points $\mathcal{P}_k = [X_k \; Y_k \; Z_k \; T_k]^T$ defining a curve $\mathcal{C}_i$ in a $4D$ space-time where T_k denotes time. Assuming a pin-hole camera, the space-time projection of the point onto the $3D$ (image) space-time satisfies

$$\widetilde{\mathbf{p}_k} \approx \mathcal{M}\widetilde{\mathcal{P}_k}, \tag{1}$$

where matrix $\mathcal{M}$ denotes the 4×5 extended camera matrix, $\approx$ denotes equality up to scale and $\widetilde{\ }$ denotes corresponding homogeneous coordinates. Although the space-time projection from $\mathcal{P}_k$ to $\mathbf{p}_k$ cannot be described by projective cameras, (1) signifies that a point in the real space-time is projected to an image space-time point by an extended affine camera. Thus the motions in the $4D$ space are projected onto images and can be observed as a set of points $\mathbf{p} = [x \;\; y \;\; t]^T$ in a $3D$ space-time on image motions extracted from an image sequence (See Fig. 2). This $3D$ space-time can be perceived as a spatio-temporal entity with two spatial dimension x, y and a time dimension t. The corresponding image coordinates $\mathbf{m}$ can be obtained from the normalized coordinates $\mathbf{p}$ with an affine transformation

$$\mathbf{m} = K\mathbf{p}, \tag{2}$$

where K is the camera intrinsic matrix [8]. These points define the image trajectory $\boldsymbol{c}_i$ of the $4D$ space-time curve $\mathcal{C}_i$.

Curves have been employed in computer vision for a very long time. However, most of the works until now refer to them in the context of shape descriptors. In this paper, the concept of 'space-time curves' is being introduced for visual servoing purpose. Such a representation offers multiple benefits. First, such features allow large class of motions to be accommodated as few constraints are enforced on the kinds of motion. As they are invariant to the changes in the object pose, they provide a stable and unique set of features for visual servoing. Moreover, the desired configuration of the manipulator can be stably defined using these invariant features. Further, such a representation avoids the complex problem of establishing frame-to-frame feature correspondences during the servoing task. The space-time curves provide a more geometric and intuitive representation of the object than other past features and thus are more interesting. Compared to finding corresponding points, corresponding curves can be easily and robustly identified using multiple cues e.g., periodicity, curvature etc.

B. Navigation Formulation

Let $\mathcal{F}_0$ be the coordinate frame attached to the target, and $\mathcal{F}^*$ and $\mathcal{F}$ be the coordinate frames attached to the calibrated cameras at the desired and current positions respectively. Let $\mathcal{F}^*$ be displaced from $\mathcal{F}$ in the Euclidean space by $\mathbf{R} \in SO(3)$ and $\mathbf{t} = [t_x, t_y, t_z]^T \in \Re^3$, where $\mathbf{R}, \mathbf{t}$ denote the rotation matrix and the translation vector respectively. Considering the angle-axis representation for rotation matrix $\mathbf{R}$, we have $\mathbf{R} = exp([\mathbf{r}]_\times)$, where $\mathbf{r} = u\theta$, is the vector containing the angle of rotation θ and the axis of rotation u, exp is the matrix exponential function and $[\mathbf{r}]_\times$ is the skew-symmetric representation of the vector $\mathbf{r}$. The relative camera pose with respect to frame $\mathcal{F}^*$ is defined by a 6×1 vector $\mathcal{E} = [\mathbf{t}^T \ \mathbf{r}^T]^T$. The points on the space-time curve $\mathcal{C}_i$ in the current frame $\mathcal{F}$ get transformed to desired frame $\mathcal{F}^*$ as $P^* = \mathbf{R}P + \mathbf{t}$, defining the curve $\mathcal{C}_i^*$. These points project onto the normalized plane $\mathcal{I}^*$ as $\mathbf{p}^*$ and their corresponding image coordinates are obtained using the relation $\mathbf{m}^* = K\mathbf{p}^*$ (similar to (2)). These image points define the trajectory $\mathbf{c}_i^*$ in the desired frame $\mathcal{F}^*$ (See Fig. 2).

In visual servoing, the goal is to reduce the error in the desired and the current features so as to drive the disparity in pose between $\mathcal{F}$ and $\mathcal{F}^*$ to zero. The objective function $\mathbf{e}$ can be defined as a function of time t as

$$\mathbf{e}(t) = \mathbf{c_i}(t) - \mathbf{c_i^*}. \tag{3}$$

Since the image features are a function of the camera pose *i.e.,* $\mathbf{c_i}(t) = f(\mathcal{E}(t))$, (3) can be redefined as

$$\mathbf{e}^{'}(t) = \mathcal{E}(t) - \mathcal{E}^*. \tag{4}$$

Thus the servoing task reduces to the problem of estimation of the partial displacement of the camera followed by a minimization of error in the relative pose parameters. We assume that the frame rate of the camera is sufficiently high for capturing one cycle of the points trajectory.

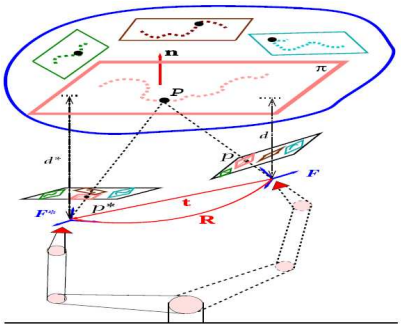


Fig. 3. Homography-based visual servoing using planar curves: The curves indicate the space-time trajectory of the points on the non-rigid object

III. Proposed Servoing Approach

The goal of our solution is to design a servoing algorithm to drive the disparity between the current and the desired camera configurations to zero. The desirable characteristics of the algorithm are –

- Immunity to non-rigid deformations and continuity in the velocity screw.
- Ability to tackle large camera displacements without being affected by degeneracies in the task space.
- Robustness to image measurement errors.
- Independence from prior knowledge of the object model and parameter initialization.
- Decoupling of camera motion and ensuring feature visibility.

We employ a hybrid homography-based formulation to achieve the above desirable characteristics. In literature, model-free hybrid approaches have been developed to deal with unknown environments [5], [13]. These methods exploit the information provided by the projective reconstruction of the scene computed only from the visual features extracted from the images.

We begin with a simple, not so strict assumption that the motion of a point on a non-rigid object can be approximated to a planar motion [7]. The most general motion of a sufficiently small element of a deformable object can be represented in three mutually orthogonal directions (*i.e.,* as a sum of translation, rotation and an extension). However, in presence of opaque objects, the visible deformations are those occurring at the object surface. These deformations can be assumed to occur locally on the plane. Hence the dominant motion of the points can be assumed to be planar in nature. Note that each curve can be planar in any orientation while the object is non-planar (See Fig. 3).

A. Homography-based Visual Servoing

Given the planarity assumption of the point trajectories, the projective transformation between the two views of the scene can be defined using a homography [8]. Several methods have been proposed in literature to estimate homography from planar curves [1]. However, most of the approaches deal with parametric curves. Since non-rigid motion can be complex, parametric methods might not be capable of estimating homography in all situations as they

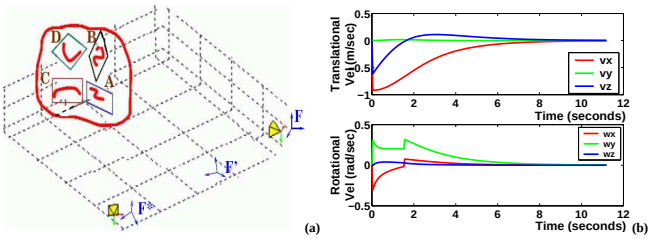


Fig. 4. Effect of degenerate configurations: (a) Servoing begun using reference curve A reaches a degeneracy at $F^{'}$ whose origin lies on the reference plane containing curve A (b) Discontinuity in the velocity screw is due to the switching of curves (A to C) at $F^{'}$

are dependent on the chosen parametrization and cannot handle changes in curve topology. To accommodate arbitrary large class of motions, we extend the notion of higher order primitives further to include an ordered collection of points on a contour. We employ the homography estimation from contours technique proposed in [11]. In this method, the homography is estimated using only an ordered set of pixels of the contour, without the requirement of explicit point to point correspondences. The algorithm converges to the actual homography in few iterations and is robust to outliers and errors in image measurements.

The recovered homography can then be decomposed to obtain the rotation matrix R, the scaled translation vector $\frac{t}{d}$ and the plane normal n using the procedure described in [6]. It must be emphasized that information from multiple homography estimates, computed for a set of point trajectories, can be utilized to unambiguously decompose H without the requirement of *a priori* knowledge or parameter initialization. The motion parameters are then used in the control law to generate the optimal velocity instruction. In [13], a similar method is discussed by Malis *et al.* wherein the homography between two views of a planar contour is estimated and the parameters obtained from its decomposition are used in a $21/2D$ control to servo the end-effector.

B. Reliable Homography Computation

A single homography estimate is not sufficient when the camera has to undergo large displacements in visual servoing as it can be affected by occlusion of visual features, tracking (drift) errors, the camera center approaching the feature plane or due to singular homographies [8]. These degenerate configurations render the homography invalid. In either of the cases, when a degeneracy is reached, a new planar curve has to be chosen as the current curve can no longer be used for estimating the two-view projective transformation. This switching causes a discontinuity in the velocity screw affecting the stability of the robotic system. In Fig. 4, the effect of switching is demonstrated, where a positioning task with respect to a non-rigid object is simulated. The above limitation is caused by the fact that only information from a single homography is being utilized at a time. To circumvent this problem, we compute a robust and reliable estimate of the homography using information from multiple homography estimates as described in [14]. In this method, the resultant homography is efficiently computed as a linear combination of four independent homographies by exploiting the rank-4 subspace constraint on homographies.

C. Control Law

Given the stable estimate of the motion and structure parameters (*i.e.*, $R, \frac{t}{d}, n$), we describe a control law to obtain a decoupled camera trajectory without loosing the visibility of features during servoing. Our approach is motivated by the controls presented in [5] and [13].

The translational velocity to move directly to the goal can be determined as $-\lambda_v(\frac{t}{d})\ d$, where λ_v is a gain factor and d is the distance to the plane (See Fig. 3). The rotational velocity is computed as $-\lambda_\omega \mathbf{u}\theta$, where λ_ω is again a gain factor and u, θ denote the rotation axis and angle that are obtained using the Rodriguez formula as $\theta = \arccos(\frac{1}{2}(tr(R)-1)),\ \ [\mathbf{u}]_\times = \frac{R-R^T}{2\ \mathrm{sinc}(\theta)}$, where $tr(R)$ indicates the trace of matrix R. A direct control in the Cartesian space might result in the features leaving the camera field of view. However, information from image features can be incorporated into the decoupled control to enforce the visibility constraint. We know from the image-based visual servoing control [10]

$$\begin{bmatrix} u-u^* \\ v-v^* \end{bmatrix} = \underbrace{\begin{bmatrix} -\frac{1}{Z} & 0 & \frac{u}{Z} \\ 0 & -\frac{1}{Z} & \frac{v}{Z} \end{bmatrix}}_{L_\nu} \underbrace{\begin{bmatrix} uv & -(1+u^2) \\ 1+v^2 & -uv \end{bmatrix}}_{L_{\omega_{xy}}} \underbrace{\begin{bmatrix} v \\ -u \end{bmatrix}}_{L_{\omega_z}} \begin{bmatrix} \nu \\ \omega \end{bmatrix}, \tag{5}$$

where $p = [u\ v\ 1]^T = [x\ \ 1]^T$, $p^* = [u^*\ v^*\ 1]^T = [x^*\ \ 1]^T$ and $Z = \mathbf{Z}(P)$ (See Fig. 3), while $[\nu\ \ \omega]^T$ denotes the velocity screw. Equation (5) can be rewritten as $x - x^* = [L_\nu\ L_{\omega_{xy}}\ L_{\omega_z}][\nu\ \ \omega_{xy}\ \ \omega_z]^T$, which yields

$$\omega_{xy} = L^{-1}_{\omega_{xy}}[(x-x^*) - L_\nu \nu - L_{\omega_z}\omega_z], \tag{6}$$

where $\nu = (\frac{t}{d})\hat{d}$ and $\omega_z = \mathbf{u_z}\theta$. In (6), the rotational motion ω_{xy} is controlled not only to minimize the differences between the current and the goal image features but also to compensate the effects caused by translation on the image. This ensures a straight-line trajectory of the features in the image. Estimates of Z and d are required in (5) and (6). This can be obtained as $\hat{Z} = \frac{\hat{d}}{n^T p}$ and $\hat{d} = \frac{\hat{d}^*}{det(H)}$, where $\hat{d}^*$ is an estimate of the constant distance to the plane π in $\mathcal{F}^*$ [13]. In general, this quantity is considered as a gain ratio and a coarse estimate obtained from a simple stereo technique is adequate [13]. Consequently, all the parameters required for the control are now available directly from the homography decomposition. In summary, the resultant expression for the velocity $\mathbf{v}$ is given as

$$\mathbf{v} = B \begin{bmatrix} \nu \\ L^{-1}_{\omega_{xy}}[(x-x^*) - L_\nu \nu - L_{\omega_z}\omega_z] \\ \omega_z \end{bmatrix}$$

$$where\ \ B = \begin{bmatrix} -\lambda_v I_{3\times 3} & 0_{3\times 2} & 0_{3\times 1} \\ 0_{2\times 3} & -\lambda_{\omega_{xy}} I_{2\times 2} & 0_{2\times 1} \\ 0_{1\times 3} & 0_{1\times 2} & -\lambda_{\omega_z} I_{1\times 1} \end{bmatrix}. \tag{7}$$

Equation (7) has only one singularity that occurs at $Z = 0$ (See expression for L_ν). However, the robust homography computation ensures that this degeneracy is always avoided.

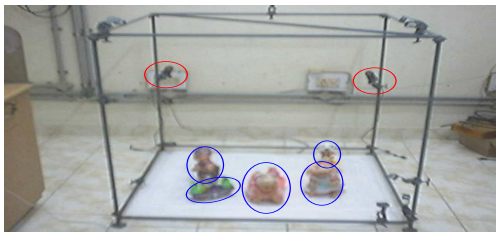


Fig. 5. Experimental Setup with the cameras and the objects used

Fig. 6. Non-Rigid Objects used in the experiments: Three sampled frames depicting the extreme positions during their motion

Thus using the space-time curves described by the points belonging to the object, the relative camera displacement can be reliably computed and used in the above control to achieve stable servoing behavior uninfluenced by the object deformations.

IV. Experimental Results and Analysis

We present a series of real and simulation results to validate the performance of the proposed technique. Real experiments were conducted to validate the proposed modeling scheme and also to verify the underlying concept behind the proposed servoing strategy *i.e.*, the estimation of projective transformation using space-time curve features and obtaining the relative end-effector displacement; while the performance of the servoing algorithm was studied in simulation. The basic implementation of our proposed algorithm can be summarized into the following steps.

1) In an off-line step, acquire images from the final goal position and extract the curves $\boldsymbol{c}_i^*$ describing the $3D$ point trajectories from these images
2) Acquire a sequence of images from the current camera pose and obtain $\boldsymbol{c}_i$
3) Estimate homography H_i induced by the space-time curves (Sect. III-A) and compute the reliable homography estimate H_{res} (Sect. III-B)
4) Decompose H_{res} to obtain the motion and structure parameters
5) Using (7), obtain the velocity instruction $\mathbf{v}$ and move the end-effector to the new pose
6) Repeat steps 2 to 5 until convergence

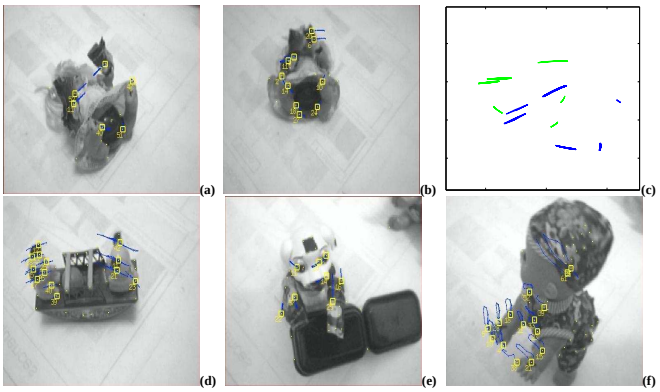


Fig. 7. Interest points tracked on the objects shown in Fig. 6. Fig.(a) and (b) show the two corresponding views of Object1. Fig.(c) plots the point trajectories obtained in case of (a) and (b) that were used to estimate the homography. Fig.(d),(e) and (f) show one of the views of the Objects 2,3 and 4 respectively, along with the interest point trajectories

Experiments were conducted using an imaging set-up consisting of a $1m$ cubical structure with holders to support CCD cameras (See Fig. 5). The object motion was tracked using the real-time GPU-based tracking system developed in [16]. The different objects used in the experiments are shown in Fig. 6. The objects possess multiple interest points moving with different velocities resembling non-rigid motion. Fig. 7 shows the tracked motion of the points on the object surface. Using these space-time curve features, the projective transformation between the two views was estimated and the relative camera displacement was computed. The computed result was compared to the ground truth obtained from a simple calibration technique. The estimated parameters were very close to the actual values in almost all the cases except in a few, when the considered point trajectory was out of plane. From this experiment, we ascertained that the relative displacement of the camera can be reliably estimated using the space-time curve features of the non-rigid object.

Simulations were conducted in MATLAB environment using a camera with a 512×512 pixel array and a sampling time of $T = 40ms$. Visual servoing was halted when the pixel error reduced below 1 pixel. An arbitrary configuration of points emulating a non-rigid object was considered. The non-rigid motion of the points was simulated using arbitrary planar curves. The image acquired at the initial and the desired camera position is displayed in Fig. 8(a). The servoing task was performed using the algorithm summarized above. Fig. 8(b) shows the smooth convergence of error norm and Fig. 8(d) displays the camera trajectory. At convergence, the camera arrives at the reference pose and the visual features coincide with the desired features. We observe that the control is stable and the translational and rotational velocities (Fig. 8(c)) converge to zero within few seconds. The proposed approach was also tested successfully on multiple initializations of curves. Further, several experiments were performed, using different initializations of the camera configurations, obtaining similar results.

Experiments were also conducted to analyze the performance of the algorithm in presence of noise in calibration

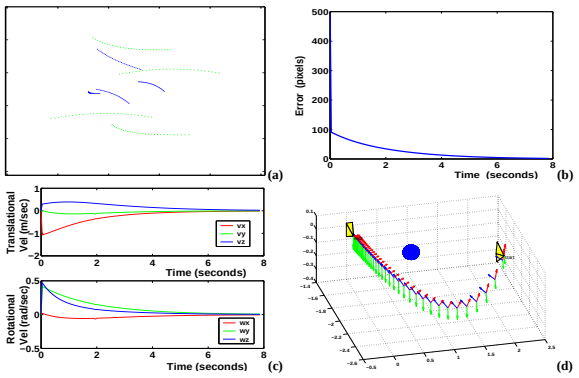


Fig. 8. Simulation Result: (a) Initial (blue) and Desired (green) images (b) Smooth convergence of Error Norm (c) Stable Velocity Screw (d) Camera Trajectory

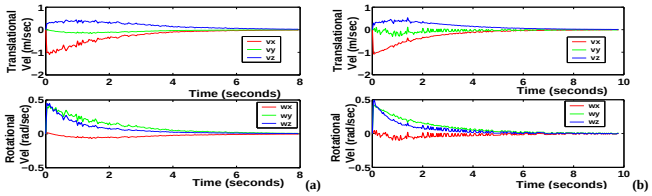


Fig. 9. Results in presence of (a) Calibration and (b) Image Measurement errors for the same case as in Fig. 8

(pose) and image measurements. An additive, zero-mean Gaussian noise with variance $\sigma = 0.1$ was considered. The parameters were varied in turn and average of the error measures was analyzed. From Fig. 9, we observe that the convergence is achieved even in presence of errors demonstrating the robust behavior of the approach.

Finally, an experiment was conducted to analyze the behavior of the robust homography computation algorithm. The degenerate case as shown in Fig. 4 was considered. The displacement that the camera had to realize was approximatively composed of a rotation of $5, -40$ and 0 degrees around camera x, y and z axis respectively and a translation of $5, 0$ and $1cm$ along those axis. Fig. 10(a) shows the variation in weights corresponding to the homographies. Observe that the weight corresponding to degenerate H tends towards the minimum value as the camera approaches the degeneracy. The smooth velocity screw in Fig. 10(b) demonstrates the stable behavior of the algorithm unlike Fig. 4(b).

V. Conclusions

A new framework for visual servoing has been proposed in this paper that accomplished the positioning task in unknown non-rigid environments. The algorithm utilized multiple homography estimates relating the current and desired camera views in conjunction with the novel non-rigid modeling scheme to accomplish the servoing task. The algorithm can handle most types of non-rigid motions and can tackle large camera displacements without being affected by degenerate configurations. A drawback with our present approach is the requirement of a high fps camera as the sensor has to perceive the target for a minimum of one cycle at every time step so as to acquire the complete space-time trajectory of the

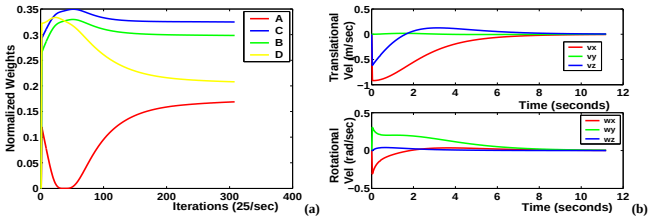


Fig. 10. Robust Homography Computation: (a) Normalized weight values (b) Velocity Screw. The weight corresponding to the degenerate curve reduces to a minimum when the degeneracy is approached, thus leading to a smooth velocity screw (unlike Fig. 4(b))

points. Note that although the points motion is cyclic with respect to a stationary camera; with a moving camera, the motion will not project onto periodic image paths due to the constantly changing camera pose relative to the point motion. However, this limitation can be overcome by using the recent developments in the field of multiple-view geometry of the space-time [9] that attempt to predict the point trajectory at the current pose using information acquired from current image and the past views. Our future work will be devoted to the development of this predictive control.

References

[1] A. Agarwal, C. V. Jawahar, and P. J. Narayanan. A survey of planar homography estimation techniques. Technical Report IIIT/TR/2005/12, International Institute of Information Technology, Hyderabad, 2005.

[2] J. K. Aggarwal, Q. Cai, W. Liao, and B. Sabata. Nonrigid motion analysis: articulated and elastic motion. *Computer Vision and Image Understanding*, 70(2):142–156, May 1998.

[3] A. D. Bue, X. Llad, and L. Agapito. Non-rigid metric shape and motion recovery from uncalibrated images using priors. *Computer Vision and Pattern Recognition (CVPR)*, 1:1191–1198, June 2006.

[4] F. Chaumette. Image moments: a general and useful set of features for visual servoing. *IEEE Transactions on Robotics*, 20(4):713–723, August 2004.

[5] K. Deguchi. Optimal motion control for image-based visual servoing by decoupling translation and rotation. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2:705–711, October 1998.

[6] O. Faugeras and F. Lustman. Motion and structure from motion in a piecewise planar environment. *International Journal of Pattern Recognition and Artificial Intelligence*, 2(3):485–508, June 1988.

[7] A. Giachetti and V. Torre. The use of optical flow for the analysis of non-rigid motions. *IJCV*, 18(3):255–279, June 1996.

[8] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2003.

[9] K. Hayakawa and J. Sato. Multiple view geometry in the space-time. *Asian Conference on Computer Vision*, 2:437–446, January 2006.

[10] S. A. Hutchinson, G. D. Hager, and P. I. Corke. A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, 12(5):651–670, October 1996.

[11] P. K. Jain and C. V. Jawahar. Homography estimation from planar contours. *International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)*, June 2006.

[12] I. Laptev. On space-time interest points. *International Journal of Computer Vision*, 64(2-3):107–123, September 2005.

[13] E. Malis, G. Chesi, and R. Cipolla. 2 1/2 D visual servoing with respect to planar contours having complex and unknown shapes. *International Journal of Robotics Research*, 22(10-11):841–854, October 2003.

[14] D. Santosh Kumar and C. V. Jawahar. Robust homography-based control for camera positioning in piecewise planar environments. *Indian Conference on Computer Vision, Graphics and Image Processing (ICVGIP)*, 906–918, December 2006.

[15] D. Santosh Kumar and C. V. Jawahar. Visual servoing in presence of non-rigid motion. *International Conference on Pattern Recognition*, 4:655–658, August 2006.

[16] S. N. Sinha, J. M. Frahm, M. Pollefeys, and Y. Genc. GPU-based video feature tracking and matching. Technical Report 06-012, Dept. of Computer Science, University of North Carolina, Chapel Hill, 2006.

Robust Homography-Based Control for Camera Positioning in Piecewise Planar Environments

D. Santosh Kumar and C.V. Jawahar

Center for Visual Information Technology
International Institute of Information Technology
Hyderabad 500032, India
{santosh@students., jawahar@}iiit.ac.in

Abstract. This paper presents a vision-based control for positioning a camera with respect to an unknown piecewise planar object. We introduce a novel homography-based approach that integrates information from multiple homographies to reliably estimate the relative displacement of the camera. This approach is robust to image measurement errors and provides a stable estimate of the camera motion that is free from degeneracies in the task space. We also develop a new control formulation that meets the contradictory requirements of producing a decoupled camera trajectory and ensuring object visibility by only utilizing the homography relating the two views. Experimental results validate the efficiency and robustness of our approach and demonstrate its applicability.

1 Robotic Vision

The use of computer vision techniques to control robotic systems has received great popularity in recent times [1]. Images captured by cameras attached to a robot provide ample information about its surroundings that assists it in efficiently navigating the environment. This field, known as *Visual Servoing* [2], has gained recent prominence due to the widespread availability of high quality cameras and low cost microprocessors. In addition to robotics, visual servoing algorithms also find interesting applications for interactive vision systems such as video conferencing, tracking, active vision, augmented reality etc. The visual feedback increases the accuracy of the overall vision system and relaxes the requirement of high precision accessories.

Many servoing techniques have been proposed and extensively studied in literature. In [3], optical flow is used to control the pose of the camera in conjunction with a Jacobian-based adaptive controller. In [4], $3D$ object pose is estimated and utilized to regulate the camera pose error. The class of algorithms similar to the former method constitute the popular *Image-based* Visual Servoing techniques while the latter pertain to *Position-based* approaches. For the relative merits and demerits of the above techniques, the reader may refer to [2]. Recently, a new group of algorithms have been proposed [5,6,7] that exploit a combination of the above methods to estimate the camera displacement between the desired and the current pose. They combine the traditional Jacobian-based control with

P. Kalra and S. Peleg (Eds.): ICVGIP 2006, LNCS 4338, pp. 906–918, 2006.

other techniques to form the class of *Hybrid Visual Servoing* algorithms. These methods yield a decoupled, straight-line camera trajectory and possess a large singularity-free task space.

Hybrid algorithms can essentially be classified into two primary categories. Algorithms in the first category are generally based on the computation of the essential matrix relating the two camera views [7,8]. Although the relative camera displacement can be obtained even for unknown (non-planar) scenes, a problem with epipolar geometry is that, it degenerates in certain critical cases (for example, when the target is planar or when the relative displacement is a pure rotation) and hence is not suitable for servoing. Note that a positioning task is accomplished only when the current and the desired images of the scene are similar, which corresponds to the degenerate case. The second class of algorithms determine the relative camera displacement by computing the homography induced by a scene plane relating the two views. However, a major drawback of these methods is the implicit assumption of the planarity of the scene, which prevents their application to real world scenarios as the world is often made up of non-planar regions. It must be emphasized that in either cases, the degeneracies critically affect the convergence and predictability of the system. Thus dealing with such degeneracies is of vital importance in the design of a stable system.

In summary, the desirable characteristics of a *hybrid* visual-control algorithm are

- Absence of degeneracies in its task space
- Applicability to both planar and non-planar environments
- Robustness to image measurement errors
- Continuity in velocity instruction and smooth convergence behavior
- Independence from prior knowledge of the object model and initialization of parameters

In this paper, we propose a new homography-based servoing algorithm that achieves the above features. Our method integrates homographies induced by multiple scene planes using geometric and subspace constraints to efficiently estimate the motion and structure parameters (Fig. 1). Another contribution of this paper is the development of a modified control law that provides the

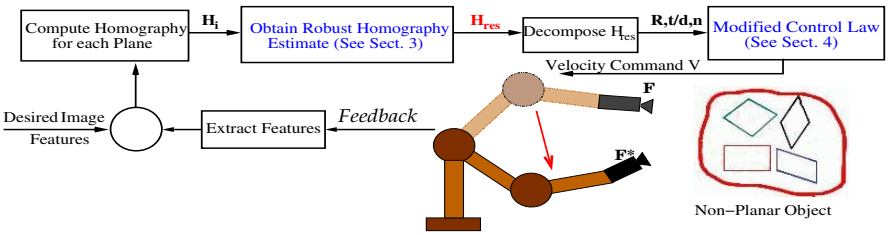


Fig. 1. Visual-feedback control: Multiple homographies are integrated to obtain a robust homography, which is used in the modified control law to gain superior performance

complementary characteristics of producing a decoupled camera trajectory and ensuring object visibility by only using the homography transformation relating the two camera poses.

2 Homography-Based Visual Control

A visual servo control compares the current image of a target with the desired image and the difference (or 'error') is used to drive the camera towards the goal position. Often the task is not just to regulate the image error but also to ensure a realizable camera trajectory. In such scenarios, homography-based control acts as a convenient option as it regulates the error in camera pose by estimating the $3D$ motion parameters only using image information.

If all the object points lie on a $3D$ plane, their coordinates in the current image I and the goal image I^* are related by a 'collineation' [9]. Assume that a point P lies on a plane whose normal vector is n as shown in Fig. 2. The point

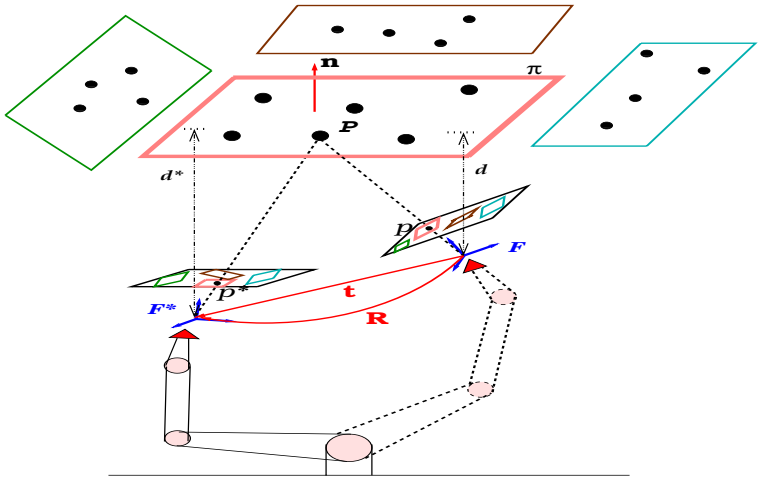


Fig. 2. Homography-based Visual Servoing

expressed in current camera frame $\mathcal{F}$ is related to goal camera frame $\mathcal{F}^*$ by a rotation matrix R and translation vector t as

$$P^* = R\ P + t = (R + t\frac{n^T}{d})P, \tag{1}$$

where $d = n^T P$ is the distance of the plane π from the current camera center. Assuming the camera intrinsic parameters are known, the image coordinates of the $3D$ points are given by $p = \frac{P}{Z}$ and $p^* = \frac{P^*}{Z^*}$ respectively. This transforms (1) to

$$\frac{Z^*}{Z}p^* = (R + t\frac{n^T}{d})p, \tag{2}$$

which can be rewritten as $\alpha p^* = Hp$ where $H_{3\times3} = R + t\frac{n^T}{d}$ is called the 'homography' matrix up to a scale factor α [9].

The recovered homography can be decomposed to obtain the rotation matrix R, the scaled translation vector $\frac{t}{d}$ and the plane normal n using the procedure described in [10]. Unfortunately, in the most general case the decomposition of H yields four different solutions (two of them being the 'opposites' of the other). They can be reduced to two solutions by applying the visibility constraint (*i.e.,* all the features must lie within the camera field of view). Further ambiguity can be resolved by decomposing an additional homography induced by another scene plane. Two pairs of solutions (S_1, S_2) and $(S_1^{'}, S_2^{'})$ are obtained respectively and a compatible pair $(S_i, S_j^{'})$ among them is found, *i.e.,* a pair with common motion $(R, \frac{t}{d})$. In general, there is only one compatible pair, and hence the unique solution can be obtained. Thus using information from multiple planes, H can be decomposed unambiguously to obtain the motion and structure parameters. These parameters are used in the control law to generate the optimal velocity instruction.

2.1 Degenerate Configurations and the Use of Multiple Planes

Some of the limitations of the existing *hybrid* techniques to estimate the relative camera displacement were reviewed in Sect. 1. Recently, another method was proposed by Malis *et al.* [6] to compute the relative orientation between the two camera views for a non-planar object using the concept of 'virtual parallax' [11]. By defining a plane using three arbitrary points on the object, they estimate the homography using this virtual plane and perform the positioning task.

A single homography estimate is not sufficient when a camera has to undergo large displacements in visual servoing as the control can be affected by degenerate configurations. Degeneracies in the task space can result either due to occlusion of the feature points, the camera center approaching the world (virtual) plane, the camera centers and the feature points arriving in a singular configuration [9] or due to singular homographies. In either of the cases, when a degeneracy is reached, the plane in consideration is switched *i.e.,* the points used to define the virtual plane are changed and a new plane using three different points is defined. This switching causes a discontinuity in the velocity command and leads to the instability of the control system. In Fig. 3, the effect of switching is demonstrated, where a positioning task with respect to a piecewise planar object was studied.

The other drawbacks in defining a non-planar object using arbitrary planes include

- **Unfavorable for planar scenes.** The methods using virtual parallax are theoretically inefficient to deal with planar objects as the epipolar geometry degenerates in this case [6].
- **Initialization of plane parameters.** In order to resolve the ambiguity in homography decomposition, *a priori* information about the normal vector of the virtual plane is required.
- **Assumption of point features.** Point correspondences are not available in many practical situations or could be noisy. Since the virtual plane is defined

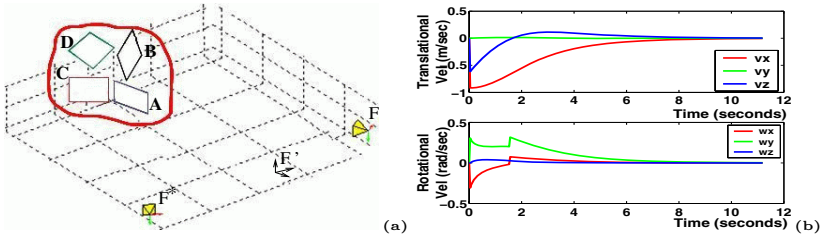


Fig. 3. Velocity Screw using virtual parallax algorithm: (a) Servoing begun using plane A reaches a degeneracy at $F^{'}$ whose origin intersects the plane (b) Discontinuity in the velocity screw is due to the switching of planes (A to C) at $F^{'}$

explicitly using the non-coplanar points on the object, these methods may not be applicable when such features are not available.

– **Effect of measurement errors.** Homography estimation is affected due to measurement errors (*'drift'*) in the correspondences. By choosing a different set of points (that are error-free) to define the virtual plane, one can obtain better results.

It must be emphasized that the above limitations are caused by the fact that only information from a single plane is being utilized to perform the positioning task. The bottleneck has been the fact that there exists no single homography relating the two camera views that can be absolutely relied upon. Nevertheless, by selectively exploiting the information available from multiple planes, one can avoid the above drawbacks and achieve superior performance.

3 Homography Estimation Using Multiple Planes

The objective of the servoing task is to drive the disparity between the current and the desired camera configurations to zero. The homographies relating the two camera poses induced by different planar regions are used to guide the positioning task.

Our approach proceeds initially by partially tessellating the non-planar scene into piecewise planar patches. This is done by a simple partitioning of the image features into homogeneous planar regions (See Fig. 6(a)). Interest regions are detected and the regions subject to planarity constraint form a set of matching regions [12]. The seed regions act as a 'driver' to guide the evolution of planar patches in the image. Any interest region detector with the ability to detect robust and stable regions can be employed here. For each pair of matching regions, a plane-induced homography is calculated.

Even though a single homography is sufficient to determine the motion parameters (*rigidity constraint*), information from multiple homographies can be combined to obtain a reliable estimate of the camera displacement. However, to avoid the estimation of multiple homographies at each instant, the constraints on homographies can be exploited to reduce the computations. Recall from (2)

that any H induced by a $3D$ scene plane is described by $H_{3\times3} \approx R + tn^T$. Given a homography matrix H_π induced by some $3D$ plane π, all other homographies H_i can be described as $\lambda_i H_\pi + tn_i^T$ for a fixed pair of cameras [9]. This observation results from the fact that all the homographies differ only in their scale λ_i and plane n_i parameters. Consider k homography matrices $H_1, H_2, \ldots, H_k$, each expressed as a column vector in a $9 \times k$ matrix. The rank of this matrix is known to be utmost four [13]. Hence the space of all homographies between two fixed camera views is embedded in a 4-dimensional linear subspace of $\Re^9$. This observation follows the fundamental fact that multiple planar patches in the scene share the common global camera geometry (*i.e.,* R, t).

3.1 Computation of the Reliable Homography

Given the rank-4 constraint, any new homography can be computed as a weighted combination of four linearly independent homographies. The four homographies are in general selected such that they are induced by planes that possess largest area and best visibility (if the centroid of the features in a planar region is within a threshold distance from the nearest image boundary, then it satisfies the visibility constraint) since they are the most reliable.

The resultant homography H_{res} is defined as

$$H_{res} = \lambda_1 H_1 + \lambda_2 H_2 + \lambda_3 H_3 + \lambda_4 H_4, \tag{3}$$

where the weights λ_i are assigned such that good homographies receive higher weights while the degenerate or errored estimates are given low priority. By appropriately choosing the λ_i's, a reliable homography can be deduced. Recall that, in general, any homography in the subspace can be expressed as a linear combination of four base homographies. In our case, H_{res} is one such 'valid' homography possessing certain desired characteristics.

The principle behind the weight assignment is to prefer valid homographies and reject singular ones in order to prevent abrupt switching of planes during a degeneracy. It must be emphasized that most of the degeneracies are not arbitrary changes and in general, can be predicted in advance. For instance, distance between a camera and a (virtual) plane gradually regresses to zero. Likewise, occlusion of planes can be anticipated by the persistent decrease in area of the planar region (or the number of features). Other degenerate cases can also be predicted in a similar manner and thus homographies that are likely to confront a degenerate configuration can be rejected.

Assignment of weights. Let us define the constraints to assign the weights and hence the parameter λ_i that is used in the computation of H_{res}.

- **Re-projection Error.** This constraint measures the accuracy of the estimated homography. A high error in re-projection indicates a poor estimate and such H should receive less weight as parameters obtained from it will be unreliable. Thus the weights are set inversely proportional to the re-projection error. This ensures that planar regions that are affected

by the cumulative tracking errors (*'drift error'*) are avoided and thereby guaranteeing the robustness of H_{res} to image measurement errors. The exact weight λ_i^e is defined by first calculating the re-projection error *i.e.,* $e = \sum_k d(p_k^*, Hp_k) = \sum_k ||\frac{p_k^*}{||p_k^*||} - \frac{H*p_k}{||H*p_k||}||$ and then assigning it using a one-sided Normal distribution $N(e_{thres}, \sigma_e)$ where e_{thres} is the tolerable re-projection error and σ_e is the variance.

- **Homography Determinant.** This quantity signifies the 'goodness' of a homography estimate. If the determinant is tending toward zero, it suggests the arrival of a degeneracy and hence such a homography should acquire low weight. Therefore the weights are set directly proportional to the value of the determinant D. This constraint ascertains the resultant homography to be free of singularities. Here again, the weights λ_i^D are set using a one-sided Normal density function $N(D_{thres}, \sigma_D)$ where D_{thres} is the minimum acceptable determinant.
- **Area of the Plane.** Occlusion of a plane can be detected by measuring the gradient of the plane area dA. If the area of the planar region decreases drastically, then it indicates a possible occlusion of this plane in the near future. Thus the λ_i's are to be set inversely proportional to the value of dA. More precisely, the weight λ_i^{dA} is set using a one-sided Normal distribution $N(dA_{thres}, \sigma_{dA})$ where dA_{thres} is the minimum acceptable gradient.

These weights are normalized and summed together to obtain the resultant weight λ_i. The final expression for H_{res} is calculated as

$$H_{res} = \sum_{i=1}^{4} \lambda_i H_i, \quad \text{where} \quad \sum_i \lambda_i = 1.$$

Hence a judicious assignment of weights using the above constraints helps in deducing a 'virtual' homography with the desirable characteristics. A change of bases might be required in case one of the H_i degenerate. However, the degenerate homography would automatically procure a low λ value and its replacement does not affect the stability of the system. This approach is applicable even if the scene consists of less than four planar regions. In such a case, the unavailable homographies in (3) acquire zero weight. It must be emphasized that the method utilizes additional homographies to obtain a reliable homography estimate rather than computing the optimal estimate. The parameters obtained from decomposition of H_{res} are used in the modified control law to compute the camera trajectory.

4 Modified Control Design

Given the stable estimate of the motion and structure parameters, our focus is to design a robust control that not only produces a decoupled camera trajectory but also guarantees feature visibility. Classical approaches such as the $3D$ control algorithms compute an optimal camera trajectory but very often violate

the visibility criteria. $2D$ controls ensure the features to remain in the camera field of view, although they suffer from non-optimal trajectory, computational complexity of calculating the Jacobian pseudo-inverse and the demand for $3D$ depth estimates. Note that providing the contradictory requirements of either controls poses a daunting challenge in the design of an optimum control scheme. Though a few attempts in this direction have been made [5,6,14], the devised controls do not satisfy all the above requirements.

Much of the information that is required for performing the positioning task is readily available from the homography transformation. The presence of multiple planes in the scene further compliments this fact. We exploit this result to fulfill the requirements of the desired optimal control.

Proposed Control. We first introduce the Cartesian ($3D$) control law and then proceed to derive the robust control. Given the parameters obtained from homography decomposition, the translational velocity to go directly to the goal is determined as $-\lambda_v(\frac{t}{d})\ d$, where λ_v is a gain factor and d is the distance to the plane (See Fig. 2). The rotational velocity is computed as $-\lambda_\omega u\theta$, where λ_ω is again a gain factor and u, θ denote the rotation axis and angle that are obtained using the Rodriguez formula for the rotation matrix R as $\theta = \arccos(\frac{1}{2}(tr(R)-1))$ and $[\mathbf{u}]_\times = \frac{R-R^T}{2\ \mathrm{sinc}(\theta)}$ [4].

However, a direct control in the Cartesian space might result in the features leaving the camera field of view. To enforce the visibility constraint, we use a single image point to control two axes of rotation (around x and y) and the final axis of rotation is controlled directly using the rotation matrix. This is done as follows: We know from the image-based visual servoing control [2]

$$\begin{bmatrix} u-u^* \\ v-v^* \end{bmatrix}_{2\times 1} = \Bigg[\underbrace{\begin{matrix} -\frac{1}{Z} & 0 \\ 0 & -\frac{1}{Z} \end{matrix} \begin{matrix} \frac{u}{Z} \\ \frac{v}{Z} \end{matrix}}_{L_\nu} \underbrace{\begin{matrix} uv & -(1+u^2) \\ 1+v^2 & -uv \end{matrix}}_{L_{\omega xy}} \underbrace{\begin{matrix} v \\ -u \end{matrix}}_{L_{\omega z}} \Bigg]_{2\times 6} \begin{bmatrix} \nu_{3\times 1} \\ \omega_{3\times 1} \end{bmatrix}_{6\times 1}, \quad (4)$$

where $p = [u\ v\ 1]^T = [x\ \ 1]^T$, $p^* = [u^*\ v^*\ 1]^T = [x^*\ \ 1]^T$, $Z = \mathbf{Z}(P)$ (See Fig. 2) and $[\nu\ \ \omega]^T$ denotes the camera velocity. Equation (4) relates the motion of image features *i.e.,* $x - x^*$ to the camera motion using the 2×6 Jacobian matrix L. It can be rewritten as $x - x^* = [L_\nu\ L_{\omega_{xy}}\ L_{\omega_z}][\nu\ \ \omega_{xy}\ \ \omega_z]^T$. Observe that a simple rearrangement of terms yields

$$\omega_{xy} = L^{-1}_{\omega_{xy}}[(x - x^*) - L_\nu\nu - L_{\omega_z}\omega_z], \quad (5)$$

where $\nu = (\frac{t}{d})\hat{d}$ and $\omega_z = \mathbf{u_z}\theta$. In (5), the rotational motion ω_{xy} is controlled not only to minimize the differences between the current and the goal image features but also to compensate the effects caused by translation on the image. This ensures a straight-line feature trajectory in the image and thereby guarantees object visibility. Estimates of the values Z and d are required in (4) that can be obtained as follows: Firstly, observe that

$$det(H) = det(R + \frac{tn^T}{d}) = det(R + \frac{t(n^{*T}R)}{d}) \quad (6)$$

$$= det(I + \frac{tn^{*T}}{d})det(R) = \frac{d + n^{*T}t}{d} \tag{7}$$

where (6) uses the fact that $n^* = Rn$ (See Fig. 2). Equation (7) can be further simplified using the result $d^* - d = n^{*T}P^* - n^TP = n^{*T}(P^* - RP) = n^{*T}t$. Hence we have $\hat{d} = \frac{\hat{d^*}}{det(H)}$. Using (7), Z can be calculated as

$$\frac{Z}{d^*} = \frac{Z}{d^*}\frac{d}{n^TP} = \frac{1}{n^Tp}\frac{1}{det(H)}. \tag{8}$$

Thus we have $\hat{Z} = \frac{\hat{d^*}}{n^Tp}\frac{1}{det(H)}$, where $\hat{d^*}$ is an estimate of the constant distance to the plane in the desired camera frame. In general, this quantity is considered as a gain ratio [6] and a coarse estimate obtained from a simple stereo technique is adequate. Consequently, all the parameters required for the control are now available directly from the homography decomposition.

In summary, the resultant expression for the velocity $\mathbf{v}$ is given as

$$\mathbf{v} = \begin{bmatrix} -\lambda_v I_{3\times3} & 0_{3\times2} & 0_{3\times1} \\ 0_{2\times3} & -\lambda_{\omega_{xy}} I_{2\times2} & 0_{2\times1} \\ 0_{1\times3} & 0_{1\times2} & -\lambda_{\omega_z} I_{1\times1} \end{bmatrix} \begin{bmatrix} \nu \\ \omega_{xy} \\ \omega_z \end{bmatrix} \left(= \begin{bmatrix} (\frac{t}{d})\ \hat{d} \\ L^{-1}_{\omega_{xy}}[(x - x^*) - L_\nu \nu - L_{\omega_z}\omega_z] \\ \mathbf{u_z}\theta \end{bmatrix} \right) \tag{9}$$

Equation (9) has only one singularity that occurs at $Z = 0$ (See expression for L_ν). However, as discussed in the earlier section, this degenerate configuration is avoided by the reliable homography computation algorithm. Thus by incorporating image features into the $3D$ control, an efficient control offering the complimentary features of object visibility and decoupled trajectory has been developed.

5 Experimental Results

In our experiments, we constructed an arbitrary configuration of planes as shown in Fig. 4(a). The projection of points belonging to these planar regions onto the image were considered as features. A perspective camera projection model was assumed. The basic implementation of the proposed algorithm given below was used to perform the positioning task.

1. Extract features from the current image and partition them into piecewise planar regions
2. Compute homography H_i induced by each region
3. Select four independent homographies induced by the regions that have the largest areas and best visibility (Only the selected regions need to be tracked in the successive iterations)
4. Determine the weights using the geometric constraints and compute the normalized weight λ_i for the selected homographies (Sect. 3.1)
5. Determine the robust homography H_{res} using (3)
6. Decompose H_{res} to obtain the motion and structure parameters (Resolve ambiguity using an additional homography)

7. Use the control law to obtain the velocity instruction $\mathbf{v}$ (See (9))
8. Repeat above steps until convergence

We analyzed the performance of our algorithm by generating several random initial camera configurations and then moving the camera to a fixed desired pose in a multi-plane scenario as shown in Fig. 4(a). Observe that a camera can frequently encounter degenerate cases during the positioning task in such a scene. However, in almost all the cases, the proposed algorithm was uninfluenced by degeneracies. In Fig. 4(c), the velocity command generated by the proposed approach for the particular scenario as tested in Fig. 3(a) is shown. Fig. 4(b) shows the variation in weights corresponding to the homographies. Observe that the weight corresponding to degenerate H tends towards the minimum value as the camera approaches the degeneracy. The smooth velocity screw in Fig. 4(c) demonstrates the stable behavior of the algorithm unlike in Fig. 3(b). Fig. 4(d) displays the camera trajectory. Note that the expression for Z in (8) requires at

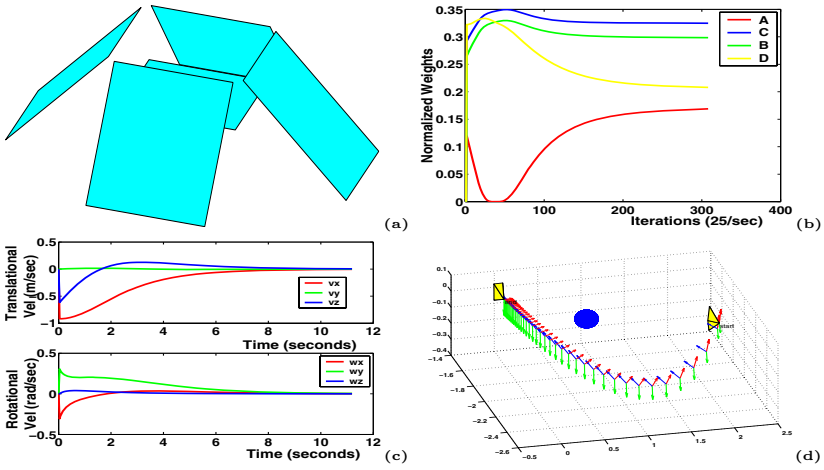


Fig. 4. (a) Non-Planar scene considered in the experiments. (b) Normalized weight values (c) Velocity Screw and (d) Camera Trajectory obtained for the scenario described in Fig. 3(a) . Smooth convergence even in presence of degeneracies confirms the stable behavior of the proposed approach.

least one feature p belonging to the planar region. However as a virtual homography is being used in our case, it might not correspond to any physical plane in the scene. In our method, we obtained this feature by finding the intersection of the plane inducing the virtual homography H_{res} with other scene planes as described in [15].

Analysis of the Control Law. The performance of the control law was analyzed in simulation. Fig. 5 shows the velocity screw and the image feature trajectory obtained during a positioning task using the proposed, $3D$ and the

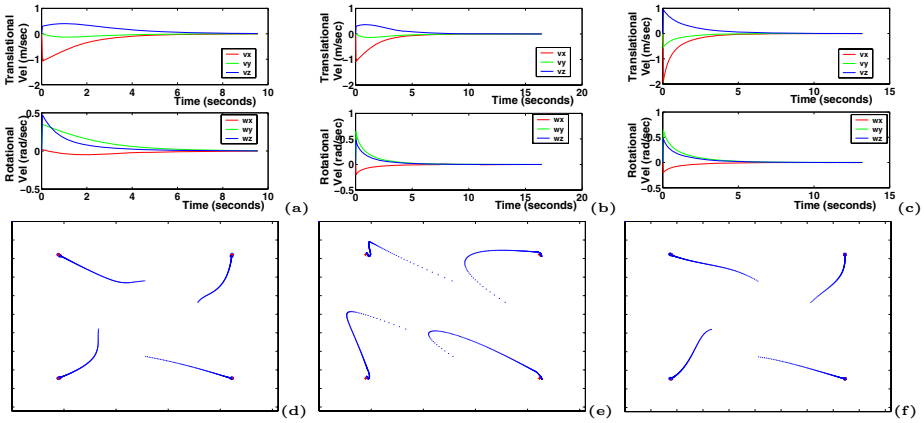


Fig. 5. Analysis of proposed control: Fig.(a),(b),(c) show the velocity screw obtained in case of proposed, $3D$ and 21/2D controls respectively while (d),(e),(f) display the feature trajectory. Similarity of velocity screws in (a) and (b) confirms the optimal trajectory behavior of the proposed control while near straight-line image feature trajectory in (d) ascertains the feature visibility.

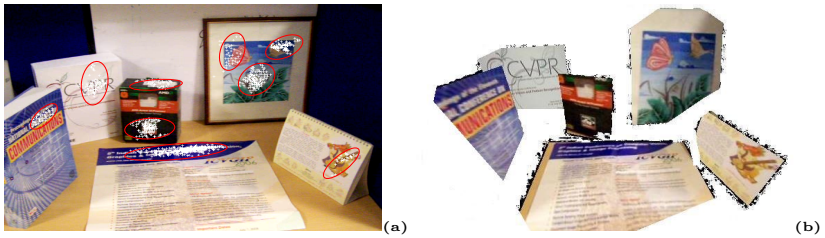


Fig. 6. Planar scene reconstruction using inter-image homographies: (a) A sample frame along with the detected interest regions on the scene planes (b) Reconstruction result

21/2D [6] controls respectively. The velocity screw obtained using the proposed control is very similar to the one obtained using the $3D$ control. Further, the feature trajectory almost follows a straight line. These two observations ascertain our claims of decoupled (straight-line) camera trajectory and object visibility using the proposed control. Inter-image homographies are an interesting tool for reconstruction of planar surfaces. The decomposition of homographies provide the $3D$ plane parameters required to reconstruct the scene. By considering a common feature belonging to two planes n_i and n, a relationship could be derived between their distances using (8) as

$$Z = \frac{d_i}{n_i^T p}\frac{1}{det(H)} = \frac{d}{n^T p}\frac{1}{det(H)} \quad i.e., \quad d_i = \frac{n_i^T p}{n^T p}d, \tag{10}$$

where p denotes the common image feature. Thus given the plane normals n_i, the $3D$ scene could be reconstructed up to a scale factor d (See Fig. 6(b)). Given an estimate of d, the exact scene can be reconstructed.

6 Conclusion

A novel homography-based control capable of positioning a camera even in presence of non-planar objects has been developed for the first time in this paper. A robust homography estimate was efficiently computed using multiple homographies by employing geometric and subspace constraints. This homography estimate was used in a modified control law to compute the optimal camera trajectory. The method performed better in comparison to existing servoing algorithms and avoided their critical drawbacks. In future, we plan to investigate further the utility of multi-plane homography-based formulations for efficiently solving other classical computer vision problems.

References

1. DeSouza, G., Kak, A.: Vision for mobile robot navigation: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence **24** (2002) 237–267
2. Hutchinson, S.A., Hager, G.D., Corke, P.I.: A tutorial on visual servo control. IEEE Transactions on Robotics and Automation **12** (1996) 651–670
3. Chaumette, F., Espiau, B.: A new approach to visual servoing in robotics. IEEE Transactions on Robotics and Automation **8** (1992) 313–327
4. Wilson, W.J., Hulls, C.C.W., Bell, G.S.: Relative end effector control using cartesian position based visual sovoing. IEEE Transactions on Robotics and Automation **12** (1996) 684–696
5. Taylor, C.J., Ostrowski, J.P., Jung, S.H.: Robust visual servoing based on relative orientation. IEEE Computer Society Conference on Computer Vision and Pattern Recognition **2** (1999) 574–580
6. Malis, E., Chaumette, F.: 2 1/2D visual servoing with respect to unknown objects through a new estimation scheme of camera displacement. International Journal of Computer Vision **37** (2000) 79–97
7. Rives, P.: Visual servoing based on epipolar geometry. IEEE/RSJ International Conference on Intelligent Robots and Systems **1** (2000) 602–607
8. Basri, R., Rivlin, E., Shimshoni, I.: Visual homing: surfing on the epipoles. IEEE International Conference on Computer Vision (1998) 863–869
9. Hartley, R., Zisserman, A.: Multiple view geometry in computer vision. Cambridge University Press (2003)
10. Faugeras, O., Lustman, F.: Motion and strucutre from motion in a piecewise planar environment. International Journal of Pattern Recognition and Artificial Intelligence **2** (1988) 485–508
11. Boufama, B., Mohr, R.: Epipole and fundamental matrix estimation using the virtual parallax property. IEEE International Conference on Computer Vision (1995) 1030–1036
12. Fraundorfer, F., Bischof, H.: Detecting distinguished regions by saliency. Scandinavian Conference on Image Analysis (2003) 208–215

13. Shashua, A., Avidan, S.: The rank-4 constraint in multiple view geometry. European Conference on Computer Vision **2** (1996) 196–206
14. Deguchi, K.: Optimal motion control for image-based visual servoing by decoupling translation and rotation. IEEE/RSJ International Conference on Intelligent Robots and Systems **2** (1998) 705–711
15. Johansson, B.: View synthesis and 3D reconstruction of piecewise planar scenes using intersection lines between the planes. IEEE International Conference on Computer Vision **1** (1999) 54–59

Appendix C

Related Publications

The work done during my masters has been disseminated to the following conferences/journals:

- D. Santosh and C.V. Jawahar, "Visual Servoing in presence of Non-Rigid Motion", The 18^{th} International Conference on Pattern Recognition (ICPR), 4:655-658, August 2006.
- D. Santosh and C.V. Jawahar, "Robust Homography-based Control for Camera Positioning in Piecewise Planar Environments", The 5^{th} Indian Conference on Computer Vision, Graphics & Image Processing (ICVGIP), pages 906-918, December 2006.
- D. Santosh and C.V. Jawahar, "Visual Servoing in Non-Rigid Environments: A Space-Time Approach", 24^{th} IEEE International Conference on Robotics and Automation (ICRA), pages 2452-2457, April 2007.
- D. Santosh and C.V. Jawahar, "Cooperative CONDENSATION-based Recognition", 8^{th} Asian Conference on Computer Vision (ACCV) 2007 (Under Review).
- D. Santosh and C.V. Jawahar, "Visual Servoing in Non-Rigid Environments", IEEE Transactions on Robotics (ITRO) 2008 (Under Submission).
- D. Santosh and C.V. Jawahar, "Robot Path Planning by Reinforcement Learning along with Potential Fields", In Preparation for the ICRA 2008.
- D. Santosh, A. Supreeth and C.V. Jawahar, "Mobile Robot Exploration and Navigation using a Single Camera", In Preparation for the ICRA 2008.

(All publications are available at http://students.iiit.ac.in/∼santosh.)

Bibliography

[1] J. K. Aggarwal, Q. Cai, W. Liao, and B. Sabata. Nonrigid motion analysis: articulated and elastic motion. *Computer Vision and Image Understanding*, 70(2):142–156, May 1998.

[2] M. Artac, M. Jogan, and A. Leonardis. Incremental PCA or on-line visual learning and recognition. *International Conference on Pattern Recognition*, 3, 2002.

[3] M. Artac, M. Jogan, and A. Leonardis. Mobile robot localization using an incremental eigenspace model. *IEEE International Conference on Robotics and Automation*, pages 1025–1030, 2002.

[4] P. Backes, K. Tso, J. Norris, G. Tharp, J. Slostad, R. Bonitz, and K. Ali. Group collaboration for mars rover mission operations. *IEEE International Conference on Robotics and Automation*, 2002.

[5] F. Bergholm, P. Trahanias, and S. Orphanoudakis. Analysis of current approaches in automated vision-based navigation. *European Research Survey*, 1997.

[6] H. Bischof, A. Leonardis, and D. Skocaj. A robust PCA algorithm for building representations from panoramic images. *European Conference on Computer Vision*, pages 761–775, 2002.

[7] G. Blanc, Y. Mezouar, and P. Martinet. Indoor navigation of a wheeled mobile robot along visual routes. *IEEE International Conference on Robotics and Automation*, pages 3354–3359, April 2005.

[8] R. Brooks. *Visual map making for a mobile robot*, chapter Readings in computer vision: issues, problems, principles, and paradigms, pages 438–443. Morgan Kaufmann Readings Series, 1987.

[9] C. Cavusoglu and et al. Control algorithms for active relative motion cancelling for robotic assisted off-pump coronary artery bypass graft surgery. *International Conference on Advanced Robotics*, July 2005.

[10] C.Furlanello, B.Crespi, and L.Stringa. A memory based approach to navigation. *Biological Cybernetics*, 69:385–393, 1993.

[11] F. Chaumette and B. Espiau. A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation*, 8(3):313–327, June 1992.

[12] Z. Chen and S. T. Birchfield. Qualitative vision-based mobile robot navigation. *IEEE International Conference on Robotics and Automation*, 2006.

[13] G. Chesi, E. Malis, and R. Cippola. Collineation estimation from two unmatched views of an unknown planar contour using visual servoing. *British Machine Vision Conference*, pages 224–233, September 1999.

[14] T.-J. Chin and D. Suter. Incremental kernel PCA for efficient non-linear feature extraction. *British Machine Vision Conference*, September 2006.

[15] W. Chin and S. Ntafos. Optimum watchman routes. *Information Processing Letters*, 28:39–44, 1988.

[16] J. Crowley and F. Pourraz. Continuity properties of the appearance manifold for mobile robot position estimation. *Image and Vision Computing*, 19(11):741–752, September 2001.

[17] D.A.Pomerleau. ALVINN: An autonomous land vehicle in a neural network. *Technical Report CMU-CS*, pages 89–107, 1989.

[18] A. J. Davison, I. Reid, N. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1–16, June 2007.

[19] K. Deguchi. Optimal motion control for image-based visual servoing by decoupling translation and rotation. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2:705–711, October 1998.

[20] F. Dellaert, W. Burgard, D. Fox, and S. Thrun. Using the condensation algorithm for robust, vision based mobile robot localization. *IEEE Conference on Computer Vision and Pattern Recognition*, 1999.

[21] X. Deng, T. Kameda, and C. Papadimitriou. How to learn an unknown environment. *Foundations of Computer Science*, page 298, 1991.

[22] G. N. DeSouza and A. C. Kak. Vision for mobile robot navigation: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(2):237–267, Febraury 2002.

[23] O. Faugeras and F. Lustman. Motion and structure from motion in a piecewise planar environment. *International Journal of Pattern Recognition and Artificial Intelligence*, 2(3):485–508, June 1988.

[24] P. Felzenszwalb and D. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

[25] C. Gaskett, L. Fletcher, and A. Zelinsky. Reinforcement learning for visual servoing of a mobile robot. *Australian Conf. on Robotics and Automation*, 2000.

[26] V. Graefe. Vision for autonomous mobile robots. *IEEE Workshop Advanced Motion Control*, pages 57–64, 1992.

[27] P. Hall, D. Marshall, and R. R. Martin. Incremental eigenanalysis for classification. *British Machine Vision Conference*, 1:286–295, 1998.

[28] J. Ham, Y. Lin, and D. Lee. Learning nonlinear appearance manifolds for robot localization. *IEEE International Conference on Robotics and Automation*, April 2005.

[29] Hans, P. Moravec, and A. Elfes. High resolution maps from wide angle sonar. *IEEE Int. Conf Robotics and Automation*, pages 116–121, 1985.

[30] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2003.

[31] K. Hashimoto, K. Nagahama, T. Noritsugu, and M. Takaiawa. Visual servoing based on object motion estimation. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1:245–250, October 2000.

[32] D. Hoiem, A. Efros, and M. Hebert. Automatic photo pop-up. *ACM SIGGRAPH*, pages 577–584, 2005.

[33] S. A. Hutchinson, G. D. Hager, and P. I. Corke. A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, 12(5):651–670, October 1996.

[34] S. Iyengar, C. Jorgensen, S. Rao, and C. Weisbin. Robot navigation algorithms using learned spatial graphs. *Robotica*, 4(2):93–100, 1986.

[35] P. K. Jain and C. V. Jawahar. Homography estimation from planar contours. *International Symposium on 3D Data Processing, Visualization and Transmission, University of North Carolina, Chapel Hill*, June 2006.

[36] M. Jogan and A. Leonardis. Robust localization using an omnidirectional appearance-based subspace model of environment. *Robotics and Autonomous Systems*, 45(1):51–72, 2003.

[37] M. Jogan, A. Leonardis, H. Wildenauer, and H.Bischof. Mobile robot localization under varying illumination. *International Conference on Pattern Recognition*, 2:741–744, 2002.

[38] B. Kalyanasundaram and K. Pruhs. A competitive analysis of algorithms for searching unknown scenes. *Computational Geometry: Theory and Applications*, 3:139–155, 1993.

[39] O. Khatib. Real time obstacle avoidance for manipulators and mobile robots. *Int. Journal of Robotics Research*, 5(1):90–98, 1986.

[40] B. H. Kim. Localization of a mobile robot using images of a moving target. *IEEE Int. Conf. Robotics Automation*, pages 253–258, 2001.

[41] S. Kimura, S. Ozawa, and S. Abe. Incremental kernel PCA for online learning of feature space. *International Conference on Computational Intelligence for Modelling, Control and Automation*, 1:595–600, November 2005.

[42] B. Krose and R. Bunschoten. Probabilistic localization by appearance models and active vision. *IEEE International Conference on Robotics and Automation*, pages 2255–2260, 1999.

[43] J. P. Laboratory. Mars exploration rover. *NASA fact sheet*, 2004.

[44] J. C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, 1991.

[45] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, November 2004.

[46] P. Maes and R. A. Brooks. Learning to coordinate behaviors. *National Conference on Artificial Intelligence*, pages 796–802, 1990.

[47] E. Malis and F. Chaumette. 2 1/2d visual servoing with respect to unknown objects through a new estimation scheme of camera displacement. *International Journal of Computer Vision*, 37(1):79–97, June 2000.

[48] E. Malis and F. Chaumette. Theoretical improvements in the stability analysis of a new class of model-free visual servoing methods. *IEEE Transactions on Robotics and Automation*, 18(2):176–186, April 2002.

[49] E. Malis, F. Chaumette, and S. Boudet. 2 1/2d visual servoing. *IEEE Transactions on Robotics and Automation*, 15(2):238–250, April 1999.

[50] E. Malis, G. Chesi, and R. Cipolla. 2 1/2 d visual servoing with respect to planar contours having complex and unknown shapes. *International Journal of Robotics Research*, 22(10-11):841–854, October 2003.

[51] T. Martinez-Marin and T. Duckett. Fast reinforcement learning for vision-guided mobile robots. *International Conference on Robotics and Automation*, pages 4170–4175, April 2005.

[52] Y. Matsumoto, K. Sakai, M. Inaba, and H. Inoue. View-based approach to robot navigation. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 3:1702–1708, 2000.

[53] Y. Mezouar. *Planification de trajectories pour l'asservissement visuel*. PhD thesis, Universite de Rennes 1, IRISA, 2001.

[54] Y. Mezouar and F. Chaumette. Model-free optimal trajectories in the image-space: Application to robot vision control. *IEEE Computer Vision and Pattern Recognition*, 2001.

[55] Y. Mezouar, A. Remezeilles, P. Gros, and F. Chaumette. Image interpolation for image-based control under large displacement. *International Conference on Robotics and Automation*, 2002.

[56] T. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

[57] T. M. Mitchell. Becoming increasingly reactive. *American Association of Artificial Intelligence*, 1990.

[58] S. Nayar, S. Nene, and H. Murase. Subspace Methods for Robot Vision. *IEEE Transactions on Robotics and Automation*, 12(5):750–758, 1996.

[59] A. Remazeilles and F. Chaumette. Image-based robot navigation from an image memory. *Robotics and Autonomous Systems*, 55(4):345–356, April 2007.

[60] D. Santosh and C. V. Jawahar. Robust homography-based control for camera positioning in piecewise planar environments. *Indian Conference on Computer Vision, Graphics and Image Processing*, pages 906–918, December 2006.

[61] D. Santosh and C. V. Jawahar. Visual servoing in presence of non-rigid motion. *International Conference on Pattern Recognition*, 4:655–658, August 2006.

[62] D. Santosh and C. V. Jawahar. Visual servoing in non-rigid environments: A space-time approach. *International Conference on Robotics and Automation*, pages 2452–2457, April 2007.

[63] A. Saxena, S. H. Chung, and A. Y. Ng. Learning depth from single monocular images. *Advances in Neural Information Processing Systems 18*, 18:1161–1168, June 2006.

[64] B. Scholkopf, A. Smola, and K.-R. Muller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

[65] S. Se, D. Lowe, and J. Little. Vision-based mobile robot localization and mapping using scale-invariant features. *IEEE Conf on Robotics and Automation*, pages 2051–2058, 2001.

[66] G. Silveira and E. Malis. Real-time visual tracking under arbitrary illumination changes. *IEEE Computer Vision and Pattern Recognition*, To appear, June 2007.

[67] R. Sim and G. Dudek. Comparing image-based localization methods. *International Joint Conference on Artificial Intelligence*, pages 1560–1562, August 2003.

[68] R. Sim and G. Dudek. Effective exploration strategies for the construction of visual maps. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 4:3224–3231, October 2003.

[69] W. D. Smart and L. P. Kaelbling. Effective reinforcement learning for mobile robots. *IEEE Int. Conf. on Robotics and Automation*, 4:3404–3410, 2002.

[70] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3D. *ACM SIGGRAPH*, 25(3):835–846, July 2006.

[71] K.-T. Song and J.-H. Huang. Fast optical flow estimation and its application to real-time obstacle avoidance. *IEEE Int. Conf. Robotics and Automation*, pages 2891–2896, 2001.

[72] A. Stentz. The NAVLAB system for mobile robot navigation. *Carnegie Melon Technical Report*, 1989.

[73] V. A. Sujan. Task directed imaging in unstructured environments by cooperating robots. *Indian Conference on Computer Vision, Graphics and Image Processing*, pages 37–42, December 2002.

[74] S. Thrun. An approach to learning mobile robot navigation. *Robotics And Autonomous Systems*, 1995.

[75] S. Thrun. An online mapping algorithm for teams of mobile robots. *International Journal of Robotics Research*, 2001.

[76] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, Cambridge, 2005.

[77] B. Tovar, L. Muñoz-Gómez, R. Murrieta-Cid, M. Alencastre-Miranda, R. Monroy, and S. Hutchinson. Planning exploration strategies for simultaneous localization and mapping. *Robotics and Autonomous Systems*, 54(4):314–331, April 2006.

[78] G. Vijaykumar, J. A. Franklin, and H. Benbrahim. Acquiring robot skills via reinforcement learning. *IEEE Control Systems*, 272(1708):13–24, 1994.

[79] C. Watkins. *Learning from Delayed Rewards*. PhD thesis, Cambridge University, Cambridge, England, 1989.

[80] J. Weng and S. Chen. Visual learning with navigation as an example. *IEEE Intelligent Systems*, 15(5):63–71, September 2000.

[81] L. Whitcomb, D. Yoerger, and H. Singh. Advances in doppler-based navigation of underwater robotic vehicles. *IEEE International Conference on Robotics and Automation*, 1999.

[82] B. Yamauchi. A frontier-based approach for autonomous exploration. *IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pages 146–151, July 1997.