Vision System for Autonomous Navigation

Thesis submitted in partial fulfilment

of the requirements for the award of the degree of

Master of Technology

in

Signal and Image Processing

by

Neelam Abhinav Karthik

(212EC6184)

Department of Electronics & Communication Engineering

NATIONAL INSTITUTE OF TECHNOLOGY, ROURKELA

राष्ट्रीय प्रौद्योगिकी संस्थान, राउरकेला

May 2014

Vision System for Autonomous Navigation

Thesis submitted in partial fulfilment

of the requirements for the award of the degree of

Master of Technology

in

Signal and Image Processing

by

Neelam Abhinav Karthik

(212EC6184)

Under the supervision of

Prof. Sukadev Meher

Department of Electronics & Communication Engineering

NATIONAL INSTITUTE OF TECHNOLOGY, ROURKELA

राष्ट्रीय प्रौद्योगिकी संस्थान, राउरकेला

May 2014

Department of Electronics and Communication Engineering
National Institute of Technology Rourkela
ROURKELA-769 008, ODISHA, INDIA

May __ , 2014

Certificate

This is to certify that the thesis titled as ***"Vision System for Autonomous Navigation"*** by **"*Neelam Abhinav Karthik*"** is a record of an original research work carried out under my supervision and guidance in partial fulfilment of the requirements for the award of the degree of ***Master of Technology*** in ***Electronics and Communication Engineering*** with specialization in ***Signal and Image Processing*** during the session 2013-2014.

_____

Prof. Sukadev Meher

Acknowledgements

First and foremost, I am truly indebted to my supervisor, Prof. Sukadev Meher, for his constant inspiration, excellent guidance and valuable discussions leading to fruitful work. From finding a problem to solving, his careful observation helped me a lot in my dissertation work and of course in due time. There are many people who are associated with this project directly or indirectly whose help, timely suggestions helped a lot in the successful completion of this project.

I would like to thank Deepak Kumar Panda, Aditya Acharya, Deepak Singh, Bodhisattwa Chakraborty, and all my friends, research members of Image processing and Computer Vision lab of NIT Rourkela and Achala Pandey for their timely suggestions and clear insight .

I am very much indebted to Prof. Sarat Kumar Patra and Prof. Kamala Kanta Mohapatra for teaching me subjects that proved to be very helpful in my work. My special thanks go to Prof. Ajit Kumar Sahoo, Prof. L.P. Roy and Prof. Samit Ari for contributing towards enhancing the quality of the work and eventually shaping my thesis. My wholehearted gratitude to my parents Ramesh and Jayasree and my sister Ritu for their love and support.

N. Abhinav Karthik

Rourkela, May 2014

ABSTRACT

Vision based navigation of robots has been an active field of research in the past decade. There are many challenges in making the vision system understand the environment in which it is placed. Such an environment can be either indoors or outdoors depending on the task at hand. Outdoor environments seldom have structure and require complex geometric models to model such environments. Little structure that exists within such an environment in the form of roads, lane markings…etc. help reduce the model complexity. On the other hand, indoor environments are relatively easier to model considering the inherent structure in building constructions, thus becoming mathematically tractable as compared to the outdoor environments. The real challenge in indoor navigation is the localization of the camera system, which involves the vision system estimating its location in the environment in which it is navigating.

This thesis explores the possibility of using a constrained indoor environment model to simplify the mathematics and computation involved. Several methods were studied, but the approach involving planar objects to estimate the pose of the camera relative to a fixed planar element was found to be computationally efficient and simple. The camera localization was modelled as a 3D pose estimation problem with the camera pose relative to the planar structures of the environment as the pose estimates.

The primary approach to navigating the robot is estimating the robots motion using the visual feed from the camera mounted atop the robot. The use of vanishing point is suggested because of the robustness even in the presence of occlusions. The vanishing point of the environment is the point at which all the 3D parallel lines viewed in perspective appear to

converge. This vanishing point gives us a coarse approximation of the 3D structure which is further improved upon using homography based techniques.

The homography based approach to navigation involves tracking the planar structures over consecutive frames to indirectly estimate the motion of the camera. The approach taken involves tracking key-points over multiple frames as the camera moves through the environment. These tracks are then analysed and fit in the environment model to estimate the motion trajectory of the camera. The environment is modelled as a '*Manhattan World*' which is a structure made completely of planar elements. This is a reasonably good assumption to make, thus involving minimal model complexity. Thus, the motion trajectory of the camera is estimated based on the video stream from the camera, thereby allowing for a feed back to control the motion of the camera. The algorithm was tested on an indoor navigation dataset and the results were analysed.

INDEX

List of Figures

List of Algorithms

Chapter 1

INTRODUCTION

1.1 Autonomous Navigation

The exact definition of the term 'Autonomous Vehicle' is still an open ended question. One may safely call it as a mobile robot capable of cleverly navigating a given environment with little or no human interaction required whatsoever. In the context of the current thesis, a robot may defined as an intelligent machine capable of interpreting inputs and responding to them accordingly. The robots environment is termed as a complex world under certain constraints and assumptions made to ensure that the robot is capable of interacting with it-with minimal hindrance. As the complexity of the world increases so does the need for a complex intelligence algorithm, better sensors ... etc. Hence it always helps to make assumptions that are true in general when it comes to man-made environments. Assumptions such as the environment being relatively flat throughout-is fairly reasonable and can reduce the complexity of the environment considerably. [17]

Fig 1.1 a) Stanley at the DARPA Desert Challenge b) CMU's Navlab Autonomous Vehicle Platform

The study of autonomous vehicles is a relatively new area of research and can be considered as a niche area of robotics that has now become possible only due to the latest technological advancements. The interest in robotics and autonomous navigation research was fuelled by the human need to control the world they live in. Their constant pursuit of making their lives simpler is also - though in part – responsible for the recent developments in the related fields of automation. The research in this field has reached a point where autonomous navigation has become possible. However, a number of issues are yet to be addressed before taking this technology to the mainstream market.

Navigation may be vaguely termed as the processing of finding a suitable and safe path between a start and a terminal point for the robot to traverse. [18]. Visual navigation with specific application to mobile robots has kindled countless contributions. This is mainly due to the rise in the possibilities for their application in autonomous mobile robot navigation. Traditionally, navigation solutions primarily based on vision are typically limited to the Autonomous Ground Vehicles (AGV). But their recent applications to UAV's has also been observed through the publication of many research papers in this field. The use of vision based navigation on UAV's is of high interest in application relating to remote surveillance, disaster mitigation, search and rescue missions and other similar situations where the observers high altitude vantage point can be thoroughly exploited. Since UAV's navigate in the three dimensional space, they are not subjected

Fig 1.2 a) Autonomous Underwater Vehicle b) Unmanned Air Vehicle

to the constraints that an AGV faces. However, the reduced size of the UAV limits its payload capabilities thereby greatly reducing the collection of sensors available for use.

In the case of navigation underwater, typical choices are inclined more towards imaging modalities that work best in fluid media such as water. The use of sound-based navigation is often sought over other sensor-based techniques owing to their effectiveness in these environments. Nevertheless, their advantages come coupled with their own set of limitation as discussed in [19]. Limitations involving the limited resolution and size of sound-based imaging systems has often been seen as a caveat in the development of Autonomous Underwater Navigation Systems (AUV's). Presently, a number of AUV's are in action serving important purposes such as installation and maintenance of deep sea communication lines, monitoring the marine eco-system … etc.

Irrespective of the type of vehicle, the navigation systems being used may be broadly classified into two classes – one with prior knowledge of the environment and the other without prior knowledge of the environment. The second class of navigation systems perceive and try to make sense of the environment as they navigate through it. Mapless navigation involves passive techniques that use visual cues obtained through optical flow, feature tracking … etc. There is no global model for the environment, the environment is perceived as the system navigates through it and makes note of the visual cues to localize itself within that environment. Map-Based Navigation systems are heavily reliant on user generated geometric models of the environment.

The restrictions put on by the definite geometric model of the environment greatly reduce the complexity of the navigation problem. However, these kind of assumption about regularity in the structure of the environment only seem to describe man-made environment and fail when the robot is in natural environments such as mountain or desert terrains. The third class of navigation which

is sub class of map based approach is the map building based navigation system. They are systems that take the help of sensors to define their own geometric models of the environment, hence using these for the purpose of navigation.

1.2 Autonomous Navigation Challenges

There are several impediments that render the many approaches to autonomous navigation useless during real time applications. The common problem encountered during real-time navigation are those pertaining to anomalies in the environment that deviate them from the typical model of the environment. Obstacle in the environment disrupt the estimates of the robots location in case of passive navigation techniques such as optical flow or feature tracking. On the other hand, wheel odometry based navigation systems suffer from wheel - slip situations resulting in location estimates that are far from the robots true location. The wheel slip situation often happens when the friction in the wheel ground interface in not enough to counter the torque from the wheels rotation. An erroneous measurement of wheel rotation is recorded via the wheel encoders resulting in a faulty robot location estimate.

In the context of vision based navigation often the problem of obstacles is encountered. The movement of persons in usually inevitable in the environment that the robot is navigating through. Errors in the sensor measurements such as that imaging sensor a.k.a the camera system due to digitization process. Also defects in the camera construction during manufacturing such as lens related aberrations and improper positioning of the image sensor also prove to be problematic during image acquisition step. These effects trickle down the motion estimation pipeline and eventually corrupt the localization output. On the other end, the vision based navigation systems

fall prey to unavailability of proper visual cues in the environment. Key features based tracking approach often suffer in environments with uniformly coloured surfaces that often lack in texture.

Presence of good amount of texture is essential in feature-tracking-based approaches to navigation. Presence of clutter in the navigating environment such as bushes and other foliage often confuse vision systems in the case of outdoor environments. Outdoor environments also pose the problem of lighting variation as the robot moves from one place to another .In addition to this the effect of shadows cannot be ignored as well. Proper localization of the robot within its environment is one of the major challenges in autonomous navigation. Several successful attempts were made to solve the problem of localization, the most significant one being the Simultaneous Localization and Mapping algorithm. Several other approaches to address the problem of localization in the context of robot navigation have been discussed in the later sections.

1.3 Related Work

The approach sought in the case of autonomous navigation is critically dependent on the type of environment. Indoor environments such as building corridors, warehouses, manufacturing plants … etc. fall under the indoor environment category rich in coherent structure. On the other hand out door environment such as road and desert like terrains put forth a different set of obstacles to overcome before one can achieve autonomous navigation. Since these are completely different problems, each had its own class of research being carried out in the context of indoor navigation.

Appearance based navigation schemes were first explored in [2] and later on much improvement was done as presented in [1] wherein the road region detection algorithms were employed to segment out the road like regions based on colour. SCARF, a colour based vision system capable

of identifying tough to recognize roads and intersections. It was successfully implemented on the NAVLAB navigation platform from CMU and tested successfully. SCARF is capable of detecting road regions with no clear demarcations of boundaries up to a decent level of accuracy. It was a pioneering system capable of detecting intersection without prior knowledge of the road intersections. The approach taken was through a Bayesian classifier to estimate the probability of the region being road-like. In [3] the approach taken differed from [4] as it used edge based shape models for road detection. The edge detected image of the scene was processed to sift the left and right road borders and then restrict the area which will eventually be parameterized into its mean and variance.

1.3.1 Outdoor Navigation

The outdoor navigation strategies involving AGV's are mostly focussed on structured and ill structured terrains. Owing to the vast differences in the road conditions from one location to the other, haphazard movement of objects on the road like terrains and changes in illumination a robust navigation strategy is needed. One of the pioneering works in this area was done by Tsugawa et al. which used a stereo camera rig to detect obstacle encountered during navigation. The CMU's NAVLAB platform [1, 2] for testing of autonomous navigation algorithms and the University of Maryland's [20.] made use of laser based approaches to obstacle detection for partially downtrodden roads. When it comes to vision based approaches to autonomous navigation in structured outdoor environments, it turned a problem of road following. Research in the field of Road Following based approach to navigation were explored in [3, 4]. One of the most successful work in the field of navigation on structures was done during the DARPA Desert challenge. The

work on self-supervised monocular road detection method [5] lead to the success of one of the first completely autonomous navigation systems ever to win at an international level. This approach used an approach that modelled the road colour to segment it and learn that model under varying lighting as well as road conditions.

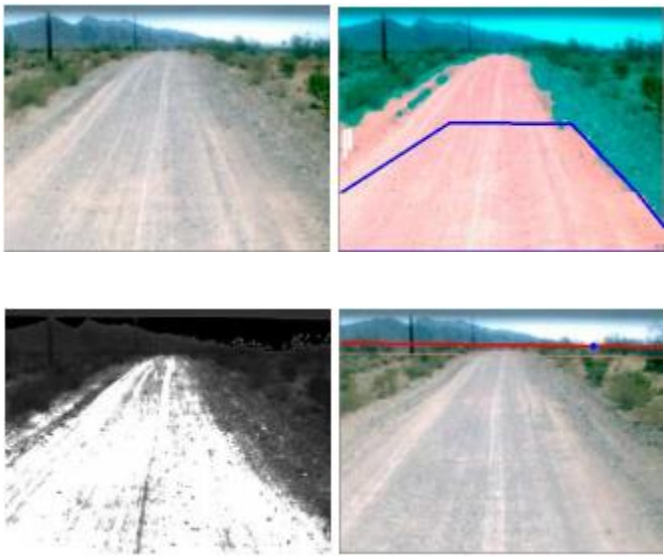

Fig 1.3 Road Segmentation in Desert terrain as in [5] with first row consisting of the raw and segmented images and the second road with the likelihood image of the pixel being a road.

The learning nature of the algorithms implementation led to the success of the system which allowed it to quickly adapt to changing road situations. The approach used was Mixture-of-Gaussian where in the road region was modelled as a collection of different Gaussians parameterized by their means and covariance's in the RGB colour space.

Probabilistic approaches to navigation in urban environments was studied in [6] where in combination of sensor data from multiple sources was used for localizing the car. This work is an improvement over its previous versions with the addition of learning based approaches to improve the accuracy of the maps built-over time. A probabilistic occupancy grid is proposed parameterized by the Gaussian mean and variance values. An offline SLAM is also suggested to improve upon

the map details on multiple passes through a particular route. With an accuracy of up to 10cm, this approach supersedes all of the previous work done in localization by a huge margin.

Fig 1.4 Stanford's Junior with a surmounted Velodyne HD-LIDAR 64-beam scanner is circled in red.

This opened up a new domain of opportunities in autonomous navigation and enabled the possibility of autonomous navigation over long distance on urban roads with any worry about localization errors.

1.3.2 Indoor Navigation

The work in the field of indoor navigation dates back to 1979 carried out by Giralt et al. [20] which was followed by the work of Moravec in the early 1980's [22,23]. Their work pointed out the ineluctable need to include some level of understanding of the environment that the computer sees. This required a rigorous modelling of the geometry of the environment in which the robot navigated. A very elaborate CAD models of the environment were used for the purpose of

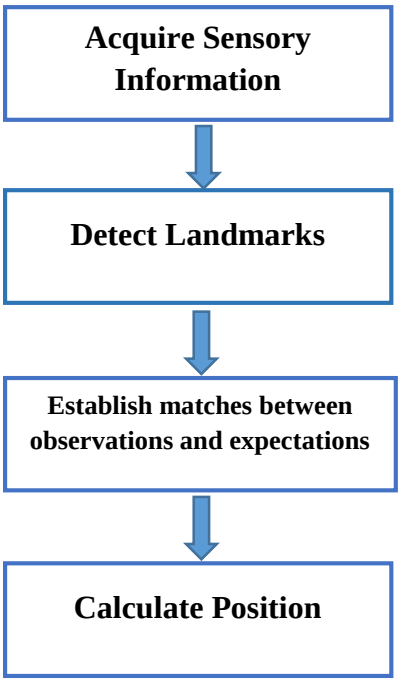

Fig 1.5 Steps involved in vision based localization

navigating the mobile platform [24] in some of the very first vision systems the approach was based on occupancy map in which the projection of the object in 3D onto the 2D space. This was further improved upon using the VFF based approach in [25]. The main notion of map-based navigation is to provide the robot with some vague knowledge of the visual landmarks it may find as it navigates through the environment. Using these landmarks the robot would then be able to localize itself with respect to those landmarks. These approach are termed as absolute localization problems. On the other hand we have incremental localization problems in which the location of the robot is updated incrementally based on the relative motion of some key features within the environment. The FINALE system developed in [26] makes use of a geometric model and stochastic prediction which is being done in the Hough transform space. In absolute localization of robots in the indoor environment, a mapping between the observation and the prediction is to be done. Any ambiguity in the localization should be resolved using Markov localization based approaches [27] or Monte Carlo Localization [28]. The fundamental notion driving the localization

problem is the ability of the vision systems to robustly recognize those features in the image that remain invariant as the robotic platform moves through a corridor. Triangulation based approach using image feature correspondences to localize the robot is also an extensively explored field. Challenges remain in localization owing to the inaccuracies in the feature location due to spatial quantization during the imaging process and uncertainties in the location of landmarks. Indoor navigation approaches in recent times make use of dynamic scene understanding was explored in [7] which used a hypothesis based indoor scene understanding for the purpose of navigation. Being an incremental process, it extends a hypothesis to children hypotheses, gradually homing on the true structure of the environment. It makes a relatively strong assumption about the environment considering it be a *Manhattan world* thus simplifying the navigation. Further improvements were done in this direction in [8] by using a Planar Semantic Model of the environment.

1.4 Motivation

The main theme of autonomous navigation is based around the localization process. Localizing the robot within its environment is a crucial task and has been extensively explored in many research papers. Vision based motion estimation or Odometry has been studied in [6] and many other papers. There is need for a highly accurate localization mechanism for error free navigation. The typical Visual Odometry Pipeline is presented in figure below as described in [9] gives us an idea of the various steps involved in the visual Odometry process. As it can be seen here, the motion estimation can be done in 3ways using 1) 2D-2D 2) 1) 3D-3D or 3) 3D-2D feature correspondences. But the complexity involved in 3D-2D and 3D-3D correspondences based

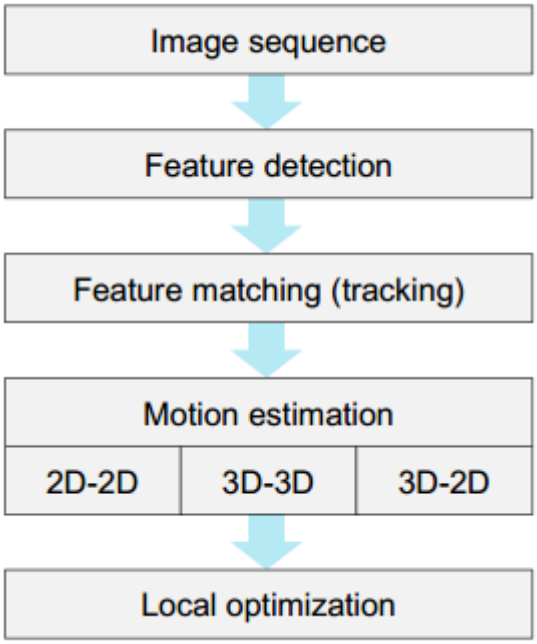

Fig 1.6 A typical Visual Odometry Pipeline

motion takes a toll on the processing speed. Also these techniques seldom take into consideration the regularity in the indoor structures. Hence to address these problems a robust and simple-yet-efficient motion estimation technique is necessary. An attempt to device such a technique keeping in mind the limited computational resources on a robotic platform and the need for near real-time processing rate has been done and discussed in this thesis.

1.5 Problem Statement

In light of all the obstacles plaguing the visual navigation process a need for an accurate and meek motion estimation system is required. Hence the problem at hand is to estimate the camera motion as the robotic platform moves along the indoor environment path. To keep track of the heading of the robot is also equally essential. An incremental pose estimation approach is to be taken to gradually update the location of the robot with respect to its starting location. A vague model of

the environment would help better navigate the robot which must also be formed. A Manhattan world assumption is made to keep the computation simple which might not be such a strong assumption as most indoor environments conform to it. Currently the environment is assumed to be free of obstacle to keep the mathematics simple. Later improvements might address this problem, but for now there are no stationary or non-stationary obstacles in the environment.

1.6 Organization of Thesis

This thesis consists of a total of five chapters organized as below—

- Chapter 1. This chapter gives a brief introduction to the current status of autonomous navigation research and the in-outs of the approach taken up by various researchers. It also highlights the different scenarios in navigation such as indoor and outdoor challenges and how people have gone about to solve the problems associated with each of them.
- Chapter 2. Will give introduction to the geometrical modelling of the 3D structure and the image formation process. Mathematical modelling of the imaging process as well as the pose estimation problem is essential, and is discussed in this chapter in detail.
- Chapter 3. This chapter discusses the various key point features available for the purpose of tracking them over consecutive frames. The extraction of key point features from the image and the technique for tracking them over consecutive frames is also explained. An evaluation of the performance of various key point features in the context of motion estimation is discussed.
- Chapter 4. Various steps in the estimation of the vanishing point from an image is discussed in detail in this chapter. The results for different frames are also present here.

- Chapter 5. The final step of camera motion estimation is a culmination of all the techniques discussed in the previous chapters. This chapter discusses the different models of the environment and describes the steps involved in the camera motion estimation.
- Chapter 6. The results of the trajectory paths obtained by applying the algorithm on different image sequences is presented in this chapter. Evaluation and analysis of these results is also presented in this chapter. Conclusion and future improvements are also discussed within this chapter.

1.7 Conclusion

This chapter gave an introduction to the different approaches taken towards autonomous navigation. Different classes of navigation such as indoor and outdoor or maples and map-based algorithms are also discussed at the beginning of the chapter. The motivation behind the work done and importance of this work has been clearly justified in section 1.4. The challenges in autonomous navigation and how far research work over the years has been able to solve them is explored in section 1.3. The final problem statement based on the current research work and the un-attended problems that still exist, has been presented along with the assumptions made in the context of motion estimation for autonomous navigation.

Chapter 2

VISUAL GEOMETRY MODELS

One of the primary steps in camera motion estimation is modeling the camera. The camera can be modeled either as a perspective camera or an omnidirectional camera. In our case the camera was modeled as a perspective camera. The perspective camera model assumes a pin-hole projection system wherein the image is formed by the intersection of light rays from the objects as they pass right through the center of a pin-hole camera. Rays from an object in the world pass through this hole to form an inverted image on the back face of the box or image plane. Our goal is to build a mathematical model of this process. The Estimated model parameters form the solution of the camera pose estimation problem.

2.1 Camera Model

Pin-hole cameras in reality, consist of an enclosure with a tiny hole in the front called a pin-hole. Rays coming off an object in the real world cast an inverted image on the rear face of the box, also called the image plane. This image formation process needs to be modelled mathematically. Since it is not intuitive to work with an inverted image we consider a *virtual imag*e which is made by placing the image in front of the pin-hole. Though this is not physically possible, it does make the things more mathematically tractable. Figure 2.1 gives a clear illustration of the image formation with the virtual image on the image plane. The optical center of the camera is assumed to be the origin of the pin-hole camera.

In real life, a pinhole camera consists of a closed chamber with a tiny hole (the pin-hole) in the front. Rays from an object in the world pass through this hole to form an inverted image on

the back face of the box or image plane. Our goal is to build a mathematical model of this process. It is slightly inconvenient that the image from the pinhole camera is upside-down. Hence, we instead consider the virtual image that would result from placing the image plane in front of the pin-hole.

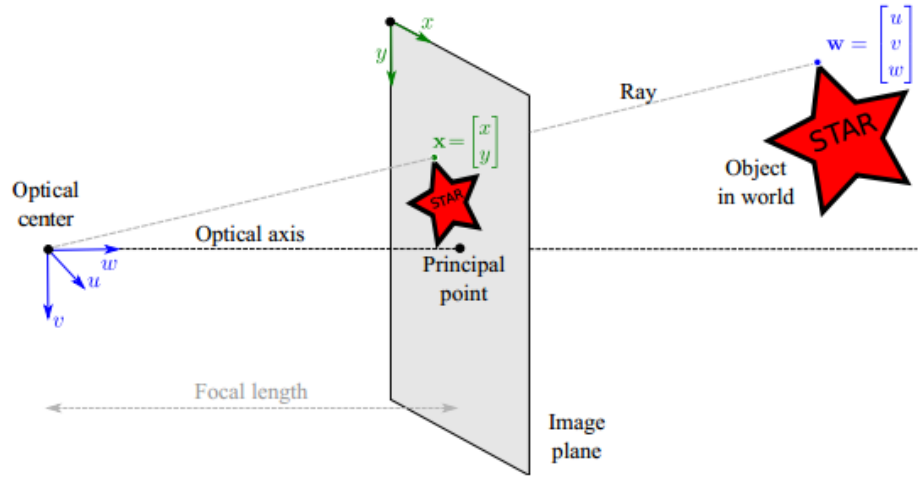

Fig 2.1 Pin-hole camera model terminology. The optical center (pin-hole) is placed at the origin of the 3D world coordinate system(u, v, w), and the image plane (where the virtual image is formed) is displaced along the w-axis. (Image taken from [36])

2.1.1 Intrinsic Parameters

The intrinsic parameters of a camera are specific to that camera and help model mathematically the process of image formation. Figure 2.1 illustrates the pin-hole camera model and also puts forth some relevant terminology. Assuming that the optical center of the camera is the origin of the world coordinate system, we proceed to represent the world coordinate as $X = [u\ v\ w]^T$.

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = KX = \begin{bmatrix} \phi_u & 0 & \delta_x \\ 0 & \phi_v & \delta_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \quad 2.1$$

Here, λ is a depth factor and ϕ_u, $\phi_v$are the focal lengths and δ_x , δ_y the image coordinates of the projection center. Now, in reality, whenever the field of view increases beyond 45^o

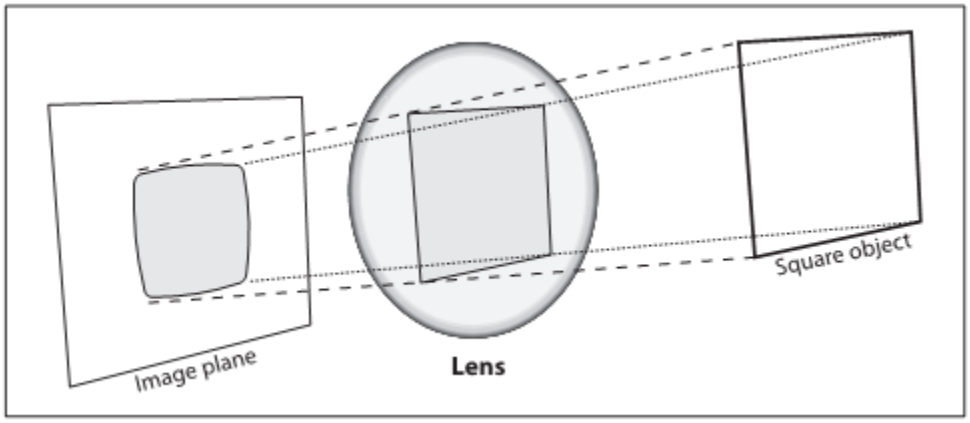

Figure 2.2 Illustration of radial distortion during image formation [learning OpenCV]

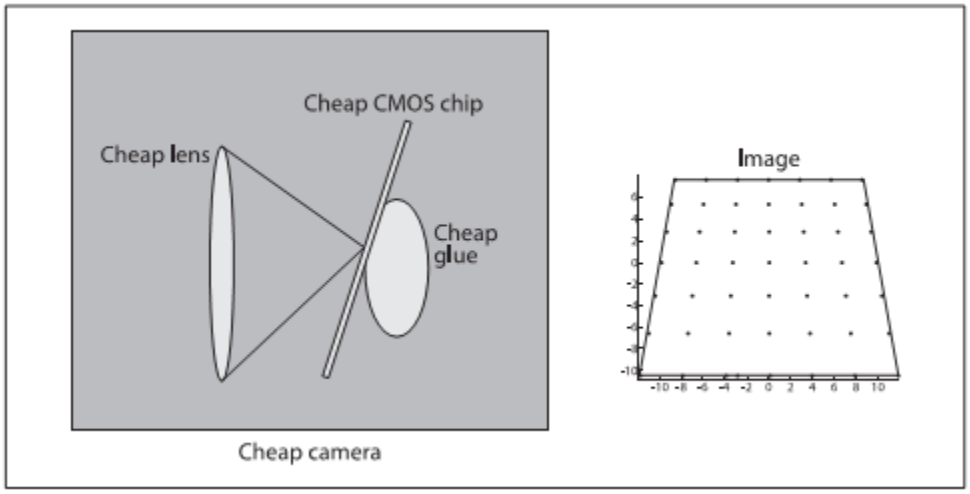

Figure 2.3 Tangential distortion due to improper placement of imaging sensor (image courtesy of Sebastian Thrun)

The effect of radial distortion comes into effect which also needs to be modeled. The effect of radial lens distortion is modelled as a second-(or higher) order polynomial. The effect of radial distortion increases with the increase in the distance from the optical center.

$$\left.\begin{aligned} u_{corrected} &= u(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \\ v_{corrected} &= v(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \end{aligned}\right\} \quad 2.2$$

Similarly, the tangential distortion parameters may also be given in the context of camera model to accurately describe the image formation process as below-

$$\left.\begin{aligned} u_{corrected} &= u + (\ 2p_1 v + p_2(r^2 + 2u^2)\ ) \\ v_{corrected} &= v + (\ 2p_1 v + p_2(r^2 + 2u^2)\ ) \end{aligned}\right\} \quad 2.3$$

Together the three radial distortion and the two tangential distortion parameters form a 5-element distortion vector which can now be used to counter the effect of defects in the camera. Both the calibration matrix K and the distortion vector can now be collectively termed as the camera's intrinsic parameters which form the camera's idiosyncrasy. Estimation of the cameras intrinsic parameters is called camera calibration and is discussed in the next section in detail.

2.1.2 Extrinsic Parameters

We must also account for the fact that the camera is not always centered at the origin of the world coordinate system with the optical axis exactly aligned with the w-axis. A rather general approach would be to define an arbitrary world coordinate system that is mutual to other cameras also. We now express the world points *w* in this coordinate system before being passed to the projection model, using the coordinate transformation below –

$$\begin{bmatrix} u' \\ v' \\ w' \end{bmatrix} = \begin{bmatrix} w_{11} & w_{12} & w_{12} \\ w_{21} & w_{22} & w_{23} \\ w_{22} & w_{32} & w_{33} \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} + \begin{bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} \quad 2.4$$

Or putting it all in the form of a matrix, we have –

$$\boldsymbol{w'} = \Omega\mathbf{w} + \boldsymbol{\tau} \quad 2.5$$

Where $\boldsymbol{w'}$ is the transformed point, Ω is a 3x3 rotation matrix, and $\boldsymbol{\tau}$ is a 3x1 translation vector.

The full pin-hole camera model can now be given by –

$$\begin{bmatrix} u' \\ v' \\ w' \end{bmatrix} = \begin{bmatrix} \phi_u & 0 & \delta_x & 0 \\ 0 & \phi_v & \delta_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} w_{11} & w_{12} & w_{12} & \tau_x \\ w_{21} & w_{22} & w_{23} & \tau_y \\ w_{22} & w_{32} & w_{33} & \tau_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \\ 1 \end{bmatrix} \quad 2.6$$

Or in a matrix form as,

$$\lambda\, \tilde{\boldsymbol{x}} = [K \;\; \mathbf{0}] \begin{bmatrix} \boldsymbol{\Omega} & \boldsymbol{\tau} \\ \mathbf{0}^T & 1 \end{bmatrix} \tilde{\boldsymbol{w}} \quad 2.7$$

Here $\boldsymbol{K}$ is the calibration matrix while Ω, $\boldsymbol{\tau}$ rotarion and translation matrices respectively. $\tilde{\boldsymbol{x}}$ is the imaged point whereas $\tilde{\boldsymbol{w}}$ *w*orld point.

2.2 Camera Calibration

Camera calibration is a process of estimating the cameras intrinsic parameters, which is typically done only once for every camera. The OpenCV calibration routines were used for the purpose of

camera calibration in our case. Calibration using these routines involves imaging a chessboard pattern held at different orientations and estimating the corners of the pattern in each of the orientations. This set of corner location can be used to estimate intrinsic parameters by solving the equations describing the camera's imaging process. The chessboard pattern, being regular, is made up of corners that are part of a regular grid. Each image has N corners, and with K such images we get N x K coordinates that form 2NK constraints that can be solved to obtain the intrinsic parameters. Under the hood, OpenCV uses Zhang's [Zhang00] method for obtaining the focal length and the offset parameters. On the other hand, it uses Brown's method [Brown71] to solve for the radial distortion parameters. Figure 2.4 clearly illustrates the process of camera calibration.

The process of camera calibration involves taking a chessboard pattern and taking pictures of it with the pattern placed at different orientations. From each of the images, the corners of the chessboard pattern are detected and the coordinates recorded for further solving. Chessboard pattern has the black and white blocks each of side 28mm. The pattern is made up of a 6 x 9 chess blocks giving a total of 40 corners in each image. The corners location are refined to sub-pixel accuracy for accurate estimation of the intrinsic parameters. The corners from 10 images were taken, arranged properly and given to the calibration rotations in OpenCV. Each of the image was of 640 x 480 in size and the calibration process resulted in the necessary intrinsic parameters like the camera matrix K and the distortion matrix d. given below is the camera calibration matrix K –

$$\boldsymbol{K} = \begin{bmatrix} \mathbf{8.58 \times 10^2} & \mathbf{0} & \mathbf{3.08 \times 10^2} \\ \mathbf{0} & \mathbf{8.47 \times 10^2} & \mathbf{1.35 \times 10^2} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix} \qquad 2.8$$

The distortion vector obtained is –

$$\boldsymbol{d} = [\mathbf{9.89 \times 10^{-2}} \quad \mathbf{-5.83} \quad \mathbf{-8.32 \times 10^{-3}} \quad \mathbf{-6.703 \times 10^{-3}} \quad \mathbf{2.62}] \qquad 2.9$$

The average re-projection error is obtained at about 2.681

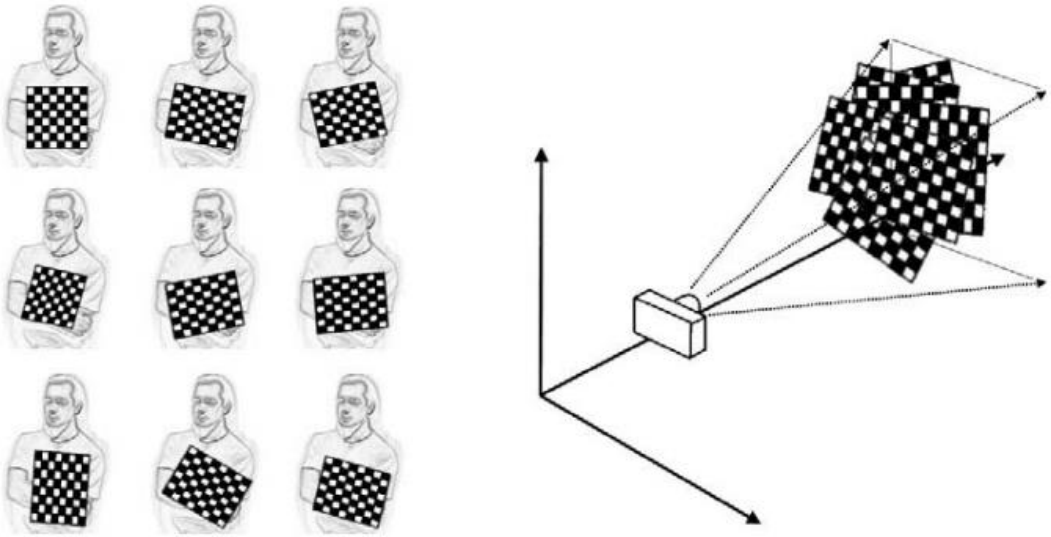

Figure 2.4 Images of a chessboard being held at various orientations (left) provide enough information to completely solve for the locations of those images in global coordinates (relative to the camera) and the camera intrinsics.

2.3 Transformation Models

Estimation of camera pose involves calcuting the extrinsic matrix that relates the world cordinate system to the camera cordinate system. The tranformation that relates the world cordindates with the camera cordinates is of many forms based on the assumptions made about the environment and the degrees of freedom allowed. Mappings between the plane and the image can be described using a family of 2D geometric transformations.

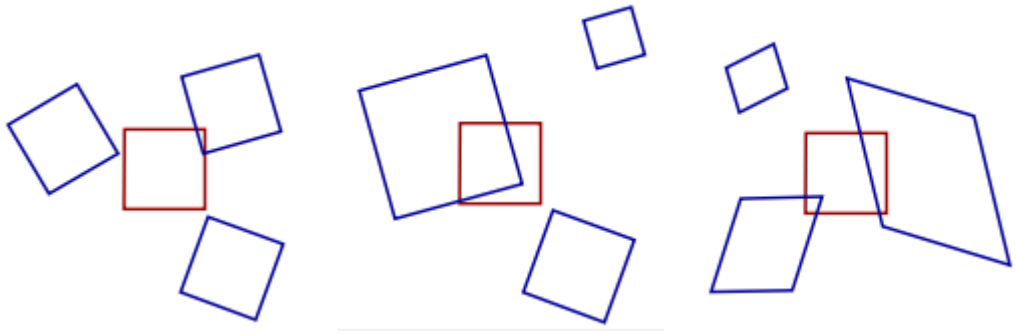

Figure 2.5 a. Eucledian class b.Similarity class c.Affine class of transformations

2.3.1 Euclidean Transformation Model

We assume that a position on the plane can be described by a 3D position w = [u; v; 0]^T, measured in real-world units such as millimeters. The *w*-coordinate, measured in real-world units such as millimeters. Applying the pinhole camera model to this situation gives-

$$\lambda\tilde{\boldsymbol{x}} = \boldsymbol{K}[\boldsymbol{\Omega}, \boldsymbol{\tau}]\tilde{\boldsymbol{w}} \qquad 2.10$$

Where $\tilde{\boldsymbol{x}}$ is the 2D observed image position represented as a homogeneous 3-vector and $\tilde{\boldsymbol{w}}$ is the 3D point in the world represented as a homogeneous 4 vector. This can be explicitly written as,

$$\lambda\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{y}\\ \boldsymbol{1}\end{bmatrix} = \begin{bmatrix}\boldsymbol{\phi_u} & \boldsymbol{0} & \boldsymbol{\delta_x} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{\phi_v} & \boldsymbol{\delta_y} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{1} & \boldsymbol{0}\end{bmatrix}\begin{bmatrix}\boldsymbol{w_{11}} & \boldsymbol{w_{12}} & \boldsymbol{w_{12}} & \boldsymbol{\tau_x}\\ \boldsymbol{w_{21}} & \boldsymbol{w_{22}} & \boldsymbol{w_{23}} & \boldsymbol{\tau_y}\\ \boldsymbol{w_{22}} & \boldsymbol{w_{32}} & \boldsymbol{w_{33}} & \boldsymbol{\tau_z}\\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{D}\end{bmatrix}\begin{bmatrix}\boldsymbol{u}\\ \boldsymbol{v}\\ \boldsymbol{0}\\ \boldsymbol{1}\end{bmatrix}$$

$$= \begin{bmatrix}\boldsymbol{\phi_u} & \boldsymbol{0} & \boldsymbol{\delta_x}\\ \boldsymbol{0} & \boldsymbol{\phi_v} & \boldsymbol{\delta_y}\\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{1}\end{bmatrix}\begin{bmatrix}\boldsymbol{w_{11}} & \boldsymbol{w_{12}} & \boldsymbol{\tau_x}\\ \boldsymbol{w_{21}} & \boldsymbol{w_{22}} & \boldsymbol{\tau_y}\\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{D}\end{bmatrix}\begin{bmatrix}\boldsymbol{u}\\ \boldsymbol{v}\\ \boldsymbol{1}\end{bmatrix} \qquad 2.10$$

Once this is obtained, we often normalize the coordinates by pre-multiplying with the calibration matrix K on both sides. This gives us the normalize coordinates –

$$\rightarrow \begin{bmatrix} \boldsymbol{x'} \\ \boldsymbol{y'} \end{bmatrix} = \begin{bmatrix} \boldsymbol{w_{11}} & \boldsymbol{w_{12}} \\ \boldsymbol{w_{21}} & \boldsymbol{w_{22}} \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{v} \end{bmatrix} + \begin{bmatrix} \boldsymbol{\tau_x} \\ \boldsymbol{\tau_y} \end{bmatrix} \quad 2.11$$

In this case the parameters $w_{11}, w_{12}, w_{21}, w_{22}$ are all constrained to fininte set of values

$$\begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \quad 2.12$$

Hence there are only three parameters- two translation parameters and a rotation θ

2.3.2 Similarity Transformation Model

The similarity transformation is a Euclidean transformation with a scaling and has four parameters: the rotation, the scaling, and two translations.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \rho w_{11} & \rho w_{12} \\ \rho w_{21} & \rho w_{22} \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} + \begin{bmatrix} \tau_x \\ \tau_y \end{bmatrix} \quad 2.13$$

The previous constraint on the parameters $w_{11}, w_{12}, w_{21}, w_{22}$ apply as usual.

2.3.3 AFFINE TRANSFORMATION MODEL

If we wish to describe the relationship between image points and points on a plane in general position the affine tranformation model is helpful. The affine tranformation is given by,

$$\lambda \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \phi_{11} & \phi_{12} & \tau_x \\ \phi_{21} & \phi_{22} & \tau_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \quad 2.14$$

Where $\phi_{11}, \phi_{12}, \phi_{21}, \phi_{22}$ are now all unconstrained, so can take arbitrary values.

$$\text{Or } \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \phi_{11} & \phi_{12} \\ \phi_{21} & \phi_{22} \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} + \begin{bmatrix} \tau_x \\ \tau_y \end{bmatrix} \quad 2.15$$

2.3.4 PROJECTIVE TRANSFORMATION MODEL

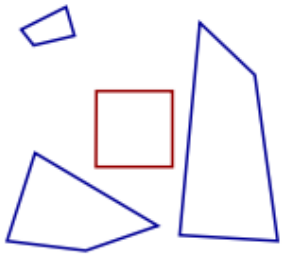

Figure 2.6 Projective class of tranformations

The projective transformation model is used to explain that class of transformations happening when a pinhole camera views a plane from an arbitrary viewpoint. The relationship between a point $w = [u\ v\ 0]^T$ on the plane and the position $x = [x\ y]^T$ to which it is projected- is

$$\left.\begin{aligned} \lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} &= \begin{bmatrix} \phi_u & 0 & \delta_x \\ 0 & \phi_v & \delta_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} w_{11} & w_{12} & w_{12} & \tau_x \\ w_{21} & w_{22} & w_{23} & \tau_y \\ w_{22} & w_{32} & w_{33} & \tau_z \end{bmatrix} \begin{bmatrix} u \\ v \\ 0 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} \phi_u & 0 & \delta_x \\ 0 & \phi_v & \delta_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} w_{11} & w_{12} & \tau_x \\ w_{21} & w_{22} & \tau_y \\ w_{22} & w_{32} & \tau_z \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \end{aligned}\right\} \quad 2.16$$

Combining the two 3 x 3 matrices by multiplying them together, the result is a transformation with the general form-

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \phi_{11} & \phi_{12} & \phi_{13} \\ \phi_{21} & \phi_{22} & \phi_{23} \\ \phi_{22} & \phi_{32} & \phi_{33} \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \quad 2.17$$

Although there are nine entries in the matrix, the homography only contains eight degrees of freedom; the entries are defined upto a predefined scale. It is easy to see that a constant re-scaling of all nine values produces the same transformation.

2.4 Camera Pose Estimation Using Homography

In this section, a 3D pose estimation method based on projection matrix and homographies is explained. The method estimates the position of a world plane relative to the camera projection center for every image sequence using previous frame-to-frame homographies and the projective transformation at first, obtaining for each new image, the camera rotation matrix **R** and a translational vector **t**. The approach followed here is based on the work by Simon et. al. [29, 30].

As explained in the previous section, as general pinhole camera model is considered along with a camera projection matrix that maps a world point $\boldsymbol{w}$ to image point $\boldsymbol{x^i}$. It is given by

$$\lambda \boldsymbol{x^i} = P^i \boldsymbol{w} = K[\Omega^i, \boldsymbol{\tau}^i]\boldsymbol{w} = K\ [\boldsymbol{r_1^i}\ \ \boldsymbol{r_2^i}\ \ \boldsymbol{r_3^i}\ \ \boldsymbol{\tau}^i]\boldsymbol{w} \quad 2.18$$

Where $\boldsymbol{r_1^i}\ \ \boldsymbol{r_2^i}\ \ \boldsymbol{r_3^i}$ are the columns of the rotation matrix Ω and K is the camera calibration matrix. As illustrated previously, the coordinates of the points on a plane have their z-coordinate zero. This results in the projection matrix P^i simplifying into $K\ [\boldsymbol{r_1^i}\ \ \boldsymbol{r_2^i}\ \ \boldsymbol{\tau}^i]$

$$P^i = K\,[\boldsymbol{r}_1^i \;\; \boldsymbol{r}_2^i \;\; \boldsymbol{\tau}^i] = \boldsymbol{H}^i \qquad 2.19$$

The thus deprived projection matrix has reduced number of parameters. It is a 3x3 projection matrix that transforms points on the world plane to the i[th] image plane which is the planar homography defined up to scale. The camera pose is hidden in the H^i matrix and can be 'estimated using the calibration matrix obtained.λ can be calculated using the following equation,

$$\left.\begin{aligned} \boldsymbol{\lambda} &= \frac{\mathbf{1}}{\|\boldsymbol{K}^{-1}\boldsymbol{h}_1\|} \\ [\boldsymbol{r}_1^i \;\; \boldsymbol{r}_2^i \;\; \boldsymbol{\tau}^i] &= \boldsymbol{\lambda}\boldsymbol{K}^{-1}\boldsymbol{H}^{\,i} = \boldsymbol{\lambda}\boldsymbol{K}^{-1}[\boldsymbol{h}_1 \;\; \boldsymbol{h}_2 \;\; \boldsymbol{h}_3] \end{aligned}\right\} \qquad 2.19$$

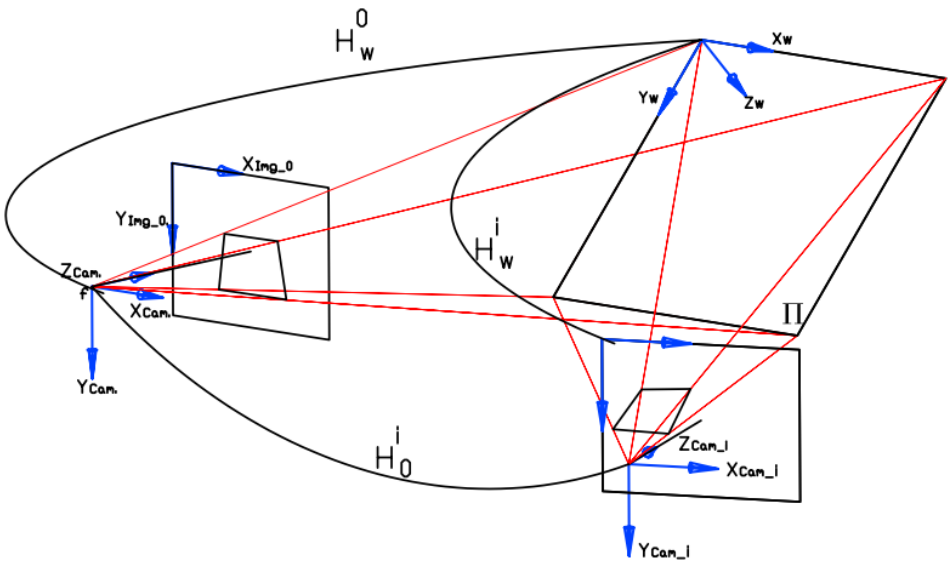

Fig 2.7 Projection model on a moving camera and frame-to-frame homography induced by a plane.

Since the columns of a camera matrix are orthonormal, the third vector of the rotation matrix $\boldsymbol{r_3}$ can be determined by the cross product of $\boldsymbol{r_1}, \boldsymbol{r_2}$. Noisy observation, however, result in rotation matrices that aren't orthonormal. Hence to obtain the best approximation Ω' of rotation matrix Ω

we intend to approximate the matrix Ω to the orthonormal matrix Ω′ in the least square sense [31, 32]. This problem may be confronted by forming the rotation matrix $\Omega^i = [\boldsymbol{r_1^i}\ \boldsymbol{r_2^i}\ \boldsymbol{r_2^i}]$ and performing a Singular Value Decomposition (SVD) to from an optimal matrix Ω'

$$\left.\begin{aligned} \boldsymbol{\Omega^i} = [\boldsymbol{r_1^i}\ \boldsymbol{r_2^i}(\boldsymbol{r_2^i} \times \boldsymbol{r_2^i})] = \mathbf{U}\,\boldsymbol{\Sigma}\,\mathbf{V}^{\boldsymbol{T}} \\ \boldsymbol{\Sigma} = \boldsymbol{diag}(\boldsymbol{\sigma 1}, \boldsymbol{\sigma 2}, \boldsymbol{\sigma 3}) \end{aligned}\right\} \quad 2.20$$

$$\boldsymbol{\Omega'} = \mathbf{U}\,\mathbf{V}^{\mathbf{T}}$$

The final solution is the Ω′ and the **t** which gives the camera's pose relative to the planar object. These would prove to be helpful in further section where in the camera pose it to be estimated repeatedly to draw inference about the motion trajectory.

2.5 Conclusion

This chapter discussed the various 3D geometric models required to interpret the images. The camera calibration process and the image formation model were also discussed. Together the intrinsic and the extrinsic parameters of a camera help understand the complete image formation process. In the final section the camera pose estimation using homography is discussed. This also introduces the decomposition of the homography matrix and how it may be used to estimate the camera pose from point correspondences.

Chapter 3

KEY POINT DETECTION AND TRACKING

Key points are of high importance in the field of vision based navigation due to their highly efficient localization properties. For any given image, making sense of its content from merely the pixel information is not simple. Hence the need for interest point descriptors that accurately describe the image. Such a description, when obtained, may be used to localize the objects in the image, or localize the imaging system with respect to the object. Either way the localization property of the key points or interest point is the corner stone vision based navigation schemes. It is very crucial for the interest point's descriptions of the image to be robust under varying lighting conditions as well as rotation and scale. The important characteristic of the key point is its relative position in the original scene must not change from one image to the other. Different features have their own pros and cons. This chapter explores the different interest point detector analysing the ins and outs of each of them. Also the problem of tracking them accurately over consecutive frames is essential and the related Lucas-Kanade Optical flow technique is discussed in detail.

3.1 Corner and Interest Point Detectors

The approach of corner detection in machine vision is of immense importance as it gives the capability to infer about the image contents from a sparse set of data. The application of the corner detection techniques is often seen in the fields of motion estimation, pose estimation, tracking, structure from motion …etc. The corners are the local maxima in the image often found at the

juncture of two edges. It is also characterised by the two dominant gradient directions in the locality of the point. The Harris corner detection algorithm [15] uses autocorrelation as a measure to classify the point as a corner. Considering a 2D image data I and taking a patch at a location (u, v) and shifting it by an amount of (x, y) we measure the squared differences between these two patches.

$$S(x,y) = \sum_u \sum_v w(u,v)\big(I(x+u, y+v) - I(u,v)\big)^2 \quad 3.1$$

Taking the Taylor series expansion of $I(u+x, v+y)$ we have –

$$I(u+x, v+y) \approx I(u,v) + I_u(u,v)x + I_v(u,v)y \quad 3.2$$

Where I_x, I_y are the partial derivatives of I along the x and y directions. Substituting it back in the above equation we obtain –

$$S(x,y) \approx \sum_u \sum_v w(u,v)(I_u(u,v)x + I_v(u,v)y)^2 \quad 3.3$$

Which may be written in the form of a matrix as given below –

$$S(x,y) \approx (u\ v)\, A \begin{pmatrix} u \\ v \end{pmatrix}, \text{ where } A = \sum_u \sum_v w(u,v) \begin{bmatrix} I_u^2 & I_u I_v \\ I_u I_v & I_v^2 \end{bmatrix} \quad 3.4$$

Based on the above Harris matrix a location is classified as a corner on the basis of their Eigen values λ_1, λ_2 –

1. If the both the eigen values are small then the location (x, y) is not a corner.
2. If $\lambda_1 \approx 0$ and λ_2 has a significantly high value, then it may be categorized as an edge.
3. If both the eigen values λ_1, λ_2 have a significantly high values, then it is a corner.

Figure below best illustrates the eigen values and the classification if regions accordingly. However, the algorithm suggested by Harris uses a different measure to sift the corner points from an image as the calculation of eigen values is computationally intensive. The metric used for the purpose of detection of corner is M_c and is given mathematically as –

$$M_c = \lambda_1\lambda_2 - \kappa(\lambda_1 + \lambda_2)^2 = \det(a) - \kappa\, trace^2(A) \qquad 3.5$$

Here κ is a control parameter for the sensitivity. The above equation is a computationally less expensive metric and may be used in real-time for the purpose of vision based motion estimation or for the purpose of pose estimation.

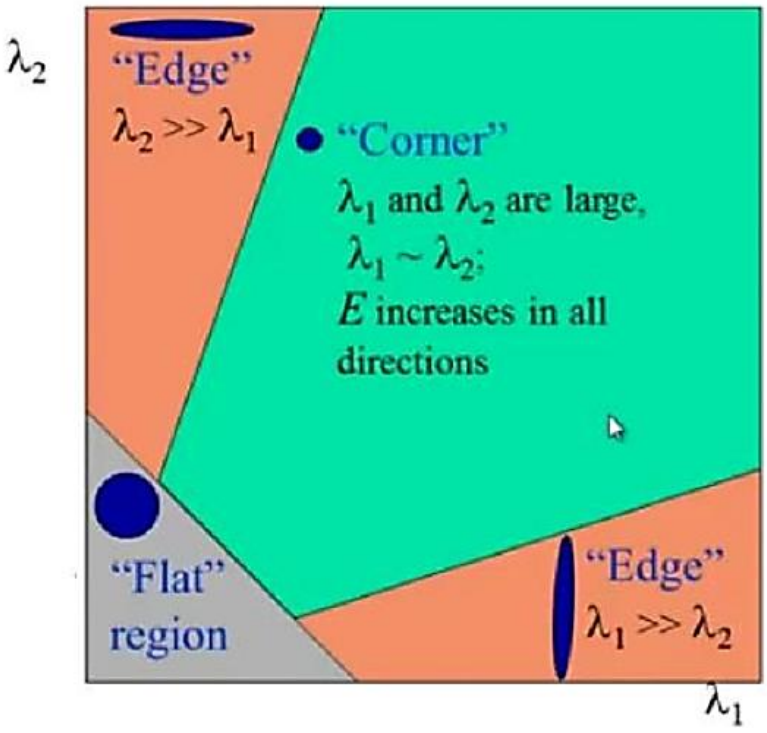

Fig 3.1 Plot of one Eigen value against the other and the region classification according to the Eigen values

3.2. Optical Flow

Optical flow in image is the dispersal of velocity, relative to the observer, over the points of an image. Optical flow carries immense amount of information regarding the structure in the scene being observed as well as the motion within the scene due to foreground objects. It is concretely defined in [14] as

"Image flow is the velocity field in the image plane due to the motion of the observer, the motion of objects in the scene, or apparent motion which is a change in the image intensity between frames that mimics object or observer motion."

Optical flow gives the velocity of every pixel in the image or also a path of a defined patch in the image. Based on the kind of output from the Optical Flow (OF) algorithm that is needed, they are two popular algorithms for measurement of OF. The Horn-Schunck Optical flow estimation algorithm and the Lucas-Kanade Optical Flow algorithm. The Horn-Schunck OF method is a dense estimation method developed in 1981 [15]. This estimation method makes use of the brightness constancy and presents a set of equations that describe mathematically the brightness constancy. These equations were solved for by imposing a smoothness restriction on the velocities v_x and v_y. This in effect tries to minimize the perturbations in the flow, giving more preference to solutions that are smoother. The flow is devised in the form of a global energy function and the solution to which is obtained by minimizing this energy function. For a 2D image signal, the energy function is given by –

$$E = \iint \left[\left(I_x u + I_y v + I_t\right)^2 + \alpha^2(\|\nabla u\|^2 + \|\nabla v\|^2)\right] dx\, dy \qquad 3.6$$

Here $I_x, I_y$and I_t are the partial derivatives of the 2D image along the x, y and t dimensions respectively. $\vec{V}$ = [$u(x,y), v(x,y)$]^T gives the optical flow vector at a location x, y in the image. The optical flow vector obtained using Horn-Schunck is a gradually varying function of x, y. The energy function E mentioned above may be minimized by solving the Euler-Lagrange equations –

$$\left.\begin{aligned} \frac{\partial L}{\partial u} - \frac{\partial}{\partial x}\frac{\partial L}{\partial u_x} - \frac{\partial}{\partial y}\frac{\partial L}{\partial u_y} = 0 \\ \frac{\partial L}{\partial v} - \frac{\partial}{\partial x}\frac{\partial L}{\partial v_x} - \frac{\partial}{\partial y}\frac{\partial L}{\partial v_y} = 0 \end{aligned}\right\} \quad 3.7$$

Here L is the integral energy equation mentioned in equation 3.6. When put in the above expression, it turns into the following form –

$$I_x\left(I_x u + I_y v + I_t\right) - \alpha^2 \Delta u = 0 \quad 3.8$$

$$I_y\left(I_x u + I_y v + I_t\right) - \alpha^2 \Delta v = 0 \text{ , } \Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \quad 3.9$$

The above system of equation may be evaluated into a simpler form as given below –

$$(I_x^2 + \alpha^2)u + I_x I_y v = \alpha^2 \bar{u} - I_x I_t \quad 3.10$$

$$I_x I_y u + (I_x^2 + \alpha^2)v = \alpha^2 \bar{v} - I_x I_t \quad 3.11$$

This is a linear equation in two variables u and v and may now be solved for each of the pixels that make up the image. This in reality may be formulated as an iterative process to account for the updating of pixels values as new frames arrive. The iterative formula for estimating the pixel wise motion of the pixels in a video sequence is given in equation 3.10.

$$u^{k+1} = \bar{u}^k - \frac{I_x\left(I_x\bar{u}^k + I_y\bar{v}^k + I_t\right)}{\alpha^2 + I_x^2 + I_y^2}$$

$$v^{k+1} = \bar{v}^k - \frac{I_y\left(I_x\bar{u}^k + I_y\bar{v}^k + I_t\right)}{\alpha^2 + I_x^2 + I_y^2} \qquad 3.12$$

The Horn-Schunck method form motion estimation if computationally intensive. It is very sensitive to local tiny local variations in the video sequence. It nevertheless gives a very dense flow vectors which can be properly exploited.

However, there is not always a need for such dense motion estimation. The information regarding the motion of every pixel of the image in a video sequence is not necessary. Motion of objects in a video sequence may very well be characterised by inferring the movement of a finite set of point within the objects contour. Such a reduced set of information by tracking a sparse set of points is often enough to characterize the motion of points. Lucas-Kanade Optical flow algorithm address the problem of sparse tracking.

3.2.1 Lucas-Kanade Tracker

As discussed previously, the Lucas-Kanade tracker or the KLT Tracker is a sparse tracking algorithm capable of tracking point features over multiple frames. This often used in the context of pose estimation, structure from motion, visual odometry, 3D reconstruction and for other related applications. It usually works well in the cases where the motion is relatively small. However, the feature tracking problem isn't as straight forward as it may seem. The different challenges faced in the feature tracking problem includes –

- Figuring out the best features that can be accurately tracked.

- Tracking these features efficiently across multiple frames.
- Dealing with changes in the feature points due to illumination, rotation and scale variations.
- Handling error accumulation due to small errors at each step.
- Ability to handle occlusion of points during one frame or a couple of frames.

The problem statement of KLT tracker is now framed as – "Given two subsequent frames and the feature points in one frame, estimate the corresponding points' translated location". Some of the key assumptions made in the KLT tracker algorithm include 'Brightness Constancy' which imposes the condition that the point looks almost the same in every frame. In addition to this there is assumed to be minimal motion of the feature points over the consecutive frames. Also, spatial coherence is also assumed implying that points move very much like their neighbours. The brightness constancy constraint can be put forth mathematically as given below –

$$I(x, y, t) = I(x + u, y + v, t + 1) \tag{3.13}$$

Now taking the Taylor series expansion of the above equation at the point (x, y, t) we have—

$$I(x + u, y + v, t + 1) \approx I(x, y, t) + I_x u + I_y v + I_t$$

$$I(x + u, y + v, t + 1) - I(x, y, t) = I_x u + I_y v + I_t$$

Hence $$I_x u + I_y v + I_t \approx 0 \rightarrow \nabla I[u\ v]^T + I_t = 0 \tag{3.14}$$

We plan on recovering the image motion (u, v) at each of the locations of the feature points. One equation is known and there are two variable to be evaluated for. Hence in order to get more equations that describe a pixels motion we need to use more locations. The spatial coherence constraint comes in handy in this situation. If we consider 5×5 neighbourhood around the feature location, it gives us with 25 such equation per location. Let i be the index of the pixel in the neighbourhood with i varying from 1 to 25 and p_i the location of the i^{th} feature point. Then the set of equations for each of the location may be put in the matrix form as given below –

$$\begin{bmatrix} I_X(p_1) & I_y(p_1) \\ I_y(p_1) & I_y(p_1) \\ \vdots & \vdots \\ I_y(p_{25}) & I_y(p_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(p_1) \\ I_t(p_2) \\ \vdots \\ I_t(p_{25}) \end{bmatrix}$$

$$\rightarrow A_{2\times2}\, \boldsymbol{d_{2\times1}} = \boldsymbol{b_{25\times1}} \qquad 3.14$$

This system of linear is over determined and not an exact solution. However, a least square solution is possible using the below mathematical manipulation –

$$(A^T A)\, \boldsymbol{b} = A^{\boldsymbol{T}} \boldsymbol{b} \qquad 3.15$$

$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix} \qquad 3.16$$

The above summations are over the $N \times N$ neighbourhood around the feature being tracked. This linear equation if solvable if $A^T A$ is invertible and there are other set of conditions.

- $A^T A$ *must be non singular*
- $A^T A$ *the eigen values of this matrix mustn't be too small.*

- $A^T A$ *should not be ill conditioned i.e* $\frac{\lambda_1}{\lambda_2}$ *shouldn'tbe too large.*

These above conditions are very similar to the ones mentioned by the Harris corner detector. The Harris corner detector featured above is used as the feature point locations to initialize the tracking algorithm. The assumption that - the movements in the video sequence are not very large - is too restrictive as in reality the movement of pixels may happen at a faster rate. To address this problem we take an iterative approach of refinement of trajectories at different scales.

We first initialize the points to be tracked and iteratively find the location of these point features at various scales to refine the track estimate. We shift the window to the estimated location (u, v) and reiterate to estimate the new location using the steps previously mentioned. We recalculate I_t and repeat the steps until the change is not significant. To handle the motion of corner features that is large, we can employ coarse to fine optical flow estimation. Corse to fine optimal estimation is done by creating an image pyramid. The number of pyramid levels to be used defines the level of accuracy of the tracks.

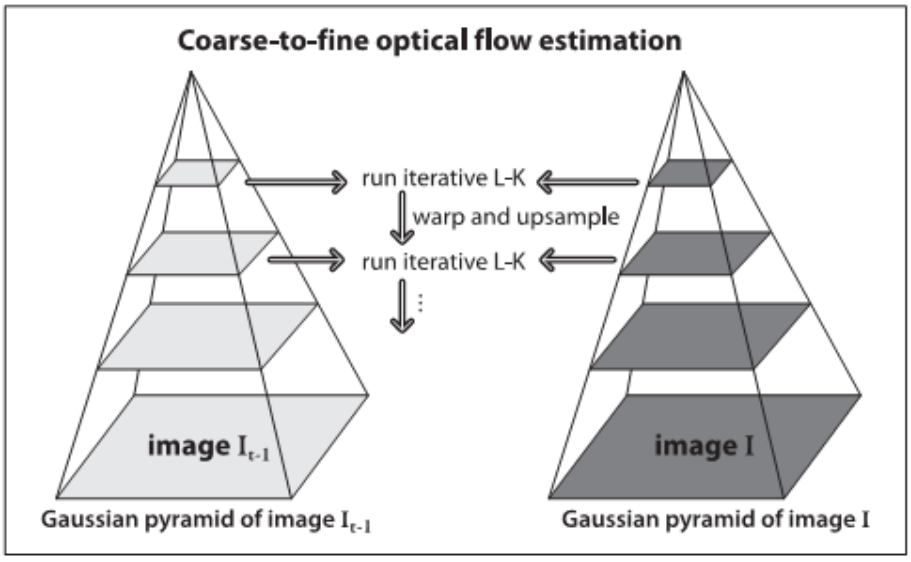

Fig 3.2 Image Pyramids of current and previous frames constructed for coarse to fine refinements of the feature tracks.

3.3 Conclusion

This chapter discussed the importance of corner features in the setting of pose estimation for the purpose of autonomous navigation. The Harris corner features works well in our case and have been successfully used for the purpose of tracking. The definition of the concept of Optical Flow was put forth and the different optical flow estimation algorithms were discussed. The dense optical flow Horn-Schunck algorithm was discussed in detail. However the use of this algorithm is not suggested for the purpose of feature tracking. Instead the sparse point tracking algorithm by Lucas, Kanade and Tomasi, the KLT Tracker was often used in several of the research works in the field of visual odometry and pose estimation. This algorithm was discussed in detail and has been used together with the Harris corner features. The Harris corner features were first detected in a frame in the image and the KLT tracker algorithm was initialized with their locations. The algorithm then gave the possible location of that corner point in the next frame.

Chapter 4

VANISHING POINT ESTIMATION

4.1 Vanishing Point

"In graphical perspective, a vanishing point is a point in the picture plane π that is the intersection of the projections of a set of parallel lines in space on to the picture plane"

Vanishing points provide important information about the structure in the environment. The location of the vanishing point is key to the robots understanding of its pose relative to the environment. This sort of information is very useful in estimating the robot pose relative to the environments such as a corridor. Estimation of vanishing point is done via the corridor which are formed by the intersection of the wall and floor surfaces. The formation of vanishing points is based on the well-known principle where in the lines parallel to each other in 3D space appear to converge at a terminal point.

This happens in the case of a pin-hole projection model. Vanishing point detection has been an area of active interest from the research community in recent years. The typical process

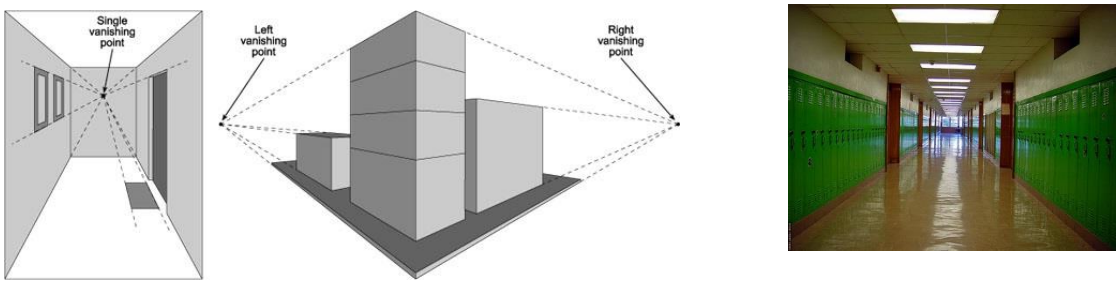

Fig 4.1 a) Illustration of the Vanishing Points in Images b) Indoor image with visible Vanishing Point

for vanishing point detection involves a pre-processing step to enhance the corridor lines. Then a standard edge detection algorithm like the 'Canny Edge detection' may be used for this purpose. It is then followed by Hough transform for the detection of straight lines amongst the edge image obtained through edge detection.

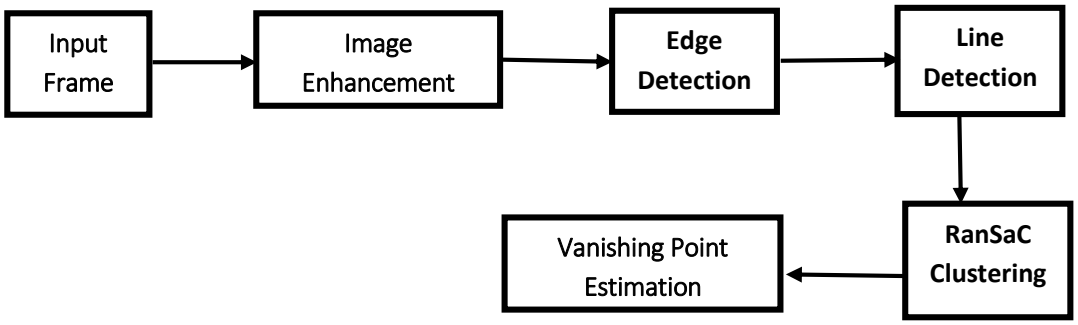

Fig 4.2. Vanishing Point Estimation Block Diagram

4.2 Corridor Line Detection

The corridor lines contain crucial information about the environment's structure making it imperative for the algorithm to make the best use of such rich set of visual cues. So an attempt to estimate the vanishing point typically involves an initial set of steps to detect the corridor lines as accurately as possible. This includes enhancing the high frequency components-the edges-in the image. Which is followed by edge detection and line detection steps. In the pre-processing step, histogram equalization works well to improve upon images of ill lit corridors that were available in the dataset being used. It also performed an enhancement of the edges thereby solving two issues at a go.

4.2.1 Canny Edge Detection

The canny edge detection algorithm has been effectively been used over the years in many computer vision application. The effectiveness of the algorithm lies in its ability to eliminate the non-edge pixels at multiple levels and then finding the weak and strong edges, which are in-turn connected accordingly to extract the truly edge like pixels. The approach involving hysteresis based thresholding helps in finding the true wedges with high level of accuracy. The canny edge detection algorithm is as follows –

Objective

Take an input grayscale image and the threshold parameters and find the edge detected image.

Algorithm

1. Blur the image to remove Noise
2. Finding the gradient images using Sobel operator
3. Non-maxima suppression
4. Hysteresis thresholding
5. Edge tracking of the Hysteresis thresholded image.

Algorithm 4.1 Algorithm for canny edge detection

The thresholding parameters are to be chosen appropriately to meet the desired requirements

-1	0	+1
-2	0	+2
-1	0	+1

Gx

+1	+2	+1
0	0	0
-1	-2	-1

Gy

Fig 4.3. Sobel Operator Kernels

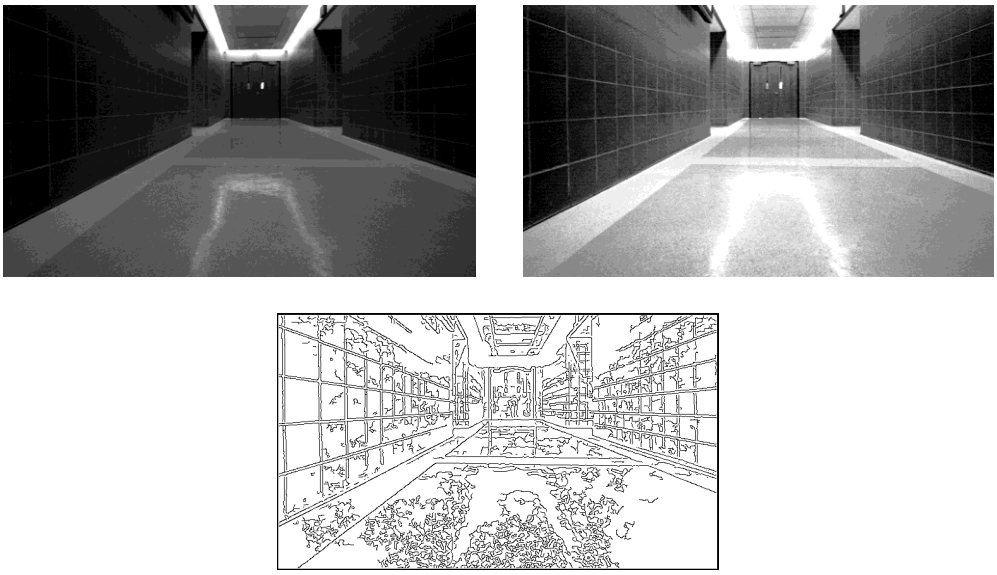

Fig 4.4 Edge Detection: a) Original image of ill-lit corridor b) Pre-processed image c) Edge detection using *Canny* algorithm

The pre-processing step was required in our case as the dataset images were of poorly lit corridors. So, to mitigate the effects of improper illumination, histogram equalization was used. Another approach to pre-processing of ill lit images is via normalizing the image colour which involves dividing the pixel values of the sum of R, G, B, values. To brighten up the images, converting the images into HSV colour space, multiplying V component by 2, and then convert it back to RGB will provide a lot more image contrast thus helpful during the features extraction stage. In our case, histogram equalization worked well enough and was hence used. The thus obtained canny edge detection output has been used to the Hough transform based line detection algorithm.

4.2.2 Line Detection Using Hough Transform

Hough transform is a parametric transform transforming the edge pixel coordinates into from Cartesian space (x, y) into polar space (r, θ). The polar feature space helps in detecting the

collection of pixels that together form straight lines. The output from the canny edge detection algorithm is fed into the Hough transform module which sifts straight lines from the edge detected image. The Hough transform approach takes the Hough space voting to classify the pixels as belonging to the straight line of not. The Standard Hough Transform (SHT) maps each of these pixels to point in the Hough space to try and estimate the line segment. Each edge pixel represents a sinusoid in the Hough space which gives all the family of lines that may pass through this point. Now if a collection of points all belong to a single line, then in all the sinusoids they represent intersect each other. Hence from the number of intersections and the location of such intersections we are able to find the parameters of the straight line. However this method only gives the parameters of the straight line but is not able to localize its position in the image by providing their end points. TO address this problem a variant of probabilistic Hough transform called the Progressive Probabilistic Hough Transform [33] was used for this purpose as it required less amount of computation and was more flexible in the selection of the right lines in the image.

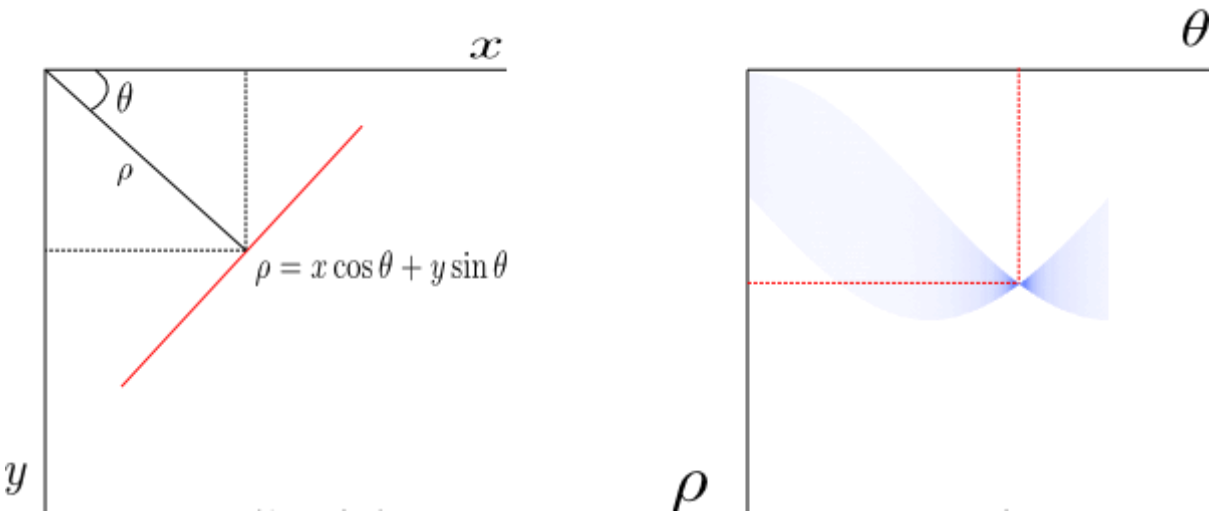

Fig 4.5 left- the Image Cartesian space and right- the Hough space in ρ and θ

Objective

Given the edge image, to find the straight lines in that image subject to some constraints of minimum length.

Algorithm

1. Check the input image, if it is empty then finish.
2. Update the accumulator with a single pixel randomly selected from the input image.
3. Remove pixel from input image.
4. Check if the highest peak in the accumulator that was modified by the new pixel is higher than threshold l. If not then Goto 1.
5. Look along a corridor specified by the peak in the accumulator, and find the longest segment of pixels either continuous or exhibiting a gap not exceeding a given threshold.
6. Remove the pixels in the segment from input image.
7. Unvote from the accumulator all the pixels from the line that have previously voted.
8. If the line segment is longer than the minimum length add it into the output list.
9. Goto 1.

Algorithm 4.2 Algorithm of the Progressive Probabilistic Hough Transform

The above algorithm directly gives the straight lines in an image by their end points. This helps us better process the image and also control the parameters such as the minimum length of the line segment and also the maximum allowable distance between two line segments. The thus obtained line segments are processed to estimate the vanishing point from the image.

4.3 RanSaC Based Vanishing Point Detection

Vanishing Point (VP) are were important features with a lot of information about the structure of environment. Detection of vanishing points in the image is a crucial step in the camera motion estimation process. To detect the vanishing point in the image several approaches were considered. The edge detected image has been processed for straight lines and the approach has been discussed in the previous section. The detected lines are available through their end-points and can be used

to represent or process accordingly. [Vanishing point detection in corridors using Hough Transform] discussed the use of K-Means based clustering approach to VP detection. Most of the algorithms available, though based on the common approach of Hough transform, differ in the parameter space being used. Most common accumulator parameter spaces are the Gaussian Sphere and the Hough Spaces. [Vanishing point detection in corridors using Hough Transform] Discussed the use of Gaussian sphere based approach to VP detection introduced by Bernard et al which has the drawback of computational complexity due to the high quantity of lines being processed. The K-Means based approach to VP detection use the Standard Hough Transform to detect the straight lines in the image. It uses the starting and ending pixel locations of a line as the feature space in which clustering is performed. The K-Means clustering is done twice and at the end of the second step we obtain the VP of the image.

However, the approach doesn't account for the presence of dominant outlying lines in the image which could potentially mess with the estimated location of the Vanishing Point. To address this problem, another approach is taken as suggested in [34] which uses RanSaC based approach to estimate the VP of the scene very effectively in the presence of outliers. This approach hypothesizes a VP coordinate by taking two lines randomly form the Hough Line Detection set. The Hough line detection algorithm described above gives us a set of line that have been detected in the image in the form of their end-point co-ordinates. Let us represent the set $\mathcal{H}$ as the set of all lines detected through the Hough line detection defined as

$$\left.\begin{aligned} \mathcal{H} &= \{l |\, l = (x_1, y_1, x_2, y_2)\ are\ the\ end\ points\ of\ a\ straight\ line\} \\ \mathcal{H}_{hor} &= \{l |\, l \in \mathcal{H} \wedge slope(l) \approx 0\} \\ and\ \mathcal{H}_{vert} &= \{l |\, l \in \mathcal{H} \wedge slope(l) \approx \infty\} \end{aligned}\right\} \quad 4.2$$

The horizontal and vertical lines are removed from H as they contribute little to the vanishing point detection and more as the outliers. Hence a new set $\mathcal{H}$ is defined by the one containing lines that are neither horizontal nor vertical. From these set of lines two lines are randomly sampled and their intersection point is calculated. This forms the initial hypothesis for the VP to be found. The VP in now checked against other lines to see to what degree it conforms to the hypothesis. A distance measure is required to quantify the degree of agreement of the remaining lines with the hypothesis. The distance measure used here is the one mentioned in [34]. It is defined as the angle between the line joining hypothesized VP and the centre of each of the remaining set of lines under consideration and that line itself. The distance $d(v, l_i)$ is described below –

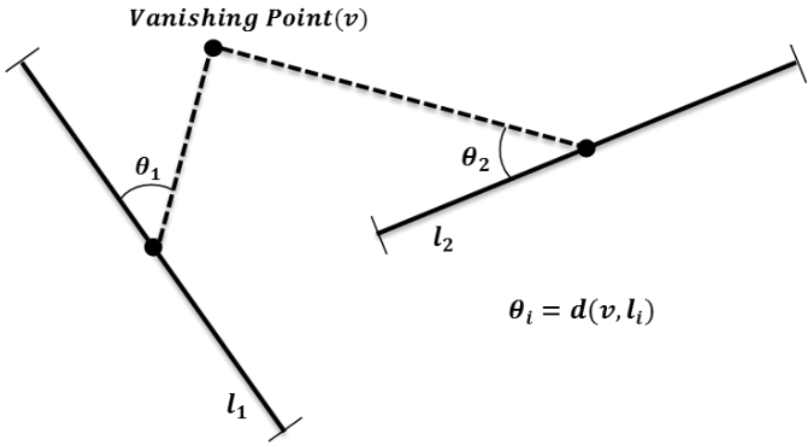

Fig 4.6. Distance metric between vanishing point and line segment

Now a random selection of two line segments l_i and l_j is done and the solution to these two lines will give their point of intersection v which becomes the initial hypothesis for VP. This is checked against the set remaining lines obtained by removing the ones used for formulating the hypothesis. An angle threshold θ_{th} is taken to consider a line as an inlying VP supporting line or an outlier. The inlier count is incremented for each hypothesis obtained by selection of two lines at random and finding their point of intersection. This is repeated for definite number of times and at each

step the hypothesis with the highest inliers so far is stored along with the corresponding inliers. At the end of all the iterations the hypothesis with the most inliers bubbles to the top along with its inliers. All the inlier lines are taken and the resulting VP is found via refinement using the SVD based approach. All the inliers when put together in the form of a linear equation can be solved to get the vanishing point.

$$[\boldsymbol{l_1} \dots \boldsymbol{l_n}]^{\boldsymbol{T}}\boldsymbol{v} = \boldsymbol{b} \tag{4.2}$$

$$\rightarrow \boldsymbol{L^T v} = \boldsymbol{b} \tag{4.3}$$

Where each $\boldsymbol{l_i}$ corresponds to the line that has been detected in the corridor image given in terms of $\boldsymbol{l_i} = [a\ b\ 1]^T$. Equation 4.3 may not have a unique solution. However, it may be solved for in the least square sense using the SVD based approach. Such an approach is required owing to the noisy observation of the vanishing lines. This results in the lines not having a unique point of intersection. The matrix L depicted in equation 4.3 is singular because the equation doesn't have a solution. This may be attributed to the noisy observations of the vanishing lines. Hence, the solution to this may only be obtained in the least square sense. The least square solution may be obtained using the SVD decomposition of the matrix L to find the pseudo inverse of the matrix. The pseudo inverse of the matrix may be obtained from its SVD decomposition as given below. The singular value decomposition of is given by –

$$L^T = U\ \Sigma\ V^T \tag{4.4}$$

$$pseudo\ inverse \quad (L^T)^+ = V\ \Sigma^+\ U^T \tag{4.5}$$

This gives the solution to the vanishing point which is the intersection of the lines in the least square sense. The thus obtained vanishing point is used in the process of segmentation of the corridor image which would be discussed in later sections.

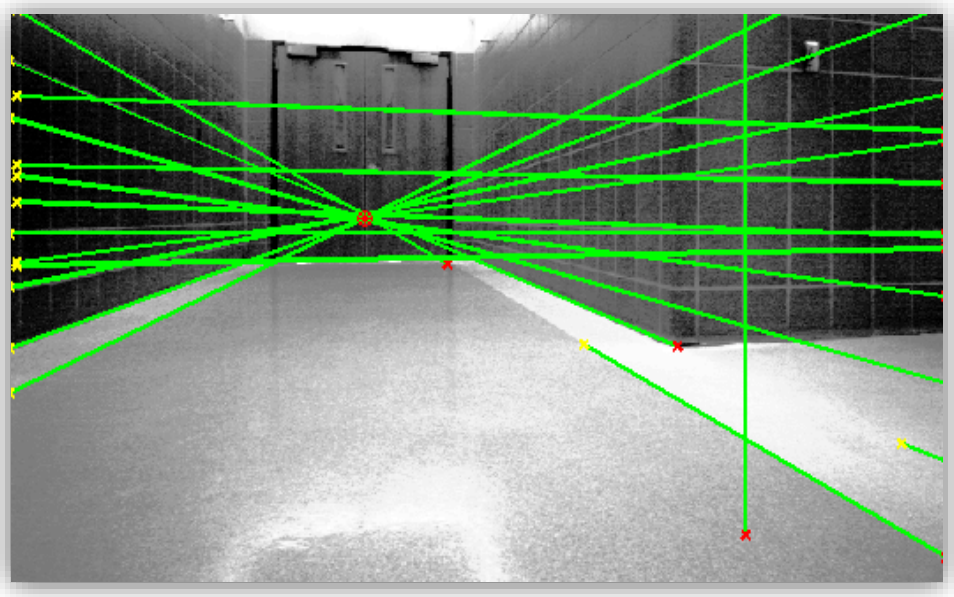

Fig 4.7 Vanishing Point detected with the inlier line segments

4.4 Conclusion

The vanishing point locations prove to be rich source of information, especially in the context of indoor navigation. Many approaches were considered for the purpose of VP estimation. This included the use of Gaussian Sphere based approach and also K-Means based clustering approach [35] to VP estimation. However, these approaches don't work well in the presence of outliers. Hence the approach suggested in [34] was used to make the best estimation of the Vanishing Point in the image. Since the approach involved RanSaC, which works well in the presence of outliers, it worked well, optimally extracting the Vanishing Point in the corridor image.

Chapter 5

CAMERA MOTION ESTIMATION

The final step is estimating the camera motion from the information obtained by tracking the key-points over several frames and fitting the feature correspondence locations into the appropriate environment models. As mentioned prior, Manhattan world assumption is taken the context of indoor navigation—

"*A **Manhattan world scene** is a term used in computer vision that* describes *a real world scene based on the Cartesian coordinate system. The scene is defined by four types of lines: random lines or lines parallel with one of the X, Y or Z axes*."

Under this assumption the world is defined by rectangular planar structures, forming a grid like arrangement. An environment modelled as a Manhattan world is easier to handle during the motion estimation process. The motion estimation is now done by finding the key points in the image frames grabbed by the camera mounted atop the robot and analysing their locations and comparing those to the model of the environment. The next section discusses finding feature correspondences and estimating the homography matrix relating these two key-point locations.

5.1 Homography Estimation

The homography matrix relates the coordinate location of key point on one plane to those on another plane. This homography matrix is evaluated using the Direct Linear Transform based approach as suggested in [11]. The homography matrix estimation needs only four point-

correspondences, $x_i \leftrightarrow x_i'$. The homography matrix relates the 2D coordinates in one plane to those in the other and is given by the following equation—

$$\mathbf{x_i'} = \mathrm{H}\,\mathbf{x_i} \tag{5.1}$$

Where $\mathbf{x_i}$ and $\mathbf{x_i'}$ are the points in plane P^i and P^{i-1} respectively and the homography matrix H relates the 2D locations given in homogenous coordinates. The homogenous coordinates are given by $x_i' = (x_i', y_i', z_i')$. The above expression in x_i', H and x_i may be written in a different form as a cross product given by—

$$\mathbf{x_i'} \times \mathrm{H}\,\mathbf{x_i} = \mathbf{0} \tag{5.2}$$

$$\mathrm{H}\mathbf{x_i} = \begin{pmatrix} \mathbf{h}^{1\mathrm{T}}\mathbf{x_i} \\ \mathbf{h}^{2\mathrm{T}}\mathbf{x_i} \\ \mathbf{h}^{3\mathrm{T}}\mathbf{x_i} \end{pmatrix} \tag{5.3}$$

Where h^{kT} is the k^{th} row of the H matrix. The cross product may now be given as,

$$\mathbf{x_i'} \times \mathrm{H}\mathbf{x_i} = \begin{pmatrix} y_i'\mathbf{h^{3T}x_i} - w_i'\mathbf{h^{2T}x_i} \\ w_i'\mathbf{h^{1T}x_i} - x_i'\mathbf{h^{3T}x_i} \\ x_i'\mathbf{h^{2T}x_i} - y_i'\mathbf{h^{1T}x_i} \end{pmatrix} \tag{5.4}$$

The above equation may be rewritten to resemble a linear equation in 9 variables of the homography matrix as given below—

$$\begin{bmatrix} \mathbf{0}^T & -w_i'\mathbf{x_i^T} & y_i'\mathbf{x_i^T} \\ w_i'\mathbf{x_i^T} & \mathbf{0}^T & -x_i'\,\mathbf{x_i^T} \\ -y_i'\mathbf{x_i^T} & x_i'\mathbf{x_i^T} & \mathbf{0}^T \end{bmatrix} \begin{pmatrix} \mathbf{h^1} \\ \mathbf{h^2} \\ \mathbf{h^3} \end{pmatrix} = \mathbf{0} \tag{5.5}$$

As said above, it may be written in the form of a linear equation in 9 variables, all of which are the elements of the homography matrix H.

$$A_i\mathbf{h} = \mathbf{0}, \textit{here } \mathbf{h} = \begin{bmatrix} \boldsymbol{h^1} \\ \boldsymbol{h^2} \\ \boldsymbol{h^3} \end{bmatrix} \text{ a } 9 \times 1 \textit{ vector and } A_i \textit{ is a } 3 \times 9 \textit{ matrix} \qquad 5.6$$

$$\text{And H } = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \qquad 5.7$$

We thus obtain a linear equation in 9 variable all of which are elements of the homography matrix. For defining the homography matrix up to scale one may choose to set the w_i' parameter to '1'. Each point correspondence in the image results in a pair of two equations in two variables that belong to the homography matrix. So, for a set of four point correspondences obtained via tracking the key points over multiple frames we obtain 8 such equations. The linear equation in elements of h i.e. A$\mathbf{h} = \mathbf{0}$ is to be solved to obtain non-trivial solutions of h as the trivial solution of h is of no relevance to us. Here A is obtained from the rows of A_i for each point correspondence $x_i \leftrightarrow x_i'$ obtained through the tracking of the points over consecutive frames. The system of linear equations thus obtained has a trivial solution but is now ignored as it is irrelevant to our use case. The non-trivial solution is obtained and is defined only up to scale. The scale may defined in such a way that the norm $\| \text{ h } \| = 1$.

Often the four point correspondences obtained are a set of noisy observation often corrupted by many factors such as image space quantization, error in the tracking process, lens distortions…etc. These noisy observations might hamper one from making a refined estimate of the homography matrix H. Nevertheless, one may attempt to make a best approximation of the H matrix by

increasing the number of observation which in our case are the point correspondences. Since each point correspondence give a pair of equations, N such point correspondences would give us 2N such equations and can be arrange into system of linear equation and can be solved accordingly. Since such a system is over determined, the approach that is to be taken is slightly different from

the typical steps used to solve a system of linear equations. In the case of an over determined system, there is a very little chance of all the equations agreeing upon a single solution. Hence, we instead attempt to estimate an approximate solution h which is optimal in terms of a cost function. To avoid the trivial solution, a condition on the h vector is levied requiring it be of unit norm. The

Objective

Given $n \geq 4$ 2D point correspondences $\{x_i \leftrightarrow x_i'\}$, to determine the homography matrix H that relates the point correspondences as $x_i = H x_i'$.

Algorithm

1. Every point correspondence $x_i \leftrightarrow x_i'$ gives two equations which are to be obtained from equation 5.5.
2. Each of the pair of equations from every pint correspondence must be put together to form a $2n \times 9$ matrix A forming a linear equation $A\mathbf{h} = \mathbf{0}$.
3. The solution for the over determined system may be obtained through the Singular Value Decomposition based approach. The matrix A is decomposed as $A = UDV^T$ where D is a diagonal matrix of singular values. The solution to **h** is obtained by arranging the singular values in descending order and taking the column of V that corresponds to that lowest singular value.
4. The thus obtained approximate solution for **h** must be rearranged into the matrix H according to equation 5.7.

Algorithm 5.1. Direct Linear Transformation for the estimation of homography matrix H

solution to this over determined system is obtained using the Singular Value Decomposition (SVD) based approach. This algorithm has been termed in the literature as Direct Linear Transformation (DLT). It is clearly illustrated in Algorithm 5.1.

5.1.1 RanSaC based Homography estimation

Often the tracks estimated through the optical flow based tracking contain spurious tracks that don't conform to the plane to plane homography during the robot motion. Due to this there are lot of outliers in the tracks that often corrupt the estimate of the homography matrix. The estimate of

Objective

Given $n \geq 4$ 2D point correspondences $\{x_i \leftrightarrow x_i'\}$ inclusive of outliers, to determine the inlier point correspondences that help estimate the homography matrix H relating the point correspondences as $x_i = H\, x_i'$.

Algorithm

1. Select four point correspondences at random from that set of n point correspondences $x_i \leftrightarrow x_i'$.
2. Computer the homography matrix H using these 4 point correspondences using the DLT algorithm mentioned above.
3. Compute the metric values to see if the equation $\| \mathbf{x_i'} - H\mathbf{x_i} \| < \varepsilon$ is satisfied, for each of the remaining point correspondences. The ones that satisfy are the inliers.
4. Repeat the above steps for a finite number of times and keep the largest set of inliers at the end.
5. Refine the homography matrix estimated to obtain a least-squares solution from the inliers.

Algorithm 5.2. RanSaC based estimation of Homography

outliers and the removal is often seen as a model fitting problem. This problem can be solved efficiently through the use of Random Sample Consensus algorithm (RanSaC). The RanSaC algorithm is capable of sifting inliers from a data corrupted by outliers. This done by randomly selecting a subset of data pints, just enough to form a model hypothesis. The remaining data points are checked to see to how well they fit into this model. This process is repeated with a new set of random data points selected to form a new hypothesis. The inlier subset with the highest cardinality is saved as the inlier set after all the iterations. The RanSaC algorithm may now be used in the context of Homography matrix estimation and is given in Algorithm 5.2.

5.2 Decomposition of Homography matrix

The homography matrix obtained above relates the coordinates on one plane to those on the other. This relation is obtained is given by the below equation 5.1. The homography matrix may be decomposed into its constituent elements as given in [12] as –

$$\mathrm{H} = \mathrm{R} + \frac{1}{\mathrm{d}}\mathbf{t}^{\mathrm{T}}\mathbf{n} \tag{5.8}$$

Where the rotation matrix R gives the rotation of the plane with respect to the origin, the vector **t** gives the translation between the two planes related by H. The normal to the plane is given by the vector **n**. Element d is the distance of the plane from the centre of coordinate system it has been set to one for mathematical tractability. Rotation matrix R is a $\mathbb{R}^{3\times3}$ matrix with $\mathrm{R} \subset \mathrm{SO}(3)$. Determinant of R is unity and is one of the constraints levied on R. The rotation matrix when applied to a position vector rotates it in 3D space.

This rotation transformation by the R matrix may be decomposed into its constituent elements as –

$$\mathrm{R} = \mathrm{R_x R_y R_z} \tag{5.9}$$

Where each of these corresponds to a transformation vector for rotation around the x, y and z axes respectively. They are specified as given below –

$$\left.\begin{aligned} \mathrm{R_x}(\theta) &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix} \\ \mathrm{R_y}(\theta) &= \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \\ \mathrm{R_z}(\theta) &= \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{aligned}\right\} \tag{5.10}$$

Each of the above specifies a transformation matrix as a function of θ, that is capable of rotating the position vector by a specified angle about the specified axis. The illustration of the homography matrix components for homography is given below is given below.

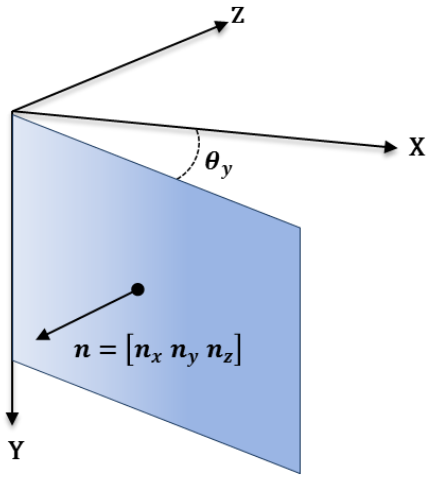

Fig 5.1 Illustration of the rotation angle and the plane normal of homography

As it can be seen above the plane normal **n** is a vector giving the direction cosines of the normal to the plane under consideration. The rotation matrix is defined by the value of the angle of rotation about the corresponding axis. Here θ_y gives the angle of rotation about the Y-axis and the rotation is given in terms of the transformation matrix as presented in equation 5.10.

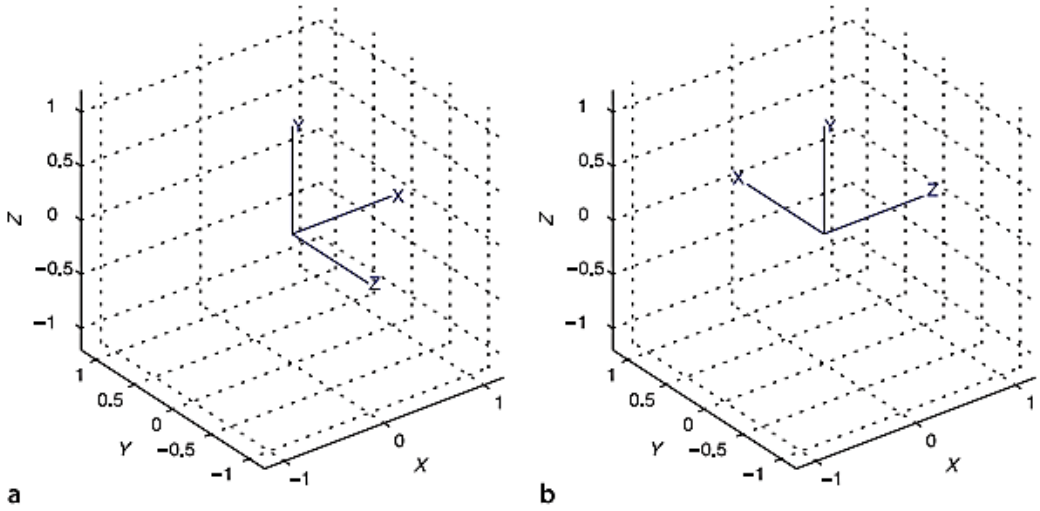

Fig 5.2 a) Reference frame rotated by $\frac{\pi}{2}$ about the x-axis b) Rotated by $\frac{\pi}{2}$ about the y-axis

The figure given in Fig 5.2 illustrates the rotation of the reference frame about the x and y axes by $\frac{\pi}{2}$ radians. The rotation part of the homography decomposition gives the relative rotation of the second plane with respect to the first plane. The translation gives the translation between the coordinate systems of the second plane and the first plane. The rotation matrix need a reference

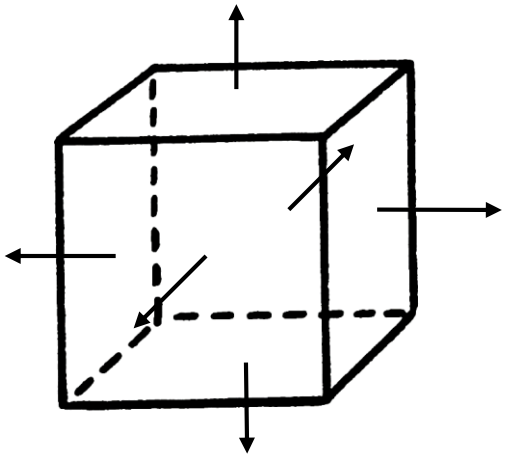

Fig 5.3 The six normals of a cube

initial orientation of the plane with respect to which it is applied. This is defined by the plane normal **n** that gives the direction cosines of the plane normal.

Different planar surfaces that make up the environment have different normal. We are interested in those planes that make up the floor and the walls. The floor and the walls are modelled in this manner and the feature tracks are fir into the appropriate models to yield the motion estimates of the camera as it moves through the corridor.

5.3 Planar Model of the Environment

The environment has been modelled as one made up on planar segments, which is typical for a Manhattan world assumption. Such an assumption as suggested in [r10] is not a strong one owing to the usual regularity in indoor structures such as corridors, hallways…etc. The planar segmentation done using the estimated vanishing point is used to for an approximate hypothesis of the planes that make up the environment. The Floor and wall planes are key her and their mathematical modelling is of utmost importance. The mathematical models of the floor and wall planes are illustrated in the next sections.

5.3.1 Floor Plane model

The floor is a characterised by the plane normal of $\mathbf{n} = [0\ 1\ 0]^T$, which implies it's a plane whose normal is along the Y –axis figure 5.4 illustrates the different elements of the homography matrix –

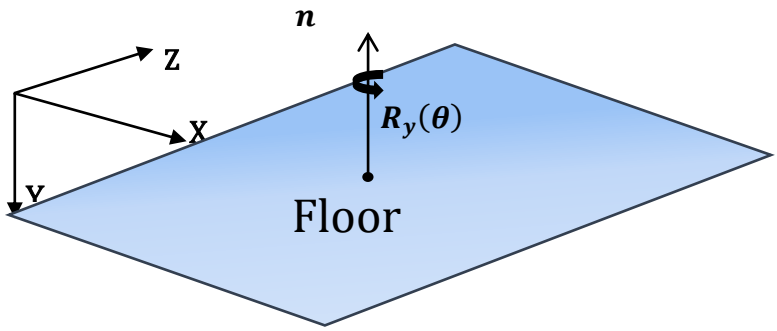

Fig 5.4 Floor plane model with its normal and the axis of rotation

The translation vector and the rotation matrix along with the plane normal form the homography matrix H. The homography matrix is decomposed as given below in equation 5.7. The plane normal in case of the floor plane is given by $\mathbf{n} = [0\ 1\ 0]^T$ which hence transforms the above equation into the following form –

$$H_i = R_i + \left[t_x^i,\ t_y^i, t_z^i\right]^T [0, 1, 0] \tag{5.11}$$

Here H_i is the i^{th} plane to plane homography estimated via tracking the key points between the i^{th} frame and the $(i-1)^{th}$. The thus obtained point correspondences are used for the purpose of estimating the homography matrix using the RanSaC based approach to remove the outliers. This is followed by the inlier estimation and refinement to obtain the best approximation homography by solving the over determined system of equations in the least square sense. The above mentioned matrix H_i is obtained via the previously mentioned steps. The decomposition of this matrix into its constituent elements as defined by equation 5.7 involves an indirect approach. The above equation when expanded transforms into the following form –

$$\mathbf{H_i} = \begin{bmatrix} \cos\left(\theta_y^i\right) & \mathrm{t_x^i} & \sin\left(\theta_y^i\right) \\ 0 & \mathrm{t_y^i} + 1 & 0 \\ -\sin\left(\theta_y^i\right) & \mathrm{t_z^i} & \cos\left(\theta_y^i\right) \end{bmatrix} \tag{5.12}$$

Here the vector $\mathbf{t} = \left[t_x^i \; t_y^i \; t_z^i\right]^T$ gives the translation undergone by the plane from one frame to the other. This indirectly gives the cameras motion trajectory. This model is compared with the homography matrix as illustrated in [13]. The comparison is not straight forward as the homography matrix is obtained is defined at some arbitrary scale.

To counter the effects of the arbitrary scale, the constraints imposed on the terms by the basic equation of trigonometry that tells the sum of squares of cosine and sine of an angle need to be one-needs to be applied here too. Hence to ensure this the obtained homography matrix estimate at the i_{th} step is normalized by diving the elements of the matrix by a constant $(h_{11} + h_{12})^{\frac{1}{2}}$. Once done the translation matrix may be directly obtained from their corresponding places in the new normalized homography matrix $\mathbf{H_i}$**.** The elements $[h_{12} \;\; h_{22} - 1 \;\; h_{32}]$ together form the translation vector of the homography matrix. The rotation matrix R is defined only by one of the three Euler angles θ_y as the robots angular pose variation along the remaining two axes has been assumed to be invalid. This is because of the *Manhattan World* assumption made in regard to the environment that restricts the floor plane to a flat and even surface. Assumption is made about the knowledge of the vertical direction thus making the pitch and roll angles θ_x and θ_z thus the images are pre rotated prior to the processing stage.

The rotation matrix R_y^i is obtained using the appropriate equation from equation 5.8 and the necessary θ value of the angular rotation may be obtained using the values h_{11} or h_{13}. Once obtained, the rotation matrix gives the instantaneous rotation of the plane as the plane underwent motion due to the robot's movement. This has to be appended to the initial pose of the robot to estimate the instantaneous absolute pose of the robot. The robot pose at $i = 0$ is to be known

beforehand. The total rotation or translation is estimated with respect to the initial pose of the robot.

5.3.2 Wall Plane Model

The wall planes that form the environment need to be modelled differently because of the fact that the normal to the plane are no more constant and vary as the robots pose changes. However, the plane normal might be put under some constraints as we have knowledge about the vertical direction. The plane normal **n** of the wall plane is of the form $\mathbf{n_i} = \begin{bmatrix} n_x^i & 0 & n_z^i \end{bmatrix}^T$ which is the instantaneous direction of the plane normal. The wall plane model is compared against the homography matrix obtained by tracking feature points on the wall plane. This approach is very similar to the one mentioned above except that the wall planes are modelled and the homography matrix obtained by tracking the features along the wall plane is compared against this model. The homography matrix is obtained by and in its decomposed form is a s given below, with the distance d set to unity–

$$H_i = R_i + \begin{bmatrix} t_x^i, & t_y^i, t_z^i \end{bmatrix}^T \begin{bmatrix} n_x^i, 0, & n_z^i \end{bmatrix} \tag{5.13}$$

The figure 5.5 better illustrates the plane normal and the rotation in the camera coordinate system. The equation 5.11 when expanded turns into the following form given –

$$\boldsymbol{H_i} = \begin{bmatrix} \cos(\theta_y^i) + n_x^i t_x^i & 0 & \sin(\theta_y^i) + n_z^i t_x^i \\ n_x^i t_y^i & 1 & n_z^i t_y^i \\ n_x^i t_z^i - \sin(\theta_y^i) & 0 & \cos(\theta_y^i) + n_z^i t_z^i \end{bmatrix} \tag{5.14}$$

Proper normalizing of the homography matrix is essential in this case. This is done by dividing the elements of the homography matrix by h_{22}.The translation vector is obtained individually first by

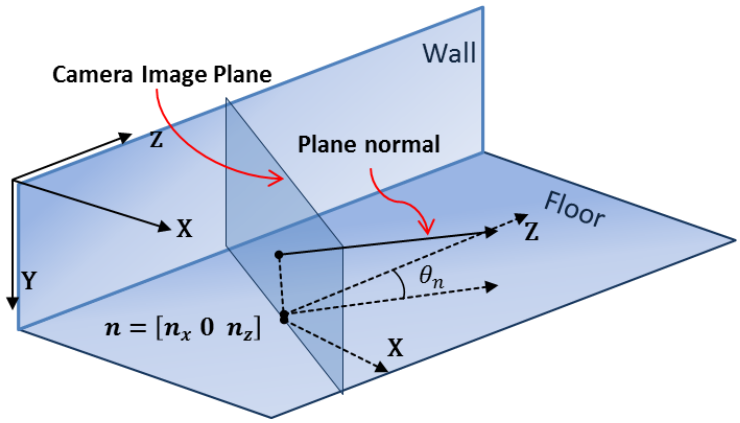

Fig 5.5. Normal of camera image plane making angle θ_n with the Z-axis

estimating using $t_y = \pm(h_{21}^2 + h_{23}^2)^{\frac{1}{2}}$. Here n_x and n_z are given by $\sin(\theta_n)$ and $cos(\theta_n)$ respectively. The two solutions may be used to estimate n_x, n_z as –

$$n_x^i = \frac{h_{21}^i}{t_y^i} \quad , \qquad n_z = \frac{h_{23}^i}{t_y^i} \tag{5.15}$$

This results in a pair of normals one each for each sign of the t_y solution. The solution to $\theta_y$is obtained by solving for the equations h_{31} and h_{33}. The proper normal among the two is the one that produces t_z in the direction of the robot's motion. The translation terms may be obtained using the following relations as illustrated in [13] using the entries of the normalized homography matrix H^i.

$$\left.\begin{aligned} t_x &= \frac{\mathrm{h}_{11}^{\mathrm{i}} - \cos\left(\theta_y^i\right)}{n_x^i} \\ t_z &= \frac{\mathrm{h}_{33}^{\mathrm{i}} - \cos\left(\theta_y^i\right)}{n_z^i} \end{aligned}\right\} \quad 5.16$$

The thus obtained translation and rotation parameters need to be appended to the previously estimated rotation and translation parameters to obtain the absolute pose of the camera mounted atop the robot. This appending operation may be done as illustrated below, which gives the total pose of the robot. Let $\mathrm{R_T}$ be the total angular pose of the robot pose of the robot. Initially it is assumed to be identity matrixI, implying that the robot is oriented at zero Euler angles i.e.—

$$\mathrm{R_T} = \mathrm{R_x}(0).\mathrm{R_y}(0).\mathrm{R_z}(0) \quad 5.17$$

The relative change is angular pose is appended as –

$$\mathrm{R_T^i} = \mathrm{R_T^{i-1}}\,\mathrm{R_i} \quad 5.18$$

Similar the translation is appended as –

$$t_T^i = t_T^{i-1} + t_i \quad 5.19$$

Here t_T^i is the absolute pose whose initial state at step $i = 0$ is known. And t_i is the relative change in the translation parameter estimated by tracking key points over consecutive frames as the robot moved. This way the robot motion was estimated using the planar model of the environment.

5.4 Conclusion

A planar model of the environment was used in the estimation of the motion trajectory for vision based steering of the mobile robot platform. This was done by using the vanishing point to form

an approximate model of the environment. The vanishing point was estimated using line detection followed by estimation of vanishing point using the RanSaC based approach.

The RanSaC based approach was used in two instance, at the VP estimation stage and then the homography estimation step. At both stages, this was done to sift the inliers from data points corrupted by spurious noise in the optical flow tracks as well as the imperfections in the line detection step during the vanishing point estimation stage. The DLT based approach to homography estimation was also discussed in detail and the RanSaC based approach in the presence of outliers is also outlined. The environment modelling done under the *Manhattan World* constraints were clearly illustrated in section 5.3.1 and 5.3.2. This was followed by comparing the obtained homography matrix to the pre-defined models of the planes to estimate the motion of the robot as it traversed the environment.

Chapter 6

EXPERIMENTAL RESULTS AND DISCUSSION

6.1 Results and Discussion

The experiments were carried out on the University of Michigan Dataset [7, 8] which consisted of long sequence of video frames taken as the robot navigated the indoor environments. A camera was mounted atop a moving platform and the ground truth was estimated used a laser range finder. The estimation of the robots ground truth pose was done using an occupancy grid based algorithm. The camera calibration matrix was also provided by the data set and the images were corrected beforehand for distortion incurred during the imaging process. The camera setup place at zero pitch and roll angles as assumed during the motion estimation stage. The camera set up of the dataset recording platform of the University of Michigan Indoor Navigation Dataset [7, 8].

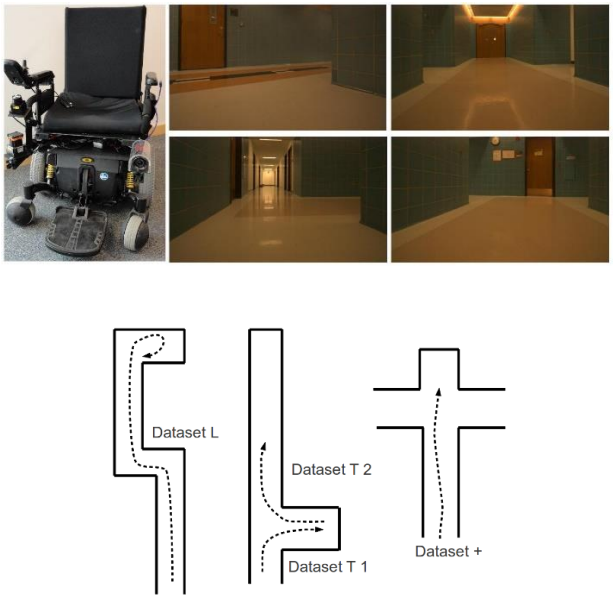

Fig 6.1 Above- the indoor video acquisition setup and snapshot of the four video sequences. Below – the top view of different trajectories taken by the mobile platform in the corridor.

For each video sequences, an estimated camera pose in each frame of the video was provided in the dataset. It provided with information of the camera's pose at each frame when it was captured. The pose was provided in the form of its $\boldsymbol{x}, \boldsymbol{y}$ and $\boldsymbol{\theta}$ values. The cameras ground truth poses were estimated using an occupancy grid mapping algorithm. The intrinsic parameters of the camera used i.e. the camera calibration matrix was also provided**.** The intrinsic parameters of the cameras are –

Focal length $fc = [\ 391.689294937162117\ 392.998178928112054\ ]$

Principal point $cc = [\ 346.792888397916727\ 145.190086296047753\ ]$

The distortion in the images has been corrected. The camera was set-up so that there was zero tilt and roll angle with respect to the ground. The camera has a fixed height (0.47 m) with the ground throughout the video.

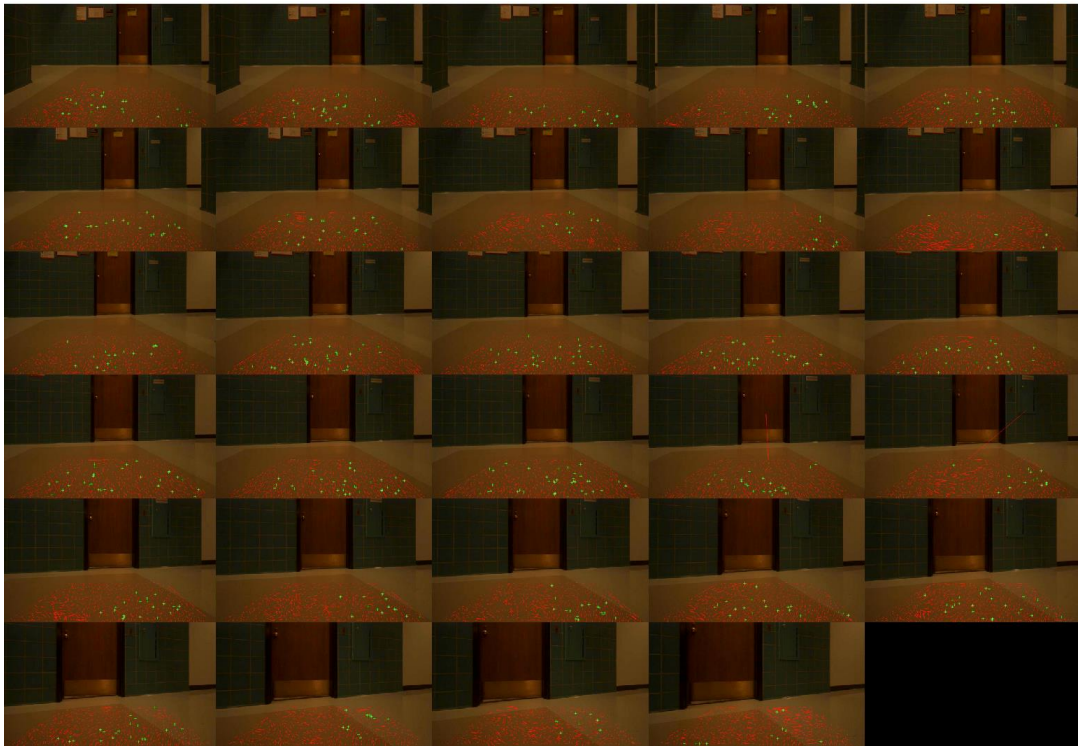

Fig 6.2 Montage of the frames and the ground plane tracks 'T 2' Sequence

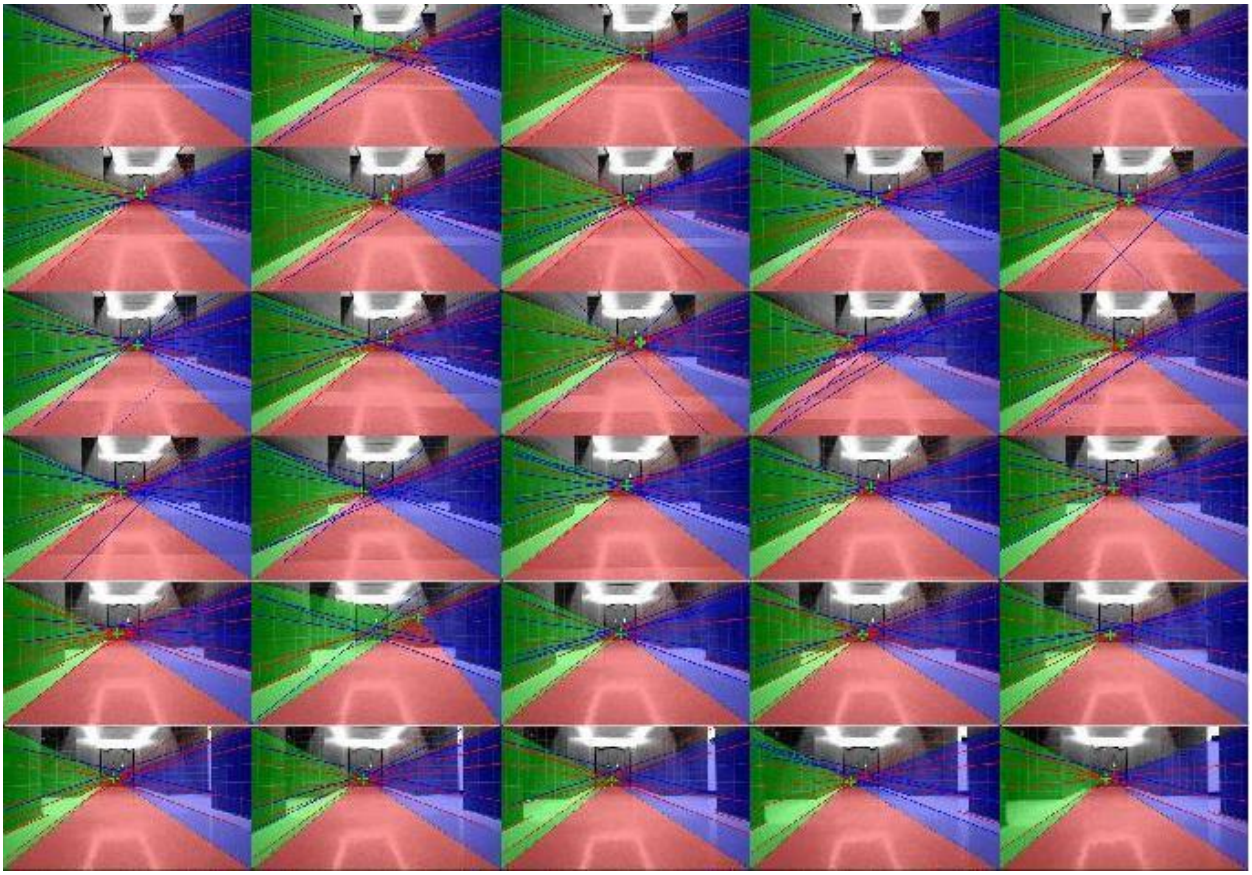

Fig 6.3 Detected Vanishing Point and the corresponding planar segmentation

The above image is a montage of several frames of the '+' sequence of the video in which the vanishing point is detected and the the planar segmentaion of the image has been done. This is done to segment the image into its planar components. The *Manhattan World* assumption works in our favour and helps make a good approximation of the structure in the environment. The Vanishing Point is detected using the RanSaC based approach as suggested in the previous sections.The thus obtained imageis segmented based on the VP location and the dominant edges in the image as suggested in the previous section. Sometimes the estimate of the VP was not accurate which can however be skipped by assuming that the distance change in the VP is not that significant over two consecutive frames.

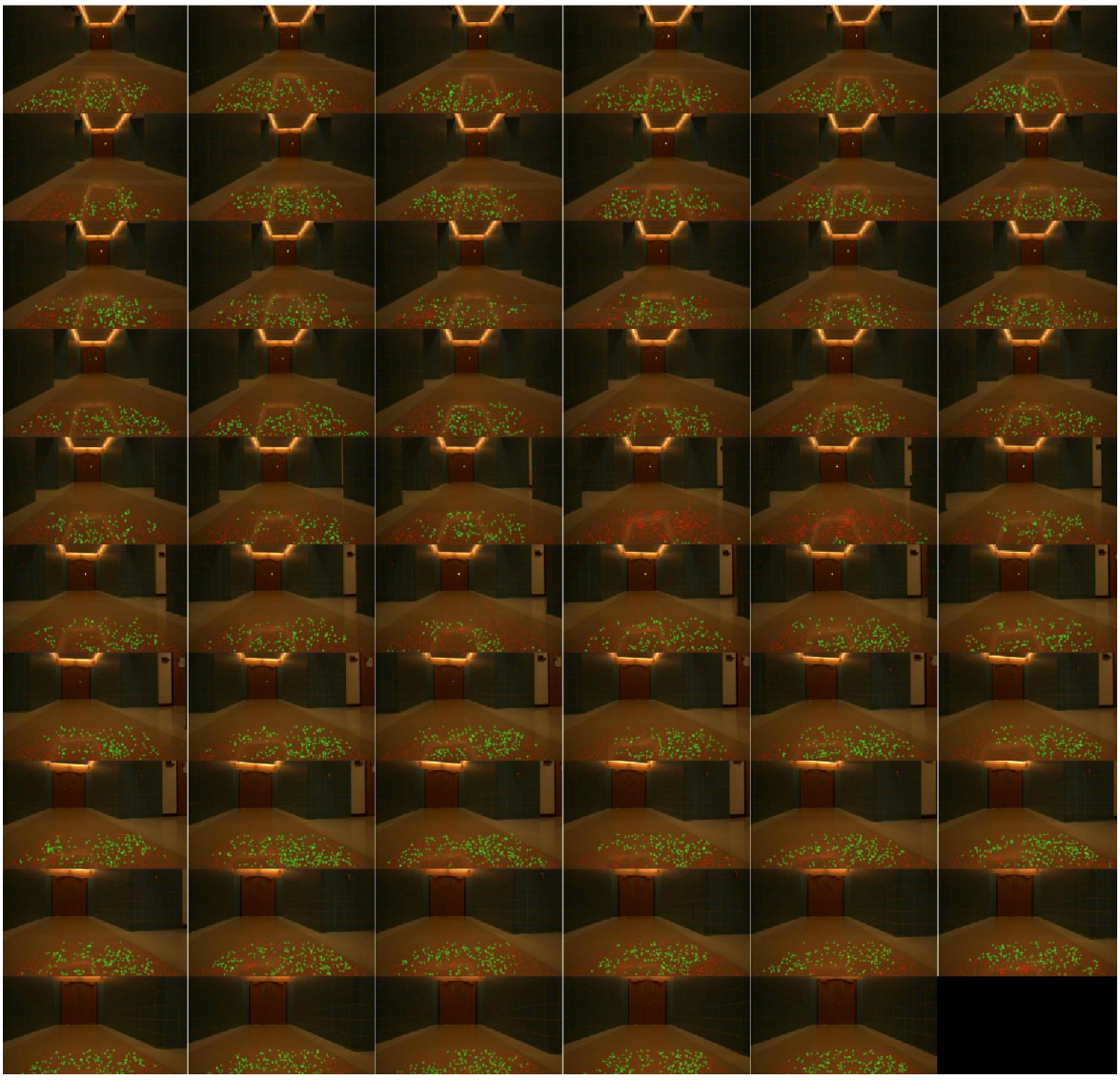

Fig 6.4 Montage of the frames and the ground plane tracks '+' Sequence

Above is a montage of the different frames of the '+' Sequence with the tracking being done only in the masked region. The masks for each frame are obtained by making use of the Vanishing Point of that frame and segmenting the image into the respective planar components as suggested and illustrated above. Each of the frame is probed for key points which are then tracked in the next frame. The tracks are processed to estimate a homography that relates the point correspondences.

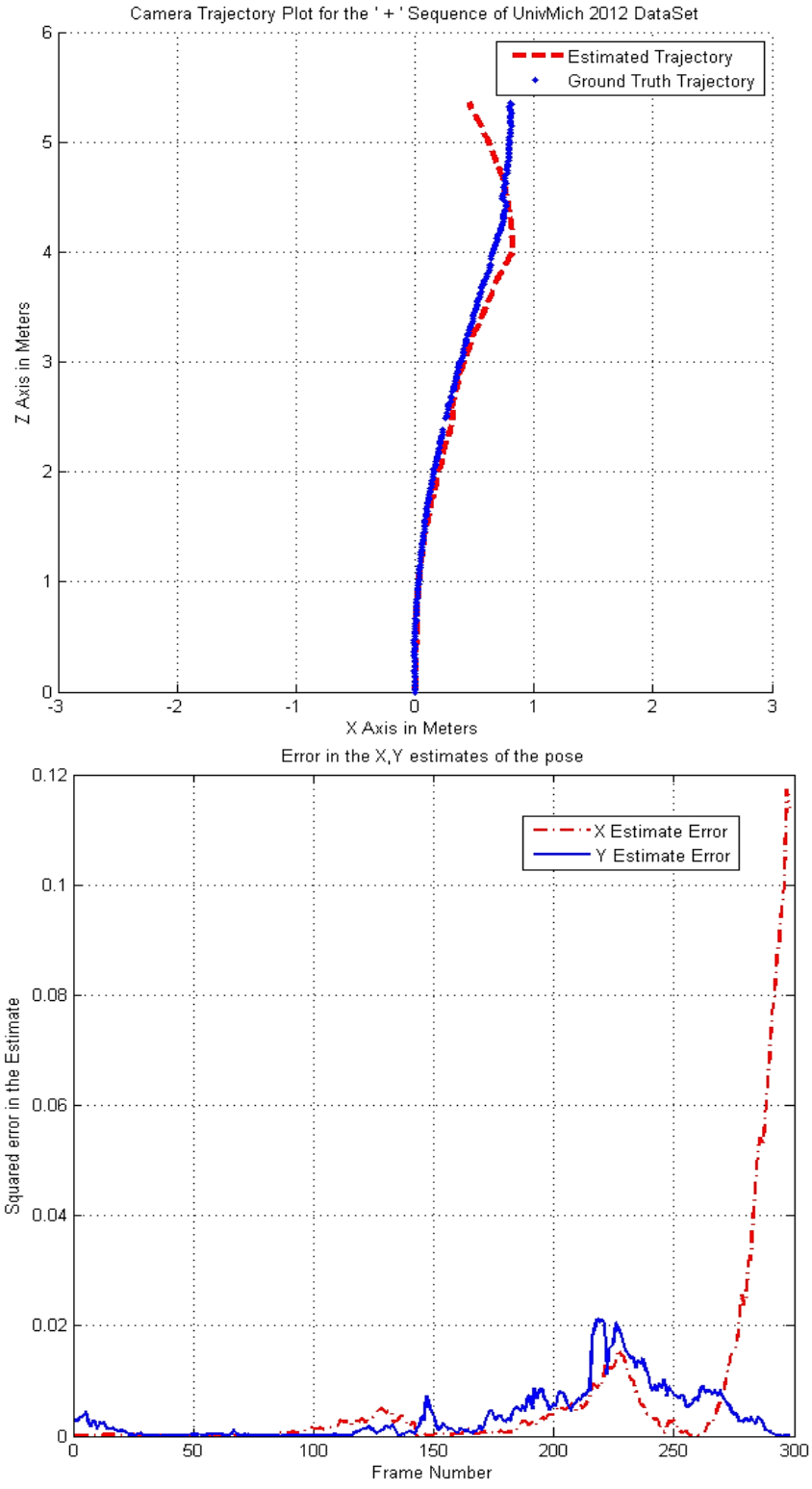

Fig 6.5 above – Plot of the Camera Trajectory (‘ + ’ *Sequence*) and below – Squared error plot of the X, Z estimates of Camera Pose.

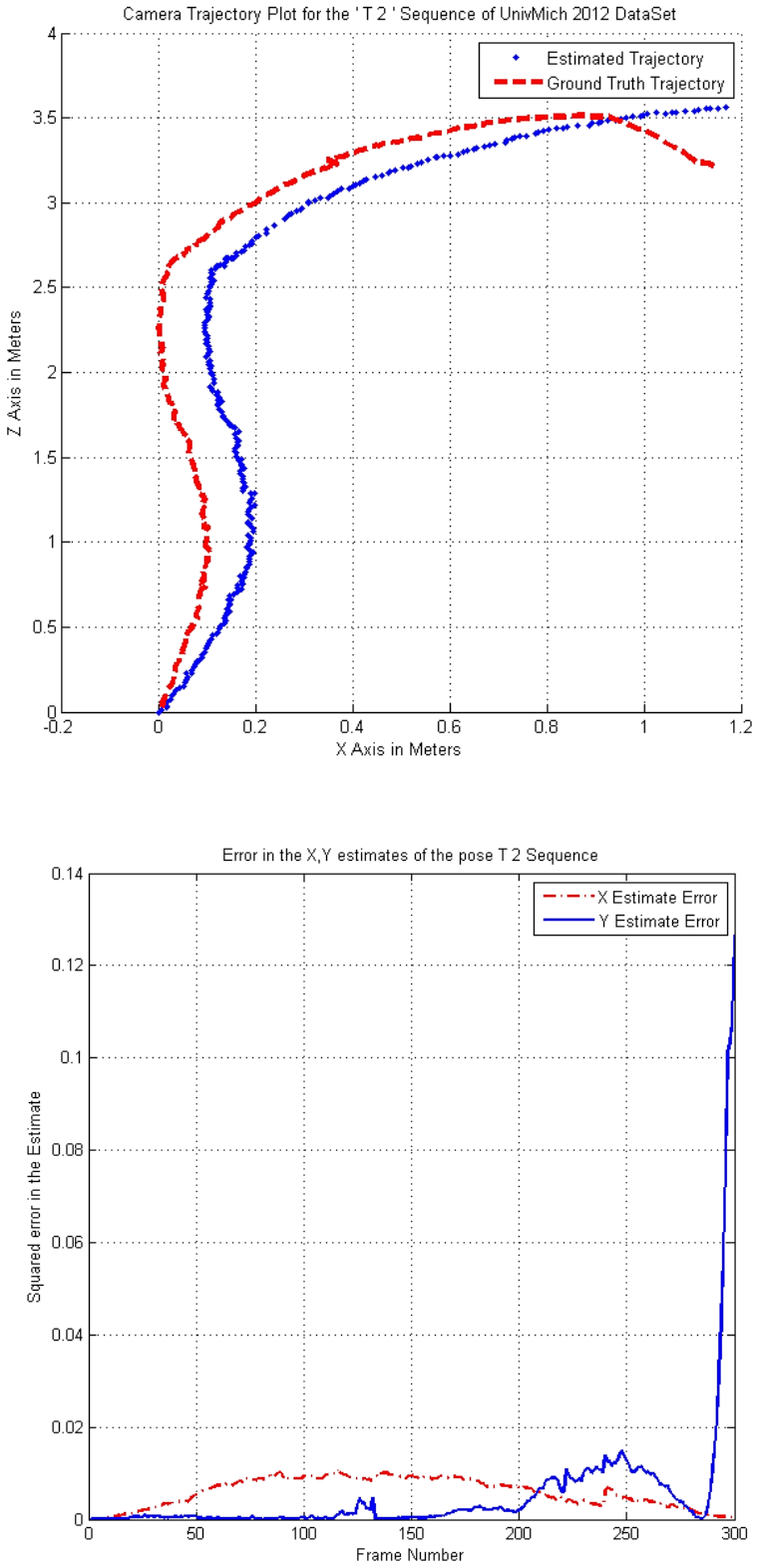

Fig 6.6 above – Plot of the Camera Trajectory (*'T 2' Sequence*) and below – Squared error plot of the X, Z estimates of Camera Pose.

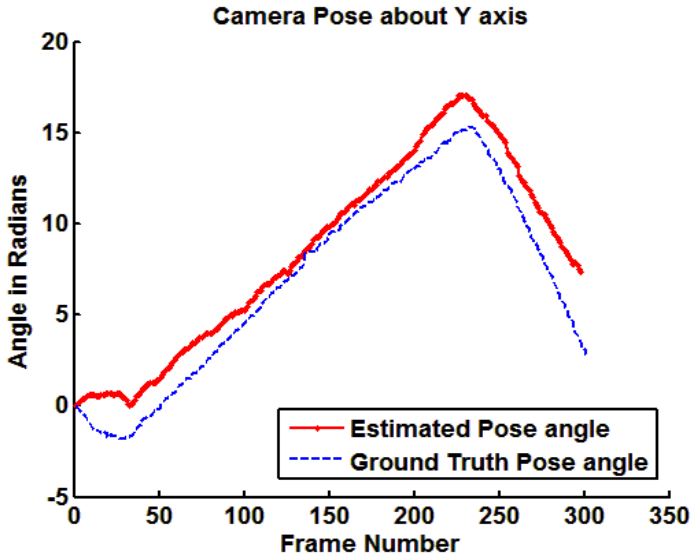

Fig 6.7. Plot of the estimated camera pose against the ground truth poses of the camera ('+' Sequence)

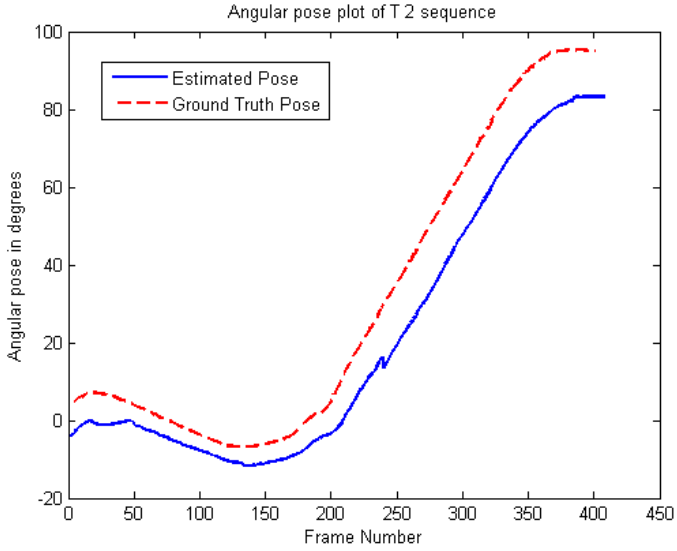

Fig 6.8. Plot of the estimated camera pose against the ground truth poses of the camera ('T 2' Sequence)

The two previous plots of the estimated and the obtained trajectories have a commonality between the two. This similarity becomes further evident in the form of the increase in error as the camera reaches intersection in the corridor paths. This is due to the accumulated direction drift resulting from the inherent error in the pose estimation. It is also a result of the lack of vanishing point location as the camera moves closer to the corridor terminus points. Since the VP location is

unknown, a proper mask for the ground plane may not the present resulting in an improper homography being fit into the point correspondences.

6.2 Conclusion and Future Work

The use of Vanishing Points in the estimation of the camera motion seems to have an advantage over the traditional ground plane based tracking approaches. This is mainly due to the solution to problem of virtual plane which has been addressed through the use of the Vanishing Points. The RanSaC based approaches used in the context of Homography estimation as well as its use in the finding of the Vanishing Point proves to be efficient in sifting the inliers even when corrupted by outliers. Most of the outliers in the Homography estimation stage have been observed to be due to spurious tracks and some due to the points that belong to other planes. The problem of error accumulation discussed above needs to be addressed. Also, the error due to lack of proper vanishing points at the end of a path in the corridor video sequences needs additional attention. The pose drift may be reduced through the use of Kalman filter based pose refinement by modelling the motion of the camera. Also the use of landmarks in the pose estimation may help further reduce the error at the endings. Absolute pose estimation may be done through key point based recognition and localization. This is necessary only at the end location of a straight path. Also, the assumption that – the environment is devoid of any obstacles – is relatively strong as in reality there may be obstacles in the environment. The use of obstacle detection algorithms may further increase the scope of the vanishing point based camera motion estimation as it increases the accuracy of the estimated camera pose trajectory.

REFERENCES

1. *J.D. Crisman, C.E. Thorpe, SCARF: a Color vision system that tracks roads and intersections, IEEE Transactions on. Robotics and Automation 9 1 1993 49–58*
2. *J.D. Crisman, C.E. Thorpe, UNSCARF, A Color Vision System for the Detection of Unstructured Roads, Proc. IEEE International Conference on Robotics and Automation,Sacramento, CA, USA, April 1991, pp. 2496–2501*
3. *Color-Based Road Detection in Urban Traffic Scenes Yinghua He, Hong Wang, and Bo Zhang SCARF*
4. *Miksik, Ondrej, et al. "Robust detection of shady and highlighted roads for monocular camera based navigation of UGV." Robotics and Automation (ICRA), 2011 IEEE International Conference on. IEEE, 2011.*
5. *Dahlkamp, Hendrik, et al. "Self-supervised Monocular Road Detection in Desert Terrain." Robotics: science and systems. 2006.*
6. *Levinson, Jesse, and Sebastian Thrun. "Robust vehicle localization in urban environments using probabilistic maps." Robotics and Automation (ICRA), 2010 IEEE International Conference on. IEEE, 2010.*
7. *Grace Tsai and Benjamin Kuipers, "Dynamic visual understanding of the local environment for an indoor navigating robot," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2012.*
8. *Grace Tsai and Benjamin Kuipers, "Toward visual semantic modeling of the local environment for an indoor navigating robot," IEEE/RSJ International Conference on Intelligent Robots and Systems Workshop on Active Semantic Perception (ASP'12), 2012.*

9. *Scaramuzza, Davide, and Friedrich Fraundorfer. "Visual odometry [tutorial]."Robotics & Automation Magazine, IEEE 18.4 (2011): 80-92.*
10. *Coughlan, James M., and Alan L. Yuille. "Manhattan world: Compass direction from a single image by bayesian inference." Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on. Vol. 2. IEEE, 1999.*
11. *Hartley, Richard, and Andrew Zisserman. Multiple view geometry in computer vision. Cambridge university press, 2003.*
12. *Malis, Ezio, and Manuel Vargas. "Deeper understanding of the homography decomposition for vision-based control." (2007).*
13. *Saurer, Olivier, Friedrich Fraundorfer, and Marc Pollefeys. "Homography based visual odometry with known vertical direction and weak Manhattan world assumption." Vicomor Workshop at IROS. Vol. 2012. 2012.*
14. *Jain, Ramesh, Rangachar Kasturi, and Brian G. Schunck. Machine vision. Vol. 5. New York: McGraw-Hill, 1995.*
15. *Horn, Berthold K., and Brian G. Schunck. "Determining optical flow." 1981 Technical Symposium East. International Society for Optics and Photonics, 1981.*
16. *Harris, Chris, and Mike Stephens. "A combined corner and edge detector."Alvey vision conference. Vol. 15. 1988.*
17. *Bonin-Font, Francisco, Alberto Ortiz, and Gabriel Oliver. "Visual navigation for mobile robots: A survey." Journal of intelligent and robotic systems 53.3 (2008): 263-296.*
18. *Choset, Howie M., ed. Principles of robot motion: theory, algorithms, and implementation. MIT press, 2005.*

19. *Dalgleish, F. R., S. W. Tetlow, and R. L. Allwood. "Vision-based navigation of unmanned underwater vehicles: a survey. Part 2: Vision-based station-keeping and positioning." Proceedings of the Institute of Marine Engineering, Science and Technology. Part B, Journal of marine design and operations. No. 8. Institute of Marine Engineering, Science and Technology, 2005.*
20. *Davis, Larry S. "Visual navigation at the University of Maryland." Robotics and autonomous systems 7.2 (1991): 99-111.*
21. *G. Giralt, R. Sobek, and R. Chatila, "A Multi-Level Planning and Navigation System for a Mobile Robot; A First Approach to Hilare ,"Proc. Sixth Int'l Joint Conf. Artificial Intelligence ,vol. 1, pp. 335-337, 1979*
22. *H.P. Moravec, "Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover," PhD thesis, Stanford Univ., Sept. 1980. (Published as Robot Rover Visual Navigation. Ann Arbor, MI: UMI Research Press, 1981.)*
23. *H.P. Moravec, "The Stanford Cart and the CMU Rover,"Proc. IEEE,vol. 71, no. 7, pp. 872-884, July 1983.*
24. *R. Chatila and J.-P. Laumond, "Position Referencing and Consistent World Modeling for Mobile Robots,"Proc. IEEE Int'l Conf. Robotics and Automation,pp. 138-145, Mar. 1985.*
25. *J. Borenstein and Y. Koren, "Real-Time Obstacle Avoidance for Fast Mobile Robots,"IEEE Trans. Systems, Man, and Cybernetics, vol. 19, no. 5, pp. 1179-1187, 1989.*
26. *A. Kosaka and A.C. Kak, "Fast Vision-Guided Mobile Robot Navigation Using Model-Based Reasoning and Prediction of Uncertainties, "Computer Vision, Graphics, and Image Processing—Image Understanding, vol. 56, no. 3, pp. 271-329, 1992.*

27. *S. Thrun, "Probabilistic Algorithms in Robotics," Technical Report CMU-CS-00-126, Carnegie Mellon Univ., 2000.*
28. *M. Isard and A. Blake, "Condensation—Conditional Density Propagation for Visual Tracking,"Int'l J. Computer Vision, vol. 29, no. 1, pp. 5-28, 1998.*
29. *G. Simon, A. Fitzgibbon, and A. Zisserman, ―Markerless tracking using planar structures in the scene, ‖ in Augmented Reality, 2000.(ISAR 2000). Proceedings. IEEE and ACM International Symposium on, 2000, pp. 120–128.*
30. *G. Simon and M.-O. Berger, "Pose estimation for planar structures", Computer Graphics and Applications, IEEE, vol. 22, no. 6, pp. 46–53, Nov/Dec 2002.*
31. *Z. Zhang, ―A flexible new technique for camera calibration, ‖ IEEE Transactions on pattern analysis and machine intelligence, vol. 22, no. 11, pp. 1330–1334, 2000.*
32. *P. Sturm, ― Algorithms for plane-based pose estimation, ‖ pp. 1010–1017, June 2000. [Online]. Available: http://perception.inrialpes.fr/Publications/2000/Stu00b*
33. *Matas, Jiri, Charles Galambos, and Josef Kittler. "Robust detection of lines using the progressive probabilistic hough transform." Computer Vision and Image Understanding 78.1 (2000): 119-137.*
34. *Huttunen, Ville, and Robert Piché. "A monocular camera gyroscope."Gyroscopy and Navigation 3.2 (2012): 124-131.*
35. *Ebrahimpour, R., et al. "Vanishing point detection in corridors: using Hough transform and K-means clustering." Computer Vision, IET 6.1 (2012): 40-51.*
36. *Prince, Simon JD, and Simon Jeremy Damion Prince. Computer vision: models, learning, and inference. Cambridge University Press, 2012.*