POSITIONAL ESTIMATION TECHNIQUES FOR AN AUTONOMOUS MOBILE ROBOT

FINAL REPORT: NASA GRANT NAG 9-361

N. Nandhakumar
and J. K. Aggarwal,
Principal Investigators

Computer and Vision Research Center
The University of Texas at Austin
Austin, Texas 78712-1084

J. K. Aggarwal, Director

Telephone: 512/471-3259

Positional Estimation Techniques for an Autonomous Mobile Robot

Final Technical Report for NASA Grant NAG 9-361

Principal Investigator: N.Nandhakumar and J.K.Aggarwal

Abstract

This report presents a description of the work done under the NASA Grant NAG 9-361. It describes techniques for positional estimation of a mobile robot navigating in an outdoor environment. A comprehensive review of the various positional estimation techniques studied in the literature is first presented. The techniques are divided into four different types and each of them is discussed briefly. In this work two different kinds of environments are considered for positional estimation; a mountainous natural terrain and an urban, man-made environment with polyhedral buildings. In both these cases the robot is assumed to be equipped with a single visual camera that can be panned and tilted and also a 3-D description (world model) of the environment is given. This world model is in the form of Digital Elevation Map (DEM) for the natural terrain case. For the urban environment, it is assumed that the 3-D descriptions of the roof tops of the buildings are given. Such a description could be obtained from a stereo pair of aerial images or from the architectural plans of the buildings. Techniques for positional estimation using the camera input and the world model are presented.

List of publications

[1] R.Talluri and J.K.Aggarwal, "Positional estimation for a mobile robot in an outdoor environment," *IEEE Workshop on Intellligent Robots and Systems, IROS '90*, pp. 159-166,Japan, July 90.

[2] R.Talluri and J.K.Aggarwal, "A Positional estimation technique for and autonomous land vehicle in an unstructured environment," *i-SAIRAS '90*, Japan, Nov 90.

[3] R.Talluri and J.K.Aggarwal, "Edge Visibility Regions - a new representation of the environment of a mobile robot," *IAPR Workshop on Machine Vision Applications, MVA '91*, pp. 375-380,Japan, Nov 90.

[4] R.Talluri and J.K.Aggarwal, "Positional Estimation techniques for an autonomous mobile robot - a review," submitted to *IEEE Transactions on Robotics and Automation.*

[5] R.Talluri and J.K.Aggarwal, "Positional Estimation of a mobile robot using Edge Visibility Regions," submitted to, *Computer Vision and Pattern Recognition, CVPR '91.*

Contents

1 Introduction

The problem of positional estimation has received considerable attention, and many techniques have been proposed for solving it. Most mobile robots are equipped with wheel encoders that give an estimate of the robot's position at every instant. However, the information from these encoders is differential, and, due to wheel slippage and quantization effects, these estimates of the robot's position contain small errors. However the errors build up quickly as the robot moves, and the position estimate becomes increasingly uncertain. So, most mobile robots use some other form of sensing, like vision or range, to sense the environment and to aid in the position estimation process.

Various techniques have been studied for estimating the position and pose of an autonomous mobile robot using different kinds of sensors. The techniques vary depending on the kind of environment in which the robot navigates, the known conditions of the environment, and the type of sensors with which the robot is equipped. Broadly the position estimation techniques can be classified into the following four types: 1) landmark-based methods 2) methods using trajectory integration and dead reckoning 3) methods using a standard reference pattern and 4) methods using the *a priori* knowledge of a world model and matching sensor data with the world model for position estimation. Section 2 describes all these different techniques briefly. More details are given in [57]

The present work falls into the fourth category i.e., the robot is aided in its navigational tasks by providing *a priori* information about the environment in the form of a preloaded world model. The basic idea is to sense the environment using on board sensors on the robot and then to try to match these sensory observations to the preloaded world model. This process yields an estimate of the robot's position and pose with a reduced uncertainty and then allows the robot to perform other navigational tasks. The problem in such an approach is that the sensor readings and the world model may be in different forms. For instance, given a CAD model of the building and a visual camera, the problem is to match the 3-D descriptions in the CAD model to the 2-D visual images.

In this work techniques are presented for estimating the position of a mobile robot in an outdoor environment. Two kinds of environments are considered; a mountainous natural terrain and an urban man-made environment consisting of polyhedral buildings.

In the former case a Digital Elevation Map (DEM) of the area in which the robot is to navigate is assumed to be given. The robot is also assumed to be equipped with a camera that can be panned and tilted, and a device to measure the robot's elevation above the datum. No recognizable landmarks are assumed to be present in the environment where the robot is to navigate. The DEM is a 3-D database. It records the terrain elevations for ground positions at regularly spaced intervals. The images recorded by the camera are 2-D intensity images. The problem is to find common features to match the 2-D images to the 3-D DEM. The presented solution makes use of the DEM information and structures the problem as a heuristic search in the DEM for the possible robot location. The shape and position of the horizon line in the image plane and the known camera geometry of the perspective projection are used as parameters to search the DEM. Heuristics drawn from geometric constraints are also used to reduce the search space [54,55].This is described in section 3.

The case of mobile robot navigating in a structured, man-made, urban environment consisting of polyhedral buildings is considered in section 4. The 3-D descriptions of the roof tops of the buildings is assumed to be given. Such a description may be obtained from a pair of stereo aerial images or from the architectural plans of the buildings. The robot uses the camera to image the surroundings, each time adjusting the tilt angle so that the roof top edges are clearly visible in the image plane. Now if a correspondence is established between the 3-D descriptions of these edges and their images the position and pose of the robot can be estimated. However establishing this correspondence is in general not a trivial problem. To alleviate this problem, it is proposed to use the geometric relations between the 3-D descriptions of the roof top edges (model edges) to prune the list of possible correspondences. The viewing plane (the plane in which the robot navigates) is divided into distinct, non-overlapping regions called the *Edge Visibility Regions* (EVRs). These EVRs essentially capture the geometric relations between the model edges with regard to their visibility from various regions in the viewing plane. Associated with each EVR is a *Visibility List* (VL) which is a list of the model edges that are visible in that EVR; also stored for each edge in the VL of an EVR is the range of orientation angles of the robot for which the edge is visible in this EVR. In this research methods for generating the EVRs given a world model are discussed. An upperbound on the maximum number of EVRs that would be generated for a given model is derived [56]. Also the use of the EVRs and their associated VLs in estimating the position and pose of an autonomous mobile robot is discussed [58]. The use of the EVR representation of the environment of a robot for other navigational tasks like path-planning are outlined.

2 Review of positional estimation techniques

In this section, various methods and techniques studied for estimating the position and pose of an autonomous mobile robot are reviewed. The techniques vary depending on the kind of environment in which the robot navigates, the known conditions of the environment, and the type of sensors with which the robot is equipped. Broadly the position estimation techniques can be classified into the following four types: 1) landmark-based methods 2) methods using trajectory integration and dead reckoning 3) methods using a standard reference pattern and 4) methods using the *a priori* knowledge of a world model and matching sensor data with the world model for position estimation.

2.1 Landmark based methods

Using landmarks for position estimation is a popular approach. The robot uses the knowledge of its approximate location to locate the landmarks in the environment. Once these landmarks are identified and the range/attitude of these relative to the robot is measured, in general, the position and pose of the robot can be triangulated from these measurements with a reduced uncertainty. The landmarks could be naturally occurring in an outdoor environment,like the tops of buildings, roof edges, hill tops, etc., or could be identifiable beacons placed at known positions to structure the environment. One of the basic requirements of the landmark-based methods is that the robot should have the ability to identify and locate the landmarks, which is, in general not an easy task. The position estimation methods based on landmarks vary significantly, depending on the sensors used: range or visual sensors; the type of landmarks, i.e., whether they are point sources or lines etc.; the number of landmarks needed.

Case [7] summarizes the landmark-based techniques for position estimation and presents new method, called *running fix* method, for position estimation. Clare D. McGillem et al [39] also describe an infra-red location system for navigating autonomous vehicles. They present an efficient method for position estimation using three infra-red beacons to structure the environment and an optical scanner on the robot capable of measuring the angles between a pair of beacons. Nasr and Bhanu [45] present a new approach to landmark recognition based on the perception, reasoning, and expectation (PREACTE) paradigm for the navigation of an autonomous mobile robot.

Sugihara [52] presents algorithms for the position estimation of a mobile robot equipped with a single visual camera. He considers the problem of a robot given a map of a room where it is to navigate. From the images taken by the robot, with the optical axis parallel to the floor, only the vertical edges are extracted. On the map the points from where the vertical edges can arise are assumed to be given. Sugihara then considers two classes of problems. In the first class, all vertical edges are identical, and he searches for the point where the image is taken by establishing a correspondence between the vertical edges in the images and those on the map. In the second class of problems, the vertical edges are not distinguishable from each other and the exact directions in which the edges are seen are not given; only the order in which they are found in the image is given. The problems are considered mainly from a computational complexity point of view.

Photogrammetry generally deals with the mathematical representation of the geometrical relations between physical objects in three-dimensional space based on their images recorded on a two dimensional medium. One of the problems of photogrammetry is to determine the location of an airborne camera from which a photograph was taken by measuring the positions of a number of known ground objects or landmarks on the photograph. This problem is sometimes known as the *camera calibration problem* . The orientation and position of the camera in the object space are traditionally called the camera's *exterior orientation parameters* as opposed to its *interior orientation parameters*, which are independent of the co-ordinate system of the ground objects. The interior orientation parameters include such elements as the camera's effective focal length, lens distortion, decentering, image plane scaling, and optical axis orientation.

The problem of estimating the position and pose of an autonomous mobile robot given a single visual camera is, in essence, similar to this camera exterior orientation problem in photogrammetry. However, since the robot is ground-based and has position encoders and other sensors on it, these can be used to constrain the possible orientation and pose. In general, the exterior camera orientation problem involves solving for six degrees of freedom, three rotational and three translational. Traditionally, in single camera photogrammetry, by observing the object's feature points on the image, it is possible to solve the exterior orientation calibration problem using a traditional method known as *space resection* (El Hassan [15]).The method is based on perspective geometry of a simplified camera model, derived from a pinhole optics, in which the image of each feature point is projected onto the image plane by a ray connecting the feature point with the pinhole lens. This collinearity condition results mathematically in two nonlinear equations for each feature point. Hence at least three non-collinear points are required to solve for the six degrees of freedom. These collinearity equations are linearized and solved in an iterative fashion. When the images are noisy, more than three points can be used, with least squares criteria, to take advantage of data smoothing. These methods are now standard in the photogrammetry literature (Wolf [64]). Iterative solutions are generally more computationally demanding, so that simplifying assumptions are usually necessary for real-time applications. Over the years, a number of alternate methods have been proposed in an effort to improve the efficiency of the camera calibration procedure.

Fischler and Bolles [18] studied the exterior calibration problem in connection with the concept of *random sample consensus* (RANSAC), a methodology proposed for processing large data sets with gross errors or outliers. Ganapathy [20] presents a noniterative, analytic technique for recovering the six exterior orientation parameters as well as four of the interior orientation parameters, 2 for scaling and 2 for the location of the origin in the image plane. He essentially decomposes the given transformation into the various camera parameters that constitute the components of the matrix. Roger Tsai [60] presents a two stage technique for the calibration of both the exterior and interior parameters of the camera. It is probably the most complete camera calibration method proposed so far. Liu, Huang, and Faugeras [35] present a new method for determining the camera location using straight line correspondences. They show that the rotation matrix and the translation vector can be solved for separately. Both linear and nonlinear algorithms are presented for estimating the rotation. Rakesh Kumar [31] argues that the rotation and translation constraints, when used separately, are very weak constraints and hence even small errors in the rotation stage become amplified into large errors in the translation stage, particularly when the landmark distances from the camera are large. So he suggests solving for both the rotation and translation matrices simultaneously to achieve better noise immunity. Haralick et al [22] summarize the various cases of the position estimation problem using point data. They argue for robust estimation procedures in machine vision. Their thesis is that the least square estimators can be made robust under blunders by converting the estimation procedure to an iterative, reweighted least squares, where the weight for each observation depends on the residual error and its redundancy number.

Most of the landmark-based approaches considered above suffer from the disadvantages of : 1) assuming the availability of landmarks in the scene around the robot; 2) depending on the visibility and the ability to recognize these landmarks from the image and to estimate the range/attitude to them from the current location; 3) requiring an approximate location to start with to check for the landmarks; and 4) needing a database of landmarks in the area to look for in the image.

2.2 Trajectory integration and dead reckoning

Another class of techniques estimate the position and pose of a mobile robot by integrating over its trajectory and dead reckoning i.e., the robot maintains an estimate of its current location and pose at all times and, as it moves along, updates the estimate by dead reckoning using sensors to sense the environment local to the new position. Features are detected from the sensory observations in one of the positions of the robot and are used to form the world model. As the robot moves, these features are again detected, and correspondence is established between the new and the old features. Usually the motion of the robot is known to a certain degree of accuracy as given by its positional sensors. These motion estimates are then used to predict the occurrence of the new positions for the features in the world model. The prediction is then used as an aid to limit the search space and to establish a correspondence between the detected features and those already in the current world model. After the positional sensors are used to establish correspondence, the motion parameters of the robot between the old and the new positions can be solved for explicitly. The solution provides a much more accurate positional estimate of the robot. The loop then continues, and the world model is continuously updated. If new features, which do not exist in the current world model, are detected, a mechanism to consistently update the world model is also provided. The type of sensing used is typically stereo triangulation or other visual sensing [41,42,37,46]. Crowley [10] uses a ring of 24 sonar sensors for a similar paradigm.

Moravec's *Cart* [41] was one of the first attempts at autonomous mobile robot navigation using a stereo pair of cameras. He defined an *interest operator* to locate the features in a given image. A coarse to fine correlation strategy was used to establish correspondence between the features selected by the interest operator between different frames. Matthies and Shafer [37] argue that a 3-D Gaussian distribution is a much more effective way to explicitly deal with stereo triangulation errors and errors due to the images' limited resolution. They detail a method to estimate the 3-D Gaussian error distribution parameters (mean and co-variance) from the stereo pair of images.They then present a method to consistently update the robot position, explicitly taking into account the Gaussian error distribution of the feature points and motion parameters and their error co-variances. A Kalman filter approach is used to recursively update the robot position from the detected features and the previously maintained world model. They assume the correspondence problem to be solved, and show by simulation data and experimental results that the Gaussian error model results in a more accurate stereo navigational paradigm.

Faugeras and Ayache [17,16,3] also address the problem of autonomous navigation. Their approach is more rigorous and mathematical. They use trinocular stereo to detect object features and, the primitives they use for the 3-D world model are line segments. They propose a paradigm to combine coherently visual information obtained at different places to build a 3-D representation of the world. The end result is a representation of the environment by a number of *uncertain* 3-D line segments, attached to co-ordinate frames and related by uncertain rigid motion. The measurements are combined in the presence of these uncertainties by using the *Extended Kalman Filtering* technique. Crowley [10] has a similar approach to Ayache and Faugeras [17]; he also uses a line segment based representation of the free space using Extended Kalman filtering techniques for dealing with error covariances. However, he uses a circular ring of 24 Polaroid ultra-sonic sensors, while Faugeras and Ayache use trinocular stereo. Chatila and Laumond [8] present a world modeling and position referencing system on their mobile robot HILARE. They take a multisensor approach of using a laser range finder for measuring depth and optical shaft encoders on the drive wheel axis for the trajectory integration.

In these type of techniques the position estimation strategy depends the representation used for the sensory observations and the environment. Miller [40] presents a surface representation for real world indoor robots, equipped with a ranging device like a sonar and the robot odometry as sensors. He assumes the world to consist of a flat, open plane on which walls and obstacles are placed. Since the robot is limited to motion on the plane of the floor, the projection of walls and obstacles on the plane of the floor captures all the relevant world information. The basic unit of the spatial representation system is the *map*, composed of linked *regions*. Regions have a local coordinate frame. Walls and obstacles are themselves represented by line segments, whose end point positions are designated by coordinates in the frame of the region. The borders of the regions are marked with labels that specify the adjoining regions. Regions can be of four types, 0-F, 1-F, 2-F and 3-F, since a floor dwelling mobile robot has 3 degrees of freedom, 2 translational (x and y), and one rotational (orientation θ). A type designation of j-F means that a sensor (here a sonar range sensor) can be used to eliminate j degrees of freedom. Having set the mapping scheme, Miller then presents methods for positional estimation as a heuristic search paradigm. The type of the region in which the robot is in determines the amount of positional information that can be calculated. If the robot is in a 0-F region, then the only positional information available would be extrapolations from the last known position, based on the robot's ability to do dead reckoning. If the robot is known to be in a region that is 1-F or greater, then positional information can be found by taking several sensor readings and conducting a heuristic search over the tree of possible matches between the observations and the edges in the map. Once a consistent matching between the observations and the map edges is found, a simple geometric construction is performed to pinpoint the robot's location and orientation.

Moravec and Elfes [14,43] use a grid based representation called "Certainty Grids" for mapping the environment a mobile robot will inhabit. The basic idea is to represent the floor by a rectangular grid and to store the information about the occupancy of different portions of the floor on this grid as probability distributions using a sonar range sensor input. The authors also develop and present a fast algorithm for relating two maps of the same area to determine relative displacement, angle, and goodness of the match. These can then be used to estimate the position and pose of the robot.

2.3 Techniques using a Standard Pattern

One other method of estimating the position and pose of the mobile robot accurately is to place standard patterns in known locations in the environment. Once the robot images these patterns and detects them, the position of the robot can be estimated from the known location of the pattern and its geometry. The pattern itself is designed to yield a wealth of geometric information when transformed under the perspective projection. Ambiguous interpretations are avoided, and a minimum of *a priori* knowledge about the camera is desirable. These methods are particularly useful in those applications where a high degree of accuracy in the positioning of the robot is required only after it is near a particular work station. Simple trajectory integration systems could be used to locate the robot near the work station, and, then, by identifying the mark (standard pattern) located near the work station, the robot can be positioned more accurately. Different researchers [19,9,36,12,26] have used different kinds of patterns or marks, and the geometry of the method and the associated techniques for position estimation vary accordingly.

2.4 Model based approaches

Some researchers consider the problem of the position estimation of a mobile robot when *a priori* information is available about the environment in which the robot is to navigate. This could be provided in terms of a CAD model of the building (or a floor map etc.) in the case of an indoor mobile robot, or a Digital Elevation Map (DEM) in the case of an outdoor robot. In these cases, the positional estimation techniques used take on a different flavor. The basic idea is, of course, to sense the environment using on board sensors on the robot and to try to match these sensory observations to the preloaded world model to arrive at an estimate of the position and pose of the robot with a reduced uncertainty.

One problem with such an approach is that the sensor readings and the world model could be in different forms. For instance, given a CAD model of the building and a visual camera, the problem is to match the 3-D descriptions in the CAD model to the 2-D visual images. This is the problem addressed by Kak et al. [27] in their work. They present PSEIKI, a system that uses evidential reasoning in a hierarchical framework for image interpretation. They discuss how the PSEIKI system can be used for self-location by a mobile robot and how their approach is used with navigational system of the autonomous mobile robot PETER. The robot's position encoders are used to maintain an approximate estimate of its position and heading at each point. However, to account for errors in the quantization effects of the encoders and slippage of the wheels, a visual sensor, in conjunction with a CAD model of the building is used to derive a more accurate estimate of the position and pose of the robot. The basic idea is that the approximate position from the encoders is used to generate, from the CAD model, an estimated visual scene that would be seen. This scene is then matched against the actual scene viewed by the camera. Once the matches are established between the features of the two images (expected and actual), the position of the robot can be estimated with a reduced uncertainty. Tsubouchi and Yuta [61] discuss the position estimation techniques, used in their YAMABICO robot, which use a color camera and a map of the building in which the robot is to navigate. The authors propose a vision system using image and map information for a mobile robot with consideration of real time requirements. This system consists of three operations. The first operation is the abstraction of a specified image from a tv camera. The image is processed and highly abstracted information called the *real perspective information* is generated. The second operation is the generation of the *estimated perspective information* by co-ordinate transformation and map information, using information about the robot's position and direction. The third operation is the establishment of the correspondence between the two perspectives. The authors use color images in their real perspective views. They argue for color images saying that they are invariant under lightness and shadow. From the color images the authors extract regions of *similar* color and fit trapezoids to these regions. From the map information trapezoids are also extracted and, in the matching process, these trapezoids from the two sources are used as matching primitives.

3 Position estimation in a mountainous terrain

In this section a solution to the positional estimation problem is discussed for an autonomous land vehicle navigating in an unstructured mountainous terrain. A Digital Elevation Map (DEM) of the area in which the robot is to navigate is assumed to be given. More details are given in [54] and [55]. The robot is also assumed to be equipped with a camera that can be panned and tilted, and a device to measure the elevation of the robot above the datum. The robot is also assumed to have a compass and knowledge of the direction in which the DEM is gridded. No recognizable landmarks are assumed to be present in the environment in which the robot navigates. The DEM is a 3-D database. It records the terrain elevations for ground positions at regularly spaced intervals. The images recorded by the camera are 2-D intensity images. The problem is to match the 2-D images to the 3-D DEM. The solution presented makes use of the DEM information and structures the problem as a heuristic search in the DEM for the possible robot location. The shape and position of the horizon line in the image plane and the known camera geometry of the perspective projection are used as parameters to search the DEM for the possible camera(robot) location. Heuristics based upon from geometric constraints are used to reduce the search space. The algorithm makes use of the computer graphics rendering techniques to generate synthetic image data from the DEM to generate a solution to the problem. More importantly, the images generated from the DEM are used to disambiguate the robot's location from the various possible ones returned by the search process.

3.1 The Approach

The basic approach is to use the height and the exact shape of the horizon line (HL) and the known camera geometry of the perspective projection to search in the DEM for the possible camera location. The actual search is a two step process. The first step is a coarser search which reduces the possible locations to a smaller set using the height of the HL in the image plane in different directions; and the second step refines this estimate using the exact shape of the HL.

From the current robot position, images are taken in the four geographic directions: N,S,E and W. In generating the four geographic views, the tilt angle of the camera is adjusted until the horizon line is clearly visible in the image. This tilt angle $\phi_i(i = N, S, E, W)$ is then measured. The approximate height H of the camera above the datum (with respect to which the DEM is specified) is assumed to be known. The height of HL at the center of the image plane in each of these four images is measured.This can be done using image processing techniques. Let this height be $h_i(i = N, S, E, W)$. The reason for using the height of the HL at the center of the image plane is that the DEM is assumed to be gridded in the same directions as those from which the images are taken. So the points that project onto the HL at the center all lie along the same grid line in the DEM. Using the approximate height of the camera H, the tilt angle ϕ_i and the HL height h_i in one of the directions e.g., north, the DEM is searched for the possible camera locations. That is, a camera location is hypothesized at each of the DEM grid points and the height of the HL h_i is back-projected onto the DEM using the camera geometry to see if any elevation points that can project to this height. If any such points exist, the camera location is marked as a possible candidate. Equation 3.1 below gives the estimated height Z_{est} for a distance

x from the hypothesized camera location.

$$Z_{est} = H + x\sin\phi + x \cdot \frac{\tan\theta}{\cos\phi} \cdot \frac{h}{I_{max}} \cdot \frac{\cos[\tan^{-1}(\frac{h}{I_{max}} \cdot \tan\theta)]}{\cos[\tan^{-1}(\frac{h}{I_{max}} \cdot \tan\theta + \phi)]} \tag{3.1}$$

However, if any grid point is of a height larger than that estimated by the back-projection, then all the camera locations between the current location and this *tall* point are marked as impossible positions and discarded. This heuristic reduces the search space significantly. Similar heuristics are also used to prune the search space by the camera height H. The results of the search process by using one of the images is thus a sparser set of possible camera locations . These are then considered as the possible set for the next search, which searches among this set with the geometric constraints extracted from the image along another direction. This process is continued by successively applying the constraints in all the four directions and the search refines the possible locations to a small set usually clustered around the actual location. For a mountainous area, the heuristics used help to prune the search space significantly, since a large number of points are discarded as being impossible camera locations.

3.2 The Derivation of Z_{est}

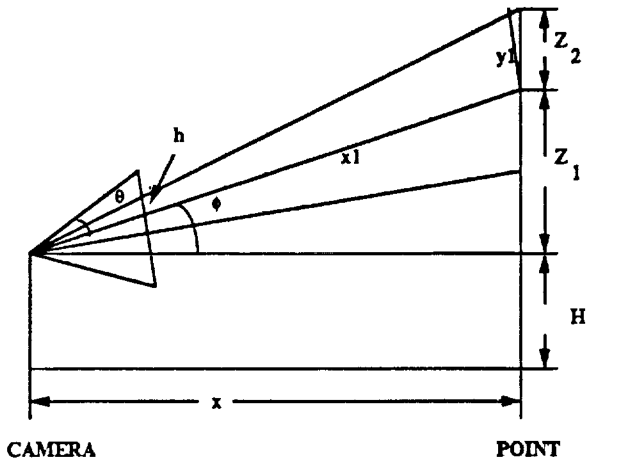
An expression for the estimated elevation Z_{est} at a point in the elevation map, for a hypothesized camera location, and for a known imaging geometry shown in Figure 3.1 is derived below:

$$\begin{aligned}
I_{max} &= \frac{Image\ Plane\ Height}{2} \\
f &= Focal\ length\ of\ the\ camera \\
\theta &= Perspective\ angle \\
h &= HLC\ height\ at\ the\ center\ of\ the\ image \\
\phi &= Camera\ tilt\ angle \\
H &= Height\ of\ the\ camera\ above\ the\ datum \\
x &= Distance\ between\ CAMERA\ and\ POINT
\end{aligned}$$

$$Z_{est} = H + Z_1 + Z_2 \tag{3.2}$$

$$Z_1 = x\sin\phi \tag{3.3}$$

$$\tan\theta = \frac{I_{max}}{f}$$



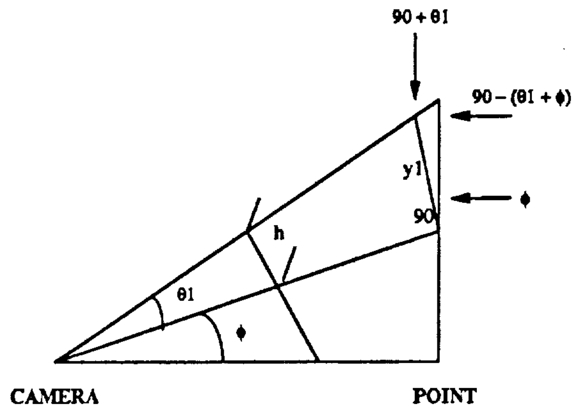


Figure 3.1: The geometry of the projection

$$\tan\theta_1 = \frac{h}{I_{max}} \cdot \tan\theta \; and \; x_1 = \frac{x}{\cos\phi}$$

$$y_1 = x_1 \cdot \tan\theta_1 = \frac{x}{\cos\phi} \cdot \frac{h}{I_{max}} \cdot \tan\theta$$

$$\begin{aligned} Z_2 &= y_1 \cdot \frac{\sin(90+\theta_1)}{\sin(90-(\theta_1+\phi))} = y_1 \cdot \frac{\cos(\theta_1)}{\cos(\theta_1+\phi)} \\ &= \frac{x}{\cos\phi} \cdot \frac{h}{I_{max}} \cdot \tan\theta \cdot \frac{\cos[\tan^{-1}(\frac{h}{I_{max}} \cdot \tan\theta)]}{\cos[\tan^{-1}(\frac{h}{I_{max}} \cdot \tan\theta + \phi)]} \end{aligned} \qquad (3.4)$$

So finally from equations (3.2), (3.3) and (3.4)

$$Z_{est} = H + x\sin\phi + x \cdot \frac{\tan\theta}{\cos\phi} \cdot \frac{h}{I_{max}} \cdot \frac{\cos[\tan^{-1}(\frac{h}{I_{max}} \cdot \tan\theta)]}{\cos[\tan^{-1}(\frac{h}{I_{max}} \cdot \tan\theta + \phi)]}$$

The search algorithm

The actual algorithm *search* algorithm is presented below. The DEM is referred to as a 2-D grid along x and y with elevation values defined at each of the grid points. The algorithm has two pointers, one pointing to the current hypothesized camera location, CAMERA, and the other pointing to a candidate grid point, POINT. The idea is to assume that the CAMERA is at a certain point (x,y). Search along this y line to see if any POINT exists which will project at the desired height h_i onto the HL. If so mark this CAMERA as a possible camera position. Repeat the procedure for all the CAMERA positions along this y line and then for all y lines.

Let CAMERA.X and CAMERA.Y denote the x and y grid values of the location CAMERA and let POINT.X and POINT.Y denote the values of the candidate point. Consider the search along one of the directions, say north, (see Figure 3.2). CAMERA.X is initialized to XMAX; POINT.X to XMIN; and CAMERA.Y and POINT.Y to YMIN. Using the known approximate camera height H and the tilt angle ϕ_N, the height of the HL, h_N, is back projected to the POINT using the geometry shown in Figure 3.1. The elevation Z_{est} necessary for this POINT to project on to the HL is estimated using Equation 3.1. The actual height Z_{actual} (POINT.Z) is extracted from the DEM database. If Z_{actual} is equal to Z_{est}, then CAMERA is saved as a possible camera position. If Z_{actual} is less than Z_{est}, then the POINT is updated to a position closer to the CAMERA(in this case, POINT.X is incremented).This process is continued until POINT coincides with CAMERA. Then POINT is re-initialized, and CAMERA is updated by moving it closer to POINT (in this case, CAMERA.X is decremented). If at some stage Z_{actual} is greater than Z_{est}, then CAMERA is not a possible location since if it were the elevation at POINT, Z_{actual} would project a height higher than h_N. It can also be said that none of the

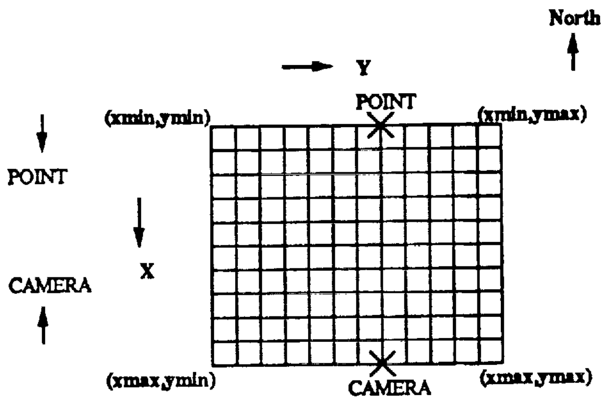


Figure 3.2: The search in the north direction

points between CAMERA and POINT are possible camera locations since at all these, the same thing would have happened. This heuristic proves very useful in pruning the search space by a large extent for mountainous terrains with many altitude variations like those considered in the illustrations. The search process is then repeated along all the y lines in this direction, i.e., for CAMERA.Y and POINT.Y ranging from YMIN to YMAX. The possible camera positions returned by this search process are then considered as inputs for the next search, which searches among this set with geometric constraints extracted from the image along another direction. The process is continued successively by applying the constraints in all the four directions and the search refines the possible locations to a small set usually clustered along the actual location. (see Figure 3.10).

Another heuristic used to reduce the search space is to use the approximate height H of the camera. Only those points of the DEM are considered as possible camera locations where the elevation Z_{actual} lies close to H, i.e., $H - \delta_H < Z_{actual} < H + \delta_H$, where δ_H is the resolution of the sensor used to measure the camera's height.

The first stage of the search algorithm is formalized below for a search in the north direction. It is similar for the other directions except that the search is carried out only in the reduced set of candidate camera positions returned by the previous search process. The direction and hence the limits of the search vary depending on the direction in which the image is taken.

3.3 The Algorithm Search

Stage 1 of the search process for searching in the north direction is described below.

Input : The DEM, h_N, ϕ_N and H.

Output : A list of possible camera locations on the stack S.

```
1 Make stack S empty.
2 POINT.Y = CAMERA.Y = YMIN.
  for (CAMERA.Y < YMAX) do
    3 POINT.X = XMIN.
      CAMERA.X = XMAX.
      for (POINT.X < CAMERA.X) do
        4 $Z_{est}$ = get-z-estimate($h_N$,$\phi_N$,
              H,CAMERA,POINT) /* computed using equation 1 */
          $Z_{actual}$ = POINT.Z. /* retrieved from the DEM */
        5 if ($Z_{est}$ = $Z_{actual}$) Push(S,CAMERA). /* save current camera location */
         else if($Z_{actual}$ > $Z_{est}$)
          Pop(S). /* discard the previously stored location and update the camera position
          */
          CAMERA.X = POINT.X.
          POINT.X = XMIN - 1.
        6 increment POINT.X
         endo
    7 decrement CAMERA.X
     endo
```

End Algorithm Search.

Stage 2 of the search process is used to further isolate the exact location from the possible ones returned by the stage 1 search. Each of these locations is considered as a possible candidate and the image that would be seen if the camera were located at that location is generated from the DEM using computer graphics rendering techniques. The HLs from these images are then extracted and correlated against the actual image HL to arrive at a measure of their disparity. The camera location corresponding to the location with the lowest disparity is considered as the best estimate of the location. The exact location can be further isolated by generating the images from the points neighboring to the estimated location and checking to obtain a zero error measure.

3.4 The error analysis

The search algorithm depends upon the errors in three parameters: the error in height H of the camera, the HL height h_i in the images plane, and the tilt angle of the camera ϕ_i. These three parameters affect the estimated elevation, Z_{est}, at a particular candidate point (POINT) for a given hypothesized camera position (CAMERA). The errors in Z_{est} directly reflect as errors in the positional estimation, since in step 5 of the algorithm, only when Z_{est} is equal to Z_{actual} is the current camera considered location as a possible candidate. So in the rest of the section, the

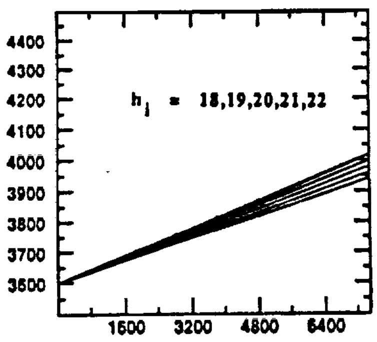


Figure 3.3: Z_{est} vs x for different h_i

effects of errors in these three parameters on the computation of Z_{est} are analyzed and then the measures taken in the algorithm to account for the worst case errors are discussed.

The equation relating Z_{est} to h_i,ϕ_i and H, is given by (1) above. For a given size of the image $(2I_{max}, 2I_{max})$ and angle of perspective projection, θ, $\frac{\tan\theta}{I_{max}}$ is a constant. So the variables are H, h_i, and ϕ_i.

Expanding using Taylor's series and neglecting the higher order terms,

$$\Delta Z_{est} \approx \Delta H \cdot \frac{\partial Z_{est}}{\partial H} + \Delta h_i \cdot \frac{\partial Z_{est}}{\partial h_i} + \Delta \phi_i \cdot \frac{\partial Z_{est}}{\partial \phi_i} \tag{3.5}$$

Errors in H

Equation 2 shows that the error in H is directly reflected as an error in Z_{est}. So to account for these a worst case error in H is estimated from the sensor used to measure H. Let this be δH. In the search algorithm, this can be accounted for by considering the estimated elevation at the candidate point to be acceptable if $Z_{est} - \delta H < Z_{actual} < Z_{est} + \delta H$.

Errors in h_i

The errors in h_i are mainly due to image quantization error and errors in the edge detector used to extract the horizon line. Since x, the distance between POINT and CAMERA, occurs as a multiplicative parameter in the second and third terms of Equation 1, the errors in h_i are magnified by this and, hence, vary depending on x. The worst error compensation should take this into account. Figure 3.3 shows a typical plot of Z_{est} versus x for different Δh_i. As can be seen with increasing distance, the error in Z_{est} grows almost linearly. Figure 3.4 shows a plot of ΔZ_{est} versus Δh_i for different ϕ_is and for fixed x and H. Ideally, $\Delta Z_{est} = 0$ should occur at Δh_i = 0. However, due to image quantization error, this occurs for $\Delta h_i \in (-1, +1)$, as can be seen for different values of ϕ_i. One way to account for these errors in h_i is to back-project a band of h_i values , $(h_i - \Delta h_i, h_i + \Delta h_i)$, instead of a single h_i in the estimation of Z_{est}. Thus, for each

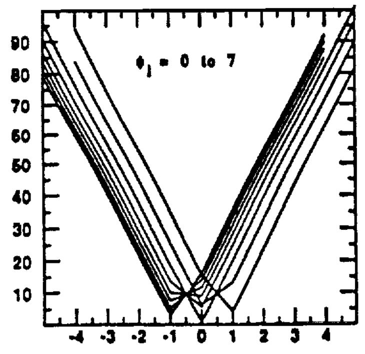


Figure 3.4: Z_{est} vs Δh for different ϕ_i

CAMERA and POINT locations a range of acceptable elevations ($Z_{est} + \Delta Z_{est}, Z_{est} - \Delta Z_{est}$) are obtained and CAMERA is considered as a possible camera location only if Z_{actual} at this POINT lies within this range. This way the effect of x on ΔZ_{est} is also implicitly accounted for.

<u>Errors in ϕ_i</u>

Similar to h_i, the effects of errors in ϕ_i are also magnified by x and, hence depend on the distance between CAMERA and POINT. The algorithm is very sensitive to effects in errors in ϕ_i. As Figure 3.5 shows, an error of 0.5 degrees in ϕ_i causes an error of about 50 meters in Z_{est} for the given x, H, and h_i. So, ϕ_i has to be measured accurately, or at least a good worst case estimate is required so that as before, Z_{est} can be compensated for. In the actual implementation of the algorithm, errors in ϕ_i are accounted for by considering the effect of these errors as errors in h_i. That is, the band $h_i \pm \Delta h_i$ is made wider to account for $\Delta\phi_i$. Hence, the range of acceptable Z_{est} values is increased. In the illustrations considered, it is found that a $\Delta h_i = 3$ pixels accounts for 0.5 degrees error in ϕ_i and for a ± 1 pixel error in h_i.

Another observation from Figure 3.5 is that $\Delta Z_{est} = 0$ does not always occur at $\Delta\phi_i = 0$. This is due to the roundoff errors in the computation of Z_{est} from ϕ_i,H,h_i, and x, which is a reasonably complicated trigonometric expression.

3.5 Illustrations

The algorithm is tested on a digital elevation map (DEM) of a mountainous area in Colorado obtained from the United States Geological Survey (USGS). The elevation data is a uniform square grid of 30m resolution and has 359 x 457 grid points. It covers an area of 148 square kms. Synthetic images for an assumed camera location are generated using the AT & T Pixel machine from the DEM. The elevation data is tessellated into polygons and surface normals are calculated at each of the elevation points using the four neighboring points. A light source position and direction are assumed and for a given camera location and perspective geometry, Gouraud shaded

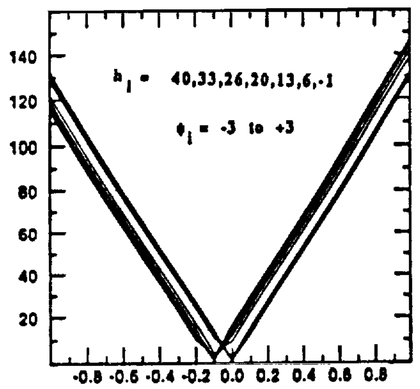


Figure 3.5: Z_{est} vs $\Delta\phi$ for different h_i

polygons are drawn and projected on to the image plane to generate perspective views of the mountain range. Figure 3.6 - Figure 3.12 illustrate a typical run of the algorithm. Figure 3.6 shows a typical picture used to test the algorithm. Figure 3.7 shows the HL extracted from this using a gradient operator from this image. Figure 3.9 shows the results of the first stage of the search process. Of the possible 164063 (359 x 457) possible locations, stage 1 of the search process using h_N and the associated camera geometry returns 473 points as possible camera locations. These successively drop in the consecutive searches using h_i in the other directions. Ultimately, there are about 8 possible camera locations as shown in Figure 3.10. Figure 3.11 and Figure 3.12 show the results of the stage 2 of the search process. The figures show the HLs of some of the images generated in the north direction for the candidate camera locations returned by the first stage of the search process. A measure of the difference between these and the actual image HL in this direction is also shown. The measure used is a mean square error measure between the two contours. The candidate with the lowest error in this case 48.3442 units is then considered as the estimate of the camera location. The actual position is (200.1,100.5) and the estimated location is (201,102). The four neighbors of this point are then considered as candidates; images are also generated in the north direction for these positions. The HLs are then extracted and correlated against the original HL, and the best estimate of the location is isolated as the one with the least mean square error.

Table 3.1 shows the results of running the algorithm at various locations on the DEM. As can be seen, in most cases the estimated position is quite close to the actual location. Zero mean Gaussian noise with a standard deviation of 5 is added to the horizon lines. This is used to represent the noise in the image formation process and the noise in the detection of the HL. Since the HLs are smoothed by a Gaussian filter and since actually a band of h_i values is used in the back projection, the effects of the additive Gaussian noise are not felt much. As the test runs illustrate, the effect of this noise is to slightly shift the estimated position in some cases. However, the estimated position is still quite close to the actual location is all cases. So, the method is quite robust to errors in the imaging process.

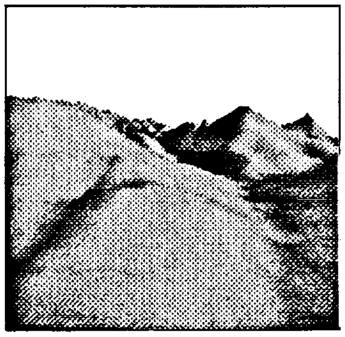


Figure 3.6: A typical view

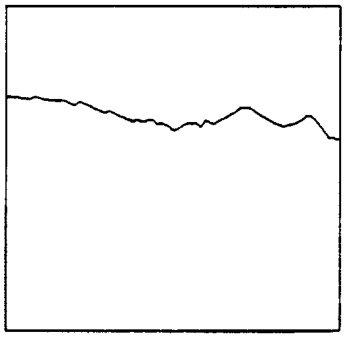


Figure 3.7: The extracted HLC

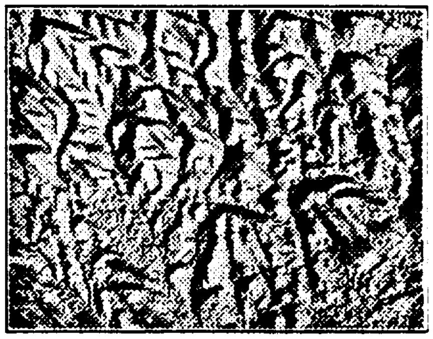


Figure 3.8: An aerial view of the DEM

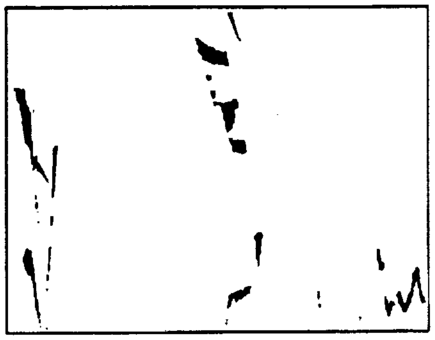


Figure 3.9: The possible camera locations after searching in one direction (aerial view)

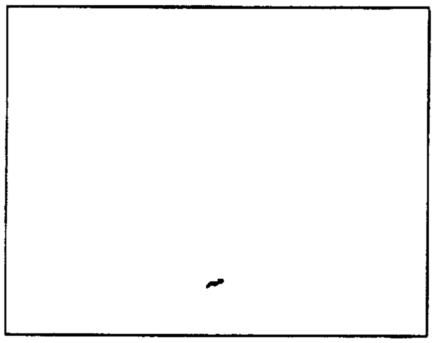


Figure 3.10: The possible camera locations after searching in all four directions (aerial view)

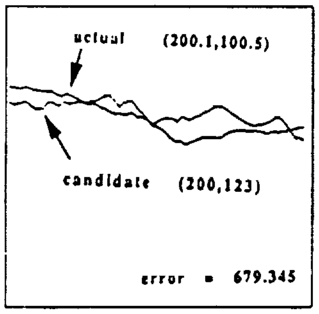


Figure 3.11: The disparity between HLCs

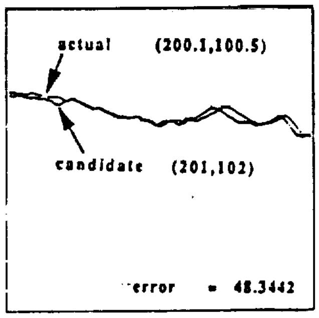


Figure 3.12: The disparity between HLCs

Table 3.1: Search Results

(1)	(2)	(3)	(4)	(5)
(150.2,100.8)	342	7	(151,101)	(151,102)
(150.0,200.0)	126	3	(151,200)	(151,201)
(150.3,300.6)	227	8	(150,301)	(152,303)
(200.1,100.5)	473	8	(201,102)	(202,100)
(200.2,200.2)	889	26	(201,201)	(200,202)
(200.2,300.8)	334	9	(201,300)	(202,302)
(300.0,100.0)	334	4	(301,100)	(301,101)
(300.5,200.8)	273	17	(301,200)	(302,202)
(300.2,300.8)	254	10	(300,300)	(301,302)

4 Positional Estimation in a man-made environment

In this section we address the navigational issues of a mobile robot navigating in an urban environment. The environment of the robot is assumed to consist of polyhedral buildings. The 3-D descriptions of the roof tops of the buildings are assumed to be given. Such descriptions could be obtained from the architectural plans of the buildings or from a stereo pair of aerial images. The robot is assumed to be equipped with a single visual camera that can be panned and tilted. The robot is *not* assumed to have the ability to recognize any landmarks and that a good estimate of the position and pose of the robot is *not* available. It is also shown that if such a capability exists it only helps in aiding the position estimation process presented. In this research a new method of representing the free space of the robot is presented. The free space is partitioned into distinct, non-overlapping regions called the *Edge Visibility Regions* (EVRs) using the stored model description of the environment. These regions have the property of capturing the geometric relations between the features (edges) in the model with respect to their visibility from different positions in the viewing plane. The idea is that the the problem of establishing the correspondence between the model features and the image features is alleviated by using these EVRs as a guide. A procedure for partitioning the free space into EVRs is described in this section. The use of this EVR representation of the free space in position estimation and path-planning are discussed in the future work. A derivation for an upper bound on the maximum number of EVRs that would be generated for a given model description in also given. Details are given in [56] and [58].

4.1 Edge Visibility Regions

This subsection presents an algorithm to divide the free space of the robot into the EVRs. The environment of the robot is assumed to consist of polyhedral buildings and the 3-D descriptions of the rooftops, of which are assumed to be given as the world model.

Let OXYZ be the world co-ordinate system (WCS) in which the world model is described. The robot is assumed to be equipped with a camera that can be panned and tilted with no roll. Let O'X'Y'Z' be the robot (camera) centered co-ordinate system (CCS). See Figure 4.1. In order to estimate the position and pose of the robot in the world co-ordinate system, the transformation matrix T that transforms WCS into CCS needs to be solved. In general, this matrix has six degrees of freedom: three rotational ψ, ϕ, θ and three translational X, Y, Z, Hence T is a function of $(X, Y, Z, \psi, \phi, \theta)$. However, in the present problem since it is assumed that 1) the roll angle, $\psi = 0$, 2) the tilt angle ϕ is measurable (a constant), and 3) the robot is on the ground plane $Z = 0$, there are only three degrees of freedom for this matrix: two translational, X and Y and one rotational, θ. So, the robot navigates in the XY-plane of the world co-ordinate system and has an orientation, θ, which is a negative rotation about the Z-axis. So $T(X, Y, \theta)$ transforms the WCS into the CCS.

Initially, we consider the restrictive case of all the buildings as having flat, convex rooftops, and of being the same height. We then develop an algorithm to generate the EVRs and their associated VLs. The extensions of this algorithm to the general case are then discussed. In the former case, only the projections of the buildings onto the XY-plane need to be considered in

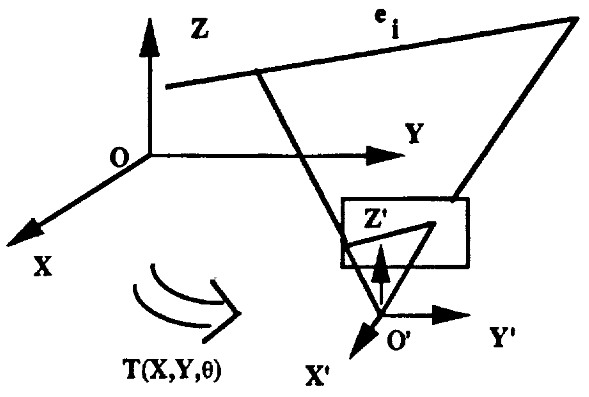


Figure 4.1: The world and robot co-ordinate systems

forming the EVRs and the VLs, since the tilt angle ϕ is assumed to be measurable.

The problem can now be formally stated as follows:

To partition a plane containing m, n-sided non-intersecting convex polygons into distinct, non-overlapping regions called the *Edge Visibility Regions*(EVRs). These EVRs are characterized by the following properties:

- associated with each EVR is a list of edges of the polygons that are *visible* in that region called as the *visibility list*(VL).
- no two adjacent EVRs have the same VL.

An edge l of a polygon is considered to be visible from a point p if there is at least one point $q \in l$ such that the line segment pq is not intersected by any other edge of any other polygon.

Note that this problem is similar in spirit to the problem of computing the *weak visibility polygon* considered in computational geometry [48,59]. Another problem bearing similarities to this problem is that of forming the aspect graph of a polyhedral object for object recognition [25,51]. However, the above methods concern mainly a single polyhedral object and divide the 3-D view space into regions based on the visibility of faces, edges, vertices, etc. of the object. The present method considers multiple objects and their occlusions. The objects considered are non-overlapping, 2-D, convex polygons, and the viewing space is a 2-D plane.

The algorithm *partition* that divides the XY plane into the desired EVRs, along with their associated VLs, is presented below. The algorithm uses three subprocesses called *split*, *project*, and *merge*. The basic idea of the algorithm is to start with the entire XY-plane as one EVR with a NULL visibility list. Each polygon is considered in turn by extending each of its edges, and the EVRs that are intersected are divided into two new ones. The new EVRs then replace the old one and the VLs of the new EVRs are updated to account for the visibility of this edge by considering the edge to be visible in one half-plane, say left of the edge, and invisible in the other. This is

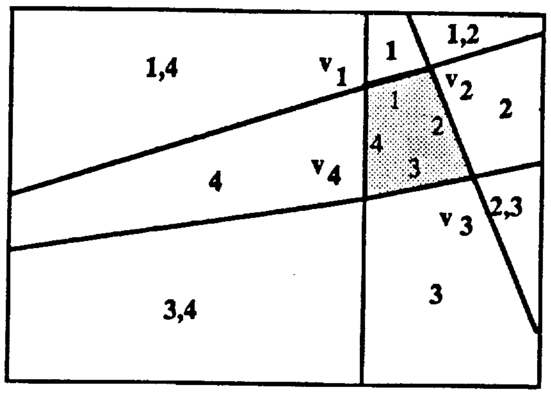


Figure 4.2: The EVRs after Split

handled by the *split* process. For each new polygon considered, the mutual occlusion of the edges of this polygon with the other existing polygons is handled by forming the *shadow region* of these edges on the other existing polygons. This is handled by the *project process.* Finally the *merge* process concatenates all the adjacent EVRs with identical VLs into one EVR. After partitioning the XY-plane into EVRs, for each model edge in the VL of an EVR the interval of orientations of the robot for which this model edge is visible in the EVR is also computed and stored. An efficient method to compute this interval is also described.

Split

Given a 2-D convex polygon in a plane and a set of existing EVRs, this process is used to update the EVR list to account for the visibility of the edges of the polygon. Consider an n-sided convex polygon P in the viewing plane defined as a collection of n vertices, $v_1, v_2, ...v_n$, and n edges, $v_1v_2, v_2v_3, ...v_{n-1}v_n, v_nv_1$, such that no pair of non-consecutive edges shares a point. Then,

1. extend each of the edges of P, $v_1v_2,...,v_nv_1$, in order and for each of these lines, and
2. check if this line cuts an existing EVR in the current EVR list. If it does, split the EVR into two along this line, copy the VL of the EVR into the two new EVRs, update the EVR list by replacing the old EVR by the two newly formed ones, and update the VL of the region to the left of this line by adding this edge, v_iv_{i+1}, to its VL.

The above procedure also returns the polygon itself as one of the EVRs. This is then removed from the EVR list by checking the list to see if any of the EVRs are the intersecting the polygon. Figure 4.2 shows the EVRs and their VLs after extending all the edges of P.

Project

This section discusses a process called *project*, which is used to account for the mutual occlusions between lines and polygons that affect their visibility from different locations in the

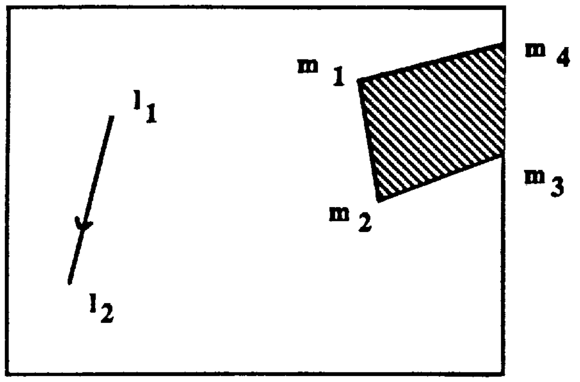


Figure 4.3: The shadow region for two lines

viewing plane. The process is first described for two line segments, and then for a line segment and a polygon.

Case 1: Consider the case of two lines, l_1l_2 and m_1m_2, in a plane as shown in Figure 4.3. We now need to find the region in the plane where l_1l_2 is not visible due to occlusion from m_1m_2. Note that m_1m_2 lies to the left of (the visible side of) the directed line segment l_1l_2. Let us refer to this region as the *shadow region* of m_1m_2 due to l_1l_2. One way to find this shadow region is to extend the line l_1m_1 and then to find the intersection of this line with the viewing plane, as m_4; similarly, extend l_2m_2 and find m_3. Now the region $m_1m_2m_3m_4$ is the desired shadow region. The lines m_1m_4 and m_2m_3 are referred to as the *shadow lines* of m_1m_2 due to l_1l_2.

Case 2: Consider the case of a line segment l_1l_2 and a convex polygon P in a plane. We need to find the shadow region of P due to the line l_1l_2 when P lies on the visible side of l_1l_2. This region is the union of the shadow regions of each edge v_iv_{i+1} of P due to l_1l_2. An easier way to compute this shadow region is to find the two vertices of P, x and y, such that the shadow region of the line segment XY due to l_1l_2 is the desired shadow region of P due to l_1l_2. The two vertices x and y are referred to as the *shadow vertices* of P due to l_1l_2. There are two sub cases to be considered:

- subcase 1 : In this subcase the entire polygon lies to the left of the line l_1l_2. That is, the extension of the line does not cut any of the edges of the polygon. Figure 4.4 shows this.
- subcase 2 : In this subcase, only part of the polygon lies to the left of the line. That is, the extension of the line cuts some of the edges of the polygon. Figure 4.5 shows this.

The determination of the shadow vertices is different in each of these subcases. So, the process *project* detects these subcases and uses a different method in each. For subcase 1, shown in Figure 4.4, the shadow vertices are found as follows: the vertex of P corresponding to the max of the angle $\angle l_2l_1v_i$, i = 1,2,....n, gives the shadow vertex x, and similarly the vertex of P corresponding to the maximum of $\angle l_1l_2v_i$, i = 1,2,....n, gives the shadow vertex y.

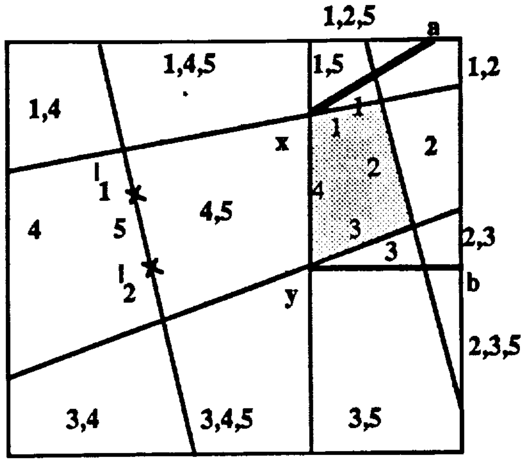


Figure 4.4: The shadow regions for subcase 1

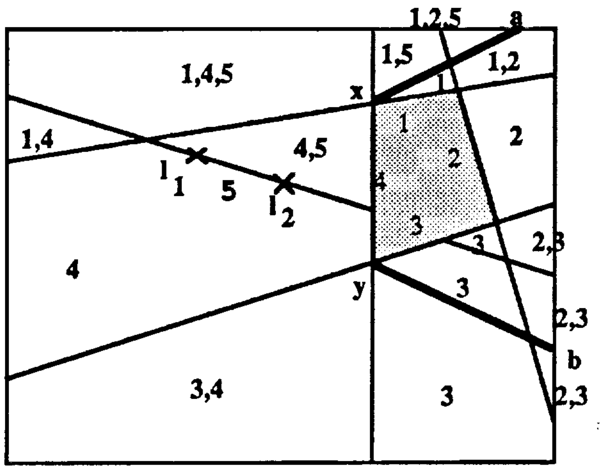


Figure 4.5: The shadow regions for subcase 2

For subcase 2, shown in Figure 4.5, it is slightly more involved to find x and y. First, the convex hull of the points $v_1, v_2, v_3, ...v_n$ and l_2 is formed. Then the two vertices of the convex hull adjacent to l_2, v_1 & v_3, are considered. Of these, the vertex to the left of the line $l_1 l_2$ is considered to be the shadow vertex y and the other to be x. Hence, $x = v_1$ and $y = v_4$. To find the shadow region, the line $l_1 x$ is extended and its intersection with the plane is found as a; similarly intersection for the line $l_2 y$ is found as b. Thus, the shadow lines are xa and yb and the shadow region is $xaby$. Once the shadow region is formed, the edge $l_1 l_2$, numbered as 5 in the Figure 4.5, is marked as invisible and, hence, removed from the VLs of all the EVRs that lie inside this shadow region. Note that in subcase 2, all of the shadow region does not lie in the visible side of the line $l_1 l_2$. As a result we have adjacent EVRs with identical VLs. However, the *Merge* process, to be discussed next, accounts for this.

Merge

Given an EVR list and the associated VLs, this process is used to search the list for adjacent

EVRs that have identical VLs. Since the EVRs are actually convex polygons, two EVRs are considered as adjacent if they have one edge common. These EVRs are then merged into a single EVR, and the EVR list is updated if they have identical VLs. This step is run iteratively, each time merging the adjacent EVRs and forming new ones until no two adjacent EVRs have identical VLs.

4.2 The Algorithm *Partition*

The algorithm *partition* is described below, which given a list of convex polygons and a viewing plane, divides the viewing plane into EVRs and forms their associated VLs. The algorithm initially assumes the entire viewing plane to be one EVR and then iteratively splits this plane into sub-regions, considering each of the polygons in turn using the process *Split*. Whenever a new polygon is considered, not only are the existing EVRs split to account for the visibility of its edges, but the occlusions of the edges of this polygon due to all the other existing polygons in the plane are also considered using the *Project* process. The *Merge* step is run after each *Project* step to merge adjacent EVRs that have the same VLs. As described before, the routine $Split(P, List)$, takes a polygon P and an EVR list *List*, and modifies the EVRs in *List* and their associated VLs depending on the visibility of the polygon P's edges. The $Project(e, P, List)$ routine takes in an edge e, a polygon P, and an EVR list *List* and modifies the List and the associated VLs to take into account the occlusion of e due to the edges of P. The $Merge(List)$ routine takes an EVR list, *List*, merges the adjacent EVRs having the same VLs, and returns the modified *List*.

Algorithm Partition
Input : A set of convex polygons in a plane.
Output : A set of EVRs and their associated VLs.

```
1. Initialize

   1.1 Initialize the $EVR\_LIST$ to contain one region,
                                 the entire plane
   1.2 Set the VL of this region to NULL.
   1.3 Initialize the $USED\_POLYGONS$ list to NULL.

2. For ( i = 1 to i < $No\_of\_ALL\_POLYGONS$ )
   do
   2.1 Set $CURRENT\_POLYGON$ =
                         $ALL\_POLYGONS(i)$
   2.2 Call $Split(CURRENT\_POLYGON, EVR\_LIST)$
   2.3 For ( j = 1 to j < $No\_of\_USED\_POLYGONS$ )
      do
      2.3.1 For ( k = 1 to k < No_of_edges_of(
                            $CURRENT\_POLYGON$) )
```

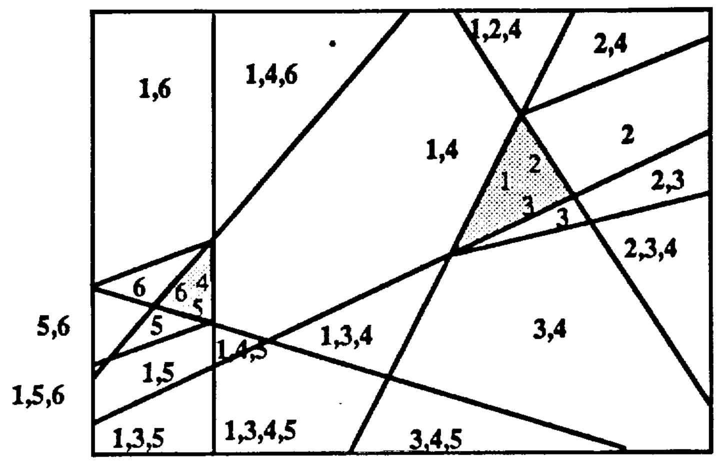


Figure 4.6: The EVRs of two convex polygons

```
        do
        2.3.1.1 Call Project(
                $EDGE\_of(CURRENT\_POLYGON, k)$,
                $USED\_POLYGONS(j), EVR\_LIST)$
        2.3.1.2 Call $Merge(EVR\_LIST)$
        endo
      2.3.2 For ( k = 1 to k < No_of_edges_of(
                        $USED\_POLYGONS(j))$ )
        do
        2.3.2.1 Call Project(
                $EDGE\_of(USED\_POLYGONS(j), k)$,
                $CURRENT\_POLYGON, EVR\_LIST)$
        2.3.2.2 Call $Merge(EVR\_LIST)$
        endo
      endo
    2.4 $USED\_POLYGONS($
            $No\_of\_USED\_POLYGONS)$
            $= CURRENT\_POLYGON$
    2.5 Increment $No\_of\_USED\_POLYGONS$
    endo
  3. Exit
```

Figure 4.6 shows the EVRs and their VLs for the case of two convex polygons.

4.3 The Estimation of the interval of orientations

For each model edge in the VL of an EVR the interval of orientations of the robot for which this model edge is visible in the EVR is also computed and stored. An efficient method to compute

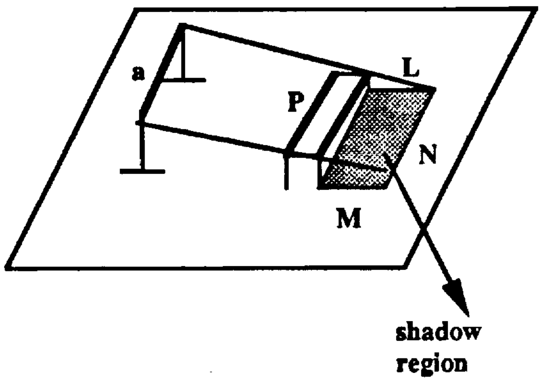


Figure 4.7: The shadow regions for buildings of unequal height

the interval is given below. Given an EVR and a model edge e_i in the VL of the EVR, we need to find the lower bound, θ_{min}, and the upper bound, θ_{max}, of the orientation angles of the robot (camera) for which this edge e_i is visible in the EVR. Let the vertices of the EVR be $v_i, i = 1,..n$. Let p_1, p_2 be the end points of e_i. Then it can be easily shown that θ_{min} is the minimum of the angles made by the lines joining p_2 to $v_i (i = 1,..n)$ and that θ_{max} is the maximum of the lines joining p_1 to $v_i, i = 1,..n$. Hence, given an EVR and a model edge e_i in its VL, the maximum and the minimum orientation angles can be estimated by considering just the vertices of the EVR.

4.4 Generalizations

In the case of the buildings with rooftops that are not convex, the non-convex polygons representing the projections of the rooftops on to the XY-plane are decomposed into a set of adjacent, component convex polygons [28,48]. Only decompositions without *Steiner points* are considered, since the modifications to the existing algorithms in this case are straightforward. The extra edges that are added in the process are considered as *dummy* edges, and their visibility is not marked in the VLs of the EVRs. Hence, the self occlusions of the edges of a non-convex polygon are handled by decomposing the polygon into component convex polygons and dealing with their mutual occlusions by using the *project* process. In the case when all the buildings are not of the same height, it is insufficient to just consider the projections of the rooftops onto the XY-plane alone when forming the shadow regions. The project process is modified to consider a Z-shadow line also. Figure 4.7 illustrates this case. Here the shadow region of line a on to the polygon P is, hence, the region bounded by L, M and N, where N is the Z-shadow line. Note that the case of all the buildings of the same height is actually a special case of unequal height buildings where the Z-shadow line is at infinity. In the case when the rooftop of a building is not flat (planar), it is decomposed into convex planar polygons and each of these is considered as a separate polygon; the partition algorithm is then modified as before, in the non-convex case. Also, the Z-shadow lines are drawn to account for these component polygons, which are now convex and flat but of different heights.

4.5 Estimating the Number of EVRs

An interesting and important question related to using this method is how many EVRs would be generated using the *partition* algorithm. If this number is very large, it would be impractical to use the method. One might think that the number of distinct EVRs grows exponentially with m and n the number of polygons and the number of sides of each polygon, respectively. We derive an upper bound on the maximum number of EVRs that would be generated and show that this is polynomial in m and n, $O(n^2m^4)$. The derivation is based on induction and is similar to that outlined by Ikeuchi and Kanade [25] in the context of aspect graphs for object recognition.

Consider the case when there are k n-sided convex polygons in a plane. Let the total number of lines in the plane be given by $L(k)$. We have

$$\begin{aligned} L(k) &= L(k-1) + n + 2 \cdot n \cdot (k-1) \\ &+ 2 \cdot n \cdot (k-1) \end{aligned} \tag{4.1}$$

where the first term indicates the number of lines already existing when $(k-1)$ polygons have been considered; the second term indicates the number of lines drawn by extending the edges of the kth polygon; the third term accounts for the shadow lines drawn from the n edges of the kth polygon onto the other $k-1$ already existing polygons; and finally the fourth term accounts for the shadow lines drawn from each of the n edges of the existing $k-1$ polygons onto the kth polygon. Recall that each edge of a polygon contributes two *shadow lines* for each of the existing polygons lying on the visible side of the edge. Solving (1) we have,

$$\begin{aligned} L(k) &= kn + 4n[(k-1) + (k-2) + \dots..1] \\ L(k) &= kn(2k-1). \end{aligned}$$

Let $T(m)$ denote the maximum number of EVRs generated when we have m n-sided, convex polygons in a plane under perspective projection. Note that $T(0) = 1$. Consider the situation when there are $m-1$ polygons already existing and in the plane. The maximum number of lines in the plane is $L(m-1) = n(m-1)(2m-3)$. Now when we consider the mth polygon, we need to perform three steps which increase the number of lines and hence the EVRs. First, each edge of the mth polygon is extended, which gives n lines. Second, for each of the n edges of this mth polygon, we draw two shadow lines onto each of the other $m-1$ polygons already in the plane; this gives $2n(m-1)$ lines. Finally, for each of the n edges of the $m-1$ polygons, we draw two shadow lines onto the mth polygon, which also gives $2n(m-1)$ lines. So, we add a total of $n + 4n(m-1)$ new lines by adding the mth polygon.

If we assume that each of these $n(4m-3)$ lines is added successively, the number of EVRs grows after each new line is added. We can derive the maximum number of EVRs by induction.

Let $T(m)_k$ denote the number of EVRs after the kth line is drawn. Since we started with $T(m-1)$ EVRs and $L(m-1)$ lines, after the first additional line is drawn, the number of EVRs, $T(m)_1$, is

$$T(m)_1 = T(m-1) + L(m-1) + 1$$

since this line is cut by the existing $L(m-1)$ lines into $L(m-1)+1$ segments (maximal case). These will divide $L(m-1)+1$ regions into two, thus adding $L(m-1)+1$ new EVRs. Proceeding in this fashion,

$$\begin{aligned}
T(m)_2 &= T(m)_1 + L(m-1) + 2 \\
T(m)_{n(4m-3)} &= T(m) = T(m)_{n(4m-3)-1} \\
&+ L(m-1) + n(4m-3) \\
T(m) &= T(m-1) + n(4m-3)L(m-1) \\
&+ \sum_{i=1}^{n(4m-3)} i \\
&= T(m-1) + \frac{[n(4m-3)]^2}{2} + \frac{n(4m-3)}{2} \\
&+ n(4m-3)[n(m-1)(2m-3)] \\
&= T(m-1) + O(n^2m^3) + O(n^2m^2) \\
&+ O(mn) \\
&= O(n^2m^4)
\end{aligned}$$

It is worth pointing out that although the maximum number of regions suggested by this derivation is of $O(n^2m^4)$, in practice it is very unlikely that so many EVRs will be generated. First, the derivation assumes that whenever a new line is drawn it cuts all the existing lines in the plane, which very rarely happens. Second the shadow lines are considered to be drawn to all the existing polygons at any given time. In practice, however, we need to draw the shadow lines only onto the polygons that lie on the visible side of the edge. Since we are considering convex polygons, the situation of all the polygons lying on the visible side of all the edges is an impossible case. Also, the *Merge* process is quite effective and reduces the number of regions by a significant amount, particularly as more more and more polygons are considered. Taking all these factors into account, we find the number of regions to be much smaller than $O(n^2m^4)$. However, this upper bound serves the useful purpose of showing that the number of EVRs is, in the worst case, still polynomial in n and m.

4.6 Positional Estimation and Path Planning

Having developed a methodology to partition the free space in the into EVRs, we now propose a procedure to estimate the position and pose of the robot in the environment. Also, we discuss the use of the EVRs in path-planning is discussed.

4.6.1 Positional Estimation

As discussed in before, the position of the robot is given by three parameters, (X, Y, θ). These can be obtained by solving the transformation matrix $T(X, Y, \theta)$, which, transforms the WCS into the CCS. This matrix can be solved for by establishing a correspondence between the features in the WCS and the CCS. The features we plan to use are the line segments that constitute the roof tops of the buildings, since the 3-D descriptions of these are assumed to be given in the world model.

To establish the correspondence between the world model features and their images, one possibility is to formulate the problem as a search paradigm [21], i.e., to form an *interpretation tree* of all the possible pairings between the model and the image features and to search this interpretation tree for a set of consistent pairings which can then be used to solve the transform $T(X, Y, \theta)$. However, since the 3-D description of the model features are given, by using the geometric relations between them, and the known perspective geometry of the camera, we can formulate constraints as to which features will be visible from different places in the XY-plane. The interpretation tree can then be pruned using these constraints so as not to consider all the possible pairings. Recall that the EVRs store all the relevant geometric information about the model features, like, the number of features visible from a given region and the interval of orientations of the robot for which these features are visible. So given a set of images features the EVRs can be used to prune the interpretation tree in establishing a consistent set of pairings between the image and model features and, hence, the position and pose of the robot can be computed accurately from this set of matches.

Usually the robot position and pose, as given by its position encoders, is available, which essentially isolates the robot position to lie within a few EVRs. So only these EVRs need to be considered in searching for the exact location Also if the robot can identify a landmark (one of the 3-D features), this ability can be used to isolate the robot position to lie only within those EVRs that contain this landmark edge in their VLs. Similarly, if a rough estimate of the θ value is available, it can be used to quickly prune some of the hypotheses.

4.6.2 Path Planning

Planning paths from the start to the goal and executing them is an important navigational task to be performed by all autonomous mobile robots [49]. The EVRs are a convex region representation of the free space. Representing the free space by convex regions has the advantage that any path chosen inside an EVR will be entirely within the EVR and, hence, will be obstacle free. So, to plan an obstacle free path between a start position S and a goal position G, the free space is represented as a graph, with the EVRs as the nodes and an arc existing between two EVRs, if they are adjacent. If S is in EVR_i and G is in EVR_j, then an obstacle free path between S and G can be found by a graph search for a connected component list between EVR_i and EVR_j. If the arcs are weighted by the distance metric or some other function to be optimized, the graph search can be used to yield the minimum distance path. Also, since each EVR has a VL of the world model edges, this can be used to establish the robot's position in each EVR accurately at every desired instant and, hence, the path can be followed more precisely.

5 Conclusions

In this report we have presented the work done under the NASA Grant NAG 9-361. The problem of estimating the position and pose of an autonomous mobile robot in an outdoor environment is considered in this research. A comprehensive review of the various positional estimation techniques studied in the literature is first presented. Then positional estimation techniques for an outdoor mobile robot in two different kinds of environments are described; a mountainous natural terrain and an urban, man-made environment with polyhedral buildings. In both these cases the robot is assumed to be equipped with a single visual camera that can be panned and tilted and also a 3-D description (world model) of the environment is given. This world model is in the form of Digital Elevation Map (DEM) for the natural terrain case. For the urban environment, it is assumed that the 3-D descriptions of the roof tops of the buildings are given. Such a description could be obtained from the architectural plans of the buildings or from a stereo pair of aerial images. Techniques for positional estimation using the camera input and the world model are presented.

In the natural terrain case the Horizon Line Contour(HLC) extracted from the camera images is used as a distinctive feature to search the DEM for the position of the robot. Various heuristics drawn from the geometric constraints are used to reduce the search space. In the urban environment, a new intermediate representation of the environment of a mobile robot called the *Edge Visibility Regions*(EVRs) is developed. This representation can be formed offline from a given stored model description of the environment. The uses of representing the environment as EVRs in alleviating the problem of establishing a correspondence between the sensor readings and the stored model for the various navigational tasks of the mobile robot like positional estimation and path planning are discussed.

References

[1] F.P.Anderson and L.S.Davis, "Visual position determination for autonomous vehicle navigation," *Center for Automation research*,Tech report, CAR-TR-100,Nov 1984

[2] M.Asada, Y.Fukui, S.Tsuji. "Representing a global map for a mobile robot with relational local maps from sensory data," *Ninth Int. Conf. Pattern Recog.*, Italy, Nov 1988.

[3] N.Ayache, O.D.Faugeras, "Building a consistent 3D representation of a mobile robot environment by combining multiple stereo views," *Proc. 10th IJCAI*, 1987.

[4] B.C. Bloom, "Use of Landmarks for mobile robot navigation," in *SPIE*, vol. 579, Intelligent Robots and Computer Vision, 1985.

[5] R.A.Brooks, "Visual map making for a mobile robot," in *Proceedings of the IEEE Int. Conf. Robotics and Automation*, pp. 824 - 829, St.Louis,1985,

[6] R.A.Brooks, "Symbolic error analysis and robot planning," *Int. J. Robotics Res.*, 1(4):29-68.

[7] M. Case, "Single landmark navigation by mobile robots," *SPIE, Mobile Robots*, vol. 727, pp. 231 - 238, October 1986.

[8] R.Chatila and J-P. Laumond, "Position referencing and consistent world modeling for mobile robots," *Proc. of the IEEE Int. Conf. Robotics and Automation*, pp. 138 - 145, St.Louis,1985,

[9] J.Courtney, M.Magee and J.K.Aggarwal, "Robot guidance using computer vision," *Pattern Recognition*, vol. 17, no. 6, pp. 585 - 592, 1984.

[10] J.L.Crowley. "Dynamic world modeling for an intelligent mobile robot using a rotating ultrasonic ranging sensor," *Proc. of the IEEE Int. Conf. Robotics and Automation*, pp. 128 - 135, St.Louis,1985.

[11] J.L.Crowley, "World modeling and position estimation for a mobile robot using ultra-sonic ranging," *Proc. of the IEEE Int. Conf. Robotics and Automation*, 1989.

[12] K.C.Drake, E.S.McVey and R.M.Iñigo, "Experimental position and ranging results for a mobile robot," *IEEE Jour. of Robotics and Automation*, vol. RA-3, no. 1, Feb 1987.

[13] M.Drumheller, "Mobile robot localization using sonar," *IEEE Tran. on PAMI*, 9(2), March 1987, pp. 325-332.

[14] A.Elfes. "Sonar based real-world mapping and navigation," *IEEE Jour. of Robotics and Automation*, vol. 3, no. 3, pp. 249 - 265, June 1987.

[15] I.M.El Hassan, "Analytical techniques for use with reconnaissance from photographs," *Photogrammetric Eng. Remote Sensing*, vol. 47, no. 12, pp. 1733 - 1738, Dec 1981.

[16] Faugeras, O.D.,N.Ayache and B.Faverjon, "Building visual maps by combining noisy stereo measurements," *Proc. IEEE Conf. Rob. Automation, San Francisco*, pp. 1433 - 1438, 1986.

[17] O.Faugeras and N.Ayache, "Building,registering and fusing noisy visual maps," *Proc. of the First Int. Conf. Computer Vision*, pp 73 - 82, June 1987, London.

[18] M.A.Fischler and R.C.Bolles, "Random Sample Consensus : A paradigm for model fitting with application to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, June 1981.

[19] I.Fukui, "TV image processing to determine the position of a robot vehicle," *Pattern Recognition*, vol. 14, no. 1-6, pp 11-19, 1981.

[20] S.Ganapathy, "Decomposition of transformation matrices for robot vision," *Proc. of the 1st IEEE Int. Conf. Robotics*, 1984.

[21] W.E.L.Grimson and T.Lozano-Perez, "Model-based recognition and localization from sparse range or tactile data," *Int. J. Robotics Res.*, vol. 3, no. 3, pp. 3-35, 1984.

[22] R.M.Haralick et al., "Pose estimation from corresponding point data," *IEEE Tran. on Sys., man and cybernetics*, vol. 19, no.6, Nov/Dec 1989.

[23] R.Horaud, B.Conio and O.Leboulleux, "An analytical solution to the perspective 4-point problem," *CVPR*, 1989.

[24] B.K.P.Horn, "Relative Orientation," *Proc. of the Image Understanding Workshop*, vol. 2, pp. 826-837, 1988.

[25] K.Ikeuchi and T.Kanade, "Applying sensor models to automatic generation of object recognition programs," in *Proceedings of 2nd ICCV*, pp. 228-237, 1988.

[26] M.R.Kabuka and A.E.Arenas, "Position verification of a mobile robot using a standard pattern," *IEEE Jour. of Robotics and Automation*, vol RA-3, no. 6, Dec 1987.

[27] A.Kak, K.Andress and C.Lopez-Abadia, "Mobile robot self-location with the PSEIKI system," *Working Notes, AAAI spring symposium on robot navigation*, Mar 1989.

[28] J.M.Keil and J.R.Sack, " Minimum decomposition of polygonal objects", *Computational Geometry*, edited by G.T. Toussaint, Elsevier Science Publishers B.V., pp. 197-216, 1985.

[29] E.Krotkov, "Mobile robot localization using a single image," *Proc. of the IEEE Int. Conf. Robotics and Automation*,1989.

[30] B.J.Kuipers and Y.T.Byun, "A robust qualitative method for robot spatial learning," *AAAI-88, The Seventh National Conf. on Art. Int.*, 1988.

[31] R.Kumar, "Determination of the camera location and orientation,"in *DARPA Image Understanding Workshop*, 1988.

[32] R.Kumar and A.Hanson, "Robust Estimation of the camera location and orientation from noisy data having outliers," *Proc. of the Workshop on Interpretation of 3-d scenes*, Austin, Nov 1989.

[33] J.Lessard and D.Laurendeau, "Estimation of the position of a robot using computer vision for a live-line maintenance task," in *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 1203-1208, 1987.

[34] T.S.Levitt, D.T.Lawton, D.M.Chelberg, P.C.Nelson, "Qualitative navigation," *DARPA Image Understanding Workshop*, 1987.

[35] Y.Liu,T.Huang and O.Faugeras, "Determination of the camera location from 2-d to 3-d line and point correspondences," *IEEE Tran. on PAMI*, vol. 12, no. 1, Jan 1990.

[36] M.J.Magee and J.K.Aggarwal, "Determining th position of a robot using a single calibration object," *Proc. of the 1st IEEE Int. Conf. Robotics*, 1984.

[37] L.Matthies and S.A. Shafer, "Error modeling in stereo navigation," *IEEE Jour. of Robotics and Automation*, vol. 3, June 1987, pp. 239 - 248.

[38] L.Matthies and A.Elfes. "Integration of sonar and stereo range data using a grid based representation," *Proc. of the IEEE Int. Conf. Robotics and Automation*, 1988.

[39] C.D. McGillem and T.S.Rappaport, "Infra-Red location system for navigation of autonomous vehicles," *Proc. of the IEEE Int. Conf. Robotics and Automation*,1988.

[40] D.Miller, "A spatial representation system for mobile robots," *Proc. of the IEEE Int. Conf. Robotics and Automation*, pp. 122 - 127, St.Louis,1985.

[41] H.P.Moravec, *Robot Rover Visual Navigation*, Ann Arbor, MI,UMI Research Press 1981.

[42] H.P.Moravec, "The Stanford Cart and the CMU Rover," *Proc. of the IEEE*, vol.71, no. 7, July 1983.

[43] H.P. Moravec. "Sensor fusion in certainty grids for mobile robots," *AI magazine*, vol. 9, no. 2, pp. 61 - 74, Summer 1988.

[44] H. P. Moravec and D.W.Cho. "A Bayesian method for certain grids," *AAAI Spring Symposium Series on Mobile Robot Navigation*, Stanford, April 1989.

[45] H.Nasr and B.Bhanu, "Landmark recognition system for autonomous mobile robots," *Proc. of the IEEE Int. Conf. Robotics and Automation*,1988.

[46] A. Robert de Saint Vincent, "A 3D perception system for the mobile robot HILARE," *Proc. IEEE Conf. Rob. Automation, San Francisco*, pp. 1105 - 1111, 1986.

[47] P.J.Rosseeuw and A.M.Leroy, "Robust regression and outlier detection," John Wiley and sons N.Y 1987.

[48] Joseph O'Rourke, *Art Gallery Theorems and Algorithms*, John Wiley, 1988.

[49] J.T.Schwartz, M.Sharir and J.Hopcroft, *Planning, Geometry, and Complexity of Robot Motion*, Ablex publishing co., N.J., 1987.

[50] R.C. Smith and P.Cheeseman, "On the representation and estimation of spatial uncertainty," *Int. Journal of Rob. Res.*, vol. 5, no. 4, pp. 56 - 58, 1987.

[51] L.Stark and K.Bowyer, "Aspect graphs and non-linear optimization in 3-D object recognition," in *Proceedings of 2nd ICCV*, pp. 501-507, 1988.

[52] K.Sugihara, "Some location problems for robot navigation using a single camera," *CVGIP*, April 1988.

[53] W.Szczepanski, *Die Lösungsverchläge für den räumlichen Rückwärtseinschnitt, Deutche Geodätische Komission* Reiche C: Dissertationen-Heft Nr, pp. 1-44, 1958.

[54] R.Talluri and J.K.Aggarwal, "Positional Estimation for a mobile robot in an outdoor environment," *IEEE Workshop on Intelligent Robots and Systems, IROS '90*, Japan, July 90.

[55] R.Talluri and J.K.Aggarwal, "Positional Estimation technique for and autonomous mobile robot in an unstructured environment," *i-SAIRAS '90*, Japan, Nov 90.

[56] R.Talluri and J.K.Aggarwal, "Edge Visibility Regions - a new representation of the environment of a mobile robot," *IAPR Workshop on Machine Vision Applications, MVA '91*, Japan, Nov 90.

[57] R.Talluri and J.K.Aggarwal, "Positional Estimation for a mobile robot - a review submitted to *IEEE Transactions on Robotics and Automation.*

[58] R.Talluri and J.K.Aggarwal, "Positional Estimation for a mobile robot using Edge Visibility Regions," submitted to, *Computer Vision and Pattern Recognition, CVPR '91.*

[59] G.T.Toussaint, "Computing visibility properties of polygons," *Pattern Recognition and Artificial Intelligence*, Elsevier Science Publishers B.V., pp. 103-122, 1988.

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

[61] T.Tsuboushi and S.Yuta, "Map assisted vision system of mobile robots for reckoning in a building environment," *Proc. of the IEEE Int. Conf. Robotics and Automation*, pp. 1978-1984, Raleigh NC, 1987.

[62] S.Tsuji and J.Y.Zheng. "Visual path planning by a mobile robot," *Proc. 10th IJCAI*, 1987.

[63] J.S-C. Yuan, "A general photogrammetric method for determining object position and orientation," *IEEE Trans. on Robotics and Automation*, vol. 5, no.2, pp. 129-142, April 1989.

[64] P.R.Wolf, *Elements of Photogrammetry*, New York, Mc.Graw Hill, 1974.