Proceedings of the 2004 IEEE
Conference on Robotics, Automation and Mechatronics
Singapore, 1-3 December, 2004

Practical Ego-motion Estimation for Mobile Robots

Shawn Schärer
Department of Electronic and Computer Engineering
University of Manitoba
Winnipeg, Canada
schaerer@ee.umanitoba.ca

Jacky Baltes
Department of Computer Science
University of Manitoba
Winnipeg, Canada
jacky@cs.umanitoba.ca

John Anderson
Department Of Computer Science
University Of Manitoba
Winnipeg, Canada
andersj@cs.umanitoba.ca

Abstract— **Accurate ego-motion estimation is a difficult problem that humans perform with relative ease. This paper describes two methods that are used in conjunction to estimate the ego motion of an intelligent autonomous vehicle from vision alone. First, a cross-correlation method is used to select a promising patch in the image. The optical flow information for this patch is used to determine linear and angular velocity of the intelligent autonomous vehicle. Lines in the image are then used to provide an estimate of the ego motion of the vehicle. The gradient of the line as well as the distance to the line allow the computation of current wheel velocities. Both methods have been implemented on real robots and have been tested in a treasure hunt competition. These methods greatly improved the exploration as well as accuracy of the generated maps of the environment.**

I. Introduction

An intelligent vehicle expected to perform autonomously in any rich domain requires many skills: for example, it must be able to accurately identify objects in the environment that affect its activities, plan paths using the information it has about the world, and be able to alter those paths in the face of changes in the environment. In order to do any of this with any accuracy, however, the mobile robot must be able to have estimates of its own motion in the world (i.e., its ego motion) - in particular, its own translational and rotational velocities with respect to the environment. This basic skill supports all higher-level motion-planning activities, and these correspondingly suffer if ego-motion estimation is done poorly and/or inefficiently.

In the domain of small mobile robots (for example mail delivery robots in an office), researchers most often rely on odometry information to as a basis for ego-motion estimation. Most commonly, shaft encoders are used to measure wheel velocities. This approach works well in simple environments where a small amount of wheel slip is tolerable and can be corrected for (such as an indoor environment with a carpet. Where errors cannot be perfectly compensated for, however, they compound, and eventually the robot's estimation of its own position is incorrect enough that path planning and following cannot ensue. The compound nature of this error makes the reliance on simple methods such as shaft encoders ineffective in complex domains such as uneven territory or where surfaces are unstable or slippery, making this an unreasonable approach to the development of general-purpose intelligent autonomous vehicles.

Shaft encoders are by no means the only method used for estimating motion: other forms of internal sensing such as accelerometers, gyroscopes, inertia navigations systems, or too some extent GPS are also used. Although these approaches are not susceptible to errors introduced by wheel slip, the required additional sensors increase the cost of intelligent autonomous vehicles, sometimes very significantly so. The use of a camera as a tool for motion-estimation, however, does not necessarily require the use of additional equipment: any sort of visual sensing on the part of the robot already requires a camera, and most intelligent autonomous vehicles also carry at least one camera to gather and/or relay data about their environment back to humans. The motivation for the research described in this paper is to allow practical method for ego-motion estimation using the camera(s) that are likely already fitted to the intelligent autonomous vehicle.

We have developed two approaches that operate in conjunction using information from a camera mounted on the mobile robot in order to estimate its ego motion. The first approach uses the optical flow of small patches in the view of the camera to determine translation and rotation from the previous to the current view. This approach is general in nature and does not rely on any specific features of the image itself. The second approach uses real or virtual lines to determine translation and rotation by measuring how the robot moves in relation to lines detected in the image. Since these two approaches rely on different information in the image, they are complimentary.

This paper describes both of these approaches in detail (Sections III, IV, and V) , as well as detailing their implementation and deployment on mobile robotic hardware. In order to examine the effectiveness of these approaches, we chose a small mobile robot platform with limited computational abilities and where ego-motion estimation was solely performed using vision, as opposed to supplemented with specialized hardware. Following this, we describe preliminary results demonstrating the effectiveness of the described methods in Section VI. We begin with a review of previous work in this area.

II. Related Work

Ego-motion estimation, the problem of determining the motion of a robot from a given a sequence of images taken during the motion, is a very popular research area in computer vision. Detecting motion across a series of images is a useful task in many applications beyond its use in ego-motion detection, from manufacturing to security, for example. While this is easy

0-7803-8645-0/04/$20.00 © 2004 IEEE

for a human to do, it is a difficult problem to approach from the standpoint of a computer program.

More formally, ego-motion estimation can be defined as determining the translation and rotation parameters of a camera view given two different views of a scene. This estimate is usually based on unconstrained motion of a calibrated camera with respect to a plane. In this case, there are eight parameters to be determined.

Most approaches use the *rigid world* assumption where the scene is considered static and all movement in the image is due to the motion of the robot. In multi-agent domains such as robotic soccer, this assumption does not hold true very often. For a local vision robot, large movements in the image may be due to other robots moving in the image as well.

Previous work in ego-motion estimation in computer vision can be divided into two categories: those that pre-suppose structuring (e.g. lines) in the image to use as a basis for detecting movement, and those that do not. As an example of the latter method, Stein et al. [1] propose a method that takes a region of an image, and uses all pixels in the region to provide support for a particular motion. Their method computes a global probability distribution function for the whole image, and is robust in the presence of many outliers. Stein reduces the number of parameters to three: translation, pitch, and yaw.

As an example of a method attempting to exploit structure in the image, Zhang [2]describes a methodology for estimating motion and structure using line segments in an image . This approach is based on the assumption that line segments overlap in 3D space, which allows a reduction in the number of motion parameters.

Szeliski andTorr [3]introduce geometric constraints early in the structure reconstruction phase to improve the overall 3D reconstruction results. This method takes advantage of the fact that, the real world has many planer structures such as vertical walls and flat ground planes and uses external geometric constraints such as plane parallelism to reduce reconstruction errors.

Sim and Dudek [5] utilize learned landmarks to estimate robot position. Their method first attempts to visually detect possible landmarks by looking for image features and then attempts to match these detected landmarks from a predefined landmark database. Robot localization is preformed by matching the detected landmarks to database landmarks and then estimation robot position.

Another example of exploiting structure for a image is from Se, Lowe and Little [4] which developed a local vision based SLAM algorithm by following visual scale invariant feature transform (SIFT) landmarks in a arbitrary environment. To estimate the ego-motion of the robot they match SIFT features between frames and preform least squares minimization to minimize the match errors, which yields a better 6 DOF ego-motion and localization estimate.

While approaches in both categories have strong theoretical foundations, there are also practical issues to be considered when they are to be deployed on mobile robots. While becoming more powerful, today's embedded systems are still limited compared to the powerful computers on which many approaches are implemented and evaluated. Consequently, most existing approaches are computationally too expensive to be performed on an embedded controller in real-time. The approach described in this paper uses a two parameter model appropriate for more basic embedded systems mobile robots. Given a fast camera calibration routine and a fast wall detection method, we are able to estimate the motion on a small embedded system. The model used in our work is described in the next Section, followed by the two specific methodologies we combine in our implementation.

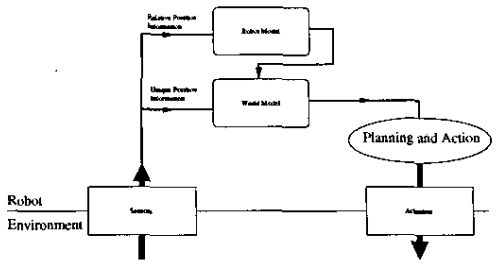

Fig. 1. System Design of our Localization Method

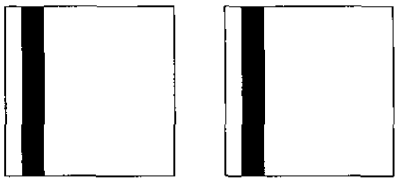

Fig. 2. A simple example where it is impossible to determine the angular, but not the linear velocity of a mobile robot. Since the relative orientation and position of the line has not changed, the robot was moving in a straight line.

III. System Design

A high-level view of the model employed in our work as shown in Figure 1. The mobile robot interacts with the environment using actuators and receives feedback from the environment via its sensors: in this case, visual feedback from a CMOS camera.

As in most other related work, the sensors are used to update the world model of the mobile robot - in our case, its orientation and position. However, not all sensor feedback is useful to update the world model uniquely. For example, assume that a robot tracks parallel to a line as shown in Figure 2. Given the image sequence, it is clear that the mobile robot has traveled in a straight line, but it is not possible to determine how far it has traveled. So, only information about the angular velocity and position can be determined uniquely.

Most other systems use dead reckoning to supplement the localization in this case. In our case, we wish to avoid relying upon methods that can only be assumed to be correct under restricted conditions (such as the use of shaft encoders). As

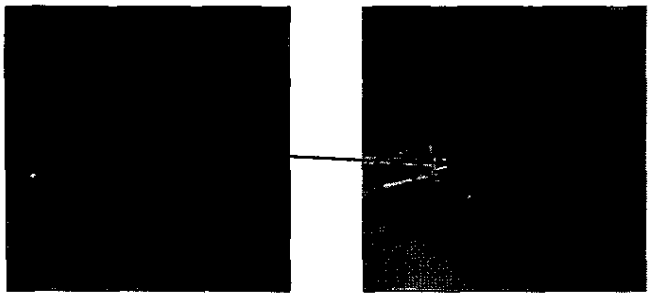

Fig. 3. Ego Motion Estimation Using Optical Flow

an alternative, our system maintains an internal model of the robot to support dead-reckoning style navigation.

In this approach, sensor feedback is used to update an internal robot model, which models the behavior of the robot to the motor commands. This relationship changes drastically over time: for example, since the batteries begin to lose their charge after only a few minutes, the robot moves a much shorter distance per unit time with a given motor setting as time passes. The robot model allows the system to consider the current reactions of the robot to different motion commands, and is adapted when new location information becomes available.

As stated in Section I, our approach to ego-motion estimation is based on two coordinating methods: one that does not assume any structure in the image, and the other that does. We correspondingly divide the description of these methods into the two sections that follow.

IV. Ego Motion Estimation Using Optical Flow

In order to make ego-motion detection through visual feedback efficient for deployment on an embedded system, we begin by considering pixels in only a limited range of the image. We track small windows (8*8 to 16*16) in the image, and compute the optical flow within these windows to calculate the ego motion.

When analyzing the differences between images to determine optical flow, the approach begins by using the current model of the robot's motion to determine the most likely direction of the optical flow. The current motion of the robot is classified into four classes: straight forward, straight back, turn left, or turn right. Seven candidate patterns are generated that favor detection in the estimated direction of motion.

For example, consider the two images depicted in Figure 3. In this situation, commands recently sent to the robot were intended to cause it to drive in a straight line. The system therefore assumes that the scene depicted in the later (rightmost) image in the Figure should be shifted down relative to the scene depicted in the earlier (leftmost) image in the Figure. The system selects seven candidate patches to track which are arranged above the center of the image. Here, these regions have a good chance of appearing in the next image as patches close to the bottom of the image. Conversely, choosing patches on the bottom of the earlier image to track would not likely be helpful, as there is a strong likelihood that those patches would drop out of view. The candidate patches chosen for movement intended to be straight back, turning left, and turning right are similar to the ones shown in the example, but rotated by the obvious angle (90 or 180 degrees).

After selecting the patches to examine, the candidate patch with the largest cross-correlation difference is selected. The motivation is that this patch is most easily distinguished from the surrounding patches and thus the chance that it is tracked successfully is increased.

We use a Mini-max search method to find the patch with the largest cross-correlation difference. This is a useful and efficient technique, since once the first minimum has been established, the remaining candidate patches can be rejected even if only one of the cross correlations is larger than the current minimum. This is because the maximum of all cross correlations for a single patch is used. In the theoretical worst case, selecting the patch requires the computation of 30 cross-correlations. In practice, however, we found that this is rarely necessary and that the Mini-max selection finishes quickly. In the example, the red patch has the maximum minimum cross correlation to the other six patches and is therefore selected.

Having found the patch with the greatest cross-correlation difference, the system then finds a patch with a small cross correlation in the next image. This is done by estimating the optical flow and starting a search for the patch in the neighborhood of the predicted position of the patch in the new image.

The robustness of this method can be improved by tracking more than one patch and by computing a best estimate given the optical flow information for multiple patches. However, this would greatly increase the computational cost of the algorithm and its associated viability on small embedded systems. Therefore, we limited ourselves to tracking a single patch.

V. Ego Motion Estimation Using Lines

This method described in the previous section is general-purpose in that it does not assume any specific structuring of the scene in the images. However, there is generally information that can be obtained from a scene that can be used to supplement the performance of a general method such as this one. One of the most obvious and striking features that appears in a wide variety of environments are lines. For example, the edges between a wall and the floor form lines in an image, as do the edges between walls themselves. While in many cases walls are the most predominant source of lines, lines also exist in terms of patterns on floors or walls (such as the many lines shown in Figure 3).

Lines are thus a natural choice for a reasonably general feature to attempt to take advantage of in order to improve general-purpose ego-motion detection approaches such as that presented above.

We explain this approach using the kinematics of a differential drive robot (summarized in Figure 4). Our approach depends on the assumption that the velocities of the right

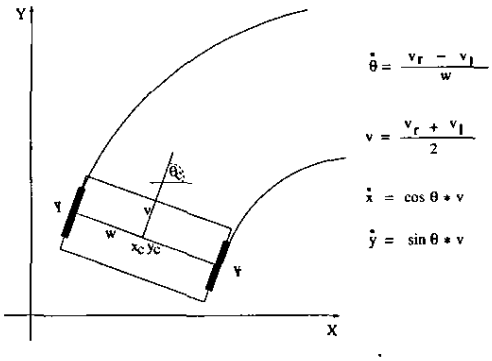

Fig. 4. Kinematic Model of a Differential Drive Robot

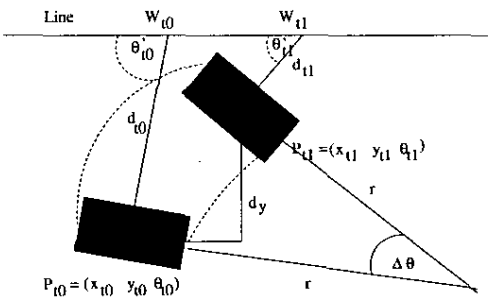

Fig. 5. Kinematic Model of a Differential Drive Robot

and the left wheel are constant during the time period for which the velocities are to be estimated. This is necessary since the approach uses the differences between the start and end locations of the robot motions to derive values for the wheel velocities, which is impossible if the wheel velocities are not constant during the time period in question. In this restriction were not made, the robot can perform any one of an infinite number of possible movements and return to the end location.

In order to motivate the analysis, consider the example depicted in Figure 5. At time $t0$, the robot is at position $P_{t0} = (x_{t0}, y_{t0}, \theta_{t0})$, and a straight line (formed by a wall) is in front of the robot. Let W_{t_i} be the point on the wall in the center of the camera view of the robot at time t_i and let d_{t_i} be the distance between P_{t_i} and W_{t_i}. Let θ'_{t_i} be the angle between the orientation of the robot θ_{t_i} and the angle of the wall θ_w.

At time $t1$, the robot is at position $P_{t1} = (x_{t1}, y_{t1}, \theta_{t1})$. The distance d of the robot to the wall as well as the angle θ' between the robot and the wall will have changed.

The problem is to derive from this information the current wheel velocities v_r and v_l for the right and left wheel respectively.

Given the kinematic model of the differential drive robot (Figure 4), it is easy to see that:

$$\dot{\theta}' = \dot{\theta} = \frac{v_r - v_l}{w}$$

Since the width w of the robot is known, this allows us to compute the difference in velocities for the two wheels. However, there is not sufficient information in the turn rate alone to determine the average velocity.

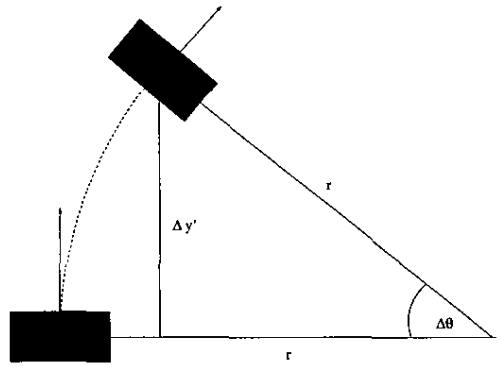

Fig. 6. Determination of the linear velocity v of a differential drive robot facing a wall

We assign a coordinate system X' with the origin at P and the x-axis parallel to the line or wall. In this coordinate system, the robot is at the origin and its orientation is θ'.

We can then derive the distance from the robot to the closest point on the wall $d_W = \sin\theta' * d$, allowing us to compute:

$$\Delta y = -(d_W(t1) - d_W(t0))$$

As can seen in Figure 6, the linear velocity of the robot can be computed from its rate of turn and the offset in the direction of the robot's $\Delta y'$.

¿From a simple geometric relationship, we can compute the radius of the circle r:

$$r = \frac{\Delta y'}{\sin(\theta'_2 - \theta'_1}$$

The linear velocity is then equal to the length of the circle segment $(r * \Delta\theta)$ divided by the time to cover this distance:

$$v = \frac{r * (\theta'_2 - \theta'_1)}{\Delta t}$$

In case the robot is facing the wall directly (i.e., $\theta' = 90^o$), and the above equation yields the correct solution for the linear velocity of the robot.

As can be seen in Figure 7, similar reasoning allows us to compute the circle distance $\Delta y'$ from a given wall distance.

¿From the figure, one can derive that:

$$\frac{c}{dy} = \sin\theta'_1 - \frac{\theta'_2 - \theta_1}{2}$$

In this case:

$$\Delta y = \Delta y' * \frac{\sin\frac{\theta_2 - \theta_1}{2}}{\cos\frac{\theta_2 + \theta_1}{2}}$$

The right and left wheel velocities can then be calculated as:

$$v_r = \frac{2v + \theta' * w}{2} v_l = \frac{2v - \theta' * w}{2}$$

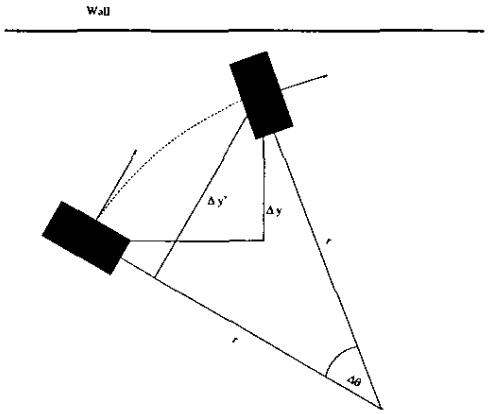

Fig. 7. Determination of the linear velocity v of a differential drive robot with an arbitrary angle to the wall

As stated earlier, any straight line segment in the image can be used as a guideline, allowing this method to be broadly applicable. While source of lines were hypothesized as walls in a typical indoor environment in the analysis above, lines are present in many important environments. In robotic soccer (and indeed, most human sports) lines are an important element of play. Even in less structured examples, lines are still present. In robotic rescue environments, for example, even though most of a structure may be collapsed, lines still exist in any remnant of standing wall, in debris with straight edges (e.g. strewn papers, lumber or other structural components), and in decorative patterns used in indoor environments. All of these provide line segments that can be used to compute the incremental location of the robot.

VI. Implementation and Evaluation

We employ the techniques described above in a number of different mobile robot platforms all based on small embedded systems (see Figure 8). In recent years, we have developed a variety of robotic platforms to support our research into intelligent mobile robotics. Because we are interested in developing broad, adaptive approaches to solving hard problems, we attempt as much as possible to force good performance to be a result of the intelligence of our algorithms rather than due to hardware that is specialized for the task at hand. To that end, our robots use cheap, off-the-shelf components such as electro motors and RC servos. The reliance on such components does not allow the use of accurate odometry sensors through shaft encoders, which in turn forces us to consider problems such as that described here, using intelligent information processing to handle what hardware cannot be generally relied upon to do.

In order to evaluate these techniques, we used a series of experiments to evaluate the effectiveness of these two approaches using our small scale mobile robots. We used both a robot employing a 33MHz 68332 based Eyebot controller, and a robot using a 300MHz Intel XScale processor. was limited to a 0.5 frames per second in its processing when

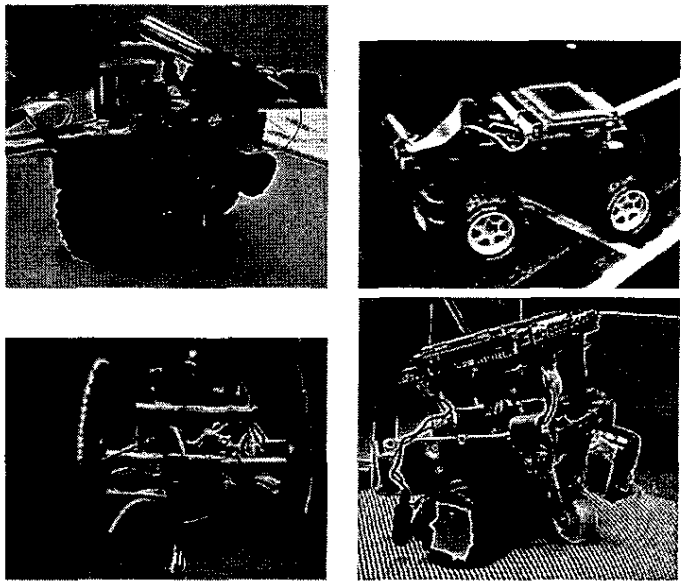

Fig. 8. Intelligent Mobile Robots at the University of Manitoba

using ego motion estimation from the optical flow. The rescue robot based on the 300MHz Intel XScale processor was able to maintain 10 frames per second.

The experiment consisted of an playing surface with few distinguishable landmarks (green carpet with markings of a soccer field), five static obstacles and one treasure. The goal of the experiment was for the robot to search the environment, find the treasure, and to produce an accurate map of the environment. The robots used estimates of its ego motion from both the optical flow and lines in the image.

Preliminary results of these experiments were encouraging. Not only was the robot able to create more accurate maps, but it also was more efficient at exploring the environment. For example, the inaccuracies in the odometry meant that the robots were driving around the environment in random directions. Using the ego motion estimation, the robot followed straight lines and 90 degree turns, making it feasible to implement search patterns such as increasing squares around a point. Overall, the robots were able to make better use of their internal maps and would not repeatedly search the same area of the environment.

VII. Conclusion

In this paper, we have presented two computationally efficient methods to estimate the motion of a mobile robot from an image sequence alone. The methods are based on general optical flow supplemented by the tracking of feature lines in the image. They are complementary in the sense that the lines are easier to track and provide better accuracy than the optical flow, but the optical flow is applicable even if no lines are found in the image.

We have tested our implementation in a treasure hunt scenario. The object of the treasure hunt game is for the robot to determine an accurate map of a simple environment consisting

of lines, obstacles, and a treasure. Preliminary results of our work are encouraging.

There is still, however, much room for future work. First, the approach would be improved if the algorithms were made more efficient for embedded processors. Despite attempting to optimize code, the robot based on the 33MHz processor was limited to 0.5 frames/second using these approaches. The 300MHz robot was able to maintain 10 frames/second, enough to be feasible for a domain where exploration is not intended to be fast-moving, such as robotic rescue, but still somewhat limited for a very fast-moving environment such as soccer.

In addition, the tracking of lines requires a calibrated camera view and assumes that the lines are parallel to the ground plane. We intend to extend this approach to three dimensional environments.

Our ultimate intention is to allow robots to use optical flow as part of more general algorithms for simultaneous localization and mapping in unstructured and complex environments such as collapsed buildings. In such environments, we believe that odometry based on wheel motions, such as the use of shaft encoders, provide little to no useful information. For example, attempting to climb over a pile of rubble results in so much wheel slippage, and the slippage itself is too unpredictable, to even attempt to maintain an accurate position using these methodologies.

REFERENCES

[1] Gideon P. Stein, Ofer Mano, and Amonon Shashua. A robust vehicle ego-motion. In *Proceedings of IV-2000*, 2000.

[2] Zhengyou Zhang. Estimating motion and structure from correspondences between line segments between two perspective images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(12):1129–1139, 1995.

[3] Richard Szeliski, P.H.S. Torr. Geometrically Constrained Structure from Motion: Points on Planes. Microsoft Research, One Microsoft way, Redmond, WA

[4] Stephen Se, David Lowe, Jim Little. Mobile Robot Localization and Mapping with Uncertainty using Scale-Invariant Visual LandMarks. University of British Columbia, Vancouver B.C. Canada

[5] Robert Sim, Gregory Dudek. Mobile Robot Localization from Learned Landmarks Centre for Intelligent Machines, McGill University, Montreal, QC

[6] Stephen Se, David Lowe, Jim Little. Vision-based Mobile Robot Localization And Mapping using Scale-Invariant Features. University of British Columbia, Vancouver B.C. Canada

[7] Brad Lisien, Deryck Morales, David Silver, George Kantor, Ioannis Rekleitis and Howie Choset. Hierarchical Simulitaneour Localization and Mapping. Carnegie Mellon University, Pittsburgh, PA 15217.

[8] Yi Ma, Jana Kosecka, Shankar Sastry. Vision Guided Navigation for A Nonholonomic Mobile Robot *AI Robotics Vision Siminar*. October 1996, Berkeley, California, USA