STRUCTURE FROM MOTION USING OMNI-DIRECTIONAL VISION AND CERTAINTY GRIDS

A Thesis

by

STEVEN REY ORTIZ

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

August 2004

Major Subject: Computer Science

STRUCTURE FROM MOTION USING OMNI-DIRECTIONAL VISION AND CERTAINTY GRIDS

A Thesis

by

STEVEN REY ORTIZ

Submitted to Texas A&M University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Approved as to style and content by:

August 2004

Major Subject: Computer Science

ABSTRACT

Structure From Motion Using Omni-directional

Vision and Certainty Grids. (August 2004)

Steven Rey Ortiz, B.S., Texas A&M University

Chair of Advisory Committee: Dr. Ricardo Gutierrez-Osuna

This thesis describes a method to create local maps from an omni-directional vision system (ODVS) mounted on a mobile robot. Range finding is performed by a structure-from-motion method, which recovers the three-dimensional position of objects in the environment from omni-directional images. This leads to map-making, which is accomplished using certainty grids to fuse information from multiple readings into a two-dimensional world model. The system is demonstrated both on noise-free data from a custom-built simulator and on real data from an omni-directional vision system on-board a mobile robot. Finally, to account for the particular error characteristics of a real omni-directional vision sensor, a new sensor model for the certainty grid framework is also created and compared to the traditional sonar sensor model.

To my loving wife, Kirsten Joy

ACKNOWLEDGMENTS

The author would like to first thank his wife for her love, support, and countless hours of building the real-life boxworld, assisting in experiments, and editing this thesis; his parents for their love, encouragement, and financial support; his mother for last-minute editing; his advisor, Dr. Ricardo Gutierrez-Osuna for many great research ideas, guidance, support, funding, and hours of editing; Dr. Nancy Amato and Dr. Reza Langari for their participation and feedback; Dr. Yoonsuck Choe for serving on the thesis committee; Dr. Shree Nayar for suggesting Remote Reality's OneShot360^{TM} as an off-the-shelf omni-directional vision system; the members of the Pattern Recognition and Intelligent Sensor Machines (PRISM) Laboratory, including Dr. Alex Perera-Lluna, Dr. Takao Yamanaka, Agustin Gutierrez-Galvez, Barani Raman, and Marco Zavala, for their help; and finally, Dr. Sherif T. Noah, undergraduate research advisor, whose excellent first research experience in the study of chaos theory inspired the author to pursue this graduate degree. Thank you all!

This work was also supported by a generous donation from Applied Materials and NSF CAREER Award 9984426/0229598.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

FIGURE Page

CHAPTER I

INTRODUCTION

Navigation is a critical ability for any robot that claims to be mobile [5]. Unfortunately, navigation is also one of the most difficult functions for a robot to perform, as recently demonstrated during the DARPA Grand Challenge, a race where teams competed to develop the first autonomous vehicle to navigate the rugged 142 mile course across the Mojave desert in fewer than 10 hours, using only on-board sensors and maps. This was the first year of the competition which was designed to push unmanned vehicle technology forward, and the difficulties of navigation were underscored when none of the 14 autonomous vehicles completed more than 7 miles of the course [6]. Navigation is also the primary reason for map making. Although it is possible for robots to navigate without maps, using a reactive control paradigm [7, 8] or topological path planning [9, 10], many mobile robots rely upon maps for localization and path planning. Therefore, map making is still an important component in mobile robotics research.

This research centers on a novel method of map-making that combines omnidirectional vision, structure from motion, and certainty grids [11, 4]. Each of these topics are covered in separate chapters of this thesis, but first the reader is presented with background material on other range finding methods, the distinction between stereo vision versus structure from motion, a review of related research, and a brief description of the remaining chapters.

The journal model is *IEEE Transactions on Automatic Control.*

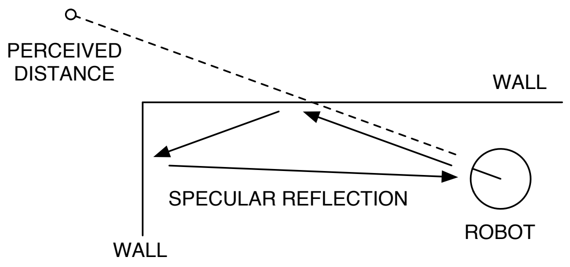
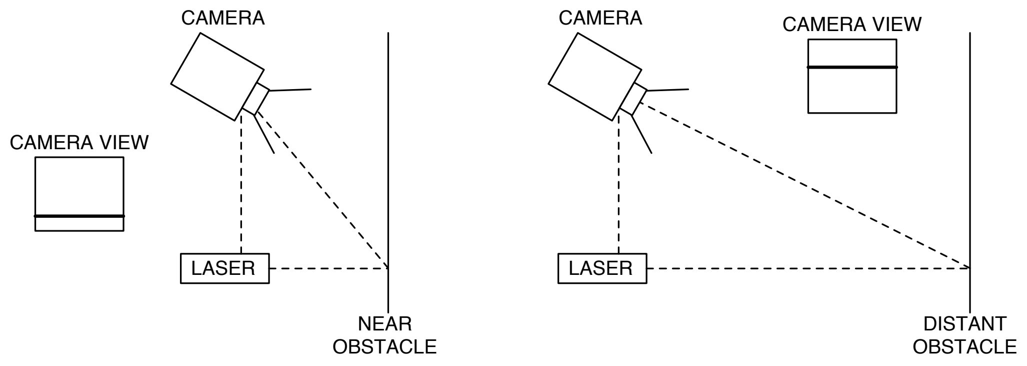


Fig. 1. Specular reflection problem

A. Range Finding

To create a map, a robot must be able to determine the distance between itself and an obstacle. This is known as range finding. Generally, range finders are based on either ultrasonics or electromagnetic irradiation [12]. Ultrasonic transducers (sonar) have traditionally been used as an inexpensive and simple means of range finding. Most operate by measuring the time-of-flight for a pulse of high-frequency ultrasound to travel from an emitter to an obstacle and back to its receiver. However, there are several drawbacks to sonar. Its detection cone is usually too wide to achieve high-resolution maps. Sonar is also susceptible to specular reflections, a problem that occurs when the ultrasound returns via an indirect path. Oftentimes, when at shallow angles of incidence, a pulse will reflect away from the receiver and either never return, or return after contacting a more distant obstacle, as shown in Fig. 1 [13]. In either case, the sensor incorrectly reports a clear path where there is actually an obstacle. Lastly, since sonar is an active sensor, it interferes with other sonar (crosstalk), which can become a problem in multi-robot scenarios [14].

Similar to sonar, some light-based range finders measure time-of-flight of a laser beam [15]. However, the speed of light is roughly 3.0×10^8 m/s, so the time measurements must be extremely accurate to recover useful distance measurements. This



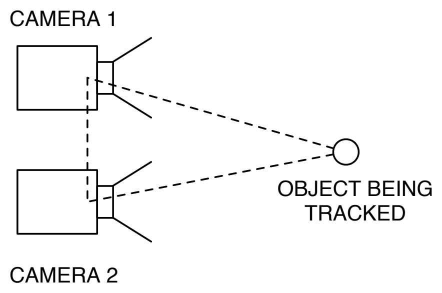
(a) The laser light appears low in the camera's image of obstacles near the robot

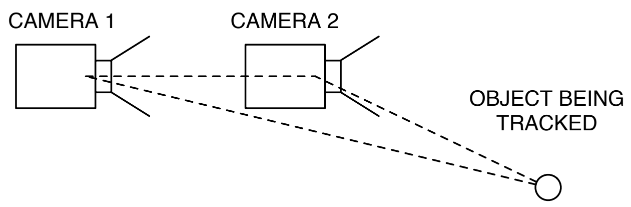
(b) The laser light appears high in the camera's image of obstacles far from the robot

Fig. 2. Active triangulation method

accuracy is possible using the same time spectroscopy technique often used in nuclear physics [16]. However, the specialized equipment to measure time disparity causes the sensor to be significantly more expensive [17]. Despite the high price tag, the better accuracy and finer resolution of time-of-flight laser rangefinders have made them the de-facto standard for high-end mobile robot platforms [18].

Rather than measuring time-of-flight, light can also be used to recover distance by active triangulation [19]. In this method, a light stripe is produced from one position and a camera views the stripe from a separate position, as shown in Fig. 2. The distance to obstacles can be calculated based on the position of the stripe in the camera image. This is one of the earliest light-based techniques and achieves good results, the accuracy of which depends mostly upon the disparity between the camera and the light source. Unfortunately, this is also an active sensor, and will cause interference in a multi-robot scenario.





(a) camera views are side-by-side, in a typical stereo vision arrangement

(b) camera views are collinear, which is only possible using a structure-from-motion method

Fig. 3. Passive triangulation method: shows the triangle geometry formed by the object being tracked and two camera positions

B. Stereo Vision vs. Structure From Motion

Light can also be used to recover distance by passive triangulation. There are two vision based methods for recovering three-dimensional data: stereo vision and structure from motion [20], both of which lead to the same geometrical relationships. However, stereo vision relies upon spatial information between two cameras, while structure from motion uses temporal information within a monocular sequence of images. Both methods result in two images from different camera positions. Correspondence is established between these images, and the distance to objects is calculated at each point of correspondence using triangulation [21]. Fig. 3 shows the geometrical relationships typical of stereo vision and structure-from-motion scenarios. Most stereo vision methods are configured within the binocular arrangement shown in Fig. 3(a), while structure-from-motion methods can use any arrangement of camera positions. The configuration shown on Fig. 3(b) would be typical of a time-to-impact sensor, where the optical flow field produced by the camera is directly used to estimate the time until impact with an obstacle [22]. To better understand structure from motion, imagine driving down a highway while looking out the side window. Objects closer to the car, such as traffic signs, appear to pass by quickly, while objects further away, like mountains or buildings, appear to move more slowly within the image plane. This phenomenon, where the perceived motion of objects across the field of view provides depth cues, is called motion parallax [23]. Structure from motion has some benefits over stereo vision. Since structure from motion requires only one camera, it can be less expensive to implement. Also, it is possible to compare multiple images from many positions and report the average distance, which may produce more robust results. However, there are also some drawbacks to using structure from motion. The implicit assumption that the scene does not change from one image to the next may not be valid, producing erroneous distance measurements for objects in motion. Furthermore, the distance to points near the line connecting one viewpoint to the next cannot be measured well because there is little change in that region as the camera moves. Therefore, a mobile robot relying upon structure from motion has poor information about obstacles along the line of travel, which is usually the area of highest concern to perform obstacle avoidance. For these reasons, there is a trend in computer vision to combine methods, using both structure from motion and stereo vision [24].

C. Literature Review

Omni-directional vision is a popular research area, with many different groups having completed a variety of studies in this field [25]. A number of studies have used individual omni-directional images to extract useful structure. Clérentin et al. [26] used a method of active triangulation to find the distance to objects using a laser striper and an ODVS. In their study, they found the omni-directional active triangulation method to produce excellent results that were better suited to segmentation than sonar data. Sekimori et al. [27] used an ODVS to distinguish between floor regions and other objects. By focusing solely on open floor space, the distance to objects was directly proportional to the amount of floor region visible in that direction. This made obstacle avoidance fast and reliable.

Yagi et al. [28] developed an omni-directional image sensor called COPIS (conic projection image sensor), and used it on a mobile robot for collision avoidance with either static or moving objects. COPIS determines the direction of the relative velocity between the robot and an object rather than determining the object's location relative to the robot. Since the conic mirror does not have a single viewpoint (refer to chapter II for further details) it is rarely used for triangulation or map-making, but knowing the relative direction of nearby obstacles provides sufficient information for collision avoidance, as demonstrated.

Winters et al. [29] used a spherical ODVS mounted high atop a Labmate mobile robot. This provided a "bird's eye view," which was used for topological navigation and visual path planning. In order to perform topological navigation, nodes of the topological map were constructed using Principal Components Analysis [30]. The robot traveled between nodes using a corridor-following behavior. To accomplish visual path planning, the robot used a closed-loop control to travel a specified course relative to known landmarks whose relative position and orientation were visible from the camera.

Stratmann [31] compared four methods of calculating optical flow on omni-directional images taken on a mobile robot: Anandan [32], Horn and Schunck [33], Lucas and Kanade [34], and Fleet and Jepson [35]. Stratmann compared the optical flow field produced by each method and qualitatively concluded that the Fleet

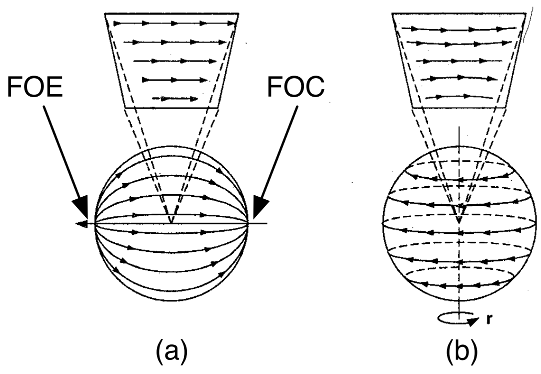


Fig. 4. Motion fields projected onto a sphere for (a) translation and (b) rotation (from [1]). Although the planar flow fields are nearly identical, the spherical flow fields are completely different, which illustrates why omni-directional cameras are more useful for extracting egomotion. Note the translational motion (a) results in a focus of expansion (FOE) and a focus of contraction (FOC) while purely rotational motion (b) has neither.

and Jepson method [35] produced the most accurate results, followed closely by the Lucas-Kanade method, and then the Horn-Schunck method. The Anandan method performed too poorly to produce a useful optical flow field. Although the Fleet and Jepson method performed the best, this method required large amounts of computation time and is not suited for real-time robotic applications. Stratmann's research did not employ any feature selection process, which could have improved the results of the Lucas-Kanade method in particular.

Fig. 4 illustrates why omni-directional vision systems are more capable of calculating ego-motion than conventional cameras. This is primarily due to their 360-degree horizontal field of view, which usually contains both the focus of expansion (FOE) and the focus of contraction (FOC) inside a single image. The FOE is the point from which the image appears to flow outward, and similarly, the FOC is the point into which the image appears to flow. Gluckman and Nayar [1], as well as Vassallo, Santos-Victor and Schneebeli [36], used omni-directional cameras to calculate ego-motion from the position of the FOE and FOC. The first step was to find the optical flow field, for which both studies employed the Lucas-Kanade method [34]. The motion field was then mapped to a unit sphere. Finally, the rotation and the direction of translation was calculated by one of several methods: Bruss and Horn [37], Zhuang et. al. [38], or Jepson and Heeger [39]. From Gluckman and Nayar's comparison of these three methods, it appears that the iterative non-linear minimization method by Bruss-Horn is the most accurate and stable method. An interesting parallel between their research and the one presented in this thesis is that the Bruss-Horn method actually removes the depth dependence of the motion field, while the aim of this work was to recover depth.

Zhang and Blum [40] studied a system of two stationary omni-directional cameras in a surveillance application. Specifically, they registered portions of the images and used triangulation to extract the 3-D coordinates of the registered object. They observed that this technique suffered from large estimation errors when the registered object was nearly aligned with the two cameras, and when the registered object was far away. Their result supports the findings of this research, as well as the proposed sensor model for the certainty grid method, which will be presented in chapter V, section B.

Kawasaki et al. [2] mounted an omni-directional video camera onto a vehicle, and recorded video as the vehicle traveled in a straight line and at a constant speed. They repeated this experiment with both a parabaloidal and hyperboloidal mirror, and created a spatio-temporal volume by stacking the omni-directional images together as shown in Fig. 5. To greatly improve the three-dimensional information recovery, they combined the structure-from-motion method with a two-dimensional map of the city. Although their results are impressive, this technique is, unfortunately, not

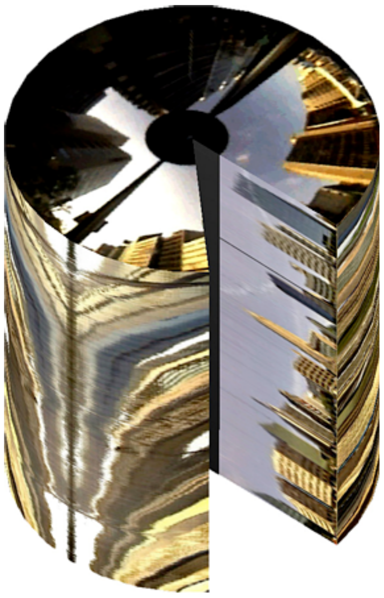


Fig. 5. Radial cross section of spatio-temporal volume (from [2])

suitable for a real-time mobile robot navigating in an unknown environment.

Chang and Hebert [41] used a single omni-directional camera to extract structure from motion, and compared their results to that of a conventional camera extracting structure from motion. Their results indicate that the wide field of view provided by the omni-directional camera generally leads to superior results. Their experimental setup and goals are the most similar to those of this thesis, but their findings left much room for exploration. The research in this thesis also differs significantly because it automatically finds and registers good features, then uses that information to build a local map that accounts for problems particular to this sensor.

This concludes the survey of research relevant to work that was done for this thesis. The best knowledge indicates that no other research in the field has explored the combination of omni-directional vision, structure from motion and certainty grids. Certainly, the work of all those mentioned above, as well as the work of many others, has made this exploration possible, and this research owes much to their contributions.

D. Thesis Organization

Having covered range finding, stereo vision versus structure from motion, and related research, this now concludes the introduction. The next three chapters of the thesis are focused on the specific components upon which this research builds, namely omni-directional vision, structure from motion, and certainty grids. Chapter V describes the specific method used by this research of combining omni-directional vision, structure from motion, and certainty grids. Chapter VI presents the experimental results, and finally chapter VII provides a discussion of conclusions and future work.

CHAPTER II

OMNI-DIRECTIONAL VISION

Since a conventional camera has a limited field of view, the designer must carefully consider to which direction the camera should point point. The most useful information for obstacle avoidance is directly ahead, but the sensor performs best when facing to the side. To overcome this difficulty, an omni-directional vision system is used to provide a complete 360-degree hemispheric view, allowing both directions perpendicular to the motion, as well as the forward and backward views, to be captured on a single image [42].

Although truly omni-directional vision systems provide a spherical 360-degree field of view, the term omni-directional vision is also used in reference to cylindrical, wide-angle and hemispheric imaging [43]. An omni-directional view field can be obtained with several techniques, which include (1) mosaicing from multiple cameras, (2) mosaicing from a single rotating camera, (3) using a wide-angle fish-eye lens, and (4) imaging the scene through a combination of mirrors and lenses, a technique known as catadioptrics [44]. Of these four techniques, the catadioptric system using a single camera is most commonly used in omni-directional vision research because of its large hemispherical viewing area and relatively straightforward implementation. Several examples of such systems are shown in Fig. 6.

A. Catadioptric Vision Systems

In a catadioptric vision system, light is emitted from objects in the world, reflected by a mirror, refracted by one or more lenses, and finally measured by an imaging device. The refraction and measurement is performed by the camera system. Specifically, the lens of the camera gathers light from particular directions, and the imaging sensor

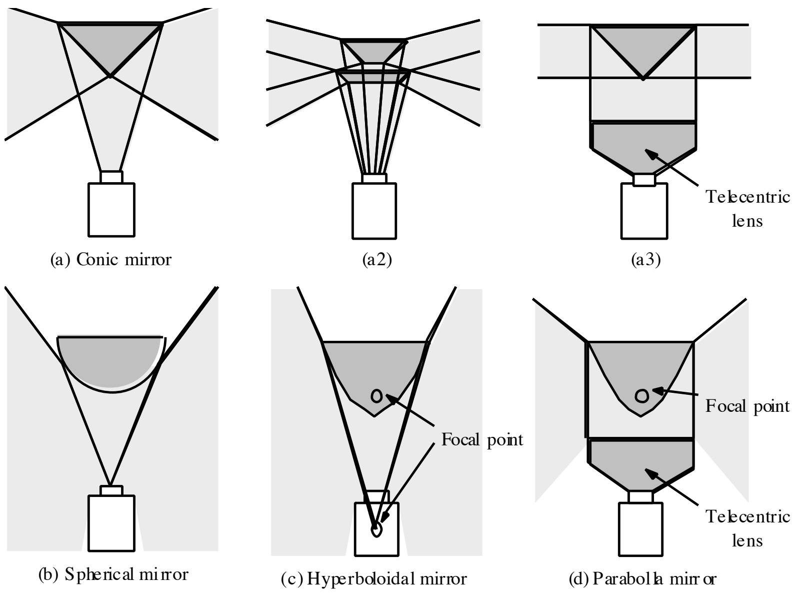
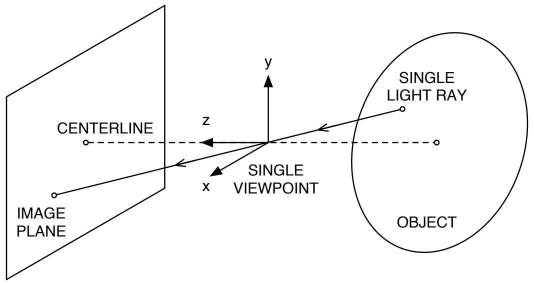
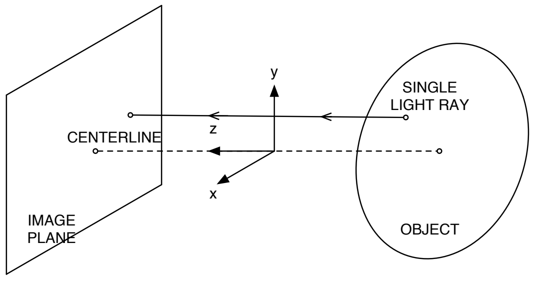


Fig. 6. Sample omni-directional mirrors (from [3]): (a) conic (b) spherical (c) hyperboloidal (d) paraboloidal



(a) perspective projection: measured light rays pass through the single viewpoint



(b) orthographic projection: measured light rays travel parallel to the certerline

Fig. 7. Comparison of perspective and orthographic projection models

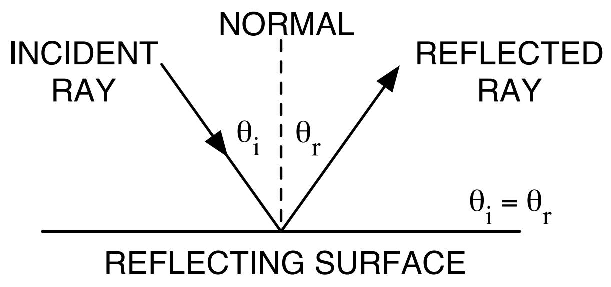


Fig. 8. The Law of Reflection, showing the incident angle θ_i equal to the reflecting angle θ_r

measures the amount of light energy on each part of of the imaging device which defines the imaging plane. The manner in which this light is gathered defines the projection model[1]. The most familiar projection model is the perspective projection shown in Fig. 7(a), where every measured ray of light travels through the single viewpoint, also known as the center of projection. Although most camera systems have a perspective projection, some cameras use a telecentric lens, which provides an orthographic projection, as shown in Fig. 7(b). The orthographic projection captures only those light rays traveling parallel to the centerline of the camera.

Another important consideration is the angle at which incident light rays reflect from the mirror's surface. This is defined by the Law of Reflection, which states that the bisector of the incident ray and the reflecting ray shall be normal to the surface, as shown in Fig. 8.

From this Law of Reflection, Baker and Nayar [44] showed that only two types of mirror shapes will result in a practical omni-direcitonal vision system with a single viewpoint: a hyperboloidal mirror with a perspective camera and a paraboloidal mirror with an orthographic camera. These mirror shapes are shown in Fig. 9, along

[1]The term projection model refers to the projection of light onto the imaging device.

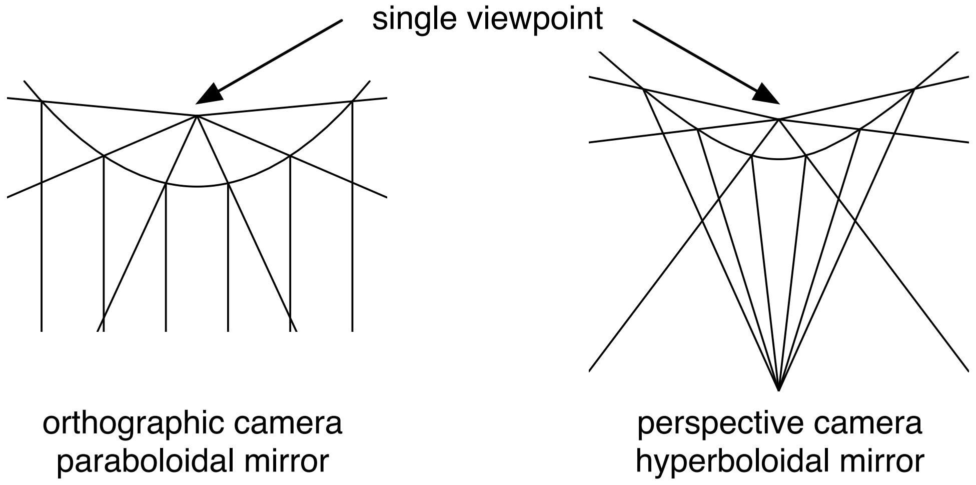
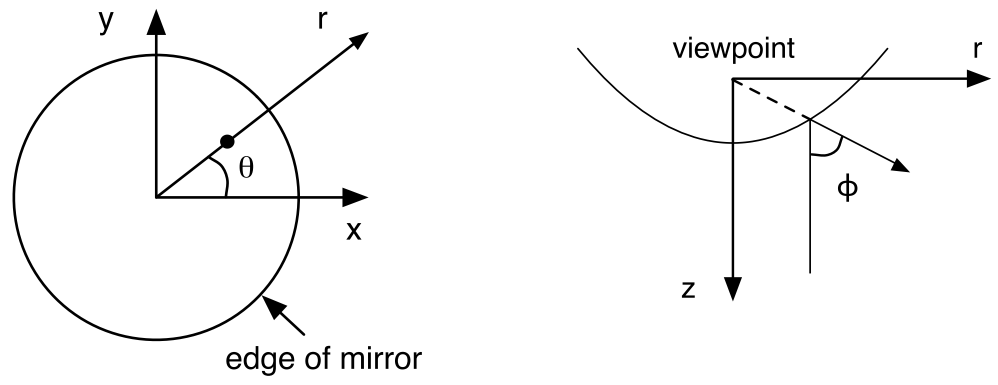


Fig. 9. Example solutions of the single-viewpoint constraint

with several examples of incident and reflected rays. This single viewpoint simplifies the structure-from-motion problem since every visible object can be traced back to this single effective viewpoint, and it is required for the creation of proper perspective images from an omni-directional image [45]. For this reason, hyperboloidal and paraboloidal are the most popular shapes, but paraboloidal mirrors have a number of advantages over hyperboloidal mirrors. The calibration of a paraboloidal mirror requires just one parameter instead of two; and since it uses an orthographic projection rather than perspective, the axis of the mirror can be translated arbitrarily, provided its centerline remains parallel to the orthographic projection. This makes the calibration and setup of the paraboloidal mirror more straightforward, and so it was chosen for this research. The tradeoff in selecting a paraboloidal mirror is that the orthographic projection requires a telecentric lens attachment, which is usually large and expensive.



(a) Omni-directional camera view

(b) Real-world geometry of mirror

Fig. 10. Conversion from image coordinates to a real-world direction. Note that θ is the same in both image space and the real-world

B. Equations for a Paraboloidal Mirror

The ODVS integrated for this thesis combines a paraboloidal mirror and lenses to provide a hemispheric view around the robot and a single viewpoint. For the paraboloidal mirror, the following equations can be used to convert from any position within the image to a direction in the real world, as shown in Fig. 10. Note that a different but equivalent set of equations could be used for a hyperboloidal mirror.

$$r^2 = x^2 + y^2 \tag{2.1}$$

$$\theta = \tan^{-1}(y/x) \tag{2.2}$$

$$z = (h^2 - r^2)/2h \tag{2.3}$$

$$\phi = \tan^{-1}(r/z) \tag{2.4}$$

where x and y are the image-plane coordinates of the pixel, r is the Euclidean distance of the pixel to the origin, which lies on the central axis of the mirror, z traces the surface of the paraboloidal mirror, and θ and ϕ define the direction of the ray in spherical coordinates. θ is the orientation, ranging from 0 to 360 degrees, and ϕ is the pitch, where $\phi = 0$ points straight down and $\phi = 90$ points out to the horizon. Finally, h is a calibration parameter of the vision system.

CHAPTER III

STRUCTURE FROM MOTION

For a structure-from-motion technique to work, it must be possible to determine the perceived motion of objects as the camera moves. This motion is vital to many computer vision applications, and is commonly referred to as optical flow. In turn, the problem of calculating optical flow leads to the problem of image registration. Since optical flow is central to calculating structure from motion, it is carefully examined in the next section. Then the details of the Lucas and Kanade method [34] are derived in detail before considering the Kanade-Lucas-Tomasi (KLT) feature tracker [46] that was used in this research. Finally, the depth by triangulation method is examined to complete the overview of the structure-from-motion process.

A. Optical Flow

The increased availability of powerful desktop computers has encouraged research in 3D computer vision. As a result, there are now a number of methods for calculating optical flow, six of which were compared in an authoritative study by Barron et al. [47]. These six optical flow methods can be grouped into four categories: differential methods, region-based matching methods, energy-based methods, and phase-based methods. They will be summarized here with special attention given to differential methods, to which the Lucas and Kanade method [34] used in this research belongs.

Differential techniques calculate optical flow from the spatiotemporal derivatives of image intensity. Examples of such techniques include the Horn and Schunck method [33], the Lucas-Kanade method [34], and the Uras et al. method [48]. These differential techniques assume that there is a conservation of intensity. In other words, as the image translates, the pixels remain at the same intensity. This is generally, but

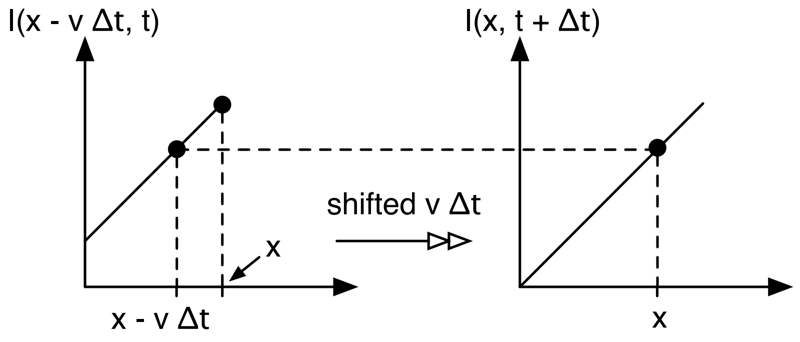


Fig. 11. Example of image translation, showing the image intensity before and after a translation of $v\Delta t$ units, where Δt is the duration of time between images and x is a position of the image

not always, a valid assumption. As a counter-example, consider the task of tracking an object that moved from behind a shadow into direct light. While the shape of this object is the same, its intensity is brighter in the direct light and so the constant intensity assumption is not valid in this particular case. Nevertheless, the changes in position and time are typically small enough so that the constant-intensity constraint is valid. Fig. 11 illustrates the constant intensity constraint, which is formulated on:

$$I(x - v\Delta t, t) = I(x, t + \Delta t) \tag{3.1}$$

where v is the optical flow vector, and $I(x, t)$ is the intensity of the image at position x and time t. To isolate the optical flow vector v, the first-order Taylor series expansion of equation (3.1) can be computed:

$$I(x, t) + (\bigtriangledown I(x, t))^T(-v\Delta t) = I(x, t) + \frac{\delta}{\delta t} I(x, t)\Delta t \tag{3.2}$$

where $\frac{\delta}{\delta t} I(x, t)$ is the derivative of intensity w.r.t. t, and $\bigtriangledown I(x, t)$ is the gradient of intensity w.r.t. x. After simplification and rearrangement equation (3.2) becomes:

$$(\bigtriangledown I(x,t))^T v + \frac{\delta}{\delta t} I(x,t) = 0 \tag{3.3}$$

from which the optimal flow vecton v can be computed.

The Horn and Schunck method and the Lucas and Kanade method both aim to satisfy the gradient constraint equation (3.3) by best-fitting a uniform optical flow over a fixed window, in the least-squares case $v = argmin[\bigtriangledown I(x,t))^T v + \frac{\delta}{\delta t} I(x,t)]^2$, as derived in the following section. However, the Horn and Schunck method calculates optical flow over a time span, and includes an optical flow smoothing term in the best-fit calculation. Instead, the Lucas and Kanade method is used between just two frames, but also uses a windowing function to give more influence to the center of the window. The Uras et al. method is a second-order method, assuming conservation of $\bigtriangledown I(x,t)$, which yields two additional equations. With three equations, the two unknown components of the optical flow vector can be calculated at any single point, rather than assuming a uniform optical flow over a window to produce a sufficient number equation. However, second-order derivatives cannot be measured very accurately and often lead to sparse and less accurate results than first-order methods.

There are several additional techniques for calculating optical flow. The most intuitive method is the region-based matching technique. This approach simply takes a window on the first frame and tries to shift it until it best matches the second frame. One example of a region-based method is the Anandan method [32], which uses a coarse-to-fine pyramid matching strategy to allow for large shifts but still converges on the best match without testing every possible shift. There are also energy-based methods, sometimes referred to as frequency-based methods. Among those, the method of Heeger [49] determines optical flow using the energy output of velocity-tuned filters in the Fourier domain. Lastly, there are phase-based methods, such as that of Fleet and Jepson [35], where velocity is determined by the phase behavior of band-pass filter outputs.

In Barron's study [47], all six of the methods mentioned above were systematically tested and compared to each other. The most accurate methods were the Lucas-Kanade method and the Fleet-Jepson method. However, the Fleet-Jepson method is very computationally demanding because it requires a large number of filters, while the Lucas-Kanade method is computationally very simple. Due to its simplicity and accuracy, the Lucas-Kanade method is often the best and most popular method for calculating optical flow.

B. Lucas and Kanade Method Revisited

For over 20 years, the Lucas-Kanade algorithm has been used to solve the image registration problem. Image registration can include rotations and deformations in the image, but in the simplest case, the image registration problem is applied to matching part of one image onto the corresponding part of another image, where some small translation has occurred between the two images. Other methods of solving this problem were discussed in the previous section, but the Lucas-Kanade method was chosen because of its speed and accuracy.

The reason this algorithm is relatively fast is because it uses an iterative gradient-based technique similar to a Newton-Raphson root-locating method. The following example may clarify the rationale behind this method. Imagine being on the side of a long, tall hill with a reliable Swiss Army Knife Altimeter Plus that displays the current elevation of 500 feet. The final destination is 600 feet and the hill has a steady 10% incline, as shown in Fig. 12. The necessary calculation would indicate an additional 100-foot climb over a distance of 1,000 feet, thus successfully using the

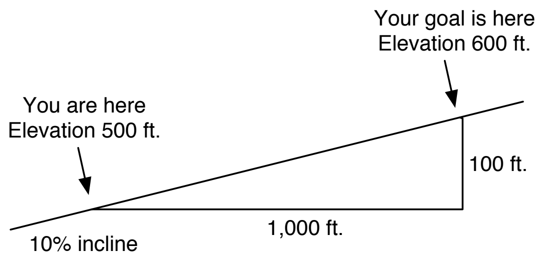


Fig. 12. Sample calculation, using a gradient technique to determine the necessary translation from your current position with an elevation of 500 feet to a desired location with elevation of 600 feet, along a steady 10% incline

gradient-based technique. Essentially, that is what the Lucas-Kanade method does at each pixel of the image that it is trying to match. At each pixel, the algorithm considers the image gradient, the current pixel intensity, and the desired intensity. It then uses the method of least squares to find a translation in both the x and y direction that will bring most of the pixels to the desired intensity. Since the gradient of most images is not constant, the method will have errors, so the process is repeated at the new position until it converges on a matching location.

In practice, a windowing function is often used in the least-squares calculation to give more influence to the center of the window. The algorithm will terminate after a fixed number of attempts. Additionally, there is often some preprocessing to provide smoother gradients throughout the image, typically a Gaussian smoothing.

Note that this method does not take into account any deformations in the image that occur due to the mirror's shape. One way to reduce errors caused by deformation is to first unwarp the image and then perform image registration, a process that is very fast. However, relatively small translations produce only slightly different deformations. In the author's experience, a small window (7x7 pixels) appears to work very well on the original omni-directional image. For this reason, there was no

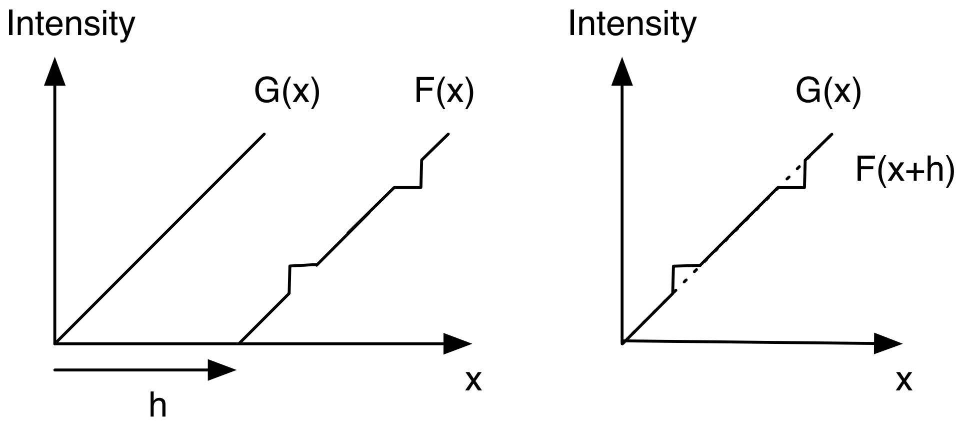


Fig. 13. Sample image registration: h is the optimal displacement such that $F(x+h)$ matches $G(x)$. Although this is not a perfect match, the dissimilarity ϵ from equation (3.4) is minimized.

need to add an unwarping step to the process used in this research.

The following is a formal derivation of the Lucas-Kanade method, which aims to minimize the dissimilarity between two images, as shown in Fig. 13. The dissimilarity ϵ is the sum of square errors:

$$\epsilon = \sum_{W} (F(x+h) - G(x))^2 \tag{3.4}$$

where h is the displacement from point x in image G to image F, and W is the tracking window. Using a first-order approximation for $F(x+h)$ produces:

$$\epsilon = \sum_{W} (F(x) + h^T \bigtriangledown - G(x))^2 \tag{3.5}$$

where $\bigtriangledown$ denotes the gradient of $F(x)$. From equation (3.5), the optimal image translation (or optical flow) h is determined by taking the derivative of ϵ w.r.t. h and setting it to zero. In the original paper by Lucas and Kanade, a minor error was discovered, so a complete derivation is presented here.

$$\frac{d\epsilon}{dh} = \sum_W 2(F(x) + h^T \bigtriangledown - G(x))\bigtriangledown = 0 \tag{3.6}$$

$\bigtriangledown$ is distributed, and the terms of the summation are rearranged:

$$\sum_W h^T \bigtriangledown \bigtriangledown = \sum (G(x) - F(x))\bigtriangledown \tag{3.7}$$

The left-hand-side is rearranged to isolate h:

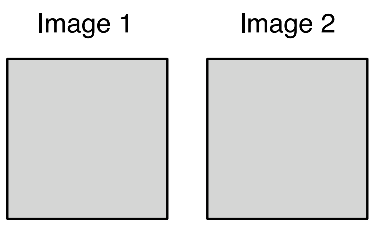
$$\sum_W (\bigtriangledown \bigtriangledown^T)h = \sum (G(x) - F(x))\bigtriangledown \tag{3.8}$$

and, finally, h is determined by premultiplying each side by $(\sum \bigtriangledown\bigtriangledown^T)^{-1}$:

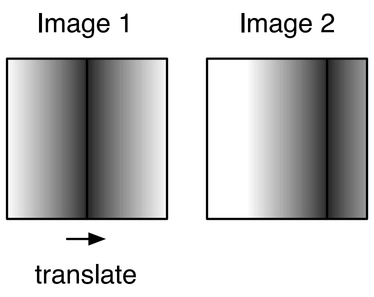
$$h = (\sum_W \bigtriangledown\bigtriangledown^T)^{-1} \sum_W (G(x) - F(x))\bigtriangledown \tag{3.9}$$

C. Good Feature Selection

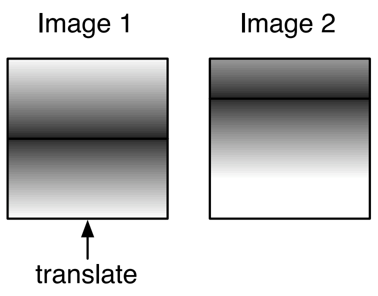
One difficulty with any method used for calculating optical flow is choosing the correct points at which to calculate the optical flow. Imagine a camera aimed at a solid white wall and translated some distance along the wall, as shown in Fig. 14(a). The second viewpoint will be a translated version of the first one, but both images will appear identical. Thus, the uniform intensity provides no useful matching information, and the calculation of optical flow is ill posed. Similarly, imagine a wall with a smooth horizontal gradient, as shown in Fig. 14(b). The camera again translates along the wall. Now it is simple to calculate the horizontal translation by shifting and matching the two images, but still there will not be enough information to tell how the camera moved in the vertical direction. Any striped pattern provides only a unidirectional component of optical flow, which may be misleading. For an image, or window of an image, to provide adequate tracking information, it must have structure in two



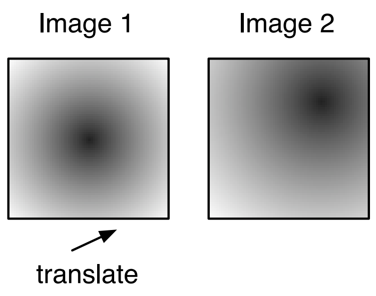
(a) no structure



(b) horizontal structure



(c) vertical structure



(d) bidirectional structure

Fig. 14. (a) Without adequate structure in two non-collinear directions, it is not possible to determine optical flow. Images (b) and (c) have unidirectional structure and can determine part of the optical flow, but there could be additional optical flow that is not visible from this structure. Only (d) has enough information to calculate optical flow in both directions.

non-collinear directions (both horizontal and vertical structure), as shown in Fig. 14(d).

Although the Lucas and Kanade algorithm solution converges quickly and works very well for tracking good features, it does not behave well at every place in the image because not all parts of the image contain enough information to be accurately positioned. To address this issue, Shi and Tomasi [46] proposed a feature selector specifically for the Lucas-Kanade method that is ideal by construction. This good feature selection is crucial to calculating a useful optical flow field. Without good feature selection, the alternative is to calculate optical flow at regularly spaced positions, and the calculated optical flow will likely be misleading at positions which lack adequate structure.

This feature tracker, commonly referred to as the Kanade-Lucas-Tomasi (KLT) [46] method, selects features that are best suited for the Lucas-Kanade tracking algorithm by finding windows where the $\sum \bigtriangledown \bigtriangledown^T$ matrix of equation (3.9) is well conditioned, meaning both of its eigenvalues are sufficiently large and of similar magnitude. Two small eigenvalues occur when all the pixels in the window are of constant intensity; one large and one small eigenvalue indicate a unidirectional pattern.

By tracking only the good features, the overall quality of the optical flow measurements is greatly improved, leading to a much more accurate recovery of three-dimensional structure. The early stages in this research used a custom-built implementation of the Lucas-Kanade algorithm, which appeared qualitatively to produce a reasonable optical flow field. However, the custom-built implementation calculated optical flow at evenly spaced intervals, and many of those windows would not contain enough information to calculate optical flow. Although there was code to recognize a complete lack of gradient and to ignore those windows, the unidirectional patterns would often cause erroneous results. After replacing this custom implemen-

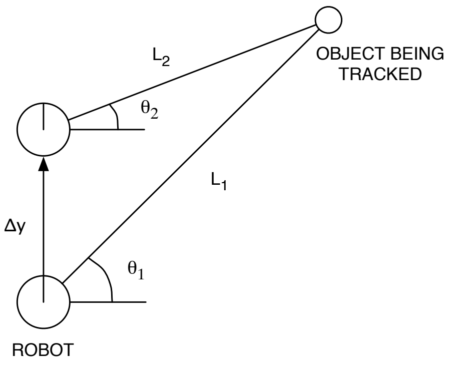


Fig. 15. Two-dimensional triangulation problem

tation with the freely available and popular Kanade-Lucas-Tomasi tracker developed by Stan Birchfield at Stanford University [50], the overall quality of the results improved, resulting in fewer errors and a more dependable optical flow field.

D. Depth from Triangulation

The previous sections have described how to select good features from an image, track these features from one frame to the next, and transform the image space coordinates to spherical coordinates. The next step is to use the spherical coordinates from each of the known positions to calculate the relative position of the object being tracked. This is done by forming a triangle between the object being tracked, the position of the robot at the time of the first image and the position of the robot at the time of the second image. First consider the two-dimensional case shown in Fig. 15. Given the distance the robot traveled by, and the angle from the robot to the object at each position, the triangle is completely defined by two of its angles θ_1 and θ_2 and the length of one side. In this case, the distances from the robot to the object L_1 and L_2

are determined by solving the following linear equations:

$$\begin{bmatrix} \cos\theta_1 & -\cos\theta_2 \\ \sin\theta_1 & -\sin\theta_2 \end{bmatrix} \begin{bmatrix} L_1 \\ L_2 \end{bmatrix} = \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \tag{3.10}$$

The three-dimensional case is more difficult to visualize, but leads to similar equations. Note that the three-dimensional case results in an over-constrained system because there are now three equations (one for each dimension of the world) and only two unknowns (one for each dimension of the image). If the rays extending from the viewpoint to the object actually intersect, there is one solution for L_1 and L_2 that will completely satisfy all three equations. However, the rays do not necessarily intersect, so the solution is determined using the pseudo-inverse to find the best fit from the following system of linear equations:

$$\begin{bmatrix} \sin\phi_1\cos\theta_1 & -\sin\phi_2\cos\theta_2 \\ \sin\phi_1\sin\theta_1 & -\sin\phi_2\sin\theta_2 \\ \cos\phi_1 & -\cos\phi_2 \end{bmatrix} \begin{bmatrix} L_1 \\ L_2 \end{bmatrix} = \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} \tag{3.11}$$

Note that the above equations assume a fixed orientation of the robot between frames. If the orientation changes, the rotation should be accounted for by adding it to θ_1, essentially changing the coordinate system of the first frame to match the second.

CHAPTER IV

CERTAINTY GRIDS

Certainty grids, also known as evidence or occupancy grids, are a method for fusing a large number of noisy depth estimates into a usable map. The certainty grid itself is a two (or sometimes three) dimensional division of the world space into cells. Each cell is associated with a probability of being occupied, $p[o]$. Since a cell can only have one of two states, either occupied or empty, the probability that a cell is empty is $p[e] = p[\neg o] = 1 - p[o]$. Therefore, the probability for either state is completely defined by one number, $p[o]$.

Each reading from a range sensor is used to update two areas of the certainty grid, as shown in Fig. 16. The area near the reported reading has its occupancy probability increased because the reading indicates there is probably an obstacle in this region. The area between the sensor and reported reading has its occupancy probability decreased because an obstacle in this area would have produced a shorter range reading, and therefore the likelihood of these cells being occupied is lower. For a computer to calculate the new occupancy probabilities, it must have a numerically defined sensor model whose shape matches either empirical observations, or a physics-based model.

To actually combine a new sensor reading with the sensor model and certainty grid probabilities requires an update rule. There are three methods for updating cell probabilities: Bayesian [51, 4], Dempster-Shafer [52], and Histogrammic in Motion Mapping (HIMM) [53]. The Bayesian method for combining probabilities was developed by Elfes and Moravec at Carnegie Mellon University [54, 11]. This is an attractive method for research due to its solid foundation in traditional probability theory and its flexible sensor model. The details of its implementation will be fully

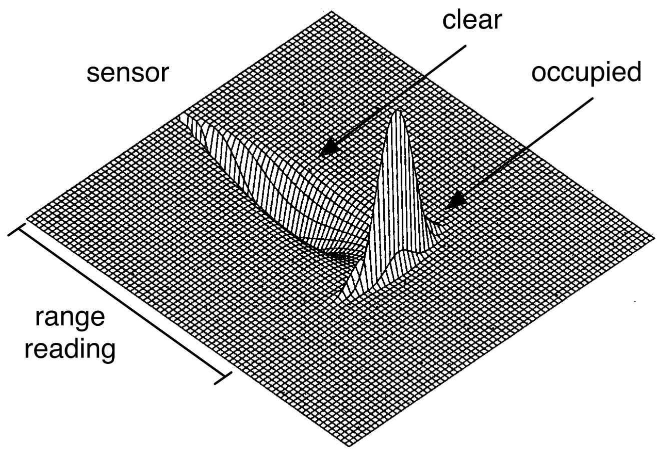


Fig. 16. Certainty grid profile for a two-dimensional Gaussian-error sensor model (from [4]). The area between the sensor and the reported range is probably clear, while the area near the reported range is more likely to be occupied.

explained in the following chapter.

A. Dempster-Shafer and HIMM Update Methods

The Dempster-Shafer method is based on belief mass instead of probabilities. One distinguishing feature of the Dempster-Shafer method is a third state to represent ambiguity. Each cell can have a portion of its belief mass in the occupied, empty or "don't know" state. To combine belief functions, this method uses the Dempster's rule of combination [55]. This rule also produces a metric called the weight of conflict, which measures disagreement between readings. Although the rules of combination are different from the Bayesian method, the results are similar and the extra information provided by the "don't know" state and the conflict metric can be used to determine when and where more sensing is required to reconcile sensor readings [56].

HIMM is an ad hoc method of fusing new measurements designed for computational efficiency. It was developed by Borenstein and Koren at the University of

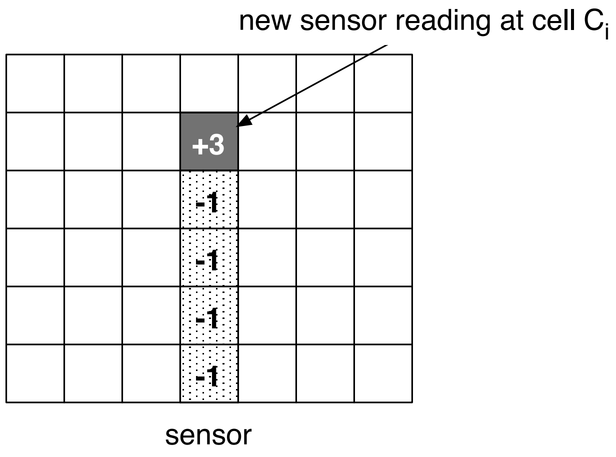


Fig. 17. Example of the HIMM update rule

Michigan and first implemented on their Cybermotion robot CARMEL (Computer-Aided Robotics for Maintenance, Emergency, and Life support). The certainty of a cell being occupied is represented as an 8-bit unsigned integer and, therefore, the range of values is integers from 0 to 15. An example of the HIMM update rule is shown in Fig. 17. Only those cells along the line of sight from the sensor to the new sensor reading at cell C_i are updated. The value of C_i is increased by 3 (up to a maximum of 15) and all the cell values between the sensor and C_i are reduced by 1 (down to a minimum of 0). Although this method lacks the theoretical underpinnings of the other methods, its flawless performance on its third and final attempt of the 1992 AAAI Mobile Robot Competition captured much attention and won the final day's main event [57].

B. Bayesian Update Method

The original method of Moravec and Elfes [11, 54] updates the probabilistic map using Bayes theorem and a probabilistic sensor model. The goal is to increase the occupancy probability of grid cells near a reported object, and to decrease the probability for those cells between the sensor and the object. Everything begins with Bayes' theorem to find the conditional probability that the state of a cell is occupied, given a set of reported sensor readings [54]:

$$p[o|\{r\}_{t+1}] = \frac{p[r_{t+1}|o \wedge \{r\}_t] \times p[o|\{r\}_t]}{p[r_{t+1}|o \wedge \{r\}_t] \times p[o|\{r\}_t] + p[r_{t+1}|\neg o \wedge \{r\}_t] \times p[\neg o|\{r\}_t]} \tag{4.1}$$

where o means the cell is in an occupied state, and $\{r\}_{t+1}$ is a history of every range reading, including the latest range reading r_{t+1}. This formula can be simplified by assuming that the range readings are strongly independent ($p[r_{t+1}|o \wedge \{r\}_t] = p[r_{t+1}|o]$). Thus:

$$p[o|\{r\}_{t+1}] = \frac{p[r_{t+1}|o] \times p[o|\{r\}_t]}{p[r_{t+1}|o] \times p[o|\{r\}_t] + p[r_{t+1}|\neg o] \times p[\neg o|\{r\}_t]} \tag{4.2}$$

Recalling that $p[o|\{r\}_t]$ is the last occupancy probability for the cell being updated and $p[\neg o|\{r\}_t] = 1 - p[o|\{r\}_t]$, the only unknown is $p[o|\{r\}_t]$. The simplest solution is to define the sensor model by a function $p[o|\{r\}_t]$. However, it is often more convenient to define the sensor model by $p[r|z]$, where z is the true distance to the obstacle. In this case,

$$p[o|r] = \frac{p[r|o] \times p[o]}{p[r|o] \times p[o] + p[r|\neg o] \times p[\neg o]} \tag{4.3}$$

To find $p[o|r]$, it is necessary to make assumptions about the world. $p[o]$ is the probability that an arbitrary cell is occupied. In a closed and cluttered environment $p[o]$ should be relatively high compared to an open environment with few obstacles. If no prior information is known $p[o] = p[\neg o] = \frac{1}{2}$ is often used, meaning a cell is just as likely to be occupied as it is to be empty.

To relate the sensor model $p[r|z_i]$ with the conditional distribution of cells, consider an arbitrary configuration of cell states. The first occupied cell C_i in front of the sensor will define the true distance z_i to an obstacle. Therefore,

$$p[r|C_i{=}o \wedge C_k{=}\neg o, \forall k < i] = p[r|z_i] \tag{4.4}$$

Since it is now necessary to consider every cell, the $p[o]$ notation has been replaced by $p[C_i{=}o]$ to denote the probability that cell i is occupied. The final piece of information in equation (4.3) $p[r|o]$ can now be found from equation (4.4).

$$p[r|C_i{=}o] = \sum_{\{G_i\}} p[C_i{=}o \wedge G_i] \times p[G_i|C_i{=}o] \tag{4.5}$$

where G_i is a specific configuration of all the cells except for cell i, and $\{G_i\}$ is the set of all possible configurations. Rather than considering each possible configuration of cell states individually, many configurations are grouped together based on the position of the nearest occupied cell for those configurations. This significantly reduces the number of computations required to obtain $p[r|C_i{=}o]$. Lastly, equation (4.5) is further simplified by assuming that the probability of a cell being occupied is independent of other cells being occupied:

$$p[G_i|C_i{=}o] = p[G_i] = \prod_{\forall k, k \neq i} p[C_k{=}s_k] \tag{4.6}$$

where s_k is the state of cell k in this configuration. Using equations (4.3), (4.4) and (4.6), $p[o|r]$ can be determined from $p[r|z]$. Hans Moravec's certainty grid implementation, which was used in this research, has a built-in two-dimensional sensor model defined by $p[r|z, \theta]$ where θ is the angle between the sensor reading and the obstacle (shown in Fig. 16). This is the sensor model originally used in this research, and later modified (see section V.B) to account for difficulty measuring optical flow near the line of travel.

$$p[r|z, \theta] = \frac{1}{2\pi\sigma_r\sigma_\theta} exp\left[-\frac{1}{2}\left(\frac{(r-z)^2}{\sigma_r^2} + \frac{\theta^2}{\sigma_\theta^2}\right)\right] \tag{4.7}$$

CHAPTER V

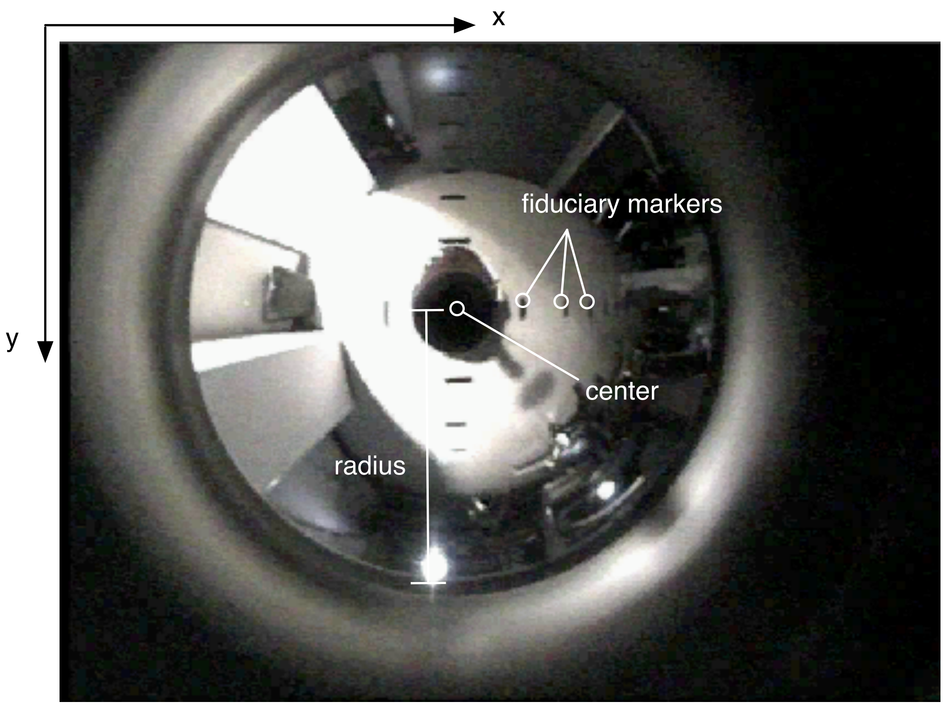
METHOD

The previous chapters have presented the details necessary to extract depth from motion using an omni-directional camera and to fuse these readings into a local map using certainty grids. This chapter will explain the camera calibration process, and then describe a proposed improvement to the certainty grid sensor model originally developed by Elfes and Moravec for sonar.

A. Camera Calibration

To calibrate the experimental vision system, fiduciary markers were placed at ground level around the robot at one foot increments in each direction, as shown in Fig. 18. Note that the camera's sensor and the mirror did not line up directly and although the mirror profile in real-life makes a perfect circle, its image is not one. To compensate for these differences, image coordinates were normalized so that $r = 1$ would outline the outer edge of the mirror. This normalization requires knowledge of the aspect ratio of the camera, the mirror radius, and the position of the mirror's centerline. The aspect ratio was determined to be the ratio of pixels between the horizontal and vertical markers. The radius of the mirror is taken to be half the number of pixels from the top to the bottom of the bounding circle. The centerline was the midpoint between each pair of markers in opposite directions.

Calculation of the h parameter was more involved. First, consider the optical flow that occurs when the robot moves one foot along a line of fiduciary markers, as shown in Fig. 19. Since the markers are already spaced one foot apart, their new image coordinates would be that of the neighboring marker. Using this hypothetical optical flow field, the three-dimensional structure of the scene can be recovered for a



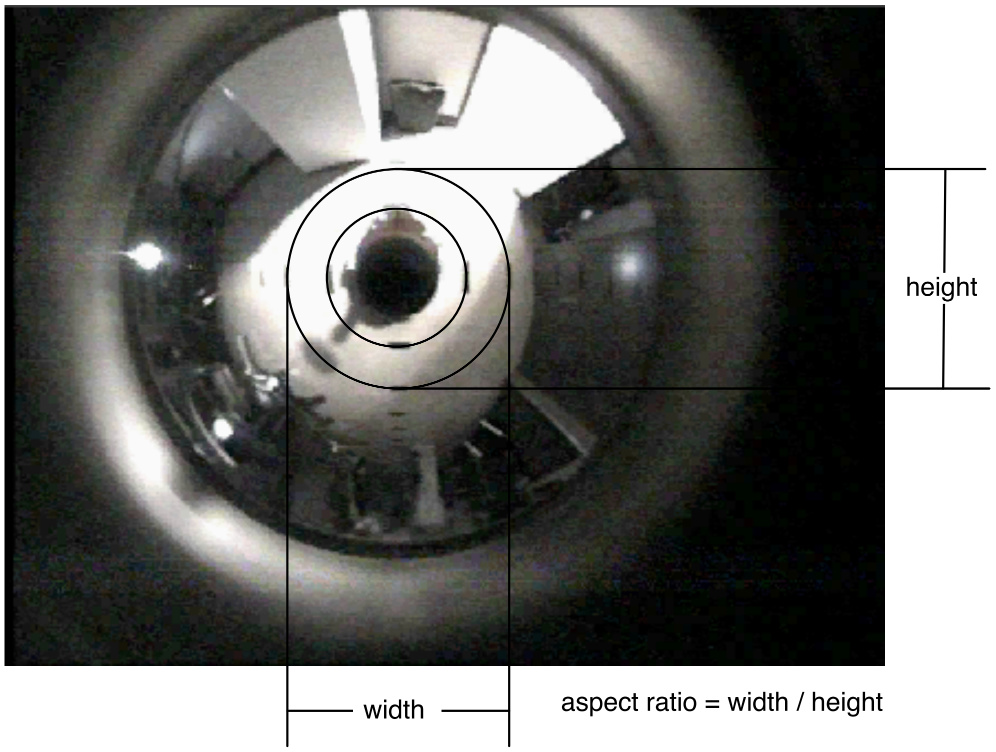


Fig. 18. Two calibration images with AmigoBot^{TM} facing two perpendicular directions

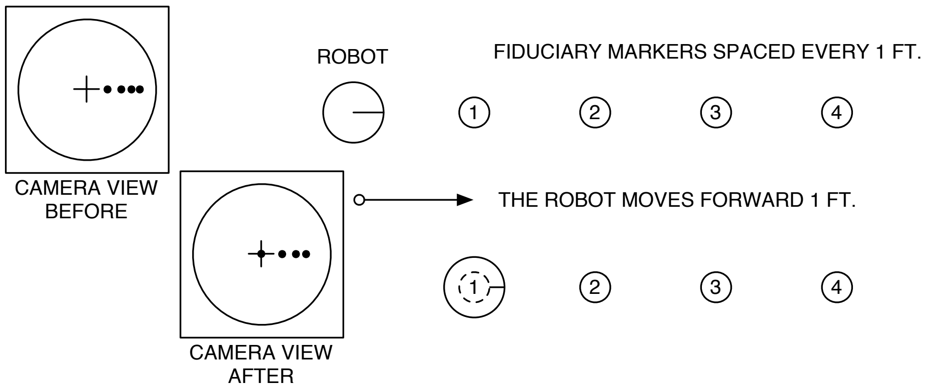

ROBOT
FIDUCIARY MARKERS SPACED EVERY 1 FT.
1
2
3
4
CAMERA VIEW BEFORE
THE ROBOT MOVES FORWARD 1 FT.
1
2
3
4
CAMERA VIEW AFTER

Fig. 19. Creating a hypothetical optical flow field from the calibration test images by pretending the robot traveled one foot forward. Note that only the initial image coordinates are known, but the image coordinates after the hypothetical move should be the same as those of the fiduciary markers, one foot closer.

specified h parameter. Since h was not known, an initial guess was made and then refined by adjusting its value until the recovered points best fit a horizontal line. This produced not only the h parameter, but also the vertical distance from the floor to the single viewpoint. Table I contains the previously mentioned calibration parameters from the real system, later described in section VI.B.

Table I. Calibration parameters

image resolution	400 x 300 pixels
center (x, y)	(180, 124) from top-left
radius	122 pixels
aspect ratio (x/y)	0.986
h	0.82
height	1.48 ft.

From equation (2.4), $h = 0.82$ and $r = 1$, the vertical field of view of the system

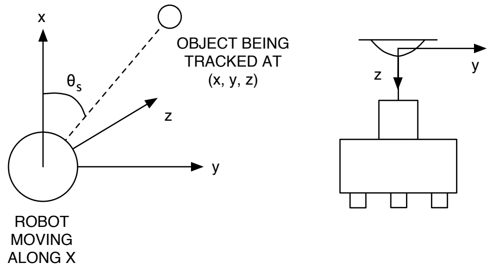


Fig. 20. Three-dimensional view of the separating angle θ_s

can be computed to be 101 degrees, which is close to the manufacturer's specification of 106 degrees.

B. Omni-directional Sensor Model

The default sensor model provided by Moravec and Elfes's certainty grid framework assumes that all sensor readings have the same uncertainty, regardless of their direction relative to the robot's heading. However, for an omni-directional vision sensor, measurements perpendicular to the direction of travel are inherently more accurate than those measurements in the direction of travel. The primary cause for this difference is the inability to measure the very small changes in optical flow near the direction of travel. As the angle separating the line of travel from the direction of the obstacle becomes smaller, so does the resulting optical flow for a fixed change in position. This separating angle θ_s is shown in Fig. 20. There are other contributing factors to the error in range, particularly from the triangulation process; however, the relationship between the separating angle and the range uncertainty is the cornerstone of the omni-directional sensor model presented in this research.

From Fig. 20, the separating angle θ_s is found to be

$$\theta_s = \tan^{-1}(\frac{\sqrt{y^2+z^2}}{x}) \tag{5.1}$$

More importantly, the sensitivity of this angle w.r.t. x is[1]

$$\frac{d\theta_s}{dx} = -\frac{\sqrt{y^2+z^2}}{x^2+y^2+z^2} \tag{5.2}$$

To approximate this sensitivity without actually knowing (x, y, z), assume (x, y, z) lies on a unit sphere, so $x^2 + y^2 + z^2 = 1$, then

$$\left(\frac{d\theta_s}{dx}\right)_{apx} = -\sqrt{y^2+z^2} = -\sin(\theta_s) \tag{5.3}$$

This assumption neglects the role of distance in the sensitivity, but captures the characteristic that objects near the direction of travel do not move as quickly across the field of view as objects to the side.

Finally, equation 5.3 leads to the fundamental form of the sensor model. The range uncertainty is inversely proportional to the magnitude of the optical flow, which is proportional to the sensitivity of the separating angle. To find the true sensitivity of the separating angle requires the exact position of the object, which is not known; however, the magnitude of the sensitivity can be approximated by $\sin(\theta_s)$.

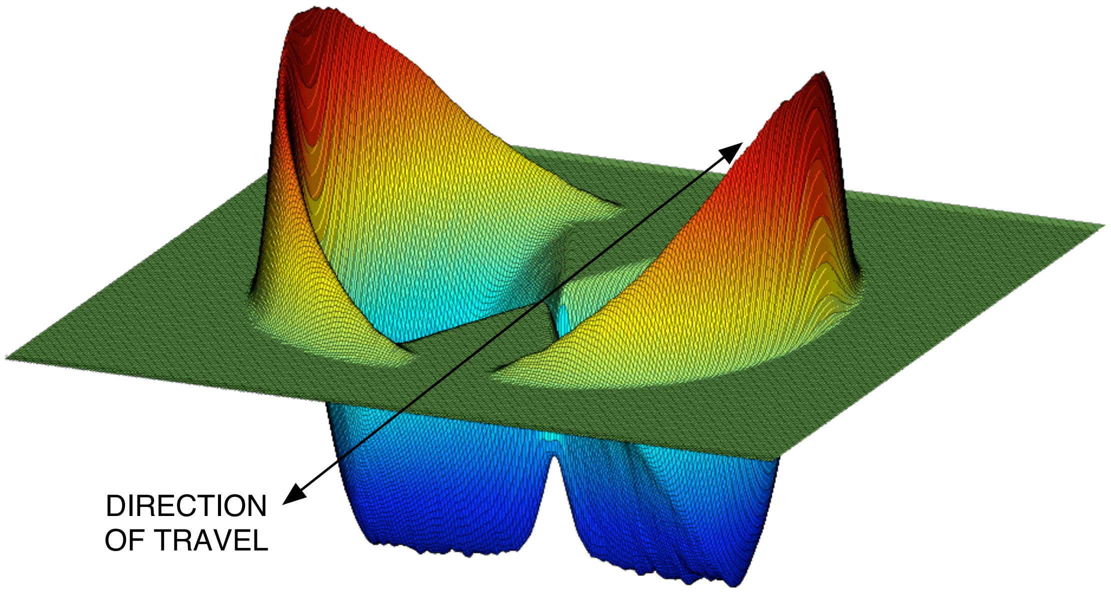
Implementation of this concept in Moravec and Elfes's certainty grid framework was relatively straightforward. Rather than creating one sensor model to use with every range reading, the possible separating angles θ_s were evenly partitioned, and a separate sensor model was created for each range of separating angles. Moravec's implementation has two adjustable parameters for the sensor model: width and height. The width corresponds to the uncertainty of the distance, and so a large width affects a larger area of the map. The height relates to the confidence that an obstacle is present in this area; a taller height leads to a larger certainty. This implementation allows the parameters to change independently; however an increase in the range's

[1] $\frac{d}{dx}\tan^{-1}(u) = \frac{1}{1+u^2}\frac{du}{dx}$

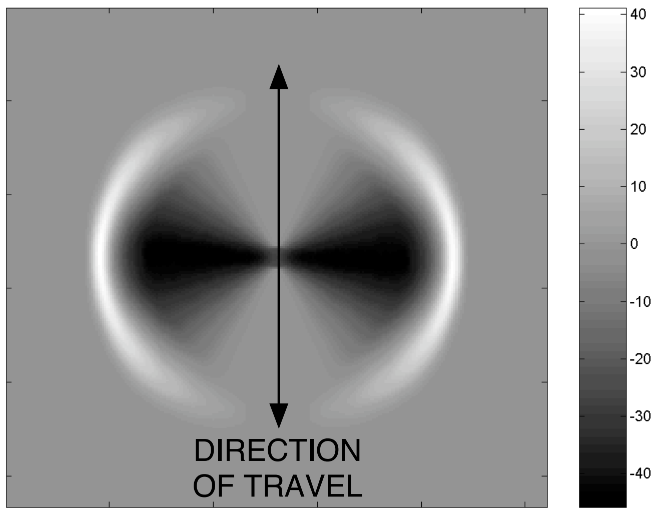
uncertainty (width), should also decrease its confidence (height). At one extreme, this uncertainty should approach a width of infinity and a height of 0.5, which indicates a complete lack of information. The other extreme has a width approaching zero and a height approaching one, which indicates an extremely precise and confident sensor reading. In practice, there is no well documented method for choosing the sensor model paramaters. For this reason, the values were chosen qualitatively for the minimum width / maximum height and maximum width / minimum height, corresponding to the best and worst performance, respectively. Once the extremes were chosen, the interpolation of these values is proportional to $\sin(\theta_s)$, in accordance with the model's characteristic shape. The minimum height (worst case) is typically chosen to be a height of 0.5, which indicates complete uncertainty along the line of travel.

Fig. 21 shows two plots of the resulting certainty grid produced by a ring of measurements surrounding the robot. The height of the profile indicates certainty that those grids are occupied. Notice the certainty of occupation is highest towards the sides and unknown in the direction of travel; the profile also widens, as it lowers. This form matches the desired characteristics very well. Also note the ring of measurements used to create this profile were taken entirely along the x-y plane, so the separating angle spanned 0 to 2π. Objects in front of the object can still be detected when they form a separating angle in the vertical rather than horizontal direction (i.e. they are above or below the path of the omni-directional viewpoint).

Other improvements to the sensor model were also considered, but ultimately rejected. The most promising of these ideas was to adjust the angle of detection for the sensor model, as shown in Fig. 22. The actual implementation used a small fixed detection angle according to the typical 7x7-pixel windows that were tracked. However, not all tracking windows fit perfectly within the constant detection angle.



(a) height corresponds to the probability of cells being occupied



(b) height corresponds to the probability of cells being occupied

Fig. 21. Certainty grid profile of proposed sensor model, for a ring of range measurements around the robot

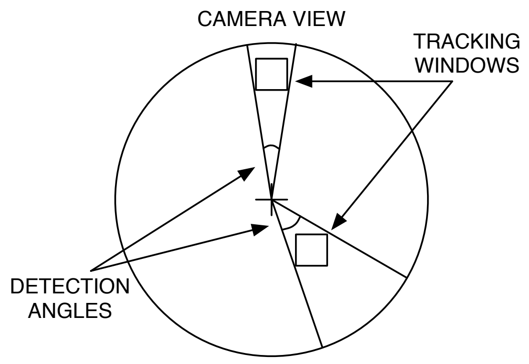


Fig. 22. Varying detection angles shown for two different tracking windows of the same size

As Fig. 22 illustrates, objects further away from the center of the image benefit from a smaller detection angle than objects closer to the center. In particular, objects near the center of the image must also be close to the robot. Widening their detection angle would help them to appear more clearly on the certainty grid, which is important for obstacle avoidance. However, it seemed to be a relatively minor improvement since very few objects would be that close, and instead, a small constant angle was chosen to be consistent with the window size of the matching algorithm and typical tracking window positions.

Another potential improvement would have been to vary the range uncertainty based upon the distance to the object. This factor was lost during simplification by assuming objects were all a unit distance away. In reality, objects further away do not move across the image as quickly and should, therefore, be more susceptible to errors in the optical flow. In fact, a similar capability was already built into the standard model, but the main problem with this idea appeared when trying to select meaningful parameters by which to vary the range. In practice, the most significant improvements occurred when range uncertainty changed as a function of the separating angle, rather than as a function of the distance to the obstacle.

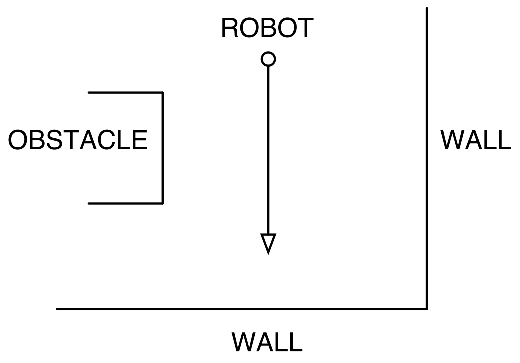
CHAPTER VI

RESULTS

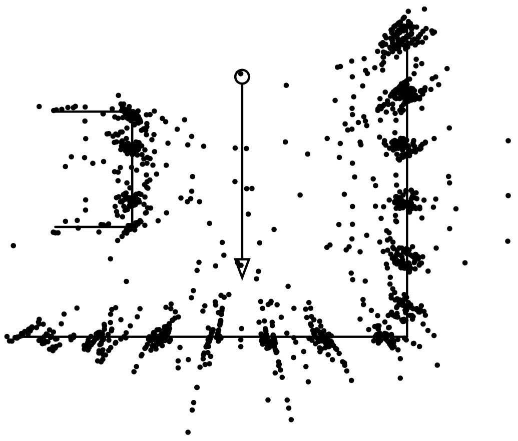
Results of the proposed computational sensor are shown from experiments on a computer graphics simulation, a simplified real-world test and, lastly, two real-world scenes. In each experiment, the omni-directional structure-from-motion technique described previously was used to create a local map and was also compared to the ground truth. Additionally, the local maps were constructed using different sensor models to compare the results of the proposed sensor model and the traditional sonar model described in section IV.B.

A. Simulation Results

An omni-directional vision simulator was created for testing purposes. The simulator is a simple ray-caster that determines the ray directions using the equations of Baker and Nayar [44]. The parameters of the simulated imaging system were designed to match those of the real-world system, as described in section VI.B. The simulated world and the robot path are shown in Fig. 23(a), where the robot traveled in a straight line, taking a total of 24 images, one every 2 inches, and tracked 100 feature points, using the KLT method. The resulting scatter plot of range readings is shown in Fig. 23(b). Most of these measurements are reasonably close to the known position of the objects, which indicates that the structure-from-motion method worked well. However, notice the much larger errors in the direction of travel, a trend that agrees with the findings of Zhang and Blum [40]. A point nearly in the direction of travel will be nearly collinear with the two viewpoints, and so the triangle used to calculate depth collapses into a line. As a result, small errors in optical flow are greatly magnified. This phenomenon is the very reason for developing the specialized sensor model in



(a) Ideal world layout as robot moves along the path



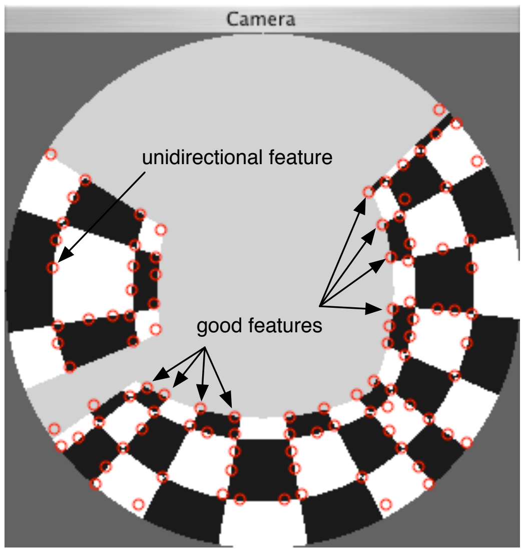
(b) Recovered structure from motion

Fig. 23. Results from the simulated block world

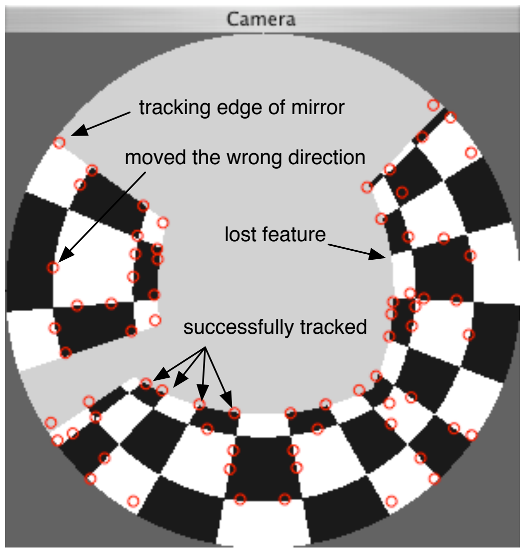
section V.B.

There was only one source of errors in this simulation: miscalculations of optical flow. Fig. 24(a) shows the simulated camera view and the features that were tracked halfway through the simulation. Good features have structure in two non-collinear directions, which makes them easier to track. As expected, corners are the favorite types of feature. However, the KLT algorithm is programmed to select the 100 best features[1], and so it also chooses some less attractive features, such as the unidirectional feature identified in Fig. 24(a). Fig. 24(b) shows that the KLT method was able to track only 70 of the features, which means 30 features were lost and were not available for the structure-from-motion calculations between these two frames. Nonetheless, most of those features with good 2D structure were tracked well. This is clear by looking for their counterpart in Fig. 24(a). Unfortunately, the unidirectional

[1]Note that the optimal number of features is dependent on the environment.



(a) 100 features selected



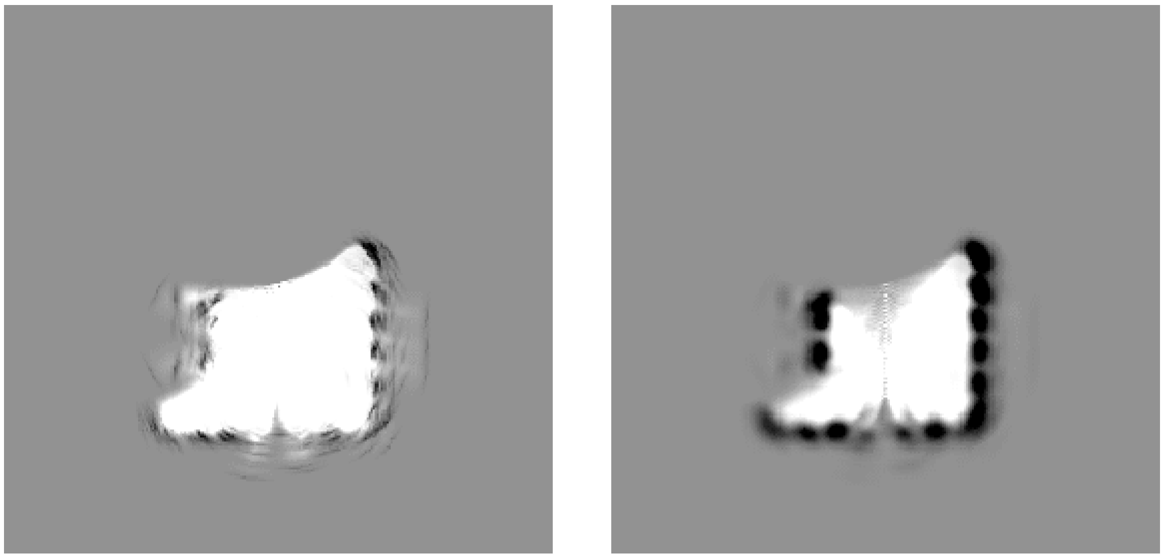
(b) only 70 features are matched on the next image

Fig. 24. Features tracked halfway through the simulation

feature mentioned before was actually marked as moving in the opposite direction. This happened because the shape of the mirror bent the location of the correct image while it straightened the piece just behind it. Since the optical flow algorithm was designed to choose the best match, it chose the newly straightened location. This deformation problem could have been solved before calculating optical flow, by unwarping omni-directional image into a panoramic image. However, the optical flow algorithm would have still failed to match the correct image because the feature was unidirectional. In general, when the feature contained adequate structure, it was either tracked correctly, or the point was lost. Points were often lost when the point traveled far and the deformation was large. When points such as these were lost, they were replaced with new features, selected from the subsequent image using the Kanade-Lucas-Tomasi method. Also notice the features being tracked at the mirror's edge. These features are an artifact of the catadioptric system, and must not be considered in the structure-from-motion process since they do not correspond to real features present in the environment. To ensure that these false features were not included in the structure-from-motion calculation, all features within seven pixels of the mirror's edge were discarded. Finally, after careful examination of the feature tracking, it was determined that the process worked well in most cases.

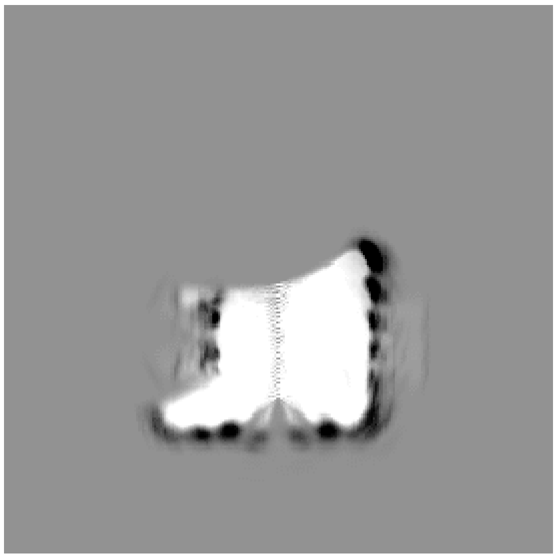
1. Selection of Sensor Model Parameters

The standard sonar model requires the selection of a fixed range uncertainty and intensity. The choice of these parameters is discussed next. Results with a low range uncertainty are shown in Fig. 25(a). Notice that obstacle-free areas are cleared, but the wall directly ahead of the robot is mostly grey rather than black, so it appears to be unknown rather than occupied. Yet from the range readings in Fig. 23(b), an obstacle was clearly detected by the computational sensor.



(a) standard model with small uncertainty

(b) standard model with large uncertainty



(c) results with proposed sensor model

Fig. 25. Certainty grid of the simulated world using different sensor models

The obstacle did appear in the certainty grid when the range uncertainty of the model was increased. Fig. 25(b) shows the same certainty grid using a sensor model with larger range uncertainty and lower confidence. The portion of the wall directly ahead of the robot appears as two blobs, but the walls perpendicular to the direction of travel also become much wider, indicating less certainty in their position. Overall, this situation is preferable to the previous problem, but still not ideal.

Lastly, Fig. 25(c) shows the proposed sensor model, which combines the best of both worlds. Notice that the wall in front of the robot is clearly identified, yet the walls on the sides appear as thin lines. None of these three sensor models performed poorly, however, the results from this new sensor model appear to be a slight improvement over the alternatives because objects ahead of the robot appear dark, while objects to the sides of the robot remain clearly and precisely defined in the certainty grid.

Note also that the path along which the robot traveled was not completely cleared by the sensor model. This is an artifact of the small angled sensors. Grey shades do not indicate an obstacle in the path; rather it means that the area is unexplored. Although the robot had many readings that pierced this area, the line of travel was always nearest the vertex of the sensor. Since the cleared area is most thin near the vertex, it leaves thin lines of unexplored area in-between the thin lines of cleared area. This could be compensated for by marking certain areas as unoccupied whenever the robot passes through. For illustration purposes, however, this trail was not cleared in this implementation.

B. Real Robot in Structured World

The robot platform employed in this research was an AmigoBot^{TM} from ActivMedia. The robot had an onboard microcontroller, and a wireless communication system to

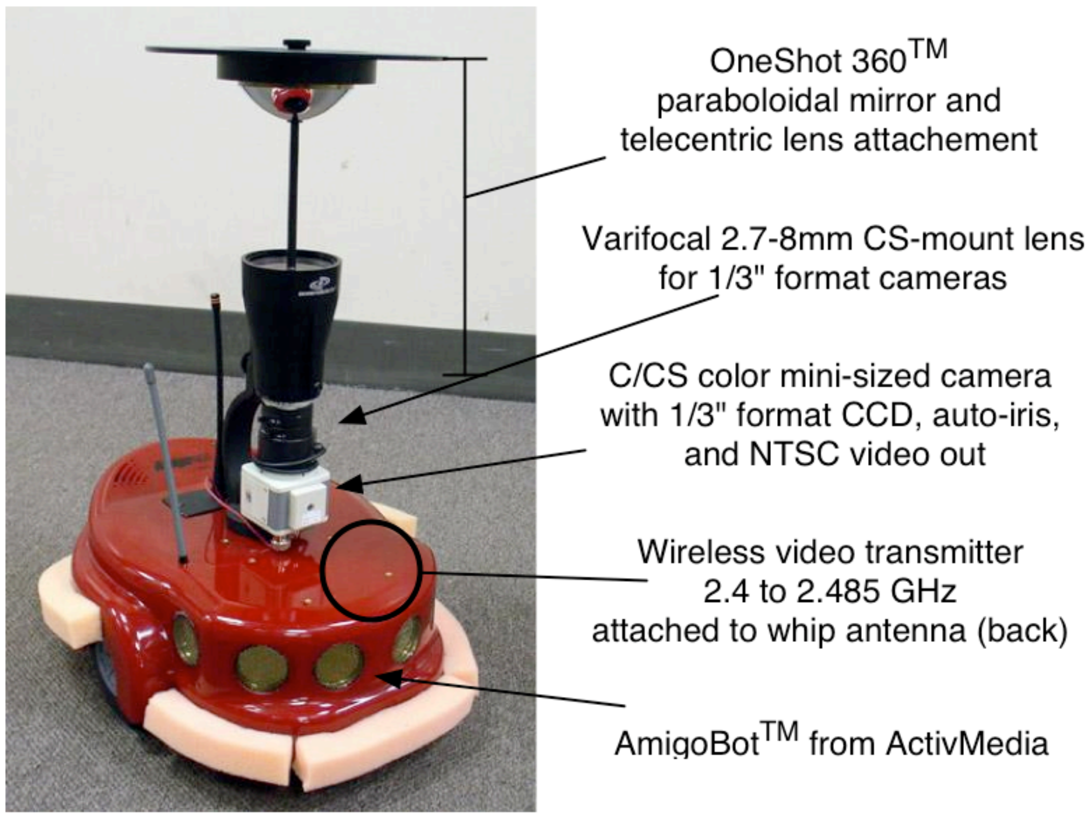

OneShot 360™
paraboloidal mirror and
telecentric lens attachement
Varifocal 2.7-8mm CS-mount lens
for 1/3" format cameras
C/CS color mini-sized camera
with 1/3" format CCD, auto-iris,
and NTSC video out
Wireless video transmitter
2.4 to 2.485 GHz
attached to whip antenna (back)
AmigoBot™ from ActivMedia

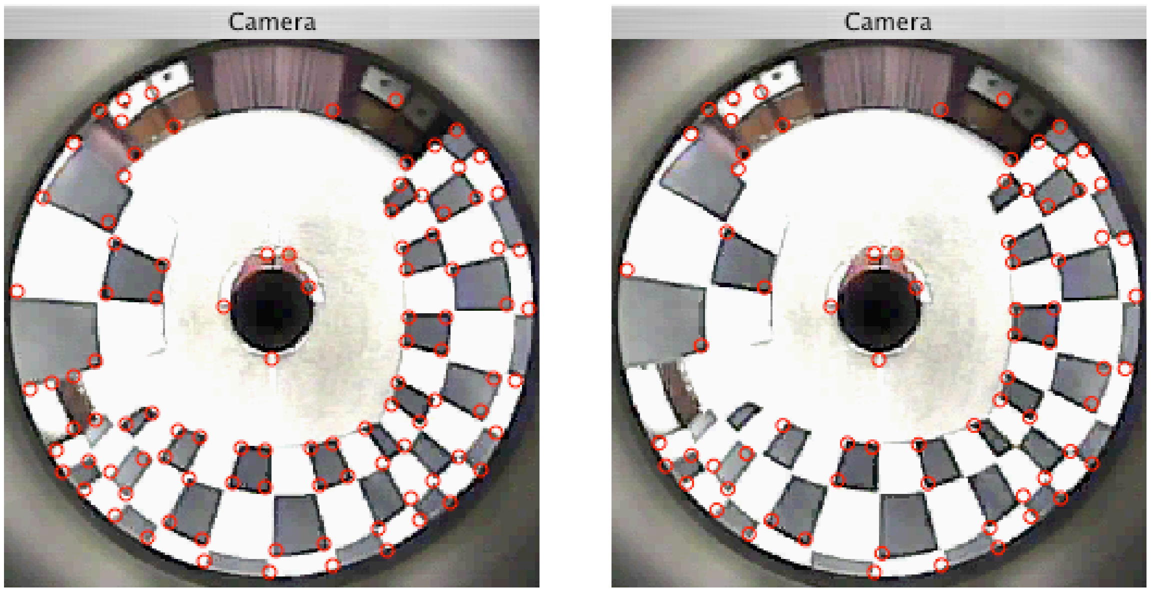
Fig. 26. Major components of the omni-directional AmigoBot^{TM}

transmit video, sonar and other information while receiving simple motion commands. The robot was controlled by an Apple® PowerBook® computer, which processed video and performed the more demanding computations that were necessary.

Since there were no complete omni-directional vision systems available at the time that this research was started, one had to be constructed. Fortunately, a paraboloidal mirror, the OneShot360™ from Remote Reality, was commercially available and highly recommended by Dr. Shree Nayar, professor of Computer Science at Columbia University, co-director of the Columbia Vision and Graphics Center, head of the Columbia Automated Vision Environment (CAVE), and author of several papers cited in this thesis [44, 1, 43, 45, 30]. Following his recommendation, the AmigoBot™ was outfitted with a custom built ODVS, constructed from a CCTV camera, a wireless video transmitter, and the OneShot360™. The complete system is shown in Fig. 26, and described in Appendix A.

Up to this time, all optical flow computations were performed off-line using pre-recorded video from the robot's ODVS. However, the calculations are not too demanding, and the author is confident that these calculations could be performed in real-time.

For testing purposes, a simple box-world with checkered walls was constructed. Its design was very close to that of the simulation, which made it easier to compare simulated and real results. Fig. 27 shows the real camera view, the selected features (Fig. 27(a)), and the tracked features in the following frame (Fig. 27(b)). Again, 70 features were successfully tracked from an initial selection of 100 features. However, a portion of these features was along the mirror boundary and had to be discounted. Several points from the distant background were also tracked. However, these points were too far away to produce significant optical flow and accurate range measurements. There did not appear to be any unidirectional features selected from this



(a) 100 features selected

(b) 70 features matched on the next image

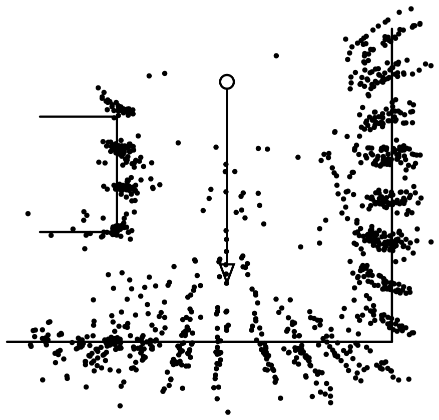
Fig. 27. Features tracked halfway through simplified real world

image, and all the features that were not lost were also correctly tracked. Overall, the image was of slightly lower resolution (244 pixel radius versus 300) and there were more uncertainties due to possible errors in position / orientation of the robot, calibrated parameters not matching true values, and background objects. Therefore, it was expected that there would be more errors in the structure calculation.

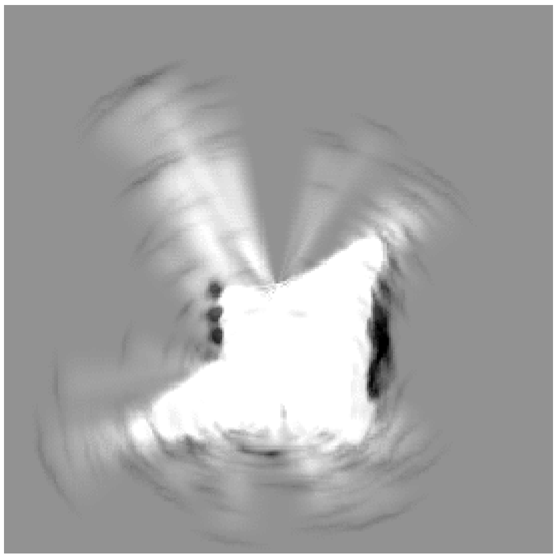
Fig. 28(a) indeed shows the same structure as that in Fig. 23(b), only it has larger errors. Nonetheless, the results clearly show the outline of the walls, with concentrated areas along the checker boundaries. The resulting certainty grids are also shown in Fig. 28. In general, they are similar to their counterparts in Fig. 25. The adaptive model also performed better than the fixed-uncertainty sensor models. The problems from each of the fixed uncertainty models were more pronounced with real-world data for two reasons: (1) the optical flow errors became larger and (2) the real environment was invariably more complex than a simulation. Note that the lower wall of these experiments is much less clearly defined, due to the difficulty measuring optical flow near the line of travel. This issue is one of the major drawbacks to using structure from motion.

C. Real Robot in Unstructured World

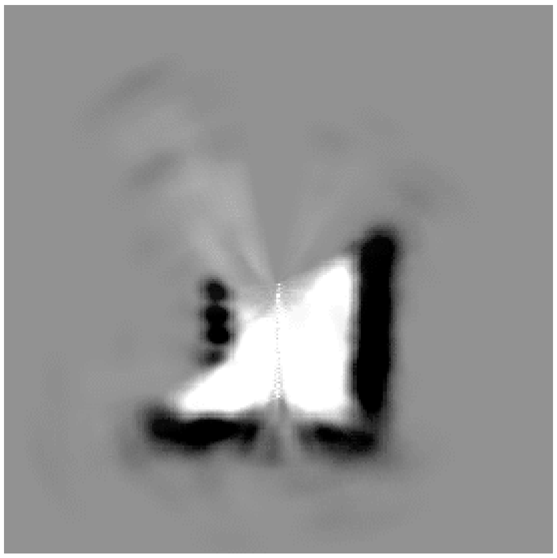
The system was finally tested on two unstructured office environments. Both tests clearly illustrated the weakness of this sensor when there was no optical flow to track, small optical flow in the direction of travel, or visually complex background objects. The tests also proved that the sensor was able to track some features of the environment very well.



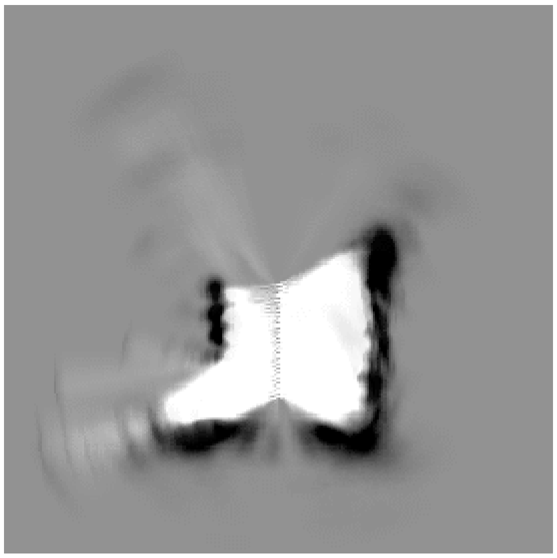
(a) Recovered structure from motion



(b) Certainty grid using small uncertainty

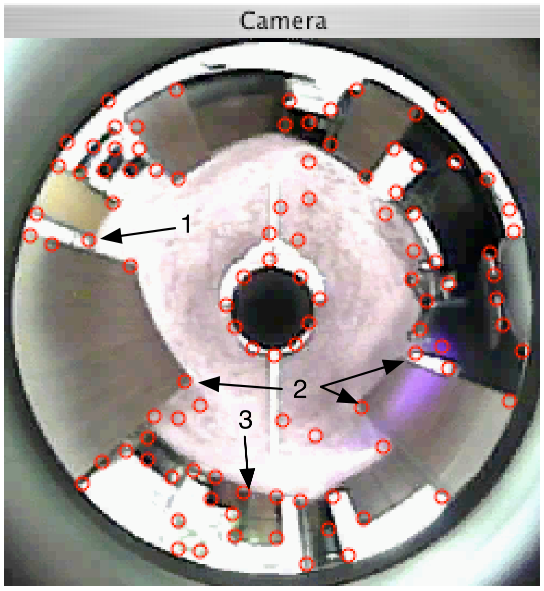


(c) Certainty grid using large uncertainty

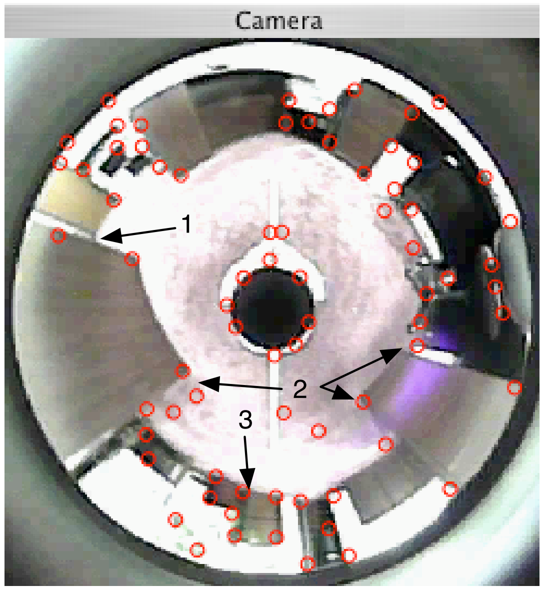


(d) Certainty grid using proposed sensor model

Fig. 28. Results from the simplified real world experiment

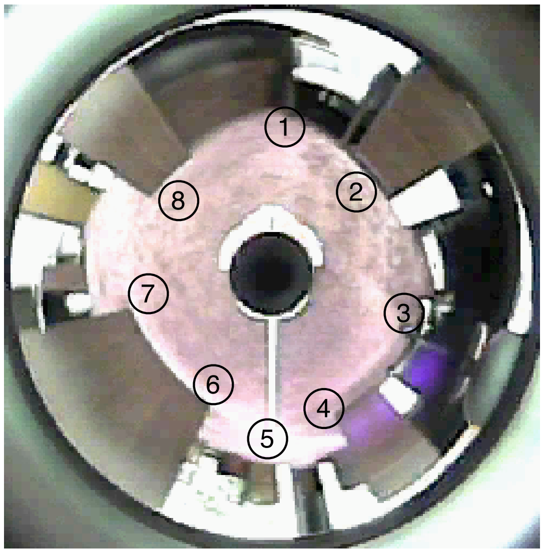


(a) 100 features selected

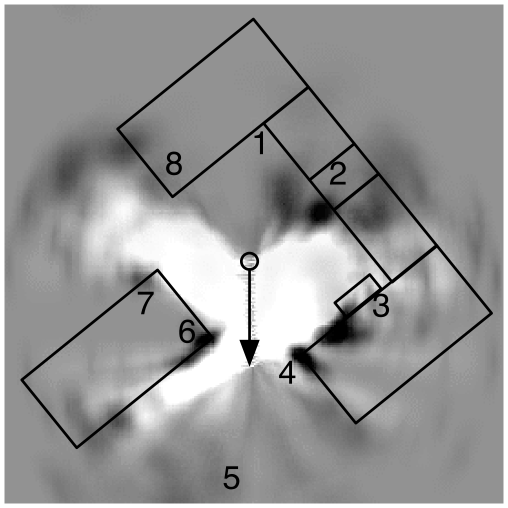


(b) 73 features matched on the next image

Fig. 29. Features tracked halfway through first office test. Feature 1 is actually composed of two separate objects and cannot be tracked. Features labeled 2 are desk corners that were tracked well. Feature 3 is tracked well, but is so far away that its motion is too small to be useful.



(a) Omni-directional view from starting position



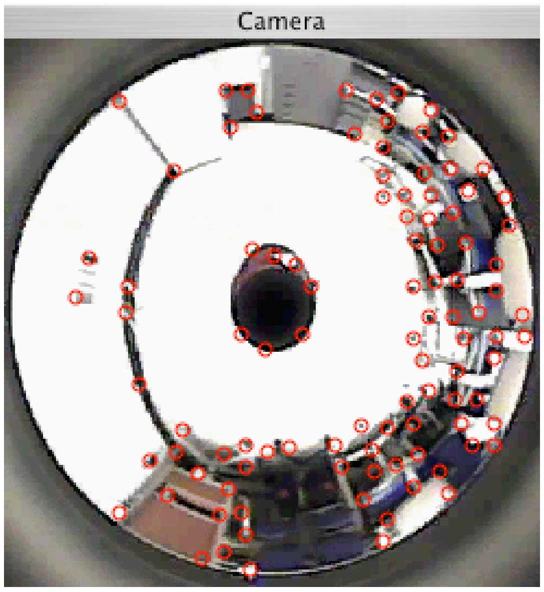
(b) Certainty grid using proposed sensor model

Fig. 30. Results from the first office test, showing the scene and resulting map with corresponding points labeled

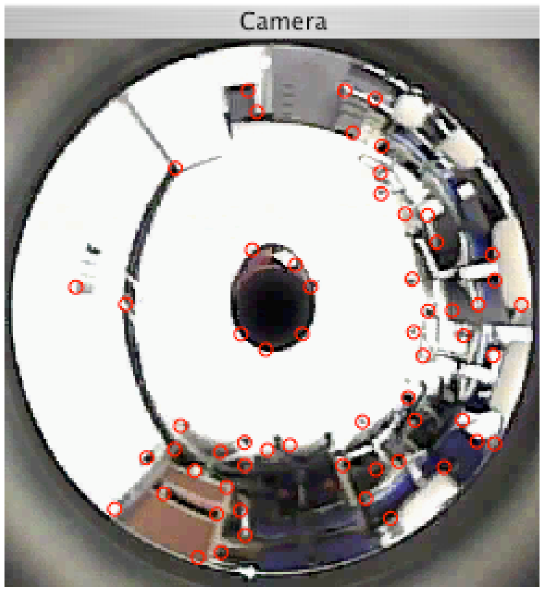
1. First Office Test

In the first office, shown in Fig. 29, the challenges of real-world complexity appeared. In this complex environment, often one object was in front of another. If the background had a solid intensity (i.e. a wall or desk), optical flow could be computed reliably. However, problems arose when the background object was visually complex. In these cases, each object moved across the view field at different speeds, and the boundary of the foreground object, which was normally an excellent location to track, did not produce meaningful optical flow because the background also changed unexpectedly. This was evident in feature 1 of Fig. 29. As expected, the corners of objects against the floor made excellent features for tracking, as generally shown in Fig. 29.

Fig. 30(a) shows the camera view as the robot was just starting its path of 4.0



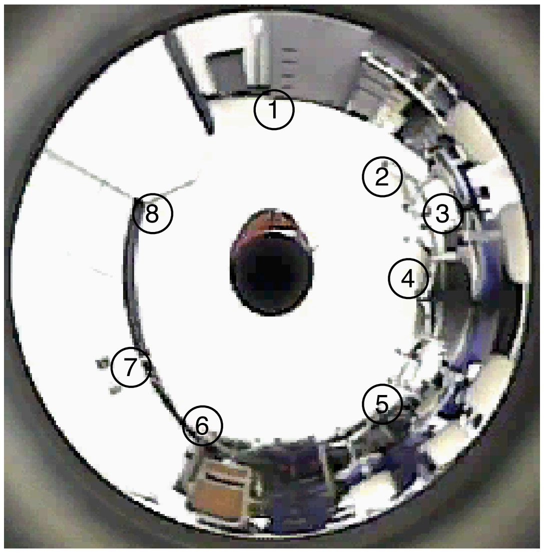
(a) 100 features selected



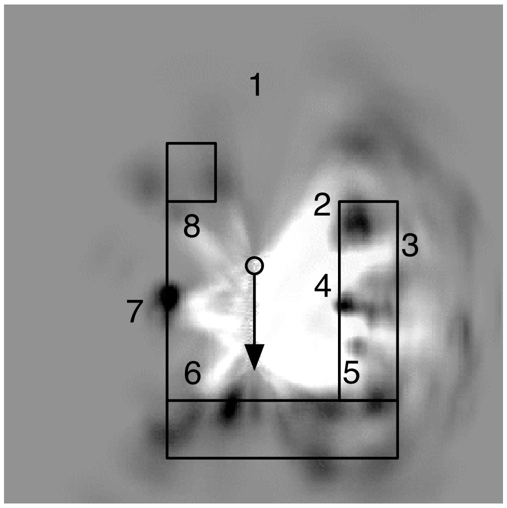
(b) 59 features were matched on the next image

Fig. 31. Features tracked halfway through the second office test

feet, recording an image every 2 inches. In this test, the robot traveled through an opening between desks of the office environment (downward in Fig. 30(b)). The results clearly displayed this method's shortcomings. Areas of solid intensity did not produce optical flow and, therefore, remain grey, signifying an unknown state. An opening to the left of the image appears clearly, and certain objects also appear to be well defined. The obstacles at the bottom of the image were too far away to produce any significant optical flow, but could be expected to become detectable as the robot approached them more closely.



(a) Omni-directional view from starting position



(b) Certainty grid using proposed sensor model

Fig. 32. Results from the second office test, showing the scene and resulting map with corresponding points labeled

2. Second Office Test

In the second office, shown in Fig. 31, there were many visually complex objects (chairs and desks) on the right and lower portions of the image, however, the left side had a plain white wall, and its only distinguishing feature was an Ethernet port. Plain white walls such as this one were challenging, but if a feature was present, the tracking process became easy because of the lack of confounding structures surrounding it. Overall, fewer features were successfully tracked in this office test than any other test. This could have been a byproduct of the scene's visual complexity.

Fig. 32(a) shows the camera view as the robot was just starting a 4.0 feet path, recording an image every 2 inches, the same as before. In this test, the robot traveled parallel to the wall and a line of chairs (to the right of the image) while approaching more chairs straight ahead (downward in Fig. 32(b)). The lower and right portions of the image contained many features to track and both appear clearly in Fig. 32(b) (as does the Ethernet port on the white wall!) The grey areas to the left of the robot's path (to the right from the robot's perspective) are a consequence of the lack of contrast on the white wall, illustrating once more the main shortcoming of the computational sensor.

CHAPTER VII

CONCLUSIONS AND FUTURE WORK

A. Conclusions

This research described and successfully demonstrated a method of combining omni-directional vision, structure from motion, and certainty grids to extract depth from camera images and to create a local map. The proof of concept was shown to work well in both simulation and an engineered real-world experiment. Tests in two unstructured office environments showed the ability of the method to create maps from real-world omni-directional images, but also showed its weakness (1) when faced with obstacles of solid intensity, and (2) when tracking obstacles along the direction of travel.

To account for the large uncertainty of range readings along the line of travel, a new sensor model was proposed and tested. Its design was based on the observation that objects in the line of travel are difficult to accurately track. From this observation, a model was formed to relate the range uncertainty with the direction of travel. Although this model made simplifying assumptions, it accurately captured the sensor's problem areas, and qualitatively worked better than the standard sensor model by properly distinguishing between high and low uncertainty range readings.

Overall, the research objective was achieved by designing, testing, and improving upon a new method of map-making, that combines omni-directional vision, structure from motion, and certainty grids.

B. Future Work

This is part of an ongoing effort, with several possibilities for future work. Currently, all calculations are performed offline, but performing this method in real-time on a high-end desktop computer is a realistic goal; and for this method to be useful in the field of robotics, this is a necessary goal. Along these lines, a future system should be designed with on-board video and processing to eliminate noise from the wireless video signal. A larger, more stable robot platform would be required, which would have the added benefit of a more stable base, thus reducing vibrations in the video and possibly providing better dead-reckoning. This system could also take advantage of the latest CS-mount omni-directional vision system from RemoteReality, the D40, which became available during the writing of this manuscript.

There is also some room for improvement in the probability model of the sensor by adapting the angle of detection to account for a higher angular-resolution near the edge of the mirror. Furthermore, the sensor model could be improved by using the obstacle's range as part of the sensor model. These changes were discussed in section V.B.

In addition to modifying the sensor model, the structure-from-motion method could also be improved. Currently, the method uses only the last two frames for matching purposes in the optical flow calculation. Instead, if it performed the calculation multiple times, using the last frame and several previous frames, then it could detect smaller optical flow measurements that are only visible after several frames. This improvement would help to significantly improve the structure-from-motion calculation near the line of travel. The redundancy might also reduce the potential for error caused by vibrations, changes in lighting, and other real-world factors.

Lastly, the method could be improved by using two omni-directional vision systems in stereo. The stereo vision would provide excellent depth recovery in the direction of travel, while the wide field of view would permit the structure-from-motion method to accurately calculate depth to the sides. Furthermore, the relative position of the two viewpoints could be used to automate the calibration process and accurately calculate ego-motion.

REFERENCES

[1] J. Gluckman and S. K. Nayar, “Ego-motion and omnidirectional cameras,” in *Proc. IEEE Sixth International Conference on Computer Vision*, Bombay, India, Jan. 1998, pp. 999–1005.

[2] H. Kawasaki, K. Ikeuchi, and M. Sakauchi, “Spatio-temporal analysis of omni image,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Hilton Head Island, SC, June 2000, pp. 577–584.

[3] H. Ishiguro, “Development of low-cost compact omnidirectional vision sensors and their applications,” in *Proc. IEEE International Conference on Information Systems, Analysis and Synthesis*, Orlando, FL, 1998, pp. 433–439.

[4] A. Elfes, “Using occupancy grids for mobile robot perception and navigation,” *Computer*, vol. 22, pp. 46–57, June 1989.

[5] R. R. Murphy, *Introduction to AI Robotics.* Cambridge, MA: MIT Press, 2002.

[6] K. Edds, “Defeat wins: Robots racing for $1 million become nobots,” *The Washington Post*, p. A03, 14 March 2004.

[7] R. Brooks, “A robust layered control system for a mobile robot,” *IEEE Journal of Robotics and Automation*, vol. 2, pp. 14–23, Mar. 1986.

[8] S. Goodridge and R. Luo, “Fuzzy behavior fusion for reactive control of an autonomous mobile robot: MARGE,” in *Proc. IEEE International Conference on Robotics and Automation*, San Diego, CA, May 1994, pp. 1622–1627.

[9] B. Kuipers and Y.-T. Byun, “A robot exploration and mapping strategy based on semantic hierarchy of spatial representations,” *Robotics and Autonomous Systems*, vol. 8, pp. 47–63, 1991.

[10] D. Kortenkamp and T. Weymouth, “Topological mapping for mobile robots using a combination of sonar and vision sensing,” in *Proc. AAAI Twelfth National Conference on Artificial Intelligence*, Seattle, WA, July 1994.

[11] H. P. Moravec, “Sensor fusion in certainty grids for mobile robots,” *AI Magazine*, vol. 9, pp. 61–74, Summer 1988.

[12] S. B. Niku, *Introduction to Robotics Analysis, Systems, Applications.* Upper Saddle River, NJ: Prentice Hall, 2001.

[13] J. H. Lim and D. W. Cho, “Physically based sensor modeling for a sonar map in a specular environment,” in *Proc. IEEE International Conference on Robotics and Automation*, Nice, France, May 1992, pp. 1714–1719.

[14] K. W. Jorg and M. Berg, “First results in eliminating crosstalk and noise by applying pseudo-random sequences to mobile robot sonar sensing,” in *Proc. IEEE International Conference on Intelligent Robots and Systems*, Osaka, Japan, Nov. 1996, pp. 292–297.

[15] B. Bury, “Proximity sensing for robots,” in *IEEE Colloquium on Robot Sensors*, London, UK, Jan. 1991, pp. 301–318.

[16] A. Kilpelä̋, J. Ylitalo, K. Mä̋ättä̋, and J. Kostamovaara, “Timing discriminator for pulsed time-of-flight laser rangefinding measurements,” *Review of Scientific Instruments*, vol. 69, pp. 1978–1984, May 1998.

[17] J. D. Tardos, J. Neira, P. M. Newman, and J. J. Leonard, “Robust mapping and localization in indoor environments using sonar data,” *International Journal of Robotics Research*, submitted for publication.

[18] Z. Xu, J. Liu, Z. Xiang, and H. Li, “Map building for indoor environment with laser range scanner,” in *Proc. IEEE 5th International Conference on Intelligent Transportation Systems*, Singapore, 2002, pp. 136–140.

[19] N. Pears and P. Probert, “Active triangulation rangefinder design for mobile robots,” in *Proc. IEEE International Conference on Intelligent Robots and Systems*, Raleigh, NC, July 1992, pp. 2047–2052.

[20] O. Faugeras, *Three-Dimensional Computer Vision.* Cambridge, MA: MIT Press, 1993.

[21] J. Oliensis, “Exact two-image structure from motion,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 1618–1633, Dec. 2002.

[22] M. Tistarelli and G. Sandini, “Direct estimation of time-to-impact from optical flow,” in *Proc. IEEE Workshop on Visual Motion*, Princeton, NJ, Oct. 1991, pp. 226–233.

[23] J. Loomis and D. Eby, “Relative motion parallax and the perception of structure from motion,” in *Proc. IEEE Workshop on Visual Motion*, Irvine, CA, Mar. 1989, pp. 204–211.

[24] Z. Zhang, “Motion and structure of four points from one motion of a stereo rig with unknown extrinsic parameters,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 12, pp. 1222–1227, Dec. 1995.

[25] K. Daniilidis and C. Geyer, “Omnidirectional vision: Theory and algorithms,” in *Proc. IEEE International Conference on Pattern Recognition*, Barcelona Spain, Sept. 2000, pp. 89–96.

[26] A. Clérentin, C. Pégard, and C. Drocourt, “Environment exploration using an active vision sensor,” in *Proc. IEEE International Conference on Intelligent Robots and Systems*, Kyongju, South Korea, Oct. 1999, pp. 1525–1530.

[27] D. Sekimori, T. Usui, Y. Masutani, and F. Miyazaki, “High-speed obstacle avoidance and self-localization for mobile robots based on omni-directional imaging of floor region,” *Transactions of the Institute of Systems, Control and Information Engineers*, vol. 16, pp. 204–213, Feb. 2003.

[28] Y. Yagi, S. Kawato, and S. Tsuji, “Collision avoidance using omnidirectional image sensor,” in *Proc. IEEE International Conference on Robotics and Automation*, Sacramento, CA, Apr. 1991, pp. 910–915.

[29] N. Winters, J. Gaspar, G. Lacey, and J. Santos-Victor, “Omni-directional vision for robot navigation,” in *Proc. IEEE Workshop on Omnidirectional Vision*, Hilton Head Island, SC, June 2000, pp. 21–28.

[30] H. Murase and S. K. Nayar, “Visual learning and recognition of 3D objects from appearance,” *International Journal of Computer Vision*, vol. 14, no. 1, pp. 5–24, Jan. 1995.

[31] I. Stratmann, “Omnidirectional imaging and optical flow,” in *Proc. IEEE Third Workshop on Omnidirectional Vision*, Copenhagen, Denmark, June 2002, pp. 104–111.

[32] P. Anandan, “A computational framework and an algorithm for the measurement

of visual motion," *International Journal of Computer Vision*, vol. 2, pp. 283–310, 1989.

[33] B. Horn and B. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 16, pp. 185–203, Aug. 1981.

[34] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. IJCAI 7th International Joint Conference on Artificial Intelligence*, Vancouver, British Columbia, Aug. 1981, pp. 674–679.

[35] D. Fleet and A. Jepson, "Computation of component image velocity from local phase information," *International Journal of Computer Vision*, vol. 5, pp. 77–104, 1990.

[36] R. F. Vassallo, J. Santos-Victor, and H. J. Schneebeli, "A general approach for egomotion estimation with omnidirectional images," in *Proc. IEEE Third Workshop on Omnidirectional Vision*, Lisbon, Portugal, June 2002, pp. 97–103.

[37] A. R. Bruss and B. K. P. Horn, "Passive navigation," *Computer Vision, Graphics, and Image Processing*, vol. 21, pp. 3–20, Jan. 1983.

[38] X. Zhuang, T. S. Huang, N. Ahuja, and R. M. Haralick, "A simplified linear optic flow-motion algorithm," *Computer Vision, Graphics, and Image Processing*, vol. 42, pp. 334–344, June 1988.

[39] A. Jepson and D. Heeger, "A fast subspace algorithm for recovering rigid motion," in *Proc. IEEE Workshop on Visual Motion*, Princeton, NJ, Oct. 1991, pp. 124–131.

[40] Z. Zhang and R. S. Blum, "Extraction of 3-d coordinates from fusion of omnicamera images," in *Proc. IEEE Thirty-Third Asilomar Conference on Signals,*

Systems, and Computers, Pacific Grove, CA, Oct. 1999, pp. 397–401.

[41] P. Chang and M. Hebert, “Omni-directional structure from motion,” in *Proc. IEEE Workshop on Omnidirectional Vision*, Hilton Head Island, SC, June 2000, pp. 127–133.

[42] J. Neumann, C. Ferműller, and Y. Aloimonos, “Eyes from eyes: New cameras for structure from motion,” in *Proc. IEEE Third Workshop on Omnidirectional Vision*, Copenhagen, Denmark, June 2002, pp. 19–26.

[43] S. Nayar, “Catadioptric omnidirectional camera,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, San Juan, Puerto Rico, June 1997, pp. 482–488.

[44] S. Baker and S. K. Nayar, “A theory of single-viewpoint catadioptric image formation,” *International Journal of Computer Vision*, vol. 35, pp. 175–196, November-December 1999.

[45] V. N. Peri and S. K. Nayar, “Generation of perspective and panoramic video from omnidirectional video,” in *Proc. DARPA Image Understanding Workshop*, New Orleans, LA, May 1997, pp. 243–246.

[46] J. Shi and C. Tomasi, “Good features to track,” in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Seattle, WA, June 1994, pp. 593–600.

[47] J. Barron, D. Fleet, S. Beauchemin, and T. Burkitt, “Performance of optical flow techniques,” in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Champaign, IL, June 1992, pp. 236–242.

[48] S. Uras, F. Girosi, A. Verri, and V. Torre, "A computational approach to motion perception," *Biological Cybernetics*, vol. 60, pp. 79–87, 1988.

[49] D. J. Heeger, "Optical flow using spatio-temporal filters," *International Journal of Computer Vision*, vol. 1, pp. 279–302, Jan. 1998.

[50] S. Birchfield, "KLT: An implementation of the kanade-lucas-tomasi feature tracker," accessed on January 31, 2004. [Online]. Available: http://vision.stanford.edu/ birch/klt/

[51] H. Moravec and D. W. Cho, "A Bayesian method for certainty grids," in *AAAI Spring Symposium Series, Symposium on Mobile Robots*, Cambridge, MA, Nov. 1988.

[52] G. Shafer, *A Mathematical Theory of Evidence.* Princeton, NJ: Princeton University Press, 1976.

[53] J. Borenstein and Y. Koren, "Histogramic in-motion mapping for mobile robot obstacle avoidance," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 4, pp. 535–539, 1991.

[54] A. Elfes, "Occupancy grids: A probabilistic framework for robot perception and navigation," Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, PA, 1989.

[55] A. P. Dempster, "A generalization of Bayesian inference," *Journal of the Royal Statistical Society*, vol. 30, pp. 205–247, 1968.

[56] R. Murphy, "Dempster-shafer theory for sensor fusion in autonomous mobile robots," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 2, pp. 197–206, Apr. 1998.

[57] C. Congdon, M. Huber, D. Kortenkamp, K. Konolige, K. Myers, A. Saffiotti, and E. Ruspini, “CARMEL vs. flakey: A comparison of two winners,” *AI Magazine*, vol. 14, no. 1, pp. 49–57, Spring 1993.

APPENDIX A

EXPERIMENTAL SETUP

The following parts were used in construction of the omni-directional vision system:

- OneShot360^{TM} from RemoteReality
- Computar varifocal lens 2.7-8mm, DC auto iris, F1.0, for 1/3" format cameras, CS-Mount, part TG3Z2710FCS
- COP 15-CC25NV day/night C/CS color mini sized camera
- AmigoBot^{TM} E-Presence from ActivMedia
- CLOVER'S Wireless transmitter PCB, part CW3800 (included with AmigoBot^{TM})
- Step-up ring 35.5mm–37mm and 37mm–46mm
- Whip antenna to replace directional antenna that comes with AmigoBot

The original AmigoBot^{TM} E-Presence from ActivMedia came with a wireless video system, which included the onboard camera and transmitter, a four-channel receiver, and a PC video capture card. However, the camera it was shipped with, the Marshall V-X0097-SE-P, had no threads nor replaceable lens, so it was not suitable for attachments such as the OneShot360^{TM}. The Computar lens reportedly had threads and it met the focal length requirements specified by RemoteReality (5.8mm for 1/3" CCD or 3.8mm for 1/4" CCD). Its auto iris lens also allowed it to automatically adjust to various lighting conditions. To accommodate this replacement lens, the

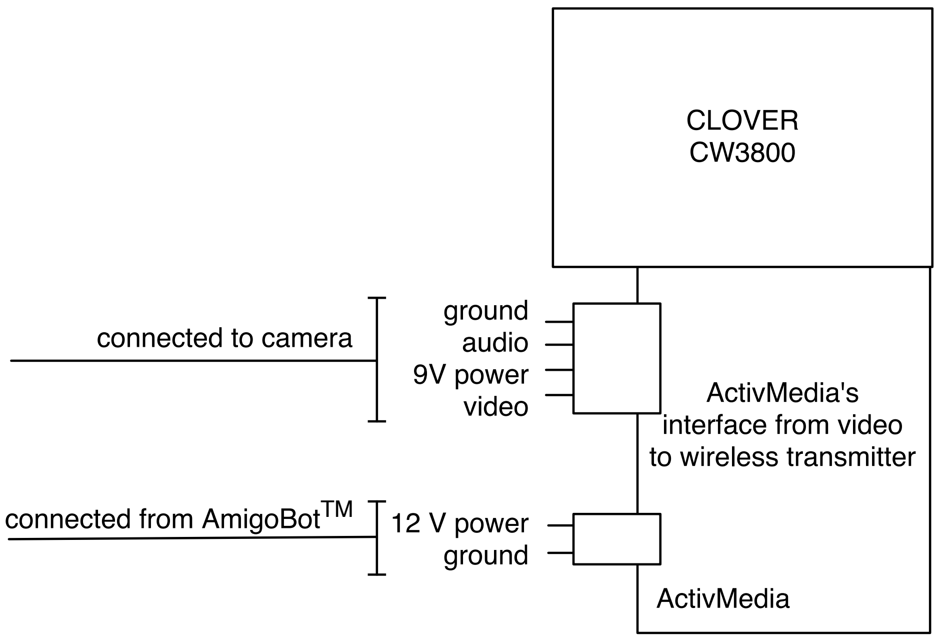


Fig. 33. ActivMedia's camera board interface, reverse engineered to replace original camera

COP 15-CC25NV camera was chosen because it was a good match for the lens being C/CS mount and having DC auto iris capability. It also produced similar output to the Marshal camera (NTSC 310 TV lines).

To connect this replacement camera, the existing camera connections were reverse engineered, as shown in Fig. 33. Since the COP 15-CC25NV camera used a 12-Volt power supply, only the 12 V, ground and video pins were used once the Marshal camera was replaced. The onboard components of the original video system were also moved inside the AmigoBot^{TM} and the directional antenna, which originally pointed upwards, was replaced with an omni-directional whip. This freed space on top of the AmigoBot to mount the camera and mirror. Replacing the antenna also made the video signal less sensitive to the robot's position.

The final system worked, but could be improved. The threads of the lens were not designed for a large attachment such as the OneShot360^{TM}; they were designed to support small lightweight filters. Hence, the lens did not mate well with the step-up ring and appeared susceptible to wear. Also, the centerline of the camera's lens did not match the center of the CCD, so the full zoom capability was not utilized, as visible in the calibration images (Fig. 18). Had the camera zoomed in closer to increase the resolution of the omni-directional image, the top part of the mirror's image would have been lost. Lastly, the wireless video signal conflicted with the wireless serial modem controlling the AmigoBot. Turning on the robot's controller produced static in the video. This was overcome by separating the channels of both the controller and the video. However, it is the author's recommendation that future systems use onboard video and processing to completely eliminate the potential for noise caused by wireless interference.

VITA

Steven Rey Ortiz was born in Midland, Texas on May 2, 1978, the son of Isaias Ortiz Jr. and Sonia Esther Ortiz. He was raised in Houston, Texas and Pittsburgh, Pennsylvania. After graduation from North Allegheny High School in 1996, he attended Texas A&M University in College Station, Texas. He married Kirsten Joy Moreth in December of 1997, and he graduated in May of 2000 with a Bachelor of Science degree in mechanical engineering. Following graduation, he joined the dot-com boom, and began full-time work as a programmer for Quetzal Consulting in South San Francico, California. In January of 2002, he decided to resume his studies at Texas A&M University, and by May of 2004 completed the requirements for his Masters of Science degree in computer science, which will be awarded in August. He and his wife moved again to California. This time, he works for the Naval Air Warfare Center Weapons Division, where he hopes to apply both his mechanical engineering and computer science education. He can be contacted by e-mail at steve@whoop.us or by post at 921 N. Sierra View, Ridgecrest, CA 93555.