Navigation, Path Planning and Homing for Autonomous Mobile Robots using Panoramic Visual Sensors

J.S. Chahl and M.V. Srinivasan
Visual Sciences
Australian National University
ACT, Australia, 2605
Javaan.Chahl@anu.edu.au M.Srinivasan@anu.edu.au

7/4/97

Abstract

Algorithms for computing the fundamental parameters of navigation have been devised; range, egomotion, and localisation. All algorithms use a panoramic strip image as their sole sensory input. All motion/deformation computations are performed using the *Image Interpolation* technique [1]. This technique has shown itself to be effective in diverse machine vision problems. It is particularly adept at computing the relative deformation between images that are deformed by non-affine processes.

The first algorithm presented computes the parameters of egomotion of a mobile robot equipped with two panoramic visual sensors. We show that egomotion can be computed using the image interpolation algorithm by interpolating the position of the image captured by one of the sensors at the robot's present location, with respect to the images captured by the two sensors at the robot's previous location. The algorithm delivers the distance traveled and angle rotated. Trajectories of over 1.5m in length have been reconstructed to within 5% accuracy using this technique.

The second algorithm computes range in the entire panoramic environment. Range estimation is based on the fact that local deformation of the panoramic image is range-dependent, regardless of the nature of the deformation. The nature of the deformation depends on azimuthal viewing direction: expansion in the direction of motion, contraction in the opposite direction, and translation for viewing directions perpendicular to that of the motion. Range in each direction is estimated by comparing the magnitude of the measured image deformation, with the magnitude of the deformation that would be produced by a surface at a known, standard, minimum range. Range can be accurately computed to about 100 times the distance moved by the sensor.

A sensor's location in both orientation and position is computed by using the image interpolation algorithm to interpolate the image captured by the panoramic sensor at that location with respect to panoramic images captured at known, reference locations. The algorithm can reliably determine the location and orientation of the sensor within a small region.

All of the algorithms described are rapid and non-iterative in determining a solution, not dependent on floating point operations, and generally computationally inexpensive.

Introduction

Applications of panoramic imaging to navigation of an autonomous mobile robot are presented. Three significant challenges in navigation are localisation, egomotion determination and obstacle avoidance. The three algorithms described here endeavor to achieve these goals with minimal computation and maximum robustness.

All algorithms use a panoramic strip image of 250 pixels taken about the horizon as their sole sensory input. In every case the image was obtained from a CCD camera facing upwards beneath an inverted cone with a mirrored finish (figure 1). The images were decoded by choosing that annulus within the

Proc. AISB workshop on "Spatial Reasoning in Mobile Robots and Animals", Manchester 1997.
Technical Report Series, Department of Computer Science, Manchester University, ISSN 1361 - 6161.
Report number UMCS-97-4-1. http://www.cs.man.ac.uk/csonly/cstechrep/titles97.html

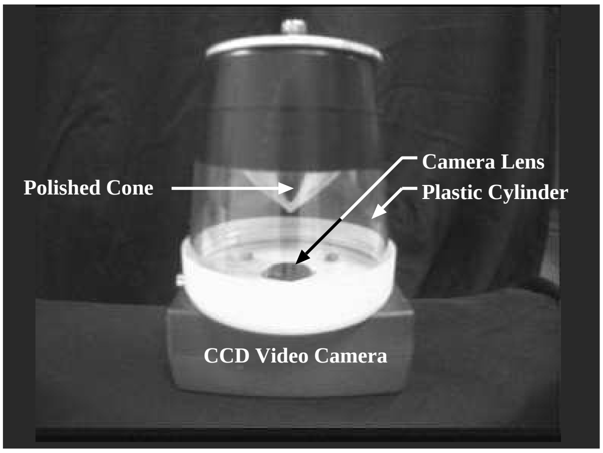

Figure 1: The panoramic images were captured using a standard CCD video camera, positioned beneath a polished aluminium cone. The cone was supported by a transparent plastic cylinder to avoid any structure being visible to the camera.

resulting image that corresponded to the horizon, and dividing it into sectors subtending equal azimuthal angles about the centre of the image.

Panoramic imaging was chosen for several reasons.

- Panoramic images usually contain some visual texture, whereas narrow field images frequently show no visual texture. A panoramic field of view therefore increases the likelihood of being able to apply vision to navigation problems.
- Few animals have narrow visual fields, suggesting an advantage to wide angle imaging when interacting with the environment[2][3].
- Panoramic images of the type used in this work do not have edges or boundaries, allowing a very uniform treatment of the image arrays, thereby increasing the inherent parallelism in the algorithms.
- Finally, panoramic images capture the entire environment in every image, thus offering the possibility of homing to landmarks regardless of bearing, and removing the requirement to articulate the camera[4].

The Image Interpolation Algorithm

A common thread running through this work is the *Image Interpolation Algorithm* [1][5][6]. This algorithm has shown itself to be versatile in diverse machine vision problems, from computing motion vectors, to computing the magnitude of complex non-affine transforms. The basic concept of the interpolation algorithm is to take two reference images, one before a transformation, and one after a maximal transformation of a given type. If the image is deformed in the same manner to an extent between that of the first and second reference images, the interpolation algorithm can determine the degree of the transformation. This feature allows the interpolation algorithm to be used in situations where the type of motion is unknown, and can only be determined by the mobile agent as it moves, or from internal calibration on-board the robot – such as two displaced cameras. This factor makes the interpolation algorithm very suitable for determining egomotion and range, where the deformation of the image is non-affine.

Consider a visual sensor that acquires an image $f(\theta)$ at some intermediate point between two similar sensors (reference sensors) that capture images $f_0(\theta)$ and $f_1(\theta)$. We would like to determine the position of that sensor relative to the two reference sensors (image motion, as opposed to observer motion is a similar problem, though easier to solve). If we assume that the image captured by the sensor will deform continuously between the positions of the two reference images, then first order linear interpolation will return the position of the sensor on the line defined by the two reference images. If this is the case then

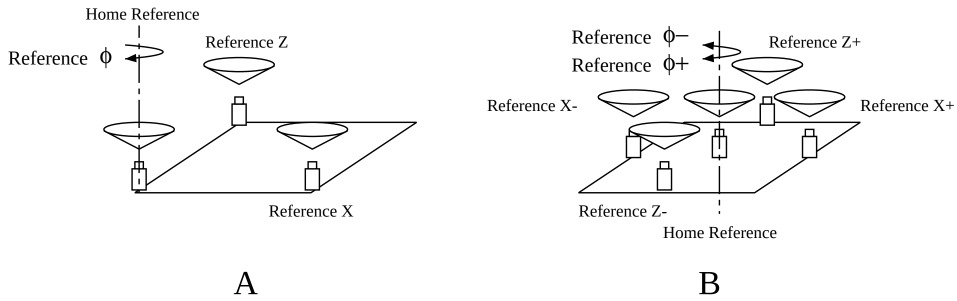

Figure 2: Selection of reference images for the image interpolation algorithm as it is applied to sensor localisation. (A) For the asymmetrical version of the algorithm the reference images are chosen from the home position, and extreme positions along the different degrees of freedom, x, z and ϕ in this case. The origin of the coordinate system is not in the geometrical centre of the region spanned by the images. (B) The symmetrical version of the algorithm takes reference images from maximum positive and negative displacements along the various axes. As a result the home reference image is in the centre of the coordinate system.

$f(\theta)$ is approximated by $\widehat{f(\theta)}$, a weighted linear combination of the two reference images

$$\widehat{f(\theta)} = f_0(\theta) + \alpha\,(f_1(\theta) - f_0(\theta))$$

where α is the position of the sensor normalised to the distance between $f_0(\theta)$ and $f_1(\theta)$. The value of α for which $\widehat{f(\theta)}$ best approximates $f(\theta)$ is determined by minimizing the integral squared error between $\widehat{f(\theta)}$ and $f(\theta)$ as follows

$$\int_0^{2\pi} \left[\widehat{f(\theta)} - f(\theta)\right]^2 d\theta = \int_0^{2\pi} \left[f_0(\theta) + \alpha\,(f_1(\theta) - f_0(\theta)) - f(\theta)\right]^2 d\theta$$

Minimising this expression with respect to α yields the following expression for α:

$$\alpha = \frac{\int_0^{2\pi} \left[(f(\theta) - f_0(\theta))\,(f_1(\theta) - f_0(\theta)\right] d\theta}{\int_0^{2\pi} \left[f_0(\theta) - f_1(\theta)\right]^2 d\theta} \tag{1}$$

Consider the problem of general motion of a panoramic sensor on a plane. In this case there are two orthogonal axes of translation on the plane as well as rotation about the vertical axis. The motion can be described by three variables, each giving the location of the sensor along a particular degree of freedom, normalised to the separation of the respective reference images. For estimating position relative to the three possible degrees of freedom in a plane (translations x and z and rotation ϕ), we need to capture (in addition to the original image $f_0(\theta)$ and the currently captured image $f(\theta)$) reference images $f_x(\theta)$, $f_z(\theta)$ corresponding to reference translations along the x and z axes respectively, and the reference image $f_\phi(\theta)$ corresponding to a reference rotation in θ. The best linear estimate $\widehat{f(\theta)}$ of $f(\theta)$ can be expressed as

$$\widehat{f(\theta)} = f_0(\theta) + \alpha\,(f_x(\theta) - f_0(\theta)) + \beta\,(f_z(\theta) - f_0(\theta)) + \delta\,(f_\phi(\theta) - f_0(\theta))$$

where α, β and δ represent the normalised displacements along x, z, and θ respectively, which are to be determined.

The validity of the linear approximation can be checked by examining the boundary conditions of the displacements, α, β and δ. If $\alpha = \beta = \delta = 0$ then $\widehat{f(\theta)} = f_0(\theta)$, as expected. If $\alpha = \beta = 0$ and $\delta = 1$, then $\widehat{f(\theta)} = f_\phi(\theta)$, as expected. Similar checks can be carried out for the two other degrees of freedom. The above expression is the asymmetrical form of the equation, in which all values for the various degrees of freedom are positive and lie between 0 and 1.0. In this case the region defined by the reference images is triangular, with the home reference image acquired at a vertex (figure 2A). This is not ideal for use in navigation as there is only room for localisation in one quadrant with respect to the home location. However, the above equation can be modified to cater for a symmetrical localisation problem, which allows each degree of freedom to vary from -1.0 to 1.0, with the home reference location at the origin of the coordinate system, as shown in figure 2B. In this case, we can approximate $f(\theta)$ by

$$\widehat{f(\theta)} = f_0(\theta) + \frac{\alpha}{2}\left(f_{x+}(\theta) - f_{x-}(\theta)\right) + \frac{\beta}{2}\left(f_{z+}(\theta) - f_{z-}(\theta)\right) + \frac{\delta}{2}\left(f_{\phi+}(\theta) - f_{\phi-}(\theta)\right) \tag{2}$$

α, β and δ can be estimated by minimising the integral squared error between $\widehat{f(\theta)}$ and $f(\theta)$:

$$\int_0^{2\pi}\left[\widehat{f(\theta)} - f(\theta)\right]^2 d\theta = \int_0^{2\pi}\Big[f_0(\theta) + \frac{\alpha}{2}(f_{x+}(\theta) - f_{x-}(\theta)) + \frac{\beta}{2}(f_{z+}(\theta) - f_{z-}(\theta)) + \frac{\delta}{2}(f_{\phi+}(\theta) - f_{\phi-}(\theta)) - f(\theta)\Big]^2 d\theta \tag{3}$$

The values of α, β, and δ for which the error is minimal is determined by taking partial derivatives of this expression with respect to each variable, and determining the value of that variable for which the derivative is zero. The resulting system of equations can be expressed in the following matrix notation

$$\begin{bmatrix} \int_0^{2\pi}(f_{x+}-f_{x-})^2 d\theta & \int_0^{2\pi}(f_{x+}-f_{x-})(f_{z+}-f_{z-})d\theta & \int_0^{2\pi}(f_{x+}-f_{x-})(f_{\phi+}-f_{\phi-})d\theta \\ \int_0^{2\pi}(f_{x+}-f_{x-})(f_{z+}-f_{z-})d\theta & \int_0^{2\pi}(f_{z+}-f_{z-})^2 d\theta & \int_0^{2\pi}(f_{z+}-f_{z-})(f_{\phi+}-f_{\phi-})d\theta \\ \int_0^{2\pi}(f_{x+}-f_{x-})(f_{\phi+}-f_{\phi-})d\theta & \int_0^{2\pi}(f_{z+}-f_{z-})(f_{\phi+}-f_{\phi-})d\theta & \int_0^{2\pi}(f_{\phi+}-f_{\phi-})^2 d\theta \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ \delta \end{bmatrix} = 2\begin{bmatrix} \int_0^{2\pi}(f_0-f)(f_{x+}-f_{x-})d\theta \\ \int_0^{2\pi}(f_0-f)(f_{z+}-f_{z-})d\theta \\ \int_0^{2\pi}(f_0-f)(f_{\phi+}-f_{\phi-})d\theta \end{bmatrix}$$

α, β, and δ can be estimated by solving this system of linear equations. The left side of the equation is a 3x3 matrix which contains terms that involve the reference images only. This matrix is also diagonally symmetrical, which means that only $\frac{n^2+n}{2}$ of the terms need to be computed, where n is the number of degrees of freedom.

The result will be accurate provided that there are sufficient changes between the images from the various reference positions, and that the spatial gradients of each degree of freedom are approximately linear in the region bounded by the reference images. To ensure that the second constraint is satisfied, the original images can be low pass filtered to attenuate high frequency components of the signal and prevent aliasing. This can be achieved by using sufficiently low resolution optics, particularly in the case of panoramic images that typically have low angular resolution.

The above equations apply equally to computing normal image motion, however in this work only panoramic images have been used.

Egomotion

The first algorithm presented is for computing the egomotion of a mobile robot equipped with two visual sensors (see figure 3 for a picture of the robot and the configuration of the sensors). Each sensor captures a panoramic view of the environment. The algorithm requires two panoramic sensors mounted coaxially along the length of a vehicle. It is also required that the rear sensor is mounted at the point about which the vehicle pivots.

Translational egomotion is computed in the following manner:

1. Reference images are captured, from the aft (f_0) and fore (f_1) cameras, before the vehicle moves.

2. The vehicle undergoes a translation.

3. After the translation another image f is captured by the aft camera at an unknown location that lies between the original locations where f_0 and f_1 were captured.

The parameter α is then determined by solving equation 1. α lies between 0.0 and 1.0 assuming that the translation of the vehicle leaves the aft camera between the original positions of the fore and aft cameras. α is the offset of the aft camera with respect to the original position of the aft camera, normalised to the distance between the fore and aft cameras.

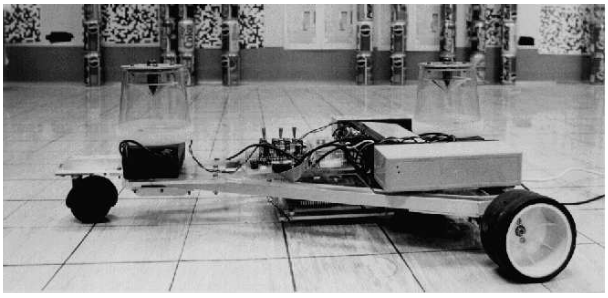

Figure 3: Photograph of the robot on which the egomotion algorithm was implemented. Note the panoramic imaging sensors on the front and rear of the vehicle.

Rotation of the vehicle about the position of the aft camera is determined similarly. However, under rotation the transformation of the image is affine and predictable, as the pattern of light on the panoramic strip image merely undergoes a phase shift equal to the amount of rotation. Thus f_1 is synthesised by software shifting f_0 which is captured by the aft camera, through a known angle. After the vehicle has rotated, the amount of rotation is computed using f_0, f_1, and the image f captured from the new orientation. The shift indicated by the value of α found by solving equation 1 is the angular position of the vehicle with respect the orientation at which f_0 was captured, and normalised to the amount of simulated rotation that produced f_1.

The feasibility of the algorithm as a navigational aid was tested by implementing it on a custom-built, mobile robot, equipped with two camera/cone assemblies, one fore and one aft. The distance between the optical axes of the two cameras was 28.5cm. The robot used a differential drive configuration, thus allowing the robot to either (i) translate along the long axis, (ii) rotate about the axle (iii) some combination of (i) and (ii). In the present implementation, however, only motions (i) and (ii) were used.

The panoramic images were smoothed by a 7 element averaging filter before applying the image interpolation algorithm. At each step of motion, the program was given information as to whether the step was a translation or a rotation, and the polarity of rotation, thus enabling the appropriate reference images to be used and/or created, and the appropriate calculations to be made based on the parameters returned by the algorithm.

The robot traveled through this environment, following an arbitrarily determined path. Three such experiments were performed, each with a trajectory encompassing 30 to 40 steps. Each step consisted of a forward translation, of magnitude distributed randomly (and uniformly) between 0cm and 25cm, or a rotation of magnitude distributed randomly between -24^o and $+24^o$. The reference rotation was 24^o in the proposed direction of travel. In each experiment, the series of elementary motion steps was integrated to recover the complete trajectory which was then compared to the actual trajectory. In all three cases (figure 4) the true path corresponds closely with that computed by the interpolation algorithm. The cumulative positional error at the end of the each trajectory was never greater than 10cm, which is less than 5% of the total path length. The final orientation of the robot, as computed by integrating the rotations computed for each step, was always within 5^o of the actual orientation.

The results are affected by a number of sources of error. Apart from the obvious errors induced by aliasing and occlusion in the environment (as the first order approximation to image deformation is not always an accurate description) there are several systematic experimental errors that are technically difficult to completely eliminate. Some of these are:

- Non-identicality of the two cameras and cones.
- Imperfect optical alignment of the cones in relation to the camera axes.
- Non-coincidence of the optical axis of the rear camera with the centre of rotation of the robot.
- Impurity of translatory and rotary motions. Random rotations during translation of up to 2.5^o were mechanically unavoidable.
- The angular reference separations for calculation of rotation were somewhat large (24^o) This was an unavoidable consequence of the relatively poor spatial resolution of the robot's motor control system.

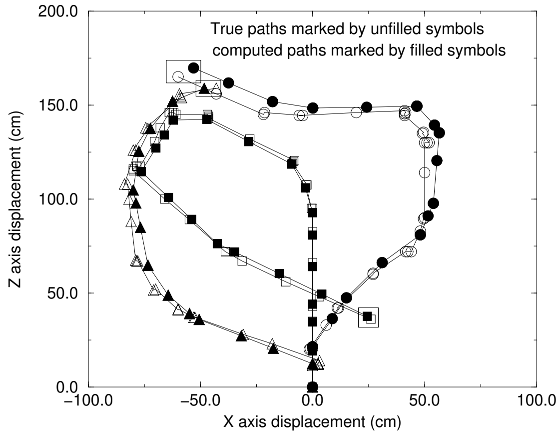

Figure 4: The robot was moved around the laboratory along arbitrary trajectories. The true path followed by the robot is shown with dotted lines and unfilled symbols. The corresponding filled symbols and unbroken lines show the trajectories computed by the algorithm. In all three cases the final position, which is indicated by boxes containing both the computed and actual endpoints, is accurately determined by integrating the responses from the algorithm.

Despite these errors, the egomotion of the robot was computed with acceptable accuracy.

We have shown that the parameters of egomotion can be computed by interpolating the deformation of the image captured by one of the sensors at the robot's present location, with respect to the images captured by the two sensors at the robot's previous location. The algorithm delivers the distance traveled and angle rotated, without the explicit measurement or integration of velocity fields. The result is obtained in a single step, without any iteration or successive approximation. No knowledge about the world is required, and the only assumption is that the world is stationary.

Range

The second algorithm computes range in the entire panoramic environment. Range estimation is based on the fact that local deformation of the panoramic image is range-dependent, regardless of the nature of the deformation. The nature of the deformation depends on azimuthal viewing direction: it is an expansion in the direction of motion, a contraction in the opposite direction, and a translation for viewing directions perpendicular to that of the motion. Range in each direction is estimated by comparing the magnitude of the image deformation measured in that direction, with the magnitude of the deformation that would be produced by a surface at a known, standard, minimum range. The estimation of local image deformation is achieved using a version of the *image interpolation algorithm*.

In the following description we will consider the case of a single mobile camera, rather than for two displaced cameras (see the causes of systematic errors in the previous section for justification of this decision). The procedure for computing range is as follows:

- A panoramic image is captured from the initial position of the camera, we call this image $f_0(\theta)$.
- The camera is moved through a known distance, and an image is captured at the new location, we call this image $f(\theta)$.
- A second reference image $f_1(\theta)$ is calculated, using $f_0(\theta)$ and the size of the sensor displacement (or for two sensors, the stereo baseline) h. This image is computed, not acquired. It is obtained by assuming that the environment surrounding the sensor at the first position is at a constant distance from the sensor in all azimuthal directions. Such a situation would prevail if the sensor were located inside a cylinder with an image of the environment painted around the inner surface, see figure 5 for a graphical representation of this process. This surface is referred to as the *virtual circle*. The

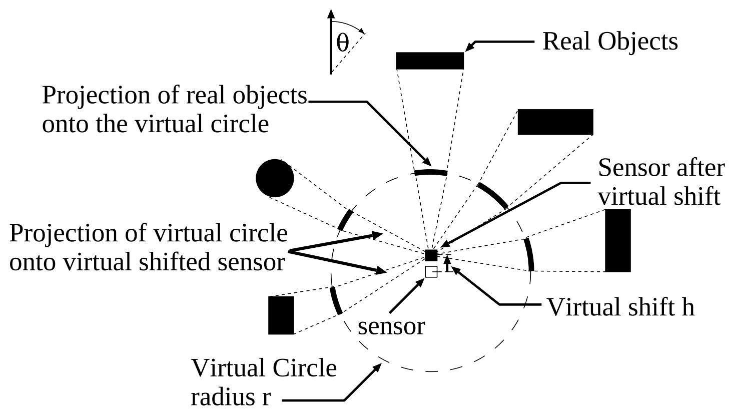

Figure 5: Procedure for generation of second reference image, f_1. The entire environment is assumed to be at the same distance as the radius r of the virtual circle. The image that would result if the sensor were to undergo a reference shift h, is used as the second reference image for the interpolation algorithm.

image that would be captured by the sensor after it is moved by a distance h, within the virtual circle (of radius r) is then computed as follows

$$f_1(\theta) = f_0\left(\theta + \sin^{-1}\left(\frac{h}{r}\sin(\pi - \theta)\right)\right) \tag{4}$$

This image provides an upper-bound to the magnitude of image deformation that can occur at any location, assuming that no object in the environment lies within the virtual circle.

- Having generated $f_1(\theta)$ the deformation between $f_0(\theta)$ and $f(\theta)$ can be compared to the deformation between $f_0(\theta)$ and $f_1(\theta)$. Local deformation between the images is computed using the interpolation algorithm. The deformation computations are performed across a small patch of the ring image, so that local range information can be computed. In our experiments overlapping 15^o and 36^o windows are used. To limit the effect of artifacts around the edges of the windows, the intensity functions within each window are weighted with triangular functions centred on the middle of each window. So after find α by solving equation 1, the nominal distance to the environment R within a given window is computed by
$$R = \frac{r}{\alpha}$$

The result of this technique is a uniform and efficient treatment of the entire panoramic image, and a robust estimation of range in every direction. Using this method a single translatory motion provides range information along every azimuthal direction.

Experimental results show that the algorithm can compute range to within 10% across overlapping regions of 36^o of azimuth, for distances of one hundred times the sensor separation. Figure 6 shows the range image computed by the technique, overlayed on an image of the environment in which the image was computed. The range sensor was moved to different locations in the room, and yet the range image returned by the technique accurately describes the geometry of the room. Window sizes can be reduced to increase the angular acuity of the range finder, however smaller windows are more likely to contain areas of no texture, and are more subject to camera noise. This is clear in Figure 7 in which a 15^o window size was used so that smaller objects could be detected by the range sensor. The small objects are accurately detected, but the walls, with their higher frequency texture (removed by the filter to a large degree), and greater distance, are not ranged accurately.

The only requirement for achieving these results is that the environment contains sufficient visual texture. Unlike ultrasonic ranging which has comparable angular resolution, this optical method provides the opportunity to determine when a range reading will not be accurate, by detecting when insufficient texture is present in the part of the panoramic image to be processed.

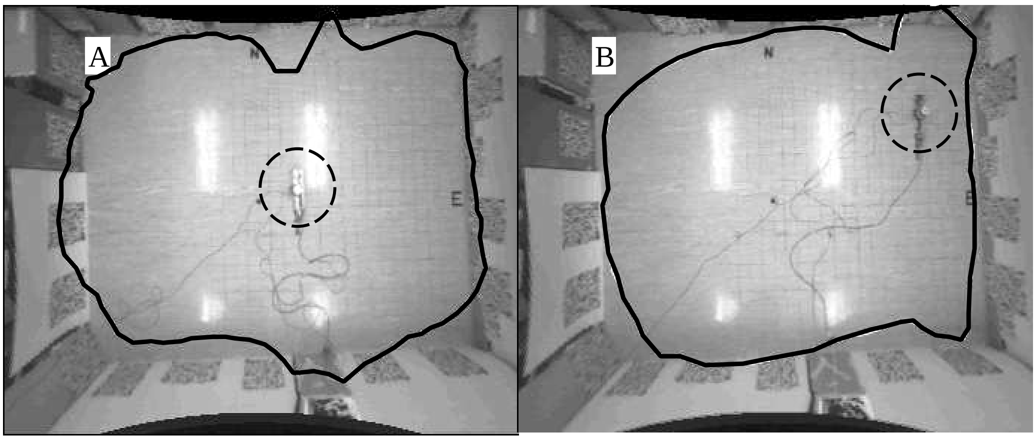

Figure 6: Images of the environment in which the algorithm was tested, overlayed with the resulting range profile. The range computation window was set to 36^o. Measurements were carried out with the sensor placed in different locations around the room to determine how closely the known range profile matched the computed range profile.

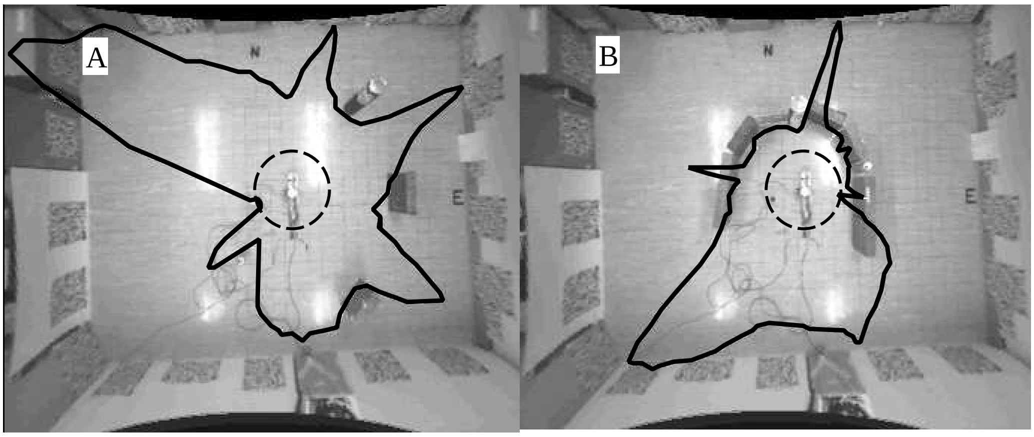

Figure 7: Range profiles for constellations of objects near the sensor. The range computation window was set to 15^o. (A) An arrangement of widely separated small objects at different ranges (yes, there are small objects at 7 o'clock and 9 o'clock – aluminium beverage cans). (B) A structure made from folders, whose ranges are accurately measured by the algorithm in most instances. However, a lack of texture leads to poor measurements at some points.

Localisation

The final algorithm determines the location of a visual sensor in a known region of space. The sensor's location is computed by interpolating the image captured by the panoramic sensor at that location with respect to panoramic images captured at known, reference locations, using the *image interpolation algorithm*. This method of computing location is, in principle, similar to the egomotion computation described earlier. Here, however, we consider more degrees of freedom of movement, and address the problem of estimating position in two dimensions, as well as orientation.

The reference images are selected at the extremes of motion for each degree of freedom, so if position is represented in three coordinates, (x, y, θ) then reference images are captured at $(1, 0, 0)$, $(-1, 0, 0)$ for x, and $(0, 1, 0)$, $(0, -1, 0)$ for y, and $(0, 0, 1)$, $(0, 0, -1)$ for θ (see figure 2).

So considering the general form of the interpolation algorithm set out in equation 2, the location in each degree of freedom is computed by solving the set of equations describing the minimum integral-squared-error for each degree of freedom (equation 3).

In tests using real images and two translational degrees of freedom, the algorithm is capable of determining the location of a camera with an accuracy better than 5cm in a region of 40cm x 40cm (figure 8F) and even better with smaller reference separations (figure 8E). One noticeable feature of the

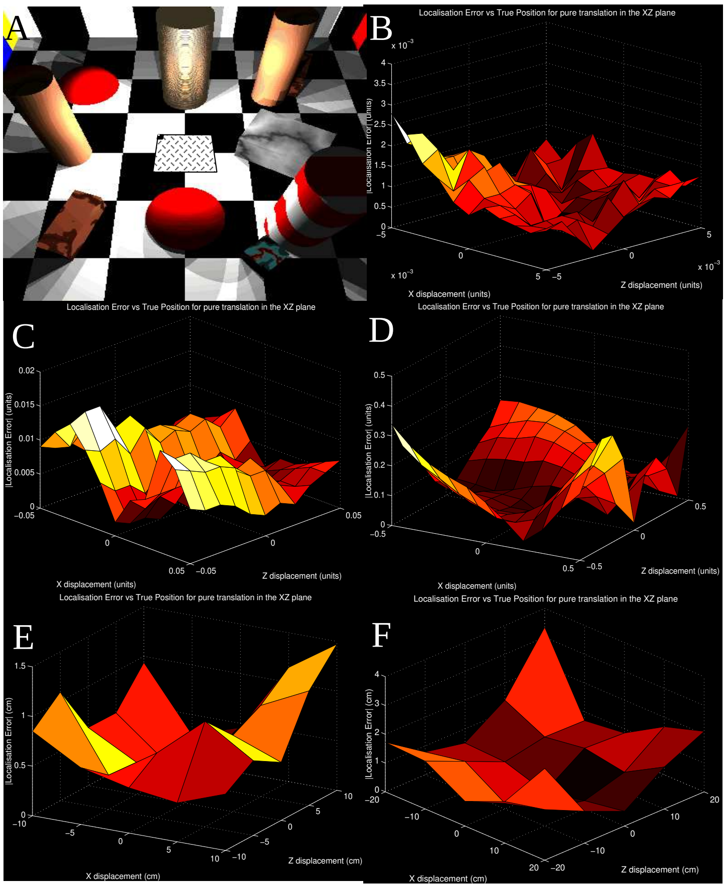

Figure 8: (A) The simulated world in which the results for (B),(C) and (D) were obtained. The textured area in the centre of the image shows a region of 1 x 1 unit, the black area in the top left corner of the large patch is 0.1 x 0.1 unit, and there is a very small area in the top left of the black area representing an area of 0.01 x 0.01 unit. The performance of the two degree of freedom interpolation algorithm is displayed as three dimensional surface plots with error as height against position on the xz plane. The plots are also coded with a "hot scale". (B) Reference separation 0.01 unit, maximum error 0.0025 units. (C) Reference separation 0.1 unit, maximum error 0.0155 units. (D) Reference separation 1 unit, maximum error 0.32 units. The results in (E) and (F) were obtained from tests in a real laboratory environment. (E) Reference separation 20cm, maximum error 1.5cm (F) Reference separation 40cm, maximum error 3.3cm.

error-surface plots is that the error is highest near the extremes of the space. This is to be expected as the first order approximation used by the image interpolation algorithm does not always accurately describe the true image deformation.

Due to mechanical constraints the 3 degree of freedom version of the algorithm could only be tested in a raytracing environment. Extensive raytraced simulations have verified the accuracy, and some of the limitations of the algorithm when operating in the three degrees of freedom possible on a plane. The graphs in figure 9A-F show the positioning error for each degree of freedom as a function of the location of the sensor on all three axes. It is apparent that the positioning error tends to be largest near the diagonals of the cube describing the three degrees of freedom.

It is proposed that the method can be used by a robot that captures reference images before leaving an area, to "home in" on a specific target location, much more rapidly than could be achieved by some sort of simple correlation technique, and perhaps more robustly than a feature recognition method.

Conclusion

All of these algorithms are computationally cheap, and well suited to implementation on mobile robots. They contain only a few divisions per computation, and no transcendental floating point operations. For ease of computation it would be appropriate to build the panoramic array as a "compound eye" of

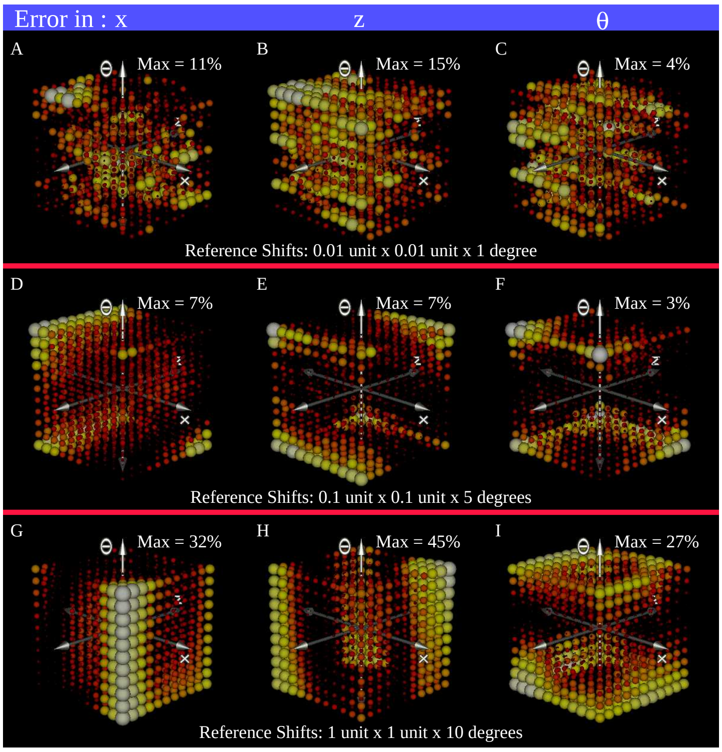

Figure 9: Plots showing four dimensional data: x, z and θ are shown with axes in Cartesian coordinates. Error, the fourth dimension is colour coded with a "hot scale" and size coded. Large errors are represented by large white spheres. The simulated environment in which the tests were performed is shown in 8A. On each row errors in x, z and θ are represented by the graphs in the left, centre and right columns respectively. The sphere sizes are all normalised to the maximum error in each dataset, which is indicated in each panel as a percentage of the reference separation. A-C show the errors for reference separations 0.01 in x and z, and 1^o in θ. D-F: 0.1 in x and z, 5^o in θ. G-I: 1 in x and z and 10^o in θ.

photodiodes rather than using a CCD array and image decoding. All of the above algorithms executed extremely rapidly after the panoramic image was extracted from the CCD image array and would run acceptably on even the slowest eight bit microcontrollers.

References

[1] M. V. Srinivasan. An image interpolation technique for the computation of optic flow and egomotion. *Biological Cybernetics*, 71:401–415, 1994.

[2] R. C. Nelson and J. Aloimonos. Finding motion parameters from spherical motion fields (or the advantages of having eyes in the back of your head). *Biological Cybernetics*, 58:261–273, 1988.

[3] J. Pichon, C. Blanes, and N. Francheschini. Visual guidance of a mobile robot equipped with a network of self-motion sensors. In *Mobile Robots IV*, pages 44–53, Philadelphia, Pennsylvania, November 1989. SPIE.

[4] J. Hong, X. Tan, B. Pinette, R. Weiss, and E. M. Riseman. Image-based homing. In *Proceedings of the 1991 IEEE International Conference on Robotics and Automation*, pages 620–625, Sacramento, California, April 1991.

[5] M. G. Nagle and M. V. Srinivasan. Structure from motion: Determining range and orientation of surfaces by image interpolation. *Journal of the Optical Society of America*, 13(1):25–34, January 1996.

[6] J. S. Chahl and M. V. Srinivasan. Visual computation of ego-motion using an image interpolation technique. *Biological Cybernetics*, 74:405–411, 1996.