Visual Target Based Wall Pose Estimation

Thomas C. Henderson and Mohamed Dekhil

Department of Computer Science
University of Utah
Salt Lake City, Utah 84112, USA.

Abstract

We compare vision and sonar techniques for the recovery of wall pose. A vision algorithm is developed which is based on known target geometry, and a sonar technique based on a wedge model is described.

1 Introduction

In any closed-loop control system, sensors are used to provide the feedback information that represents the current status of the system and the environmental uncertainties. Building a sensor system for a certain application is a process that includes the analysis of the system requirements, a model of the environment, the determination of system behavior under different conditions, and the selection of suitable sensors. The next step in building the sensor system is to assemble the hardware components and to develop the necessary software modules for data fusion and interpretation. Finally, the system is tested and the performance is analyzed. Once the system is built, it is difficult to monitor the different components of the system for the purpose of testing, debugging and analysis.

This work was supported in part by NSF grant CDA 9024721

It is also hard to evaluate the system in terms of time complexity, space complexity, robustness, and efficiency, since this requires quantitative measures for each of these measures.

In previous work (Dekhil & Henderson, 1996a), we have developed techniques to evaluate systems, and in this paper we use these techniques to compare two alternative methods to recover wall pose. One computes wall pose from a single image using the known geometry of a visual target, while the other uses two sonar readings (Henderson *et al.*, 1996c).

2 Wall Pose Estimation

The following example illustrates the use of the proposed framework to model and analyze two alternatives for determining flat wall position and orientation: one using vision and one using sonar sensors (Dekhil & Henderson, 1996b; Henderson *et al.*, 1996b; Henderson *et al.*, 1996a; Henderson *et al.*, 1997). The camera and sonar sensors are mounted on a LABMATE mobile robot designed by Transitions Research Corporation. The LABMATE was used for several experiments in the Department of Computer Science at the University of Utah. It was also entered in the 1994 and 1996 AAAI Robot Competition (Schenkat *et al.*, 1994) and it won sixth and third place, respectively. For that purpose, the LABMATE was equipped with 24 sonar sensors, eight infrared sensors, a camera and a speaker. [1] Figure 1 shows the LABMATE with its equipment.

In this example, we consider two different logical sensors to determine wall pose and find the corresponding errors and time complexity for each. The first Instrumented Logical Sensor System (ILSS) uses a camera and known target size and location. The second ILSS deals with the sonar

[1]The LABMATE preparations, the sensory equipments, and the software and hardware controllers were done by L. Schenkat and L. Veigel at the Department of Computer Science, University of Utah.

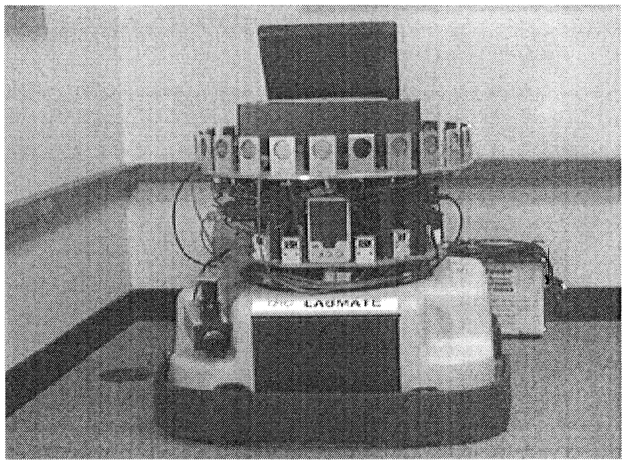


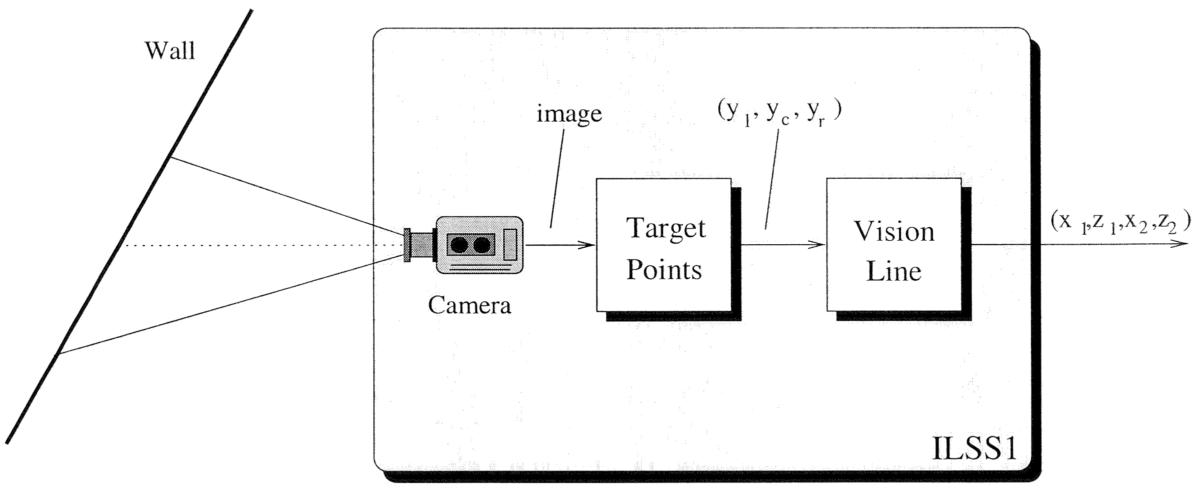
Figure 1: The LABMATE robot with its equipment.

sensor as a wedge sensor (i.e., it returns a wedge centered at the sonar sensor and spread by an angle 2θ.) Figure 2 shows the two logical sensors. (See (Henderson *et al.*, 1996a) for an overview of the sonar pose recovery technique, and (Henderson & Dekhil, 1997) for target-based calibration.)

In this figure, *image* is the 128x128 gray scale image acquired by the *Camera*, and r_1 and r_2 are the two sonar readings generated from $Sonar1$ and $Sonar2$, respectively. *Target Points* extracts three reference points from the *image*, while *Vision Line* produces two points on the line of intersection of the wall with the x-z plane of the camera system. $Wedge\_Sonar\_Line$ takes the two range values r_1 and r_2, and the spread angle of the sonar beam θ, and returns two 2D points on the line representing the wall.

2.1 System Modeling and Specification

The main ideas behind our visual target technique are shown in Figure 3. The basic idea can be



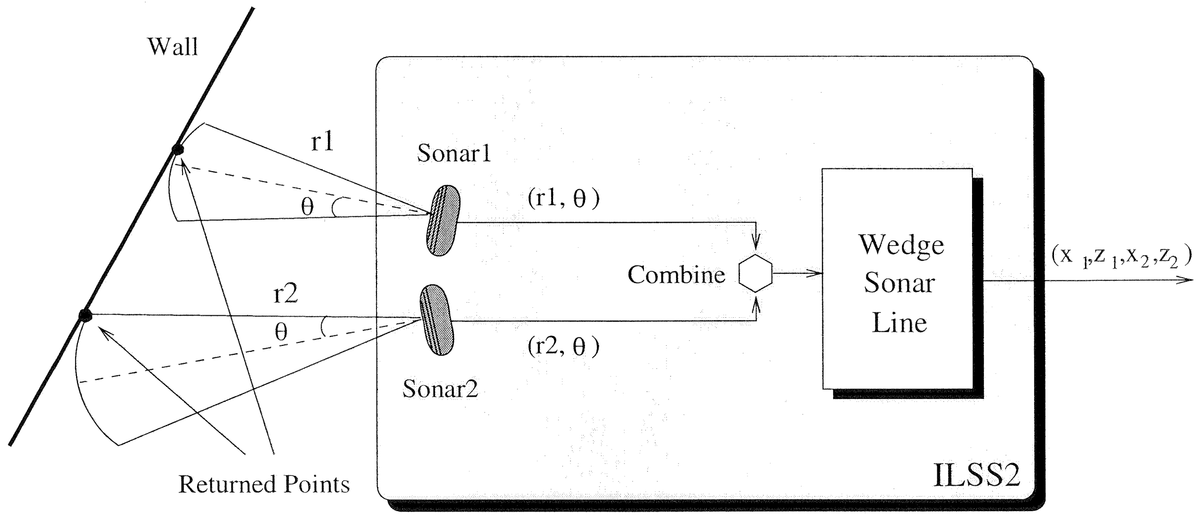


Figure 2: Two Instrumented Logical Sensors for determining wall position.

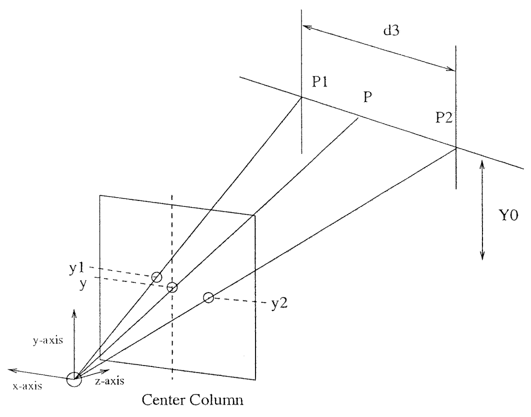


Figure 3: Visual Target Method

described by assuming the standard unit focal length camera model:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

which implies that the coordinates on the image plane will be:

$$\begin{bmatrix} \frac{x}{z} \\ \frac{y}{z} \end{bmatrix}$$

We put a horizontal line at a known height, Y_0, on the wall, with vertical lines crossing it at a known distance apart, d_3. Then we can determine the range of points on the horizontal line using the fact that

$$z = \frac{Y_0}{y}$$

where y is the y location of the line in the image plane.

Using the perspective relation, we recover the range to the projections, P_1' and P_2', onto the $x-z$-plane of the two points marked P_1 and P_2 in Figure 3, call those z_1 and z_2, respectively. We then find the range to the projected point, P', of the horizontal target line for the center column; call that z. Next we consider the triangle $\triangle P_1P_2P_3$ shown in Figure 4. The orientation of the wall is found from the fact that

$$sin(\alpha) = \frac{z_1 - z_2}{d_3} = \frac{\overline{P_1P_3}}{d_3}$$

and the range is then given by:

$$\rho = \overline{FP_4} = zsin(\frac{\pi}{2} - \alpha) = zcos(\alpha)$$

Finally, the line is represented by two points: $(0, z)$ which is the intersection of the wall plane with the camera's $z - axis$ and $(cos(\alpha), z + sin(\alpha))$, a point a unit distance along the wall on the line of intersection of the camera's $x - z$ plane and the wall.

As shown in Figure 2, ILSS1 is composed of three modules, a *Camera* module, a *Target Points* module and a *Vision Line* module. On the other hand, LSS2 has three modules, two *Sonar* modules and a *Wedge_Sonar_Line* module followed by a $Combine$ operator.

Each ILSS is defined in terms of a set of components that characterize the module. The data and the corresponding performance measures start from the $Camera$ or $Sonar$ module and propagate upward until they reach the COV of the main ILSS. On the other hand, the commands start from the main ILSS and propagate downward until they reach the $Camera$ or $Sonar$ module. The COV is composed of two parts: *data* and *performance measures*. For example, COV_{out} for $Sonar1$ is

$$(\{r_1, \theta\}, \{t, \Lambda_{r1}, \Lambda_\theta\})$$

where t is the time taken to execute the module and Λ_{r1} and Λ_θ are the error variances for r_1 and θ, respectively. In this example, each module has only one alternate subnet, therefore, the select function is trivial.

2.2 Performance Semantic Equations

Using worst case analysis, the performance semantic equations of the *time* and *error* for ILSS1 and ILSS2 can be written as:

$$time(ILSS1) = time(Serial(Camera, TargetPoints, VisionLine))$$

$$error(ILSS1) = error(Serial(Camera, TargetPoints, VisionLine))$$

$$time(ILSS2) = time(serial(combine(Sonar1, Sonar2), Wedge\_sonar\_line))$$

$$error(ILSS2) = error(serial(combine(Sonar1, Sonar2), Wedge\_sonar\_line))$$

Now, we need to calculate the time and error for the subcomponents. Assume that t_{sonar1}, t_{sonar2}, t_{camera}, $t_{TargetPoints}$, $t_{VisionLine}$ and $t_{wedge\_sonar\_line}$ are the time for the subcomponents, and Λ_{r1}, Λ_{r2}, Λ_{y_l}, Λ_{y_c}, Λ_{y_r} and Λ_θ are the error measures for r_1, r_2, y_l, y_c, y_r and θ, respectively. The time for LSS1 and LSS2 can be easily calculated using the propagation operations discussed earlier as follows:

$$time(ILSS1) = t_{camera} + t_{TargetPoints} + t_{VisionLine}$$

$$time(ILSS2) = max(t_{sonar1}, t_{sonar2}) + t_{wedge\_sonar\_line}$$

Propagating the error requires more elaborate analysis for each component. For ILSS1, we start with the error in the physical sensor which is the camera in this case. The camera generates two-dimensional arrays of intensity values, $P(x, y)$, where P is an $m \times n$ matrix. The error we are concerned abound in this example is the error in position (x, y) of a point on the CCD array (which corresponds to rows and columns in the image.) This error is affected by the resolution of the camera and the distance between the CCD elements. Let's assume that the error is Gaussian with mean 0 and variance (Λ_x, Λ_y) at any point (x, y). This can be written as:

$$error(Camera) = \{(\Lambda_x, \Lambda_y)_{m \times n}\}$$

This error translates directly into the second component, $Target\_Points$, which extracts the y value for three different points in the image; y_l, y_c, and y_r. Assuming that the variance in the y direction (Λ_y) is the same at any pixel, the error at this stage will be:

$$error(Target\_Points) = \{\Lambda_y, \Lambda_y, \Lambda_y\}$$

The last component in ILSS1, $Vision\_Line$ performs several operations on these three values to generate the two points of the line representing the wall. First, the corresponding z value is calculated for the three points using the equation:

$$z_i = \frac{Y_0}{y_i}, \qquad i = l, c, r$$

where Y_0 is the height of the physical point and is a known constant in our example. The error associated with z_i can be calculated as follows:

$$\Lambda_{z_i} = \left(\frac{\partial z_i}{\partial y_i}\right)^2 \Lambda_{y_i}$$

By calculating the derivative in the above equation we get:

$$\Lambda_{z_i} = \left(\frac{-Y_0}{y_i^2}\right)^2 \Lambda_y = \frac{Y_0^2}{y_i^4}\Lambda_y$$

which shows how Λ_{z_i} depends on the value of y_i. Second, the angle between the robot and the wall (α) is calculated with the function:

$$\alpha = sin^{-1}\left(\frac{z_l - z_r}{D_0}\right)$$

where D_0 is the known distance between the two physical points p_l and p_r. Therefore,

$$\Lambda_\alpha = \left(\frac{\partial \alpha}{\partial z_l}\right)^2 \Lambda_{z_l} + \left(\frac{\partial \alpha}{\partial z_r}\right)^2 \Lambda_{z_r}$$

$$= \left(\frac{1}{\sqrt{1-\left(\frac{z_l-z_r}{D_0}\right)^2}}\right)^2 \Lambda_{z_l} + \left(\frac{-1}{\sqrt{1-\left(\frac{z_l-z_r}{D_0}\right)^2}}\right)^2 \Lambda_{z_r}$$

After simplifying the last equation we get:

$$\Lambda_\alpha = \frac{D_0^2}{D_0^2-(z_l-z_r)^2}(\Lambda_{z_l}+\Lambda_{z_r})$$

Finally, we calculate two points on the line representing the wall as shown in Figure 5. Take the first point p_1 at $(0, z_c)$ and the second point p_2 at one unit distance from p_1 along the wall which gives the point $(\cos\alpha, z_c + \sin\alpha)$:

$$x_1 = 0, \qquad z_1 = z_c$$

$$x_2 = \cos\alpha, \qquad z_2 = z_c + \sin\alpha$$

From these equations, the error for the two points will be:

$$\Lambda_{x_1} = 0, \qquad \Lambda_{z_1} = \Lambda_{z_c}$$

$$\Lambda_{x_2} = sin^2\alpha\ \Lambda_\alpha, \qquad \Lambda_{z_2} = \Lambda_{z_c} + cos^2\alpha\ \Lambda_\alpha$$

Now, we can write the error of ILSS1 as:

$$error(ILSS1) = \{\Lambda_{x_1}, \Lambda_{z_1}, \Lambda_{x_2}, \Lambda_{z_2}\}$$

Notice that we can write the error in terms of $\Lambda_y, Y_0, D_0, y_l, y_c$, and y_r. For example, let's assume that $\Lambda_y = 1mm^2, Y_0 = 500mm$, $D_0 = 300mm$, and $y_l = y_c = y_r = 10mm$ (α is zero in this case), then the error will be:

$$error(ILSS1) = \{0, 25mm^2, 0, 25mm^2\}$$

Now we analyze ILSS2 in a similar manner. At the first level, we have the physical sonar sensor where the error can be determined either from the manufacturer specs, or from experimental data. In this example we will use the error analysis done by Schenkat and Veigel (Schenkat *et al.*, 1994) in which there is a Gaussian error with mean μ and variance σ^2. From this analysis, the variance is a function of the returned distance r. To simplify the problem let's assume that the variance in both sensors is $\Lambda_r = 4.0mm^2$. Therefore we can write the error in the sonars as:

$$error(Sonar) = \{\Lambda_r\}$$

In the $Wedge\_Sonar\_Line$ module, there are five possible cases for that line depending on the values of r_1 and r_2 (Henderson *et al.*, 1996a). In any case, the two points on the line can be written as:

$$x_1 = r_1 \cos \alpha_1, \qquad z_1 = r_1 \sin \alpha_1$$

$$x_2 = r_2 \cos \alpha_2, \qquad z_2 = r_2 \sin \alpha_2$$

where the values of $\alpha 1$ and α_2 are between $-\theta$ to θ (see Figure 6).

Considering the worst case error, we can set $\alpha_1 = \alpha_2 = \theta$. Assuming that the error in θ is zero, then the error in the calculated points is:

$$\Lambda_{x_i} = \left(\frac{\partial x_i}{\partial r}\right)^2 \Lambda_r$$

$$\Lambda_{z_i} = \left(\frac{\partial z_i}{\partial r}\right)^2 \Lambda_r$$

which results in:

$$\Lambda_{x_1} = \cos^2\theta\, \Lambda_r, \qquad \Lambda_{z_1} = \sin^2\theta\, \Lambda_r$$

$$\Lambda_{x_2} = \cos^2\theta\, \Lambda_r, \qquad \Lambda_{z_2} = \sin^2\theta\, \Lambda_r$$

Finally, the error function for $ILSS2$ is:

$$error(ILSS2) = \{\Lambda_{x_1}, \Lambda_{z_1}, \Lambda_{x_2}, \Lambda_{z_2}\}$$

As an example, if $\Lambda_r = 4.0mm^2$, and $\theta = 11^o$ (approximately correct for the Polaroid sensor), we get:

$$error(ILSS2) = \{3.85mm^2, 0.15mm^2, 3.85mm^2, 0.15mm^2\}$$

This example illustrates the importance and usefulness of the ILSS library since all these analyses can be performed once and put in the library for reuse and the user does not have to go through these details again. For example, if a different sonar sensor is used, then the same error analysis can be used by supplying the sensor's error variance. In addition, given that the error range has been determined, redundancy can be added using different sensor pairs to sense the same wall and a monitor can be added to detect error discrepancies.

2.3 Experimental Results

We do not have a very good model of our camera, and therefore actual experiments were required to compare the pose error for the two proposed techniques. The two instrumented logical sensors were used with the LABMATE to find the location of walls using real data. The goal of the experiment was to use the framework to obtain measures to help choose between a vision based wall pose technique and a sonar based wall pose estimator.

First, we calibrated the range of our visual target (a horizontal line at a known height, Y_0 with vertical stripes regularly spaced 34.2mm apart) with its y-location in the image. This was done by aligning the z-axis of the mobile robot camera to be normal to the wall; the mobile robot was then backed away from the wall a known distance and the image row number of the horizontal target line

Test No.	*Measured ρ*	*Measured θ*	*Sonar ρ*	*Sonar θ*	*Vision ρ*	*Vision θ*
1	919	-21	915.6	-20.6	888	-29.66
2	706	-27	715.4	-22.7	667	-35.51
3	930	20	924.0	23.2	783	23.99
4	1,242	0	1,226.3	4.6	1,128	10.27
5	764	32	778.5	46.1	593	43.62
6	1,164	-11	1,164.9	-13.7	1,084	-13.33
7	1,283	6	1,277.4	3.7	979	-6.53
8	1,319	-10	1,300.8	-9.8	1,084	-13.33

Table 1: Pose Results from Measured Data, Sonar, and Vision Techniques.

recorded. Figure 7 shows the results of this step. (Note that we digitized a 128x128 image; greater resolution would produce more accurate results.)

Once the target range calibration was done, the robot was placed in eight different poses with respect to the wall and the visual target acquired. Each image was constrained to have at least two vertical stripes and neither of them could be centered on the middle column of the image. The test images are shown in Figure 8.

Sonar data was also taken at each pose. The actual pose of the mobile robot with respect to the wall was independently measured by hand. Table 1 gives the hand measured, sonar and image calculated results.

The error values of the sonar and vision results with respect to the handmeasured data are plotted in Figures 9 and 10.

These results allow the user to decide whether to use one technique or the other given the global context. For example, our application was a tennis ball pickup competition in which we were using vision to track tennis balls anyway, and we needed to locate a delivery location along the wall; if we can get by with pose error of less than $0.3m$ range and 15^o angle, then ILSS1 will suffice. If less error were required, then a costly sonar system with hardware and software would need to be added to the robot, or else the use of higher resolution imagery could be explored. However, decisions made with respect to all these considerations would now be defensible and well documented. (For another detailed example comparing two alternative sonar sensor techniques to obtain wall pose, see (Henderson *et al.*, 1997).)

Note that, to keep things simple, we did not consider the error in the sonar location and orientation. However, these errors can be incorporated into the model in the same manner.

3 Conclusions

In this paper we presented a theoretical and empirical analysis of two wall pose estimation techniques.

Acknowledgment

We would like to thank Kevin Linen of North Carolina A & T for help with the experiments.

References

Dekhil, M., & Henderson, T. C. 1996a (December). Instrumented Sensor Systems. *Pages 193–200 of: IEEE International Conference on Multisensor Fusion and Integration (MFI 96), Washington D.C.*

Dekhil, M., & Henderson, T. C. 1996b (December). Optimal Wall Pose Determination in a Shared-Memory Multi-Tasking Control Architecture. *Pages 736–741 of: IEEE International Conference on Multisensor Fusion and Integration (MFI 96), Washington D.C.*

Henderson, T. C., Dekhil, M., Bruderlin, B., Schenkat, L., & Veigel, L. 1996a (February). Flat surface recovery from sonar data. *Pages 995–1000 of: DARPA Image Understanding Workshop.*

Henderson, T. C., Bruderlin, B., Dekhil, M., Schenkat, L., & Veigel, L. 1996b (April). Sonar sensing strategies. *Pages 341–346 of: IEEE Int. Conf. Robotics and Automation.*

Henderson, T. C., Dekhil, M., Bruderlin, B., Schenkat, L., & Veigel, L. 1997. Wall reconstruction using sonar sensors. *To appear in the IEEE International Journal of Robotics Research.*

Henderson, Thomas C., & Dekhil, Mohamed. 1997 (July). *Visual Target Based Wall Pose Estimation.* Tech. rept. UU-CS-97-008. University of Utah, Department of Computer Science.

Henderson, Thomas C., Bruderlin, Beat, Dekhil, Mohamed, Schenkat, Larry, & Veigel, Larkin. 1996c. Sonar Sensing Strategies. *Page to appear of: IEEE Robotics and Automation Conference.* New Jersey: IEEE.

Schenkat, L., Veigel, L., & Henderson, T. C. 1994 (Dec.). *EGOR: Design, Development, Implementation – An Entry in the 1994 AAAI Robot Competition.* Tech. rept. UUCS-94-034. University of Utah.

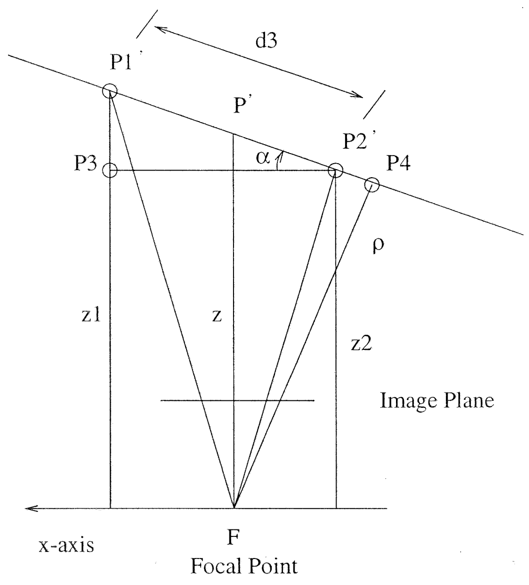


Figure 4: Recovering Orientation and Range

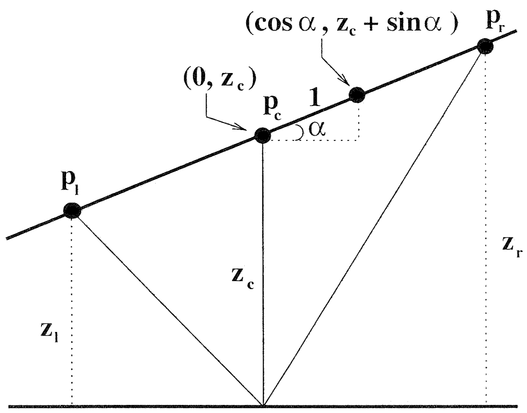


Figure 5: The two points on the line representing the wall

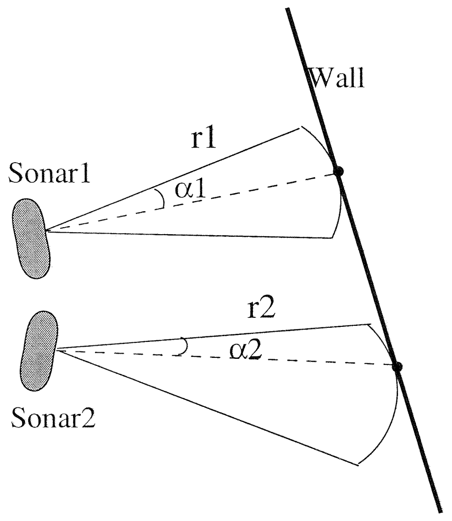


Figure 6: The general case for the points returned by the wedge_sonar_line.

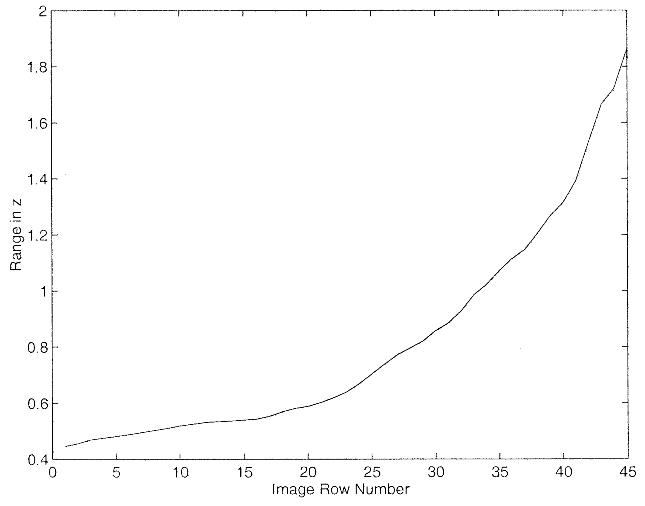


Figure 7: Row vs. Range

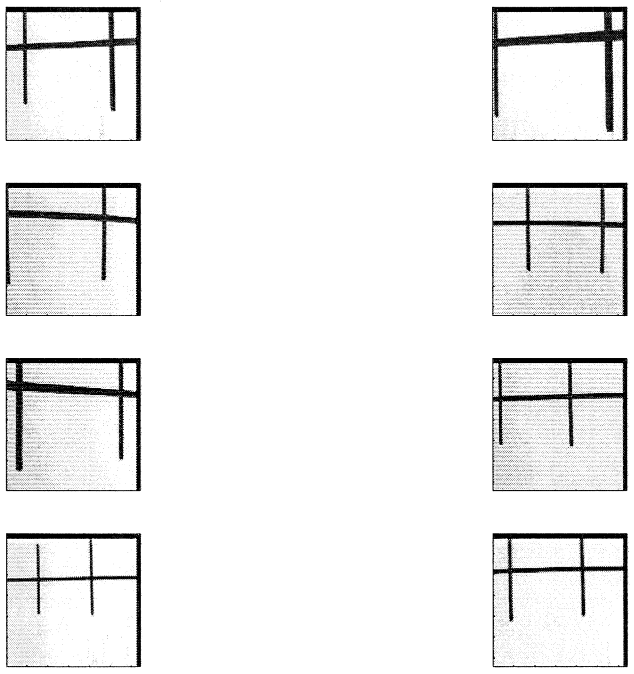


Figure 8: Visual Target Test Images

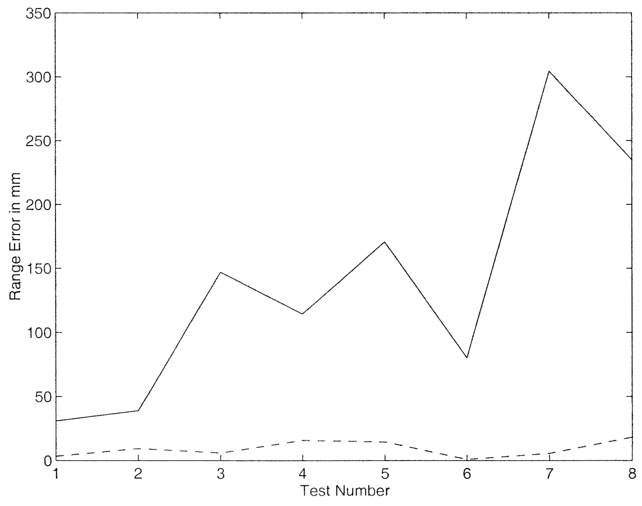


Figure 9: Error in ρ for sonar (dashed line) and vision

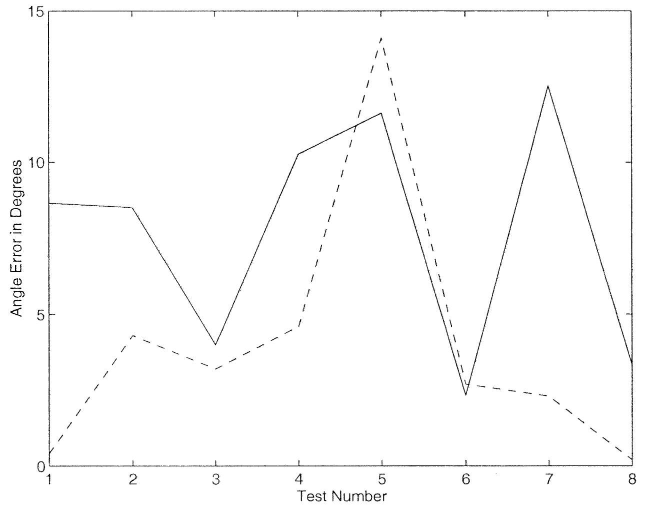


Figure 10: Error in θ for sonar (dashed line) and vision