# VISION BASED OBJECT POSE ESTIMATION FOR MOBILE ROBOTS

Annie Wu, Clint Bidlack, Arun Katkere*, Roy Feague, and Terry Weymouth
Artificial Intelligence Laboratory
University of Michigan
Ann Arbor, MI 48109-2110
aswu@engin.umich.edu

## Abstract

Mobile robot navigation using visual sensors requires that a robot be able to detect landmarks and obtain pose information from a camera image. This paper presents a vision system for finding man made markers of known size and calculating the pose of these markers. The algorithm detects and identifies the markers using a weighted pattern matching template. Geometric constraints are then used to calculate the position of the markers relative to the robot. The selection of geometric constraints comes from the *typical* pose of most man made signs; such as the sign standing vertical and the dimensions of known size. This system has been tested successfully on a wide range of real images. Marker detection is reliable, even in cluttered environments, and under certain marker orientations, estimation of the orientation has proven accurate to within 2 degrees, and distance estimation to within 0.3 meters.

## Task description

Humans are very dependent on their sense of sight for navigation. People use both natural and man-made landmarks to help them determine where they are and which way they want to go next. What humans can do with the greatest of ease, however, can be very difficult for robots. Mobile robot navigation using visual sensors typically requires that the robot be able to obtain pose information from a camera image. This task often includes recognizing markers or other known objects in the image and calculating the object pose from the size and appearance.

There are several tasks that a robot navigating by vision must deal with: the robot must to be able to extract markers from a complex environment; the robot has to recognize these markers from many different points of view; and the robot must determine, from it's view of the marker, the pose (3D position and orientation) of the marker. In addition, for all practical purposes, the robot should be able to perform all of the above tasks relatively fast (less than a few seconds in most cases).

This paper describes a vision system that was implemented for the AAAI 1993 Robot Competition in Washington D. C. on July 11-16, 1993. All vision processing was performed onboard the robot using a 80486 PC DOS based computer. A complete description of the design of the University of Michigan entry can be found in [1].

The vision system is divided into a marker extraction and identification step, and a pose estimation step. Marker extraction finds predefined markers (black 'x's and '+'s on a white background) in the environment and determines their pose relative to the robot. Thus, a robot using this system should be able to navigate autonomously using visual sensors in a semi–constrained environment. The required geometric constraints are: the marker must stand vertical; the marker and camera contain no roll; the focal length of the camera and the camera's location relative to the robot are known; the robot is oriented in the plane perpendicular to the marker; and the width and height of the marker are known. Though these constraints may seem restrictive, they are typical of most man made signs such as traffic signs and office door markers.

## Marker detection

The marker detection phase is composed of two main routines: the connected components routine and the marker identification routine. The detection phase must be both fast and accurate for the system to be useful for most real world tasks.

To maximize speed, we make only one pass through the entire image. During the pass, the image is thresholded and connected components are found and labeled. One pixel components are ignored and not labeled. Size thresholding then filters out most of the non-marker components. Only one pass is made through all possible connected components. Figure 1 shows sample output from this stage. The possible markers are outlined with a bounding box.

To identify or reject the remaining markers, a weighted pattern matching template is used. An $nxn$ template matrix is created for each marker (see Figure 2). Increasing $n$ increases the resolution of the template, but also increases the process time. We found $n = 7$ to be a good compromise. This weighted template indicates which areas are expected to be black and which ones white. The weights for our matrix are currently determined by trial and error, but we could easily replace these with machine gener-

*Currently at University of California, San Diego

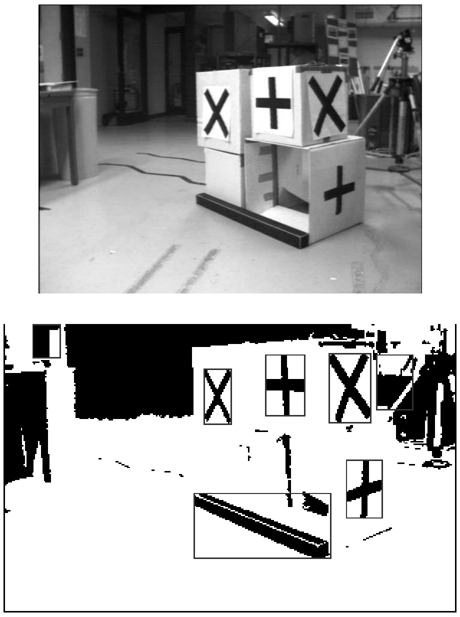

**Figure 1:** The first image is a typical input image. The second image shows the markers that are detected by the connected components routine. These markers will be identified as x, +, or neither.

| | | | | | | |
|---|---|---|---|---|---|---|
| b | w | w | w | w | b | b |
| b | b | w | w | b | b | w |
| w | b | b | b | b | w | w |
| w | w | b | b | b | w | w |
| w | w | b | b | b | b | w |
| w | b | b | w | w | b | b |
| b | b | w | w | w | w | b |

Sample marker

$$Certainty_x = \frac{\sum_r \sum_c f_x(r,c)}{\sum_r \sum_c |x_{rc}|} = \frac{92}{96} = 0.9583$$

$$Certainty_p = \frac{\sum_r \sum_c f_p(r,c)}{\sum_r \sum_c |p_{rc}|} = \frac{50}{140} = 0.3571$$

$$Certainty = max(Certainty_x, Certainty_y \tag{1}$$

$$f_x(r,c) = \begin{cases} |x_{rc}| & \text{if correct color} \\ 0 & \text{otherwise} \end{cases}$$

$$f_p(r,c) = \begin{cases} |x_{rc}| & \text{if correct color} \\ 0 & \text{otherwise} \end{cases}$$

**Figure 3:** Sample marker with calculated x and + certainty values. "b" indicates a black pixel; "w" indicates a white pixel. $x$ refers to the x template; $p$ refers to the + template. $r$ counts rows; $c$ counts columns. For this example, the program is 95.8% certain that the sample marker is an x and 35.7% certain that it is a +.

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 1 | -2 | -8 | -2 | 1 | 1 |
| 1 | 2 | 0 | -1 | 0 | 2 | 1 |
| -2 | 0 | 3 | 1 | 3 | 0 | -2 |
| -8 | -1 | 1 | 8 | 1 | -1 | -8 |
| -2 | 0 | 3 | 1 | 3 | 0 | -2 |
| 1 | 2 | 0 | -1 | 0 | 2 | 1 |
| 1 | 1 | -2 | -8 | -2 | 1 | 1 |

x template

| | | | | | | |
|---|---|---|---|---|---|---|
| -4 | -6 | 0 | 8 | 0 | -6 | -4 |
| -2 | -3 | 0 | 8 | 0 | -3 | -2 |
| 1 | 0 | 0 | 8 | 0 | 0 | 1 |
| 2 | 3 | 5 | 8 | 5 | 3 | 2 |
| 1 | 0 | 0 | 8 | 0 | 0 | 1 |
| -2 | -3 | 0 | 8 | 0 | -3 | -2 |
| -4 | -6 | 0 | 8 | 0 | -6 | -4 |

+ template

**Figure 2:** Weighted pattern templates for the x and the + markers. Positive values indicate expected black areas; negative areas are expected to be white. Certainty increases with magnitude.

ated weights if a learning program were implemented. The marker template which a component most resembles is selected as the "guess" for that component. The program generates a certainty measure with each guess (see Figure 3) and uses this measure to accept or reject the guess.

Each marker can have one or more templates. The additional templates may be used to improve marker recognition from other views.

Two types of heuristic information is also used in identifying the markers. Some heuristics were known before the program was written. Knowing that all +'s have a vertical line down the center of the bounding box, no matter what the robot's relative position, has strongly emphasized the importance of the center line in the template. Other heuristics were not learned or incorporated until after the program had been tested. Diagonal lines often scored high enough certainty values to be considered x's. Adding a specific test to verify that each possible x is not a diagonal line solved this problem.

## Pose estimation

The three dimensional position and orientation (pose) of the markers is also determined. Such information is useful for performing further analysis.

One possible application of the pose estimation algorithm is the detection of road signs. Once a sign's pose is calculated the pixels corresponding to the sign can be mapped to an orthographic projection. Since virtually all character recognition algorithms assume an orthographic projection, this would allow for *much* improved character recognition.

For the robot competition, the pose of the markers also represents the pose of the box to which the marker is attached. One phase of the competition requires the robot to autonomously move the box from one location to another. The marker pose is used to guide the robot to the box such that the box can be pushed to the appropriate location.

Geometric constraints are used to calculate the position of the markers relative to the robot. First, the marker is expected to be mounted on a planar surface and that the four corners of the marker are detected from the low level image processing (marker extraction and identification). The markers dimensions are also know in advance. Second, the marker is standing vertical. As mentioned before, this is not an unreasonable constraint as many man made signs stand vertical. Finally, the calibration parameters of the camera are known, including orientation of the camera relative to the robot and the camera's focal length. Also, there should be minimal* camera roll (rotation about the $Z$ axis).

These geometric constraints form a set of 24 equations in 18 unknowns defining the position of the four corners of the markers. This provides an overconstrained set of equations which is solved using the method of least squares. The final result are the 3D position of the four corners of the markers. For the given application, the orientation of the markers and the distance to the center of the marker are calculated from the four 3D positions. These two values are used by the robot to navigate to the markers so that more accurate identification and pose calculations can be made.

## Utilizing Geometric Constraints

Figure 4 depicts the geometry of the imaging process with the bounding box of a '+' marker being mapped to the image plane. Both the width ($w$) and height ($h$) of the markers are known. The three dimensional unit direction vectors $\vec{n1}$, $\vec{n2}$, $\vec{n3}$, and $\vec{n4}$, which are directed from the known focal center of the camera $\vec{F}$ towards the unknown marker position vectors $\vec{P1}$, $\vec{P2}$, $\vec{P3}$, and $\vec{P4}$, are calculated. This calculation is feasible given the position of the focal center of the camera $\vec{F}$, and given the four sensor plane 2D position vectors $\vec{p1}$, $\vec{p2}$, $\vec{p3}$, and $\vec{p4}$. These 2D vectors correspond to the mapping of the corners of the markers onto the sensor plane. Due to the imaging process, distances $d1$, $d2$, $d3$, and $d4$ are unknown (where $dn$ is the distance in 3D space from $\vec{pn}$ to $\vec{Pn}$). Figure 5 shows the coordinate frame assigned to the camera's sensor plane $\Psi_s$ and its relation to camera's 3D coordinate frame $\Psi_c$, the image coordinate frame $\Psi_i$, and the robot coordinate frame $\Psi_r$.

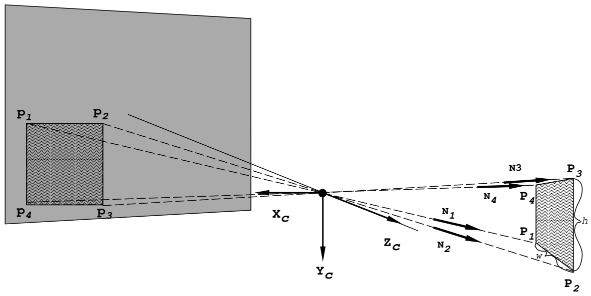


**Figure 4:** Mapping of objects onto the image plane.

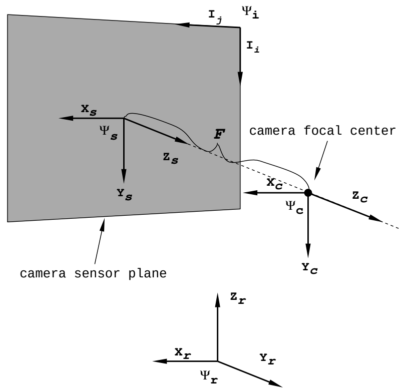


**Figure 5:** Locations of coordinate frames

It is assumed that the camera focal length is known and that the pose of the camera relative to the robot is also know. Then all points are transformed to the robot coordinate frame $\Psi_r$. This results in the following equations of known vectors:

$$\vec{n1} = [-p1_x, -p1_y, f] \quad (2)$$
$$\vec{n2} = [-p2_x, -p2_y, f] \quad (3)$$
$$\vec{n3} = [-p3_x, -p3_y, f] \quad (4)$$
$$\vec{n4} = [-p4_x, -p4_y, f]. \quad (5)$$

The vector equations with unknowns are:

$$\vec{P1} = d1 \times \vec{n1} \quad (6)$$
$$\vec{P2} = d2 \times \vec{n2} \quad (7)$$
$$\vec{P3} = d3 \times \vec{n3} \quad (8)$$
$$\vec{P4} = d4 \times \vec{n4}. \quad (9)$$

In addition the following constraint equations arise given the marker is standing vertical and that the camera and marker have no roll (rotation about the $Z$ axis). Here $d1$, $d2$, $d3$, and $d4$ are the distances from the camera focal center to the unknown 3D points $P1$,

*Current experimentation indicate that both a marker tilt and marker (or camera) tilt of up to 10 degrees do not significantly effect the calculation of the position of the marker. In addition, the effects on the orientation also seem negligable relative to other errors. Further testing is being performed.

$P2$, $P3$, and $P4$.

$$
\begin{aligned}
P2_x &= d1 \times n1_x + w \times nw_x && (10)\\
P2_y &= d1 \times n1_y + w \times nw_y && (11)\\
P3_x &= d4 \times n4_x + w \times nw_x && (12)\\
P3_y &= d4 \times n4_y + w \times nw_y && (13)\\
P4_z &= d1 \times n1_z + h && (14)\\
P3_z &= d2 \times n2_z + h && (15)\\
d1 \times n1_z &= d2 \times n2_z && (16)\\
d4 \times n4_z &= d3 \times n3_z && (17)\\
P4_x &= P1_x && (18)\\
P4_y &= P1_y && (19)\\
P3_x &= P2_x && (20)\\
P3_y &= P2_y. && (21)
\end{aligned}
$$

These equations can be expressed as an overconstrained system of linear equations with the above 24 equations and the 18 unknowns of $d1$, $d2$, $d3$, $d4$, $\vec{P}1$, $\vec{P}2$, $\vec{P}3$, $\vec{P}4$, and $\vec{nw}$. The two dimensional unit vector $\vec{nw}$ has an $x$ and $y$ component. $nw_x$ corresponds to the $x$ component of the vector pointing from $\vec{P}1$ to $\vec{P}2$, and $nw_y$ corresponds to the $y$ component of this vector. There is no $z$ component to $\vec{nw}$ since the markers, and the camera, are assumed to have no roll.

Equations 2 thru 21 result in the matrix equation

$$\vec{y} = A\vec{x}, \tag{22}$$

where $\vec{y}$ is the 24 element known vector

$$\vec{y}=(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,H,H,0,0,0,0) \tag{23}$$

and $\vec{x}$ is the 18 element unknown vector

$$\vec{x}=(P1_x, P2_x, P3_x, P4_x, P1_y, P2_y, P3_y, P4_y, P1_z, P2_z, P3_z, P4_z, d1, d2, d3, d4, nwx, nwy) \tag{24}$$

and the matrix $\mathbf{A}$ is the following:

$$
\begin{bmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&-n1x&0&0&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&0&0&-n2x&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&0&0&0&0&-n3x&0&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&-n4x&0&0\\
0&0&0&0&1&0&0&0&0&0&0&0&-n1y&0&0&0&0&0\\
0&0&0&0&0&1&0&0&0&0&0&0&0&-n4x&0&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0&0&-n3y&0&0&0\\
0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&-n4y&0&0\\
0&0&0&0&0&0&0&0&1&0&0&0&-n1z&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&1&0&0&0&-n2z&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&1&0&0&0&-n3z&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&-n4z&0&0\\
0&-1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&-1&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&-1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&0&0&-1&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&-n3z&n4z&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&n1z&-n2z&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&1&0&0&-n2z&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&-n1z&0&0&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&0&-n1x&0&0&0&-W&0\\
0&0&0&0&0&1&0&0&0&0&0&0&-n1x&0&0&0&0&-W\\
0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&-n4x&-W&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&-n4y&0&-W
\end{bmatrix}
$$

## Results

The accuracy of the pose estimation algorithm is measured by the error between the estimated and true marker distance and orientation. Robustness refers to the program's ability to detect markers and make reasonable pose estimations in complex situations such as cluttered images, tilted camera, uneven floor, etc. A set of experiments have been performed which test these measures.

The testing of this vision system has produced promising results. Marker extraction and identification is very accurate, even in cluttered images. Markers can be extracted at orientations of up to 60 degrees. Pose estimation is possible in the range of one to seven meters. Distance can be determined to within .2 meters when the marker is at an orientation of 50 degrees. Marker orientation can be as accurate as 1 degree; the ground truth measurements of orientation is approximately 1 degree, so any error at this resolution could be a factor of either the vision system or the ground truth measurements of the marker orientation. These results were obtained on low resolution images of 315 by 200 pixels. Figure 6 shows two sample images with the calculated marker pose projected onto the images.

The system should be able to extract only and all markers in an image. If a tradeoff must be made, then it is prefered to that non-markers be identified as markers. The robot can then approach *false* markers and perform further analysis to determine that indeed this marker is not a false positive. To make

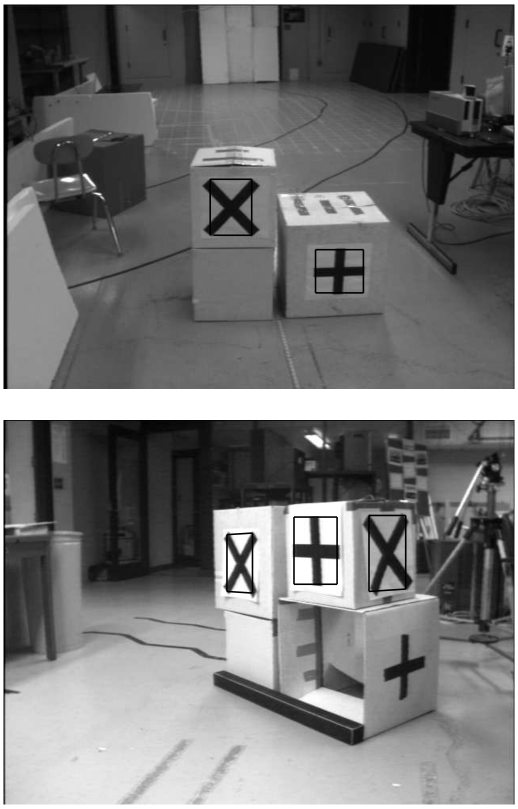

**Figure 6:** Two sample images with the calculated marker pose projected on top.

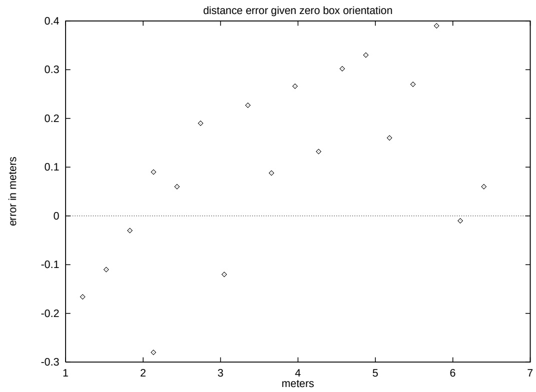


**Figure 7:** Plot of the error in calculated distance as actual distance increases and with zero box orientation.

such an analysis more tractable, the vision system should output a confidence value with each marker sighting, which would be used by the robot to determine which markers need further analysis. With each classification, the marker detection algorithm generates a certainty value as given in equation 1, and the pose estimation algorithm generates $\delta$, the residual from the least squares fit as

$$\delta = A\vec{x} - \vec{y} \tag{25}$$

These two values, *residual* and *certainty*, are available to the robot to help determine how to accept the marker and its pose.

The experiments involved processing of 42 images, each having two to four markers. Only once did the marker detection step identify a non-marker object as a marker (a false positive). The program only missed existing markers when oriented at angles greater than 50 degrees and often detects markers up to 70 degrees.

The original purpose of the marker size threshold was to eliminate obvious non-marker components as soon as possible and reduce the number of connected components that are processed by the marker identification routine. If the user can set the threshold to limit the size of the markers to a small range, fewer extraneous components are then processed by the marker identification routine, reducing the chance of false positives. Unfortunately, a small range also limits the distance at which markers can be recognized. During testing, it was found that a narrow size threshold was not crucial for accurate identification. Marker sizes in the distance images ranged from about 50 pixels at seven meters to over 1000 pixels at one meter. Even with such a wide size range, the program returned a false positive only once, while successfully finding over 100 markers in 42 test images.

Figures 7, 8, and 9 are plots of some of the experiments. The first plot displays the calculated distance error as a function of distance to the marker. All tests resulted in an error of less than 0.4 meters and over half being less than 0.2 meters. As expected, the results show that the error generally increases as the distance from the object increases. The main exception being the two data points around 6 meters that have a very small error. More data points are needed to determine if this is the not due to some unforeseen anomaly of the algorithm, or just chance, as we suspect it is.

Figure 8 displays the results from the experiment to test the distance accuracy as a function of marker orientation. The marker detection algorithm can not reliably segment markers at orientations above 60 degrees, hence the orientation plots only extend from zero to 60 degrees. An orientation of 0 degrees corresponds to the marker being perpendicular to the imaging plane. All these tests were from a distance of 2.16 meters. The distance error is within 0.13 meters with a marker orientation between zero degrees and 50 degrees.

Figure 9 represents the experiment to test the orientation calculation accuracy as a function of marker orientation. All the tests were from a distance of 2.16 meters again. This plot displays the interesting feature that the error is minimal between 30 and 60 degrees. Also, the error increases from 30 degrees back to 0 degrees of marker orientation. This effect is due to the perspective transformation; when objects are perpendicular to the imaging plane, small perturbations in the objects orientation make even smaller changes in the view as mapped to the imaging plane. The small perturbation effects increase as the angle increases (object becoming less perpendicular to the imaging plane). This effect causes fairly large

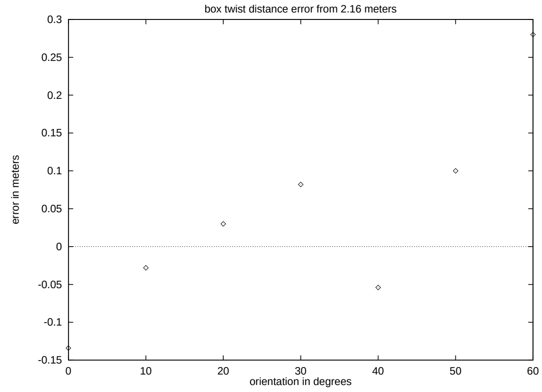


**Figure 8:** Plot of the error in calculated distance as box orientation changes and at constant distance of 2.16 meters.

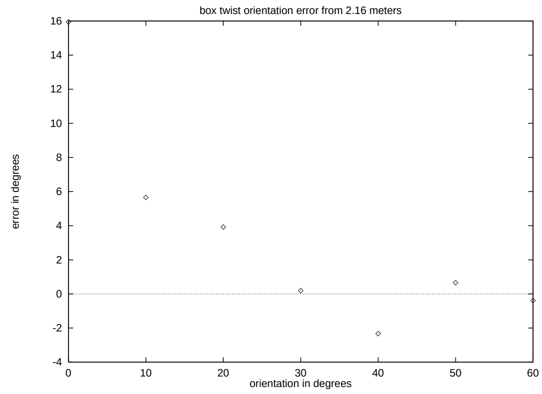


**Figure 9:** Plot of the error in calculated box orientation as actual box orientation increases.

changes in the orientation of the markers (when the object is almost perpendicular to the imaging plane) to account for small changes in the mapping of the marker onto the image plane. Hence, small changes in marker orientation go unnoticed by the algorithm when the object orientation is much less than 30 degrees. Perhaps more appropriately, small errors in the pixel locations of the four corners of the marker result in large changes in the computed object orientation when orientations are less than 30 degrees. Errors in the marker detection algorithm become more crucial under small orientation angles, with our experimental results showing this to be true as well. This *marker orientation sensitivity* can be shown analytically as well. Figure 10 shows a two dimensional representation of the problem. For our experiments, the variables $f$, $D$, and $L$ are known and have values of 0.0085 meters, 2.16 meters and 0.23 meters respectively. $f$ corresponds to the camera focal lenght, $D$ the distance from the camera to the marker, and $L$ the width of the marker. The following equations are basic geometry equations from Figure 10:

$$L_p = Dl/f \tag{26}$$

$$\theta = 180 - atan(f/l) \tag{27}$$

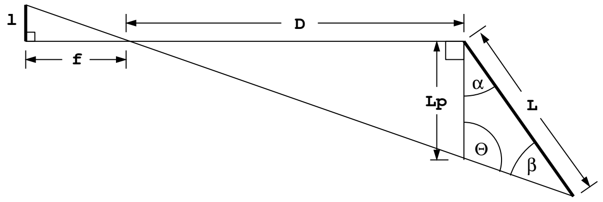


**Figure 10:** The two dimensional representation of the orientation of the marker relative to the imaging plane.

$$\beta = arcsin((Dl/(fL))sin(\theta_1)) \tag{28}$$

$$\alpha = 180 - \theta - \beta. \tag{29}$$

Now solving for $\alpha$ as a function of $l$,

$$\alpha(l) = \arctan(\frac{f}{l}) - \arcsin(DL^{-1}\frac{1}{\sqrt{1+\frac{f^2}{l^2}}}), \tag{30}$$

and its derivative with respect to $l$ is

$$\frac{d\alpha(l)}{dl} = -fl^{-2}\left(1+\frac{f^2}{l^2}\right)^{-1} - Df^2 * \tag{31}$$

$$\frac{1}{\sqrt{1-D^2L^{-2}\left(1+\frac{f^2}{l^2}\right)^{-1}}}L^{-1}\left(1+\frac{f^2}{l^2}\right)^{-3/2}l^{-3}.$$

Figure 11 represents the plot of $\alpha(l)$ for values of $l$ from zero to $\frac{L_p f}{D}$, and Figure 12 is a plot of $\frac{d\alpha(l)}{dl}$ for the same range of $l$. Notice the sharp *knee* in $\frac{d\alpha(l)}{dl}$ at $l \approx 0.0007$. This shows that for $l < 0.0007$ meters the magnitude of the rate of change of $\alpha$ with respect to $l$ is fairly constant and small. However, for $l > 0.0007$ meters, the magnitude of this rate of change increases very rapidly, meaning that small perturbations in the length $l$ (the measured width of the marker) result in large changes in the marker orientation. When the marker detection process introduces small spatial measurement errors, for example, due to quantization of the image and the due to the marker segmentation process itself, then the resulting estimated orientation errors may be very large when $l > 0.0007$ meters. This corresponds to the experimental results as shown in Figure 9. Also, from the plots in Figure 12 and 11, the location of the *knee* at 0.0007 meters corresponds to an angle of approximately 0.6 radians or 34 degrees. This in turn, corresponds to the experimental findings that the orientation error increases for values of marker orientation less than approximately 30 degrees.

## Conclusions

Results from this project indicate that it is possible to obtain useful pose information from a camera image in real time on a general purpose computer such as a 80486 based PC. Additional tests on the sensitivity of pose estimation to various parameters such as focal length value perturbations and marker size are planned. In addition, we will be studying the tradeoffs between process time (i.e. image resolution) and accuracy.

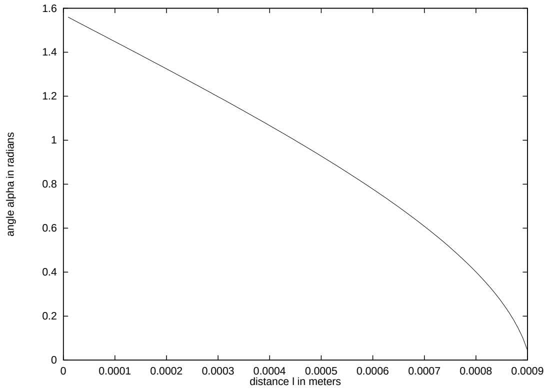


**Figure 11:** Plot of $\alpha(l)$.

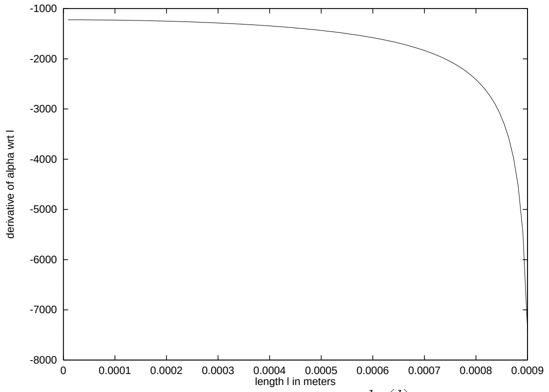


**Figure 12:** Plot of $\frac{d\alpha(l)}{dl}$.

## References

[1] David Kortenkamp, Marcus Huber, Frank Koss, William Belding, Jaeho Lee, Annie Wu, Clint Bidlack, and Seth Rogers. Mobile robot exploration and navigation of indoor spaces using sonar and vision. In *Conference on Intelligent Robots in Field, Factory, Service and Space*, March 1994.