A New Method for Estimating the Pose of Objects for Material Handling

Fatih Ugurdag and Frank L. Merat

Center for Automation & Intelligent Systems Research and Department of Electrical Engineering and Applied Physics
Case Western Reserve University
Cleveland, Ohio 44106

Abstract

Automated material handling equipment, especially that operating in unstructured environments, requires an ability to estimate the pose (position and orientation) of a prospective load rapidly and accurately even when the object is partially obscured by dust, dirt or intervening small objects such as blowing debris. It is difficult to estimate the orientation of an arbitrary object in real-time if it may be anywhere in a three-dimensional workspace. However, an *a priori* known pattern on the object can be exploited to provide pose estimation from a single image. This paper introduces a robust method called the Pi4 method which estimates the object pose from a single grey scale image of a striped pattern on the object. This Pi4 method produces pose estimates which are insensitive to partial losses of the target pattern. The performance of the Pi4 method in the presence of obscuration will be presented.

Introduction

The goal of this research is to be able to accurately estimate the pose of an arbitrary object using a single grey-scale image of the object even when the object is obscured by dirt, snow or some objects. The key words in this description are pose and obscuration. Pose is defined as the position and orientation of an object (target pattern's) coordinate frame with respect to a world coordinate frame. Pose is expressed as in the robotics literature by the homogeneous transformation matrix [**n s a p**] where the vectors **n, s, a** express the rotational and **p** the positional transformation between the object and world coordinate frames. Obscuration is defined as "the act of not being clearly seen" and can appear in several forms when dealing with real-world materials handling problems. One such form is image noise which can be caused by dirt, snow or blowing debris. Another form of obscuration is a filtering process due to atmospheric rain, fog or haze. This paper will deal only with obscuration expressed as image noise. Such obscuration due to noise can be further classified as being either constructive or destructive. This classification is based upon the image processing of the target pattern. Destructive obscuration always results in pixels corresponding to the target pattern being classified as background; constructive obscuration results in background pixels being classified as target pixels. This paper will restrict itself to negative obscuration; however, Ugurdag [1] has considered both positive and negative obscuration.

This paper describes a pose estimation system which uses a single grey scale camera and a known, high contrast target pattern on an object to estimate its pose (Fig. 1.) The image processing and resultant pose estimates of a highly visible target pattern with computer added destructive obscuration will be described.

Previous Work

The problem of pose estimation using known geometric information is not new. There have been several reported methods which use a single image and an *a priori* known target pattern to determine the pose of an object:

- *The conic method* [2]: This method uses an ellipse, a parabola or a hyperbola as the target pattern. It can handle obscuration because the algorithm does not require the entire target pattern; however, the algorithm is iterative and may take a long time to converge to a good pose estimate.
- *The three-dot method* [3,4]: If the image locations of three fixed points on the object are known, then the actual coordinates of the three points may be written in terms of three scaling parameters which can be iteratively determined using the *a priori* known distances between the three points on the object.
- *Yuan's method* [5]: Yuan used four rectangular points instead of three triangular points. He formulated the problem in a manner similar to the three-dot method but solved for pose using a different iterative method.
- *Haralick's method* [6]: Haralick, like Yuan, used four rectangular points but solved for the pose analytically.
- *The linear regression method* [7,8,9]: This method requires six or more points on the object. Pose is expressed as a twelve-element coordinate transformation matrix which can be determined by applying linear regression to the known image locations of the points on the object.

Except for Yuan's method, all the methods described above are mathematical methods which were not implemented in the real-world. They do not consider the problem of identifying the correspondence between the image points and the points on the target pattern. And, except for the conic method, they all use points to construct their target patterns.

A method which provides an analytic solution for pose is preferred because of the potentially long convergence times of an iterative method. Only Haralick's method and the linear regression method have analytical solutions. However, the linear regression technique does not utilize the known orthogonality of the target coordinate frame and may require extensive image processing to identify the correspondence between object and image points. As a consequence, Haralick's mathematical method was investigated for pose estimation in the presence of obscuration.

Wolfe [7] implemented Haralick's mathematical method as a practical method using a target pattern consisting of four small circles and appropriate image processing to identify

CH2767-2/89/0000-0605 $1.00 © 1989 IEEE

the image to object point correspondence. Wolfe's implementation can only make pose estimates over a restricted range. The assumptions that Wolfe used and their consequences are discussed in detail in [8]. This paper describes a different implementation of Haralick's method called *the Dot4 method* which works over a wide range of object poses.

The Dot4 Method

The Dot4 method uses a target pattern composed of four squares in a rectangular pattern (Figure 2) The basic asymmetry of this target pattern simplifies identifying the image-object correspondence.

The Dot4 method proceeds in two distinct computational stages: image processing and pose estimation. Image processing reduces the input grey-scale image to four distinct points with a known image to target pattern correspondence.The second stage applies Haralick's method to the centroids of the four points to yield the pose of the target pattern. The detailed derivation of Haralick's formulation of pose may be found in [1].

Image processing in the Dot4 method may be divided into four phases: thresholding, blob-coloring, segmentation, and identification. Thresholding transforms the input grey-scale image of the target pattern into a binary image where pixels are classified as either object or background pixels. Blob-coloring groups adjacent object pixels by assigning the same blob-index to them.

Segmentation is the most complex image processing phase as obscuration of the target pattern may result in more than four blobs in the image. Segmentation merges blobs which belong to the same square of the target pattern together to yield four blobs. This is done using the fact that the size of a square in the target pattern is less than the gap between the squares. Figure 3 shows an image of a Dot4 pattern before and after segmentation.

The identification step identifies the correspondence between the four squares on the image plane and the four squares on the object. As one of the squares in the Dot4 pattern is larger than the others the blob with the largest area is identified as square #1 and its centroid used as the position of square#1. The centroids of the remaining blobs are calculated and labeled as #2, #3 and #4 as they are traversed in a clockwise direction starting from square#1.

The Need for a New Pose Estimation Method

Wolfe's and the Dot4 implementation of Haralick's algorithm fail when obscuration completely covers one or more squares of the target pattern. If the target pattern squares are simply made larger the probability of a target square being completely obscured decreases; however, it becomes more probable that the target pattern squares will be partially obscured. This results in the calculated centroid locations of the target pattern squares being different from the actual centroid locations and in consequent pose estimation errors.

Overcoming the problems associated with obscuration requires a pattern whose elements are not susceptible to complete loss and whose positions can be accurately estimated even in the presence of severe obscuration. Lines fit this criteria. The position of a long, thin line can be accurately estimated even if many parts of the line are missing. Furthermore, because a line has an asymmetric shape it is unlikely that random obscuration will completely cover a line. If a line is partially covered by obscuration as seen in Fig. 4, it can still be recovered without any error. Lines are used as the basis for a new pose estimation method, *the Pi4 method.*

The Pi4 Pose Estimation Method

The Pi4 method uses an E-shaped target pattern (Fig. 5.) which looks somewhat like a Π with an extra middle leg; hence, the name Pi4.

The Pi4 method requires three parallel lines in the target pattern. The perspective transform of the camera lens maps these parallel lines into three lines which intersect at a common point on the camera focal plane. This common point is called the *vanishing point* and the coordinates of this vanishing point, through the *vanishing point theorem* [10], uniquely determine the orientation of the corresponding parallel lines in 3-D space. Although only two parallel lines are necessary to determine the vanishing point, the presence of the third line simplifies identifying the correspondence between the true lines on the target pattern and their images. As parallel lines vanish at the same point on the image plane, three of the four lines on the image plane must intersect at the same point on the image plane when they are extended. Thus, the line which does not go through the common intersection point of the other three lines is the horizontal line labeled line#4 in Figure 5.

The Pi4 method goes through two phases: image processing and pose estimation. The image processing for the Pi4 method is very similar to that described for the Dot4 method and consists of thresholding, blob-coloring, segmentation and identification. The Pi4 method uses the same thresholding and blob-coloring procedures as the Dot4 method. However, the segmentation is different than the Dot4 method merging the blobs into lines using a beamwidth criterion. Consider a line fit through the axis of least inertia of each blob. If two blobs belong to the same target pattern line, the lines through their axes of inertia should be very close to parallel; however, this merging criterion is not sufficient. A more robust criterion results from connecting the centroids of the two blobs and comparing that connecting line with the axes of each blob. The minimum angle that spans these three lines is called *the beamwidth.* Merging is done by calculating all pairwise beamwidths between blobs and merging until only four groups of blobs corresponding to the four lines of the target pattern remain.

A final line is then fit to each group of blobs along their axis of least moment of inertia. As explained above the fourth line (identified as line#4) is identified as the single line which does not intersect at a common vanishing point. The remaining lines are identified by traversing line#4 in the direction such that the centroids of the other lines are to the right. The lines crossed while traveling in this direction are labeled line#1, line#2 and line#3 in the order they are crossed.

After the identification is completed, some geometry and Haralick's algorithm is used to compute the pose of the Pi4 pattern.

Performance of the Pi4 and Dot4 Methods

Both the Pi4 and Dot4 methods give accurate results when no obscuration is present in the image. However, the Pi4 performs better than the Dot4 method in the presence of obscuration. This was demonstrated by measuring the sensitivity of pose estimates computed by the Pi4 and Dot4 methods to obscuration. Images of Pi4 and Dot4 patterns were obtained using a 244x248 pixel CID camera with a 25 mm, f1.9 lens. The camera had a grey-scale resolution of 6 bits. The target patterns were produced using an Apple LaserWriter™ printing the target patterns on ordinary white paper. The light source was standard room lighting, i.e. fluorescent lamps in a suspended tile ceiling. The pattern was horizontal and below a camera pointed down at the floor. Obscuration was artificially generated by computing random patterns with specified total noise area and superimposing them onto the target pattern images. Typical obscuration patterns are shown in Figure 6.

To test the effect of dot size upon pose estimates in the presence of obscuration two different sized Dot4 patterns referred to as Dot4(1) and Dot4(2) were evaluated. Whenever a pattern dot or line was completely obscured the pose estimation was recorded as a failure since there was not enough pattern information to calculate the pose. The three target patterns were imaged in three different poses to detect any algorithm sensitivity to a particular pose and to verify the pose range over which pose estimates could be reliably made. 400 different obscuration patterns were prepared for the performance test. 200 patterns obscured 7% of the image by area; the remaining 200 obscured 20% of the image area. The obscuration patterns used for these experiments were collections of rectangular blobs representing only destructive obscuration, i.e. if an obscured pixel overlapped a pixel corresponding to the target pattern that pixel was classified as a background pixel.

Pose estimation performance of the Pi4, Dot4(1) (the Dot4 method with a Dot4 pattern of small squares) and Dot4(2) (the Dot4 method with a Dot4 pattern of large squares) methods with 7% and 20% obscuration is summarized in Tables 1 and 2. The normalized positional error was calculated as

$$\frac{|\mathbf{p}|}{\text{range}} \times 100\%$$

where |p| is the magnitude of the vector distance between the origins of the target pattern's actual and estimated coordinate frames and range is the distance from the camera to the target pattern (approximately 1.15 meters in these experiments). The orientational error was calculated as

$$\frac{\Delta\mathbf{n} + \Delta\mathbf{s} + \Delta\mathbf{a}}{3}$$

where $\Delta\mathbf{n}$, $\Delta\mathbf{s}$ and $\Delta\mathbf{a}$ are the angles between the known and estimated orientational vectors of the target pattern. The three is used to average the orientational error over the three axes. The *number of blobs* column indicates how many blobs were in the obscured image after segmentation but prior to merging. Whenever obscuration blobs completely covered a square or a line failure occurred and the *number of failures* column records how many times this happened.

Conclusions

The experimental results demonstrate that the Pi4 method does not need the entire line to calculate line position. Mathematically, this may be regarded as the superiority of a method using second-order statistics (line fitting) as opposed to a method using first-order statistics (locating a centroid). The Pi4 uses relations rather than length or area (as in the Dot4 method or Wolfe's method) making it very difficult to incorrectly identify the stripes of the Pi4 pattern removing another source of error. Finally, the asymmetric shape of the Pi4 pattern stripes prevents the method from failure. In general, random obscuration will not result in complete loss of a line and as long as the line is not completely lost the line fitting provides a very accurate estimate of the line's location.

References

[1] H. F. Ugurdag; *A New Method for Estimating the Pose of Objects for Material Handling*; M.S. Thesis, EEAP Dept., Case Western Reserve University, May 15, 1989.

[2] R. M. Haralick and Y. H. Chu; *Solving for Camera Parameters from the Perspective Projection of a Parameterized Curve*; **Pattern Recognition,** Vol. 17, No. 6, 1984, pp. 637-645.

[3] M. A. Fischler and R. C. Bolles; *Random Sample Consensus*; **Proc. of DARPA Image Understanding Workshop,** Arlington, VA, 1980.

[4] R. Eason, M. Abidi and N. Alvertos; *A Method for Camera Calibration Using Three World Points*; Technical Report TR-EE-84-54, University of Tennessee, Nov. 30, 1984.

[5] L. C. Yuan; *An Automated Least-Error Calibration Procedure for Computer-Directed Laser Ranging Systems;* M.S. Thesis, EEAP Dept., Case Western Reserve University, Jan. 15, 1987.

[6] R. M. Haralick, *Determining Camera Parameters from the Perspective Projection of a Rectangle,* Manuscript, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, 1980.

[7] K. S. Fu, R. C. Gonzalez and C. S. G. Lee; **Robotics: Control, Sensing Vision and Intelligence,** McGraw-Hill, 1987, pp. 323-325.

[8] B. K. P. Horn; **Robot Vision;** The MIT Press, McGraw-Hill, 1986, pp. 312-315.

[9] D. H. Ballard and C. M. Brown; **Computer Vision,** Prentice-Hall, 1982, pp. 482-483.

[10] A. Fiumicelli and V. Torre; *On the Understanding of Line Drawings*; **Proc. of SPIE,** Vol. 726, Intelligent Robots and Computer Vision, 1986, pp. 158-165.

Dr.Merat wishes to acknowledge the guidance and technical support of Mr. Irv Rosen and Dr. Stephen Sousk who originally pointed out this problem while he was a Summer Faculty Member at the U.S. Army's Belvoir RD&E Center.

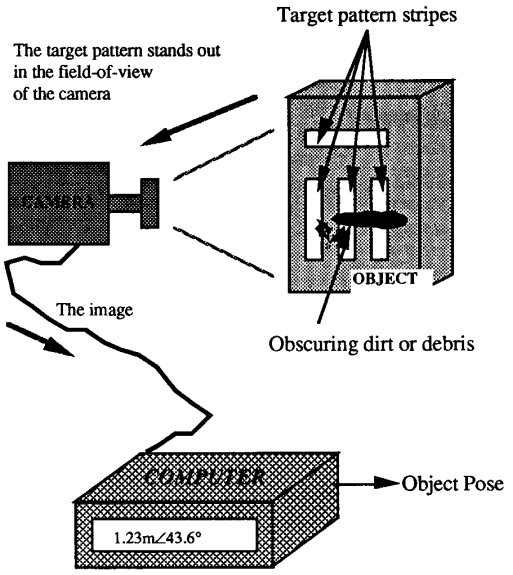


Figure 1 - Schematic Diagram of Pose Estimation

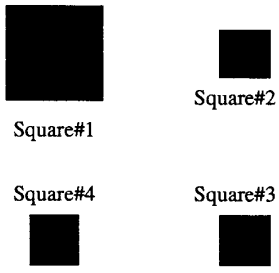


Figure 2 - Dot4 Pattern

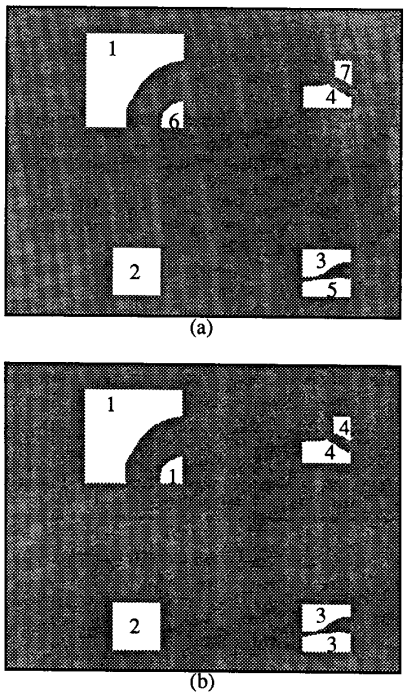


Figure 3. (a) The blob-colored image of a Dot4 pattern; (b) The segmented image. The background is labeled 0 in both images.

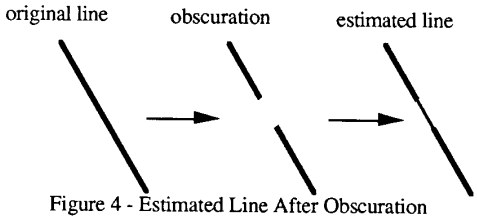


Figure 4 - Estimated Line After Obscuration

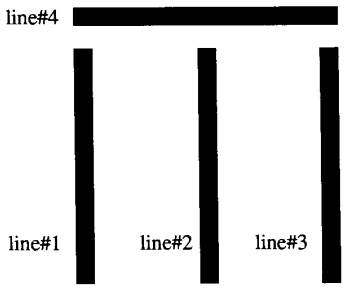
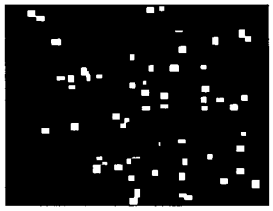
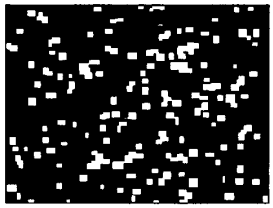


Figure 5 - Pi4 Target Pattern



(a)



(b)

Fig. 6: Two typical noise patterns. The white blobs are obscuration. The pattern in (a) has 7% obscuration whereas the pattern in (b) has 20% obscuration.

7% Obscuration		Error in orientation	Error in position	number of blobs	number of failures	total number of noise patterns
Pi4	average	0.22°	0.04 %	7	0	600
	maximum	2.53°	0.4 %	12		
Dot4(1)	average	3.62°	1 %	4	58	600
	maximum	176°	37 %	5		
Dot4(2)	average	3.67°	0.8 %	4	0	600
	maximum	30.8°	8 %	7		

Table 1 - Pose Estimation Results for 7% Obscuration

20% Obscuration		Error in orientation	Error in position	number of blobs	number of failures	total number of noise patterns
Pi4	average	0.43°	0.007 %	13	0	600
	maximum	3.82°	0.4 %	20		
Dot4(1)	average	4.54°	1 %	4	174	600
	maximum	180°	90 %	5		
Dot4(2)	average	7.93°	4 %	4	1	600
	maximum	118°	1270 %	8		

Table 2 - Pose Estimation Results for 20% Obscuration