Robust robotic manipulation

Ovidiu Ghita[1] and Paul F. Whelan[2]

Vision Systems Laboratory, School of Electronic Engineering,
Dublin City University, Glasnevin, Dublin 9, Ireland

ABSTRACT

Robotic systems based on visual recognition are increasingly more difficult to implement in real time as the complexity and the number of target objects increases. Flexible robotic implementations require a 3-D scene understanding. To obtain robust 3-D information an active sensor based on depth from defocusing technique is implemented. The algorithm involves calculating distance to points in a scene using the relative blurring between two images detected with two different focal settings. These images are acquired by splitting the image of the scene and capturing them with two CCD sensors with a physical known distance between sensor planes. The recognition algorithm uses a variant of eigenimage analysis. This approach computes the eigenspace determined by processing the eigenvalues and eigenvectors of the each image set. Each image set is obtained by varying the pose of each object in space. Also, details of the local image processing used for localisation and the centering of the object are presented and discussed. The presented system has been successfully implemented and tested on several real world scenes.

Keywords: 3-D, centroid, depth from defocus, real-time, eigenimage

1. INTRODUCTION

In industry a large number of robotic applications are implemented widely as a way to improve productivity by removing the human operator from monotonous or dangerous tasks. At present, a wide range of tasks such as manufacturing, assembling or packing are still carried out manually. An important goal of an intelligent robotic vision system is to locate, recognise and manipulate complex objects as close to real time as possible. Due to the permanent improvements in computer technologies (hardware and software), new complex and efficient vision-based algorithms that require a significant amount of computation power have been possible to be develop.

In this paper we present an approach for a robotic application which is able to locate, recognise and manipulate different complex objects using multi-sensorial information. One CCD camera is used only to locate the objects and another two are involved in 3-D recovery and object recognition. In order to remove the background information and the noise an adaptive threshold is applied. The scene is segmented by an application of labelling algorithm, the background has the label (value of the pixel) zero and objects are represented by non-zero labels. The co-ordinates of each object are determined by an application of centroid detection for each colour except black that represents the background. Now, the objects are located and the next stage involves the object recognition algorithm. The last stage is represented by the object manipulation.

In general two-dimensional approaches usually consider geometrical aspects and for some applications such as sorting and assembly require a relatively simple implementation. Many applications like bin-picking cannot be implemented using only 2-D information, therefore three dimensional (3-D) information is necessary to be achieved [7]. For this implementation the 3-D structure is obtained using an active 3-D range sensor based on depth from defocusing technique. This approach presents some advantages in comparison with other methods such as stereo or laser scanning but also has some limitations.

The vision recognision task is performed by an algorithm based on the use of eigenvectors (eigenimage analysis). This approach can be briefly described as a method, which computes the eigenspace determined by processing the eigenvalues and corresponding eigenvectors of the image set. The eigenspace is determined in order to obtain a low dimensional subspace.

[1] Ovidiu Ghita, Email: ghitao@eeng.dcu.ie, WWW: http://www.eeng.dcu.ie/~whelanp/vsg/vsgper.html
[2] Paul F. Whelan, Email: whelanp@eeng.dcu.ie, WWW: http://www.eeng.dcu.ie/~whelanp/home.html

For an unknown input image, the recognition algorithm projects this image to eigenspace and the object is recognised using a space partitioning method. The space partitioning method determines if the object is included in the database and also its position in space according with the number of images that describe the position from the image set. In this implementation we assume that the objects are not occluded.

Section 2 presents the system overview. In section 3 is briefly presented a simple solution to identify and locate complex objects. Section 4 describes the 3-D active range sensor. In section 5 an algorithm for object recognition and pose estimation is presented. Section 6 shows some experimental results and evaluates the stability of the system. Section 7 concludes this paper.

2. THE OVERALL SYSTEM

We will present and discuss a system organised in a multi-sensorial robotic workcell. The goal of the system is to identify, locate, recognise and manipulate complex objects placed on the robot's workspace using the remote information achieved from different sensors. An important problem generated by the use of multiple sensors is the way to interconnect (integrate) the sensorial information in order to obtain an optimal implementation [10]. Several problems concerning calibration and positioning for each sensor, can appear when the system is implemented. Furthermore, when the system has a strong interaction with the external environment (lights, noise, vibrations, etc.) more problems may arise and an optimal solution to solve these problems can vary from case to case. The proposed system can be divided into several modules (blocks) and its block diagram is shown in figure 1.

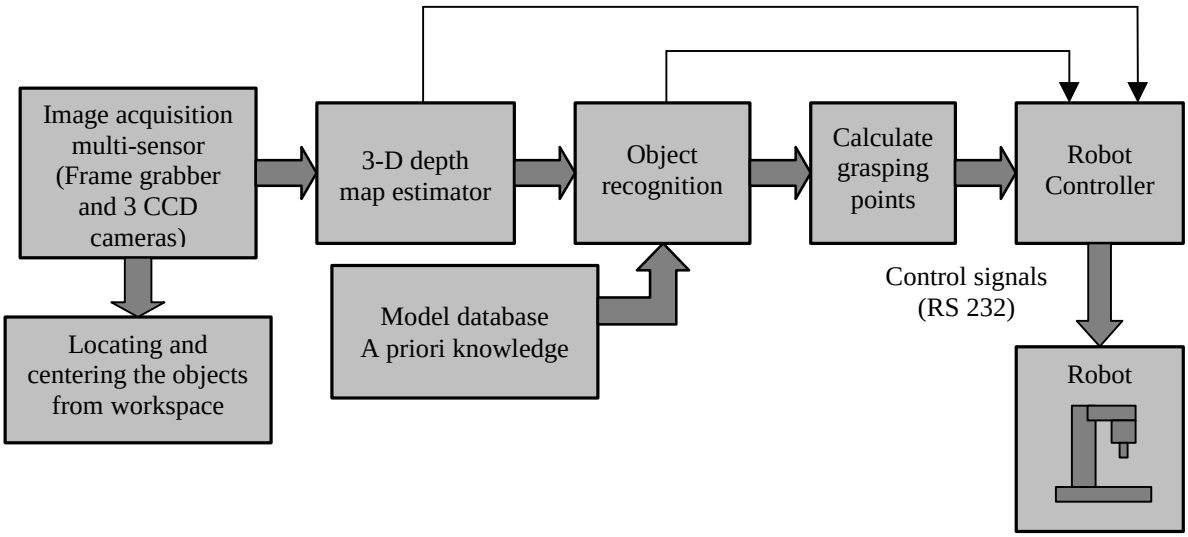


Figure 1. The block diagram of the system

For the sake of computational efficiency the system was design in such way to limit to minimum the data transferred between different modules. The image acquisition multi-sensor contains a VISION*plus* AT frame grabber [9], 3 CCD cameras (256 by 256, 256 grey levels) and a light projector used for 3-D recovery. The robot used in this implementation is an ADEPT manipulator system which is a four-axis robot. The communication between computer and the robot is carried out using the serial port.

3. IMAGE INTERPRETATION

The task of the image interpretation is to locate every object placed on the workspace. In order to solve this task a strategy that involves 2 CCD camera is used. The first camera gives a general view for workspace and is not attached to the robot's arm, while the second which is involved in the 3-D sensor yields a local view. The use of a separate camera which acquires an image of the entire workspace is a certain advantage once this view is enough to determine the approximate location of every object. After that every object can be precise located using the information obtained from the second camera. In order to remove the noise and to normalise the background an adaptive threshold is applied. This is followed by the application of the labelling algorithm that assigns a unique label for each independent object. The initial position for each object is

determined using a procedure which computes the centroids for each object. These co-ordinates are translated into the robot's domain by a simple scaling operation that returns the approximate positions of the objects. At this stage the second CCD camera which is attached to the robot's arm is involved in order to center the object. These operations are illustrated in the figure 2.

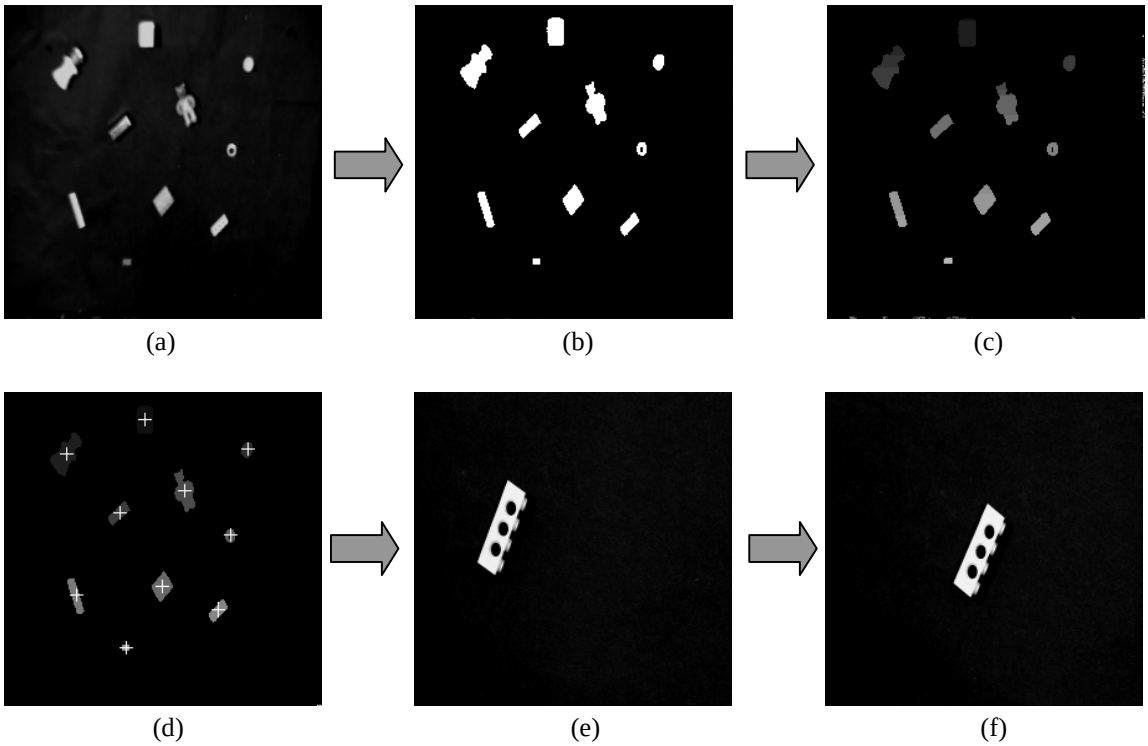


Figure 2. Operations involved to locate the objects, (a) the initial image, (b) adaptive threshold is applied, (c) image after labelling, (d) the centroids for each object are determined, (e) and (f) the robot's arm centers the object in two stages

4. THE RANGE SENSOR

The range sensor used in this implementation is based on depth from defocusing technique. The depth of the field can be determined from the degree of image blurring. The relative blurring in two images is computed using a high pass operator (in this approach is used the Laplacian filter) which yields the 3-D structure of the scene [21].

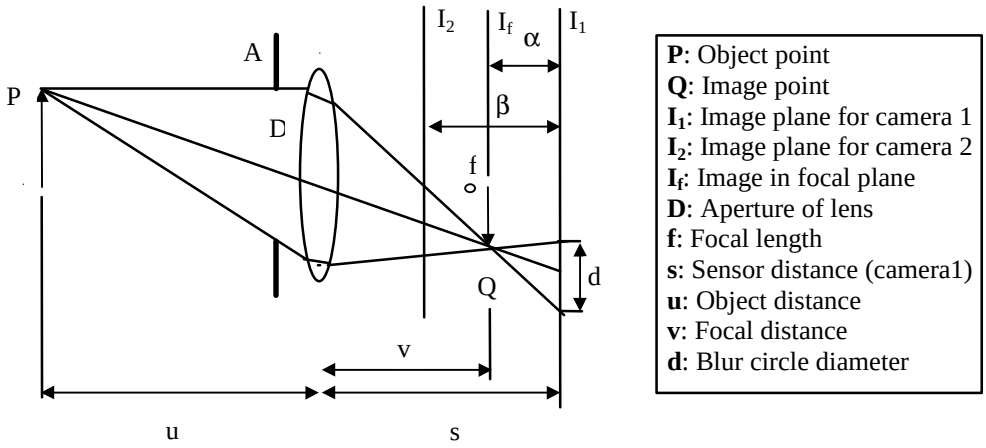


Figure 3. Image formation and the principle for depth from defocus

The depth of the field is given by a relationship between focused and defocused images. In this present implementation one of the image is near focused and other is far focused. The optical settings and the basic image formation geometry is shown in figure 3. The distance *u* of the focused object depends on the focal length *f* of the lens and the distance *v* between the lens and the position of the focused image. The relationship between *f, u, v* is expressed by the Gaussian law:

$$\frac{1}{f} = \frac{1}{u} + \frac{1}{v} \tag{1}$$

Each point of the object plane is projected onto a single point on the image plane and the result is a focused image I_f. If the CCD sensor is not placed in the focused position the energy received from P by the lens is distributed over a patch on the sensor plane, instead of single point for the focused image. Thus the image formed by the lens in any other position than focused is blurred and the measure of the blurring being given by the distance from focused plane. By the use of similar triangles the blur circle (the diameter of patch on the sensor plane) is given by:

$$\frac{D/2}{v} = \frac{d/2}{s-v} \qquad d = Ds(\frac{1}{v} - \frac{1}{s}) \tag{2}$$

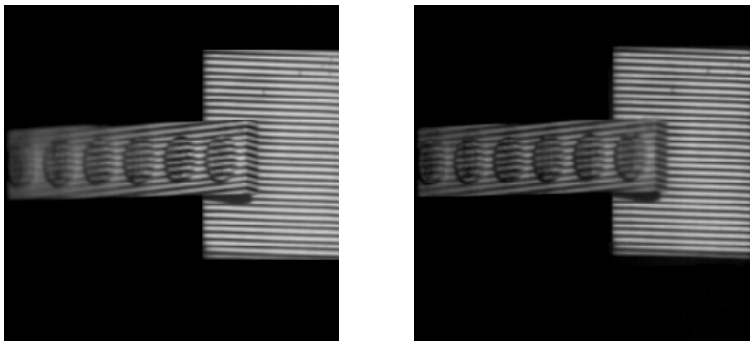
Using the previous formula we can rewrite in terms of *D*, *s* and *u*.

$$d = Ds(\frac{1}{f} - \frac{1}{u} - \frac{1}{s}) \tag{3}$$

A simple solution to determine is achieved by using two images I_1 and I_2 separated by a known distance ↕. The problem is reduced to obtain a relationship between relative blurring of each point in the two images and to compute the distance ↦ of its focused image. Using *v = s* - ↦, the lens law yields the depth u of the scene point. The transformation from the focused image into defocused image is linear shift invariant operation. This operation is called point spread function and gives an indication regarding the amount of blurring. The point spread function *h(x,y)* is cylindrical and is given by the following relation:

$$h(x,y) = \begin{cases} \frac{4}{p d^2}, & x^2 + y^2 \le \frac{d^2}{4} \\ 0 & otherwise \end{cases} \tag{4}$$

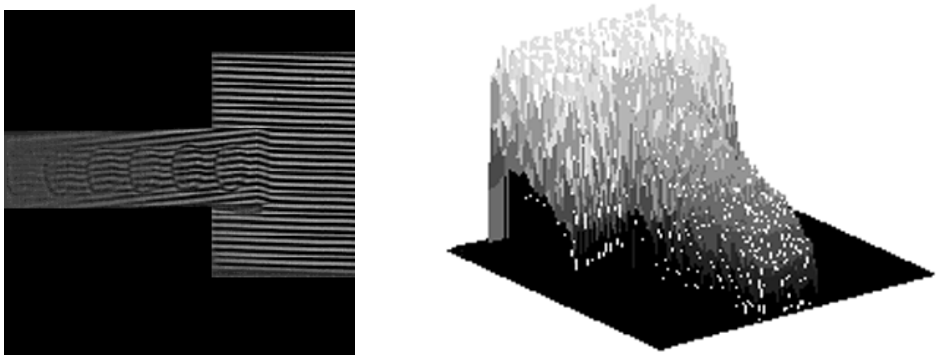
This function performs as a low pass filter where the low frequencies are passed unaltered while higher frequencies are reduced in amplitude in accordance with *d* that represents the degree of blurring. Because the point spread function is a low pass filter, the depth structure is obtained by the application of a high pass filter. For this implementation the Laplacian filter is used. After the application of the Laplacian for near and far focused images the depth information is obtained by computing the difference between these images.



(a) (b)

Figure 4. Near (a) and far (b) focused images

The difference between near and far focused images is very subtle and in order to enhance the features of the scene an active illumination is used (more details in [17]). For this purpose a structured light is projected that uses a simple stripes grid (MGP-40) used in Moire contour generation [19].



(a) (b)

Figure 5. Computed depth map (a) and 3-D representation (b)

This approach requires a simple calibration between sensors and the algorithm involved is relatively simple. Although this method presents some limitations (performs modest for specular objects and a precise depth recovery is obtained only for nearby objects), the depth from defocusing technique offers an interesting solution to determine the 3-D structure fast and precise at low cost.

5. OBJECT RECOGNITION

One of the main aims of an intelligent robotic vision system is to recognise objects and to compute their position in space. The vision recognition task must be performed as close to real time as possible. Therefore the recognition algorithm must be computationally inexpensive. In this paper we present an approach based on the use of eigenvectors (eigenimage). This can be briefly described as a method, which computes the eigenspace determined by processing the eigenvalues and eigenvectors of the image set (see also [6], [13], [14], [15], [22], [26]). For our practical implementation in order to decrease the number of images, the image set is obtained by varying pose while maintaining a constant level of illumination. The eigenspace is determined using the eigenvalues (eigenvectors) of covariance matrix (i.e. a symmetric matrix) in order to obtain a low dimensional subspace. For an unknown input image, the recognition algorithm projects this image to eigenspace and the object is recognised using a space partitioning method, which determines the object and also its position in space according with the number of images that describe the position from the image set [13].

The image set is normalised in brightness and the background noise removed in order to eliminate the redundant information. The eigenspace for the image set is built by computing the largest eigenvalues (eigenvectors) of the set. The next step is to project all the images onto eigenspace and we obtain a set of points that characterise the object and the pose in space. In order to increase the resolution for position estimation we connected these points in eigenspace and an unknown position can be better approximated. In our approach we assume that the objects are not occluded. Another important problem can be the texture reflectance and this problem is directly connected to the illumination conditions.

In comparison with other approaches based on geometry recognition (see also [1], [8], [11], [23]) this method is suitable for a large number of objects with different shape geometry.

5.1 Eigenspace technique

The image I(x,y) is represented by a two-dimensional 256 by 256 array of 8-bit intensity values. This image can also be considered as a vector of dimension 65536. This is obtained by reading pixel brightness values in a raster scan manner. To reconstruct an image we map a collection of points (in our case eigenvalues) in this huge space. The main idea of the principal component analysis (Karhunen-Loeve expansion) is to find the vectors that can better describe the image (or the entire space).

The image I can be represented as:

$$I = [i_1, i_2, i_3, \ldots, i_N] \tag{5}$$

where $i_1, i_2, \ldots, i_N$ are the pixel values.

This vector represents an unprocessed image. The reflectance proprieties of the object shape can vary from scene to scene. If the illumination conditions of the environment are maintained constant, the appearance is affected only by object position.

The image set for an object is obtained as:

$$[I_1, I_2, I_3, \ldots, I_P]^T \tag{6}$$

where P is the number of considered positions in space.

Because we do not want our images to be affected by the variations in the intensity illumination or the aperture of the imaging system (lens), we normalise each image so that the total energy contained in the image is unity [13]. The normalised image is obtained by dividing each pixel by the following number:

$$i_n' = \frac{1}{B} i_n, \qquad B = \sqrt{\sum_{n=1}^{N} i_n^2} \tag{7}$$

Before computing the eigenspace we need to calculate the average image (A) of all images:

$$A = \frac{1}{P} \sum_{i=1}^{P} I_i' \tag{8}$$

The image set will be obtained by subtracting the average image (A) from each normalised image:

$$S = [I_1' - A, I_2' - A, I_3' - A, \ldots, I_P' - A]^T \tag{9}$$

The dimension of matrix S is P x N, where P is the number of positions (images) and N the number of pixels. The next step involves the computing the covariance matrix:

$$C = S^T S \tag{10}$$

This matrix is very large (65536 x 65536) and it will be extremely difficult to compute the eigenvalues and the corresponding eigenvectors. If the number of images P is smaller than N it is easier to construct the P by P matrix $Q = SS^T$, but in this case the dimension of space is maximum P. The eigenvalues and the corresponding eigenvectors are computed solving the following well known equation:

$$Qu_i = v_i u_i \tag{11}$$

where u_i is the i^{th} eigenvector and v_i is the corresponding eigenvalue.

In this implementation the eigenvalues and eigenvectors of the covariance matrix are computed using the Householder algorithm in order to obtain a simple tridiagonal form. This is followed by an application of the QL algorithm [20]. The eigenspace is obtained by multiplying matrix of eigenvectors (eigenmatrix) with the matrix S:

$$E = US \tag{12}$$

where $U=[u_1,u_2,\ldots,u_P]^T$, U is P x P dimensional. As we expect the matrix E that represents the eigenspace is P x N dimensional.

Using this approach the number of calculations is significantly reduced but in this case the number of eigenvectors is relatively small (up to 36). For an exact description we need N eigenvectors, but for recognition purpose a smaller number is sufficient to describe a low dimensional subspace.

5.2 Object recognition and position estimation

Once the eigenspace is computed we can project all images from the set on this subspace ($E=[e_1, e_2,\ldots,e_p]^T$). The result will be a collection of points that describe the object and its position. Before projecting the image set onto eigenspace, we subtract the average image from the image set.

$$h_i = [e_1, e_2, \ldots, e_P]^T (I_i^{'} - A) \tag{13}$$

where $e_1, e_2, \ldots, e_p$ are the eigenspace vectors and each vector is N dimensional.

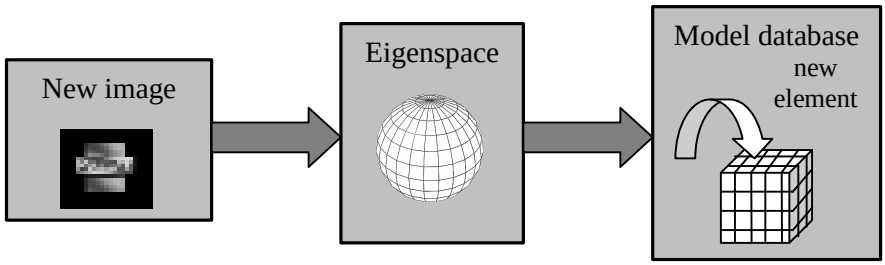


Figure 6. Building the eigen database

As we expect each point is P dimensional and this appears to be a sever restriction in image reconstruction. But for the purpose of recognition, this allows us a fairly accurate method under the condition of maintaining a constant illumination. If these points are connected in P space to give us the possibility of estimating positions of the object that are not included in the image set. A direct connection between position of the object and the projection on the eigenspace (or object space) is obtained.

An unknown input image will be projected onto eigenspace and as result we will obtain a P dimensional point. This vector can be used in a standard recognition algorithm and the simplest method to determine the best matching is to compute the Euclidean distance [13], [25].

$$d_i^2 = \left\| h - h_i \right\|^2 = (h - h_i)^T (h - h_i) = h^T h - h_i^T h_i \tag{14}$$

where $i=1,2,\ldots,P$ and h is the projection of the input image onto eigenspace. A better representation in space is offered by Mahalanobis distance (for more details see [24]), which is related to the image set, but for the sake of computational efficiency the Euclidean distance was used. The Mahalanobis distance is given by:

$$d^2 = (h - h_{m,X})^T C_X^{-1} (h - h_{m,X}) \tag{15}$$

where $h_{m,X}$ is the mean of the class X and C_X^{-1} is the inverse of the covariance matrix. The use of the Mahalanobis distance removes several limitations of the Euclidean distance, such as a better correlation between features (in our case P-dimensional points) that belong to different classes and is invariant to any nonsingular linear transformation. If the features are uncorrelated in this case the Mahalanobis distance becomes similar to the Euclidean distance.

We consider the recognition task to find the object and its position. The object is in the collection when the object gives us the minimum distance and this distance is smaller than a threshold value.

$$d_i = \min \left\| h - h_i \right\| \le J \tag{16}$$

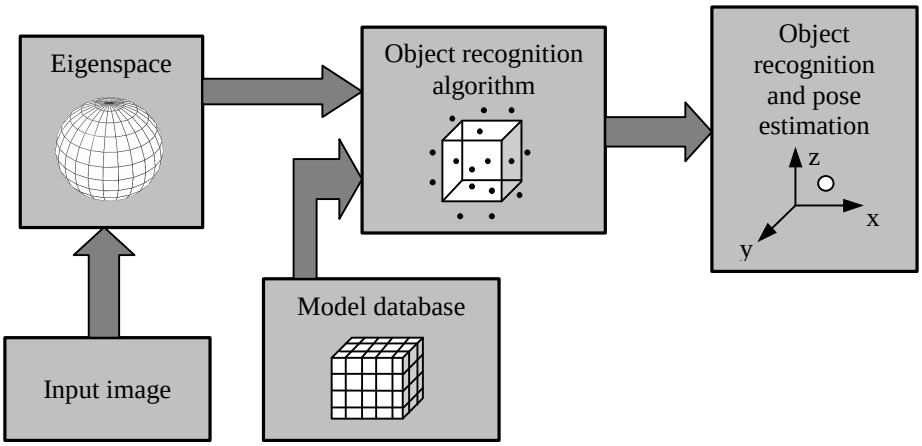


Figure 7. Data-flow diagram of the object recognition system

If we have a large collection of objects the best approach is to compute the universal eigenspace which contains all images and all positions. The first step is to recognise the object projecting the input image on this universal eigenspace and after this we can estimate precisely the position by projecting the input image on the object eigenspace.

For a large database a better solution is an algorithm based on space partitioning. *Hashing and indexing* techniques are the simplest implementation, but the index table increases exponentially with the dimension of space. Space partitioning techniques provide a practical solution for multi-dimensional search implementation. One of the most popular space partitioning technique is the *k-d tree* developed by Bentley and Friedman [4]. The *k-d tree* structure partitions the space using the hyperplanes perpendicular to the co-ordinate axes. The database has a root node, and the rest of the points are lying on one hyperplane. All the points in database are connected by passing through the root node, the new points are added to the left child (the left child is the point derived from the root node). This process is applied on the left to right until all the points from database are classified. This algorithm is illustrated in the figure 8.

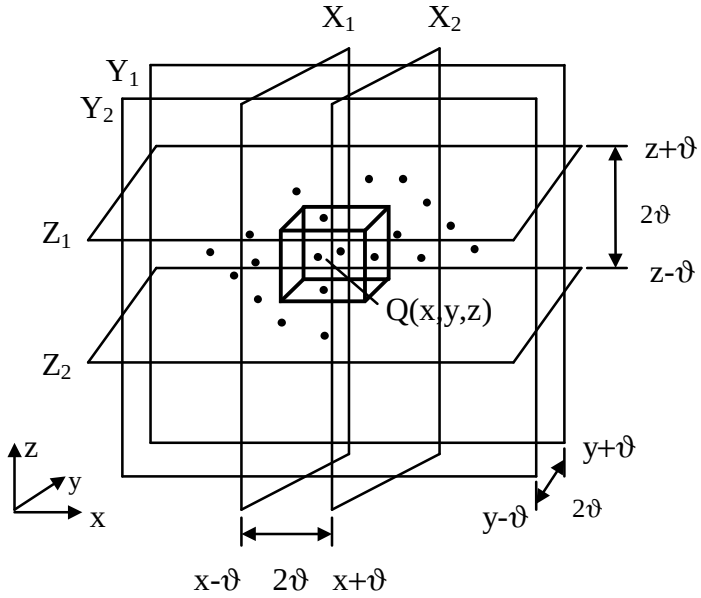


Figure 8. The space partitioning algorithm (based on [18])

In order to simplify the representation in figure 8, the space is partitioned only for the first three co-ordinates *(x,y,z)* while the recognition algorithm uses a multi-dimensional space (up to 36). The candidate list structure is organised into a database, the method uses a 1-D binary searches to find the points between a pair of parallel hyperplanes.

In figure 8, a correct matched point (for a class of objects) is inside the cube of size 2ϑ. The first step of the algorithm is to recognise the object projecting the input image on this universal eigenspace. After this, we can estimate precisely the position by projecting the input image on the object eigenspace (the position in space is determined by the point position in the hypercube).

Certainly the universal eigenspace will be difficult to compute since the number of images for a large collection of objects is huge. For this reason we will compute the largest P eigenvalues and the corresponding eigenvectors. But the advantage in recognition terms is very important because we can determine the object from collection in only one step (see also [2], [12]). The second step is to estimate the position of the object. Otherwise we have to search each object's eigenspace one by one and this method is far from efficient.

6. EXPERIMENTS AND RESULTS

Designing an industrial application requires accuracy and speed. In industry 75% of applications are in small to medium batches [3]. Also, Delchambre [5] highlights the fact that 98% of products are made of fewer than 25 parts. A key issue is to integrate the multi-sensorial information in order to obtain a compact and robust implementation. In this present implementation the object recognition task represents the most complex part and therefore our goal was to implement an efficient algorithm that performs close to real time. Thus, if the number of objects and corresponding positions are small (less than 100), the best implementation is to construct only a universal eigenspace. Clearly this is an approximate position estimation for an object set that contains more than 4 objects, but suitable for a range of applications. If the accuracy or the number of objects is large we are required to construct the universal eigenspace and the object eigenspace for each object. An important problem is the dimension of the eigenspace. As we discussed before this number is limited to P which is the number of positions for each object. This number can vary from object to object and is in direct connection with the number of objects. For this present algorithm we have obtained poor results if the dimension of the eigenspace is less than 8. We increase the eigenspace dimension to 36 step by step but the rate of recognition is not affected visibly after 15 when the recognition rate is near to 100%.

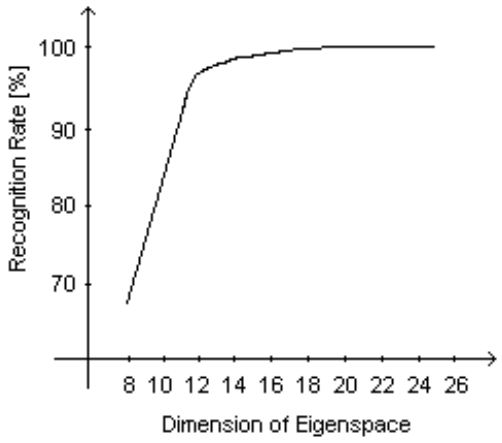


Figure 9. Recognition rate as a function of the dimension of eigenspace

A key issue is maintaining the illumination at a constant level. For 3-D detection we project a structured light pattern to obtain a better representation of shape. Because the bin-picking problem is not tackled in this approach, i.e. the object are not occluded, the 2-D recognition is used. The 3-D information is used only to determine the distance from the object to the robot's gripper. Object reflectance and occlusion can cause errors in the recognition process (more details concerning object reflectance can be found in [16]). If the level of light is not constant for each position, then we have to acquire the same image with different degree of illumination. It is clear that in this case the number of images is huge and to solve this problem is computationally intensive.

In our implementation we used a set of 5 objects that are presented in figure 10. All objects contained in the object set have almost the same colour and reflectance properties but different shapes.

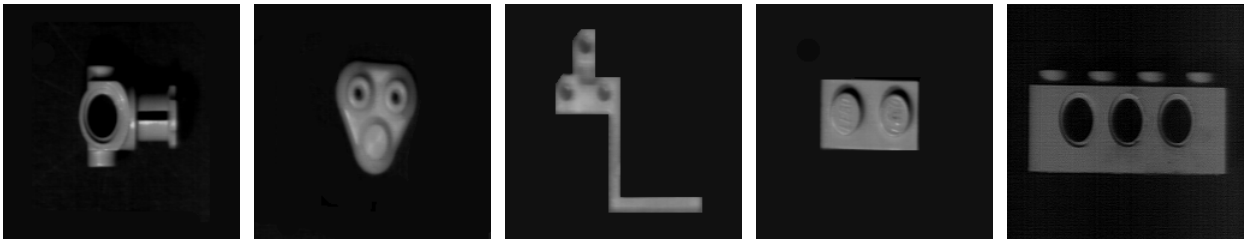


Figure 10. Object set. All images are background normalised

As a result we have obtained correct recognition for objects from an image set (universal image set) and position estimation with an error of less than 5 degrees for an object set which contains 36 images. If the image set has a small number of positions (less than 10), the results concerning pose estimation are unreliable.

7. CONCLUSIONS

We have presented a practical approach useful for many industrial applications. In comparison with other similar implementations this approach is simple, fast and reliable. It is very important to remember that for this method the rate of recognition is equal to unity and the position estimation is in accordance with the number of images contained in the image set. Therefore, this estimation can be as precise as we want, the single bottleneck is that the number of images is very large and the real time condition is very difficult to be accomplished. Furthermore, this approach can be a possible solution for bin picking and the future research will be focused in this direction. Also, future work will concentrate on recognising objects when the illumination conditions are changing in direction and intensity of the light source.

Our experiments show that the eigenspace technique can be successfully used in robotic applications where the object recognition and position estimation is determined with a high degree of accuracy.

8. ACKNOWLEDGEMENTS

This research was supported by Motorola BV, Dublin, Ireland.

9. REFERENCES

1. P. Besl and R. Jain, "Three-dimensional object recognition", *ACM Computing Surveys*, 17(1), pp. 75-145, 1985
2. T. Bailey and A. K. Jain, "A note on distance-weighted k-nearest neighbour rules", *IEEE Trans. Syst. Man and Cybern.*, vol. 8, no. 4, pp. 311-313, 1978
3. B.G. Batchelor, "Intelligent image processing in Prolog", *Springer-Verlag*, London, 1991
4. J.L. Bentley and J.H. Friedman, "Data structures for range searching", *Computing Surveys*, vol.11, no. 4, pp. 397-409, Dec. 1979
5. A. Delchambre, "Computer-aided assembly planning", *Chapman & Hall*, 1992
6. O. Ghita and P.F. Whelan, "Object recognition using eigenvectors", *Proceedings of SPIE*, vol. 3208, pp. 85-91, Pittsburgh, Oct. 1997
7. T. Henderson, "Efficient 3-D object representation for industrial vision systems", *IEEE Trans. on Pattern Anal. and Machine Intell.*, PAMI 5, no. 6, pp 609-617,1983
8. K. Ikeuchi, "Generating an interpretation tree from a CAD model for 3-D object recognition in bin-picking tasks", *Inter. J. Comp. Vision*, vol.1, no.2, pp. 145-165, 1987
9. Imaging Technology Inc., "Vision*plus* – AT®, OFG hardware reference manual", 1990
10. A. Kak and A. Kosaka, "Multisensor fusion for sensory intelligence in robotics", *Proceedings of Workshop on Foundations of Information/Decision Fusion*, Washington, August 1996
11. D. Kriegman and J. Ponce, "On recognizing and positioning curved 3-D objects from image contours", *IEEE Trans. on Pattern Anal. and Machine Intell.*, PAMI 12, no. 12, pp 1127-1137,1990
12. J. Macleod, A. Luc and D. Titterington, "A re-examination of the distance-weighted k-nearest neighbour classification rule", *IEEE Trans. Syst. Man and Cybern.*, vol. 17, no. 4, pp. 689-696, 1987

13. H. Murase and S. K. Nayar, "Visual learning and recognition of 3-D objects from appearance", *International Journal of Computer Vision*,14, pp. 5-24, 1995
14. B. Moghaddam and A. Pentland, "Face recognition using view-based and modular eigenspaces", *Automatic Systems for the Identification and Inspection of Humans*", SPIE, vol.2277,1994
15. H. Murakami and V. Kumar, "Efficient calculation of primary images from a set of images", *IEEE Trans. on Pattern Anal. and Machine Intell.*, PAMI 4, no. 5, pp. 511-515, 1982
16. S. K. Nayar and R. Bolle, "Reflectance based object recognition", *Inter. J. Comp. Vision* vol. 17, no.3, pp.219-240,1996
17. S. K. Nayar, M. Watanabe and M. Noguchi, "Real-time focus range sensor", *Columbia University, Tech. Rep. CUCS-028-94*, 1994
18. S. A. Nene, S. K. Nayar, H. Murase, "SLAM: A software Library for Appearance Matching", *Proc. of ARPA Image Understanding Workshop*, Monterey, Nov. 1994
19. Newport Catalog, "Machine vision optics guide", *Newport corporation*, no. 500
20. W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, "Numerical recipes in C", *Cambridge University Press*, 1992
21. A. P. Pentland, "A new sense for depth of field", *IEEE Trans. on Pattern Anal. and Machine Intell.*, PAMI-9, no.4, pp. 523-531, 1987
22. L. Sirovich and M. Kirby, "Low-dimensional procedure for the characterization of human faces", *J. Opt. Soc. Amer.*, vol. 4, no.3, pp. 519-524, 1987
23. F. Stein and G. Medioni, "Structural indexing efficient 3-D object recognition", *IEEE Trans. on Pattern Anal. and Machine Intell.*, PAMI 14, no. 2, pp. 125-145,1992
24. C.W. Therrien, "Decision estimation and classification. An introduction to Pattern Recognition and Related Topics", *John Wiley & Sons*, 1989
25. M. Turk and A. Pentland, "Eigenfaces for recognition", *Journal of Cognitive Neuroscience*, vol. 3, pp. 71-86, 1991
26. M. Turk and A. Pentland, "Face recognition using eigenfaces", *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp.586-591, June 1991