Complete Object Inspection Using CAD Models and Robotic Manipulation

Ellis T J, Moukas P, West G A W.

Measurement and Instrumentation Centre
The City University
Northampton Square
London EC1V OHB

ABSTRACT

This paper describes the development of an object inspection system for use in flexible manufacturing systems for the inspection of complex machined components. During operation the system first locates and identifies instantiations of the object in the field of view of a camera system. A robot arm is then used to manipulate the object in the field of view of a sensor in order to check for manufacturing defects e.g. surface finish, dimensions, part integrity. Robot manipulation is necessary to enable all views of the object to be interrogated. Much of the information required to plan these operations is obtained from the CAD model of the object (a CSG representation) e.g. possible instantiations, potential gripping points etc.

1.0 INTRODUCTION

The use of automatic inspection in industrial applications has been restricted to custom systems mainly for inspecting essentially 2-dimensional objects (11). The problem of inspecting 3-D solid objects for quality assurance presents considerable problems if simple static camera systems are used. Obtaining appropriate "ideal" views of particular features on the object would require a multiplicity of cameras in order to derive the views. To overcome this problem we could consider obtaining the appropriate views by one of two ways: either by moving the camera around the object or by moving the object under the camera. Obviously, one advantage of moving the object is that it allows the complete inspection in situations where moving a camera might be restricted. For example, if the object were resting on an opaque surface, then the underside of the object would be hidden from view. In some situations it would not be feasible to move the object (e.g. for very large objects) and hence a moving camera might be more suitable. However many manufactured components are of a size where they could be picked-up and manipulated by a robotic arm in order to perform the inspection task.

A number of advantages can be derived from using a robot manipulator to present the necessary views of the object to an inspection sensor of some kind (1). In particular, the control of the local environment (e.g. lighting) and the selection of "ideal" views. In addition, the object can be positioned by the robot for alternative inspection methods using different sensors e.g. into a coordinate measuring machine for the accurate measurement of manufacturing tolerances or under a camera to assess surface finish and quality.

The overall aim of our research is the development of an inspection workstation capable of recognising and performing pre-determined inspection strategies on a wide variety of manufactured components. This pre-determined strategy would be generated either automatically or interactively from analysis of the 3-d components, the requirements of the inspection, environmental constraints etc. and of course cost. It is envisaged that the inspection workstation would be used within a flexible manufacturing system (FMS) to perform appropriate checks on the successful completion of previous stages of manufacture. In such a system the identification of the objects may be managed by simple tagging, or by keeping track of the object as it moves between separate manufacturing workstations. However, we have adopted a more general solution to this problem and assumed that it is necessary for us to recognise the object before the inspection task can proceed.

Our approach to the recognition and inspection of the object is based mainly on information extracted from the CAD model of the component. From this model, all of our knowledge of the part is derived. This is mainly based on a wire-frame description of the object. From this information we can determine those edges that will be visible from a particular view and those surfaces with which the robot manipulator can grasp the object. An imaged view of the object is then matched against transformed views of the edge model in order to perform the recognition. To simplify this matching task we introduce a simple constraint: we assume that the object is resting on a flat surface. This allows an enormous reduction in the number of possible views of the object that we would otherwise need to consider.

The currently envisaged implementation of the inspection system uses a single TV camera positioned vertically above the object with the object resting on a flat surface. Recognition is accomplished using either binary or grey scale processing or a combination of both depending on the constraints that can be applied to the environment or those present in the environment. Processing methods currently being used are based on boundary analysis and Hough transform processing.

Following successful recognition and location of the part a robot is used to pick up the part and begin the inspection task. This is achieved by manipulating the object with the robot and an additional gripper. This sequence is repeated until all the required views of the object have been inspected.

This paper is mainly concerned with the recognition methods that have been developed. As these rely heavily on information provided by the CAD data base, the way we use this data is outlined including the prediction of possible instantiations of the object in the image and identification of features that are used for recognition. The binary and grey scale processing methods that have been developed are described and results presented.

AVC 1987 doi:10.5244/C.1.15

2.0 UTILIZATION OF THE CAD MODEL

The CAD system we used to design our solid models is BOXER, an implementation of the Leeds University Solid Modeller NONAME (2) by Pafec Ltd. (3). BOXER represents solids as "constructions" or "combinations" of solid "primitives" such as blocks cylinders etc. via regularised set operations, i.e. by employing Constructive Solid Geometry (CSG) (4). A model is built with a series of symbolic declarations obeying a specific syntax. At any stage the model may be displayed as a 2-dimensional projection from any desired viewpoint: BOXER forms the wire-frame description of the model, projects it and removes the hidden lines if desired.

Although BOXER and CSG are convenient tools for designing solids, they are not particularly suited to model-based vision work: generally speaking CSG representations do not give direct access to the boundary of the model (its geometry and topology). BOXER in particular gives a very redundant and loosely structured description of the model. A very concise representation of the topology of the boundary (wire-frame representation) of polyhedra is given by the Doubly Connected Edge List (DCEL) (5). The DCEL is a list of the oriented edges of the polyhedron represented as pairs of vertices. Each entry in the list also contains pointers to the two faces adjacent to the edge as well as pointers to edges incident to the two vertices. The advantage of this representation is that the ordered loop of edges or the ordered set of vertices of a particular face or the ordered set of edges incident to a particular vertex is easily obtained with a single pass along the DCEL by following the appropriate pointers. The geometry of the polyhedron is complete with an additional list of the coordinates of the vertices. The DCEL was originally developed to describe convex polyhedra. However it is suitable to describe any complex polyhedron if the curved surfaces are approximated by flat polygonal patches. Holes in a faces can be also catered for by using more than one edge loop for the face in which the hole is present.

Software has been written that takes the standard output of BOXER and generates the DCEL. The software copes with objects constructed from blocks and cylinders but it cannot cope with surfaces that are curved in more than one direction because BOXER cannot supply information describing these as planar patches. However the techniques developed can be generalised to surfaces that are curved in more than one direction as these can be described by the DCEL although this would be larger than one for planar objects.

The following sections utilise the BOXER output and DCEL representation to extract alternative descriptions of the CAD model, which are to be used later for matching the model with the image data.

2.1 Stable Orientations

A solid lies on a flat surface in one of a (usually finite) set of Stable Orientations (SO). This fact can be exploited to facilitate identification of a solid if the viewing angle is known. Each stable orientation defines a stable "view" of the solid i.e. a 2-d model and is the result of the projection of the 3-d model in a direction given by the SO. Lieberman (6) found the set of stable orientations from solid models by first evaluating the Convex Hull (CH) of its vertices and subsequently projecting the centre of mass onto each of the faces of the CH. A face of the CH and hence the corresponding orientation is stable if the projection lies inside the face. The SO is clearly defined as the unit normal vector of the face of the CH. The set of all SO's therefore defines the set of the 2-d models (resulting from the projection of the 3-d model along the corresponding SO) of all the expected "views" of a particular solid. The energy required to tip over the solid when lying on a stable face is used to rank the set of SO's in order of decreasing stability. Gift-Wrapping (7) is the most commonly used algorithm for the evaluation the CH of a set of points. Our implementation of this algorithm (8) efficiently evaluates the DCEL of the CH of parts designed on BOXER and copes with problems arising from the fact that the CH of typical industrial parts are very seldom simplicial polytopes. Figure 1 shows an object designed using BOXER and figure 2 shows the convex hull of this object. The set of SO's is fed to BOXER to evaluate the corresponding projections i.e. to generate the view of the object for each of the SO's for the object sitting on a flat surface with the observer directly above.

Table (1) shows the stable orientation view vectors for the object of figure 1. Seven of these objects lying in stable positions are shown in the

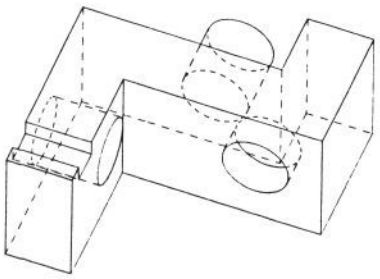

Figure 1. BOXER-designed widget.

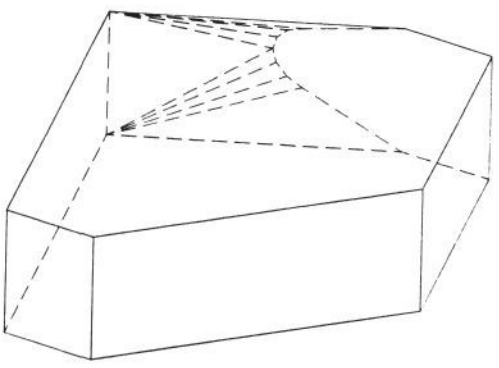

Figure 2. Convex hull of widget in fig. 1.

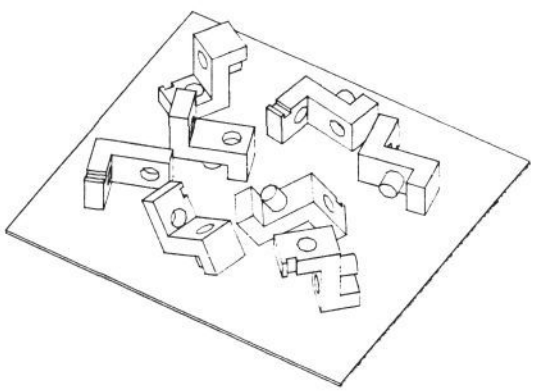

Figure 3. Table-full of widgets in their most likely stable orientations (1-7 from table 1).

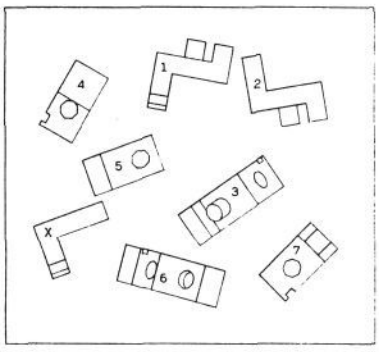

Figure 4. Plan view of widgets in fig. 3 (from camera position).

scene of figure 3 which also includes an additional object. Figure 4 shows the corresponding top view with all the visible edges. The numbers correspond to those of the table (the object marked X is not a stable view of the object of figure 1). The order of the SO's in table (1) is critical as they are ranked in order of stability with the most stable at the top. This information is useful in the recognition stage as the most stable orientation is most probable and hence will be hypothesised first. The last two SO's are impossible and result from the polygonal approximation of the curved edges such that there is a face (very narrow and of small area) upon which the object will sit. An insight into this can be gleamed from examining the convex hull of figure 2. These SO's will, of course, not occur in reality.

View	Stability	X	Y	Z
1	0.1240E+01	0.0000E+00	0.1000E+01	0.0000E+00
2	0.1157E+01	0.0000E+00	-0.1000E+01	0.0000E+00
3	0.6486E+00	-0.5547E+00	0.0000E+00	-0.8321E+00
4	0.3500E+00	0.1000E+01	0.0000E+00	0.0000E+00
5	0.2018E+00	0.0000E+00	0.0000E+00	0.1000E+01
6	0.1745E+00	0.5547E+00	0.0000E+00	0.8321E+00
7	0.1642E+00	-0.1000E+01	0.0000E+00	0.0000E+00
8	0.1043E+00	0.2594E+00	-0.5718E+00	0.7783E+00
9	0.7001E-01	0.2594E+00	0.5718E+00	0.7783E+00
10	0.3024E-01	0.4080E+00	-0.2945E+00	0.8642E+00
11	0.2620E-01	0.4080E+00	0.2945E+00	0.8642E+00

Table 1. View vectors for stable orientations ofthe object.

2.2 Extraction of object descriptions for matching.

The silhouette matching process requires a description of the occluding boundary of each instantiation (stable orientation) of the object consisting of an ordered set of vertices (in an anticlockwise sense) with straight lines between adjacent corners. The grey level processing approach requires a description of all the corners and straight lines that are visible on the object for each stable orientation. Presently, these descriptions are extracted from files produced by BOXER for plotting the CAD output as line drawings. Each one of these contains the end coordinates of each straight line to be plotted (arcs etc. are described by a polygonal approximation). Hidden line removal has been performed by BOXER so that only visible lines are in the plot file.

The first stage of processing consists of removal of multiple lines and the generation of phantom vertices. Multiple lines occur where the edges of a plane are superimposed because the plane is normal to the image plane. Phantom vertices are generated by one plane occluding another part of the object. In the second stage the occluding boundary is found by boundary following around the object from a known

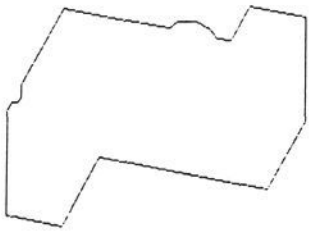

Figure 5 Silhouette boundary extracted from BOXER output (SO 9 in table 1).

vertex on the outer boundary (the first corner found by raster scanning through the plot file). At each vertex, all the lines at that vertex are examined and the one that forms the outer boundary followed. This process continues until the first vertex is again reached. For grey level processing the output of the first stage is in the correct form. Figure 5 shows an example of a silhouette boundary of one of the views of the object shown in figure 1 and referenced as view 9 in table 1.

2.3 Determining object gripping points

To facilitate grasping of the object by the robot gripper, suitable surfaces on the object must be identified. These gripping points are currently selected by matching pairs of opposing faces on the model and calculating the size of the overlapping surface (orthogonal to the face direction). Comparing the size of this overlap with the gripper faces allows us to identify and locate suitable gripping points on the model. After successful matching of object and model, the most appropriate grip point for that view is selected and passed on to the robot. For each valid gripping point, the relative orientation (normal to the paired faces) and centre X,Y coordinates are stored for later user. Although this is a rather simple scheme is has been fairly effective on the planar objects we have considered to date, but would certainly be unsatisfactory for more complex objects, in particular those with significant concavities or without parallel faces.

3.0 OBJECT RECOGNITION

In order to able to pick-up an object for inspection we must first be able locate the object in the image field, determine its orientation and then to recognise it. Recognition will be accomplished in a variety of ways dependent on the nature of the object(s) to be recognised and the environmental constraints. Binary processing will be used where good segmentation of the scene is possible (using backlighting or structured lighting) and each instantiation of the object is distinct. Ambiguities will be dealt with by further processing either using other binary methods or grey scale methods to examine the internal structure of the object. In this case binary processing will reduce the search space to only a few instantiations. If necessary purely grey scale processing will be used in situations where binary processing is not possible e.g. an opaque supporting surface preventing back-lighting.

In general, the binary processing techniques are more efficient than the equivalent grey-scale ones and might be preferred in industrial applications where a major is that operations be performed in "real-time".

3.1 Binary Processing

Where high-contrast images and uniform illumination are available, the first stages of recognition are based on entirely binary methods of image analysis. A number of advantages accrue from employing binary processing. The position and

orientation of the object can be accurately determined; gross features (area, perimeter length etc.) can be easily extracted; efficient boundary analysis methods, which reduce the 2-D image to a 1-D sequence of ordered vectors, may be used; and finally, the operation of these binary-based algorithms is quick and efficient.

Object recognition is achieved by a three-stage process: size features (area and perimeter length) are used to search the model/view database and select a subset of views which approximately match. Corners detected on the object boundary are then matched against the vertices of this subset of model views to further reduce the subset. Finally, internal edges in the object are searched for, based on expected positions of such edges in each model view. If ambiguities remain (i.e. more than one model view in the subset) then these must be resolved by picking up and manipulating the part to identify features not visible to the overhead camera.

A high contrast image of the object is obtained using either backlighting or an appropriate high-contrast background. This silhouette image is then thresholded, based on an automatic analysis of the grey-level histogram in order to provide a well-defined image for locating the object in the image field. Binary objects in the image field are uniquely located and the boundary of each object (if there is more than one) is encoded using a simple 4-way curvature code (the first difference of the Freeman 4-way chain code(12)). In addition, the area and perimeter length of the objects are measured. Since the digitised image has been calibrated (i.e. there is direct correspondence between inter-pixel distances with real world dimensions) we can directly compare these features with those calculated from the model SO's. These features are compared with the pre-computed set of features for each model view in order to identify potential matches. From this operation a subset of possible matches is derived.

The second stage of recognition proceeds by processing the extracted curvature code of the binary object. The curvature code is filtered in order to identify points of high positive and negative curvature (points of inflection on the boundary - corners). The filter is triangular-shaped and is calculated by convolving two rectangular (box-car) filters, which is computationally efficient as it allows a filter (of any length) to be updated with only 4 additions/subtractions. Figure 6a shows a plot of the output of the filtered curvatures (box-car length of 15) from the binary object in figure 6b. From this data corners are detected by thresholding and local peak detection (Table 2). A threshold value of 10 was used for case.

Corner No.	Distance	Size
1	1	18
2	257	11
3	339	17
4	507	-17
5	752	-17
6	915	11
7	997	16
8	1076	-18
9	1116	-12
10	1196	11
11	1270	17
12	1347	-17

Table 2. Corner detection based on filtered chain codes. Distance given in pixels from start.

A simple structural description of the object is then derived which encodes the object silhouette shape as a string of primitives which represent the type (straight line segment, arc segment), length and its relative orientation to the next primitive in the boundary sequence (note: currently only straight line segments are detected). This description is then matched to each view in the model/view subset (which has been similarly encoded) by correlating object and model corners in boundary sequence. Possible errors in the correlation due to invalid or undetected corners in the object are reduced by performing the correlation from a fixed reference point (the first vertex) when comparing successive corners from the image and model. Only those corners which are approximately equal distances (around the perimeter) from this reference are compared, and a count of matched corners and a cumulative error are used to rank possible matches with each model view. This matching also determines the relative orientation between model view and image, enabling the actual orientation of the part to be determined.

The third stage of processing takes each model/view from the subset (in ranked order - highest first) and attempts to detect internal lines (which are explicitly marked in the model) in the object. For this purpose a SOBEL operator is applied to the original grey-level image, and internal edges are detected by searching in the SOBEL image along a line between appropriate pairs of vertex points in the model which mark the endpoints of the edge. Results of the matches of these internal lines are used to update the second stage rankings. The final selection of the best matched model/view will be based on that which achieves the highest-ranking which exceeds a pre-determined threshold. If two or more such views are found which have equal ranking, then this would indicate that the view was non-unique, and the object can only be uniquely identified by selecting a unique view for the object using the robot manipulator.

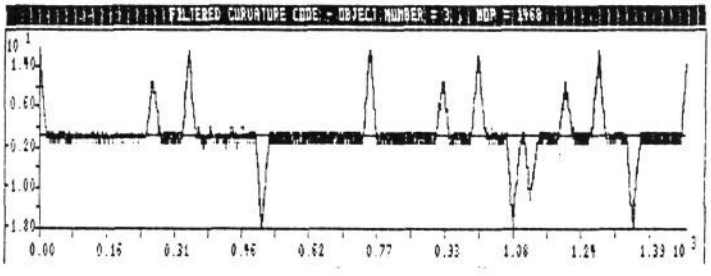

Figure 6. (a) Filtered curvature code for binary object in (b).
(b) Binarised image of widget.

The three-stage hierarchy allows an efficient means of pruning the search of the model database. The ordered sequence of corners derived from the boundary code require only a maximum of (n x m) corner comparisons (n model corners, m image corners). The current implementation does not deal explicitly with curved segments of the boundary, though the matching algorithm at the second stage of processing can skip over the polygonal segments in the model, comparing only corners. Whist this strategy seems to work where corner features predominate, models with significant curves would probably cause the matching process to fail. Work is currently in hand to include arc descriptions in the boundary encoding.

3.2 Grey level processing

The recognition of the object and it's stable orientation is obtained by matching features obtained from the image with those generated from the model. These features consist of corners and straight line segments that are obtained using the following two stages of processing.

In the first stage of processing the features are obtained from the image by firstly detecting edge points using the Sobel edge detector followed by thresholding the magnitude at a low level (to detect the majority of edge points at the expense of noise). A low level description of the object to be identified (line segment description) is generated by using the radius-theta Hough transform (9) on small regions of the image in which there are a significant number of edge segments. The description then consists of a number of straight lines found in each of these regions each described by end coordinates. The use of small regions has the following advantages over using the Hough transform on the whole image. These are (1) a reduction in the size of the accumulator and (2) a reduction in time taken to search that space because there will only be a small number of lines present in each region. In fact the lines are detected by searching for maxima in the accumulator that are a significant distance apart. It also enables the straight line Hough transform to detect components of circular arcs because in the region the arc approximates to a straight line. However this only works for curves of large radius. Figure 7 shows the grey level image for SO 9 and figure 8 shows, overlaid on the image, the regions in which the Hough transform is computed and the lines corresponding to the most significant peaks in the accumulator space. These lines, which extend right across the regions, are matched with the thresholded Sobel image to determine the extent of the lines resulting in the description shown in figure 9a.

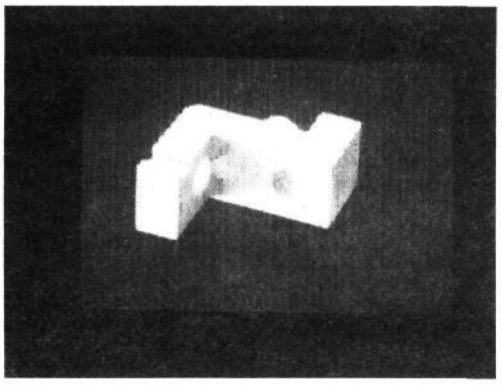

Figure 7. Photograph of the digitised widget in one of its stable orientations (SO 9).
- 256 x 256 pixels, 6 bits/pixel.

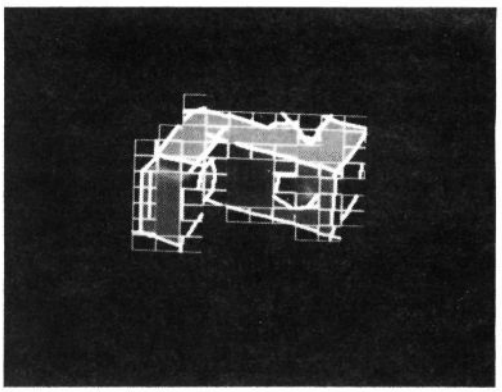

Figure 8. Photograph showing result image cells and lines corresponding to peaks in the accumulator.

The second stage of processing extracts a higher level description of the image by detecting lines and corners in the low level description obtained from the above processing. Initially the line segments are pairwise examined and combined into a longer straight line segment if two conditions are satisfied: (1) the ends are close together and (2) the angle between them is close to zero. Both these conditions use heuristically derived thresholds to give reasonable results on the images so far examined. A number of iterations of the algorithm produces a description of the image consisting of a small number of lines that reasonably describes lines in the image. Finally corners of approximately 90 degrees are detected using similar conditions for linking line segments. These are: (1) the ends are close together and (2) the angle between is approximately 90 degrees. Again heuristically derived thresholds give reasonable results on images. The result of this processing is a description of the image consisting of a list of lines (with the coordinates of the ends) and a list of corners (with the coordinates). The corner list also contains pointers to the two lines forming the corner so that it is an easy matter to move between the corner and the line list. The result of performing this processing on typically images results in an incomplete description of the image with lines and

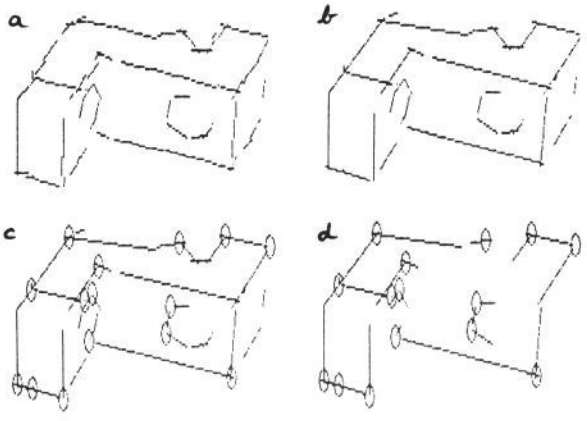

Figure 9. (a) Image lines from Hough transform output.
(b) Result of detecting straight lines.
(c) Corners and all lines.
(d) Corners and corresponding lines.

corners missed because of noise in the image, poor line detection in the Hough transform etc. However the sparse description is adequate for recognition using the method described below. Processing the data of figure 9a results in figure 9b that shows the detected straight lines, 9c the corners and 9d the corners and those straight lines that form the corners.

The description of the views for each of the stable orientations of the model can be obtained using the same processing as for the image. In this case only the corner detection process is needed as the line description is already available with each straight line joining two corners already described as one line. Figure 10a shows the lines for SO 9, 10b the corners and 10c the corners and lines that form the corners. As can be seen this is the model that corresponds to the image data of figure 9.

The matching algorithm works in image space by transforming the model description in terms of translation, rotation and scale to align it with the image. The transformation parameters are determined by initially hypothesising that a particular model feature matches with a particular image feature. The feature considered is a straight line connecting a pair of corners. Then the degree of match is determined by counting the number of one-to-one correspondences between the positions of image and model corners and lines. A match occurs if a model feature is closer to an image feature than a pre-defined distance. The best match is then simply that model and transformation with the largest percentage of model features matched. A reduction in processing time is obtained by only transforming the model data shown in figure 10c i.e. the corners and those lines that form the corners.

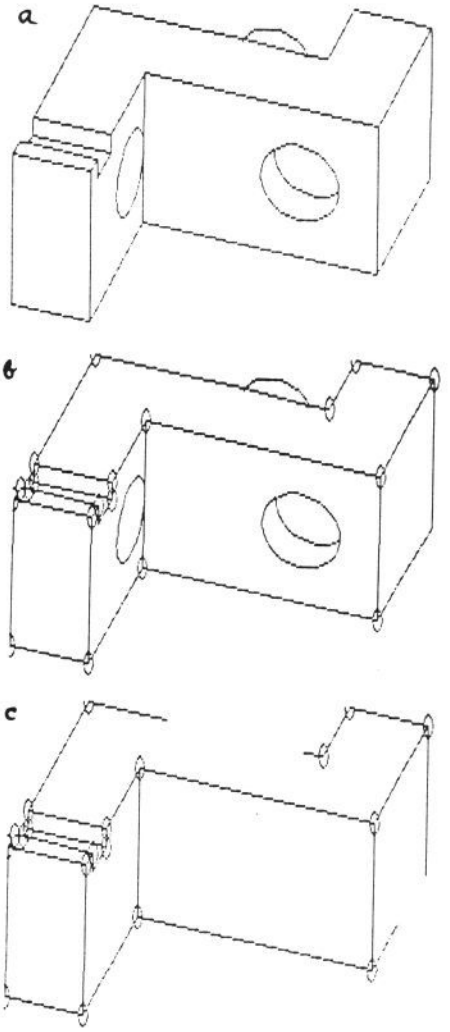

Figure 10. (a) Model lines for SO 9.
(b) Corners and all lines.
(c) Corners and corresponding lines.

The above strategy has been shown to work on various objects that can be described mainly by straight lines and corners. Figure 11 shows two examples of the results of matching for two SO's of one object. The circles at two of the corners indicate the pair of corners and connecting line of the model matched with those of the image. As can be seen there is a reasonable match with inaccuracies caused by the fact that errors in the initial match of one part of the model to the are magnified across the image. The correct stable orientation is determined for each of the possible images. Work is currently in hand to improve the matching by considering curved segments and other features e.g 'Y' junctions and other corners, that can be obtained from the image. The matching is an exhaustive search algorithm that computes a match for each model line matched to each image line in turn i.e. of order M*I where M is the number of model lines and I is the number of image lines. The search space can be significantly reduced by only allowing transformations that produce a feasible scaling of the model. This assumes that the size of the object is known in advance.

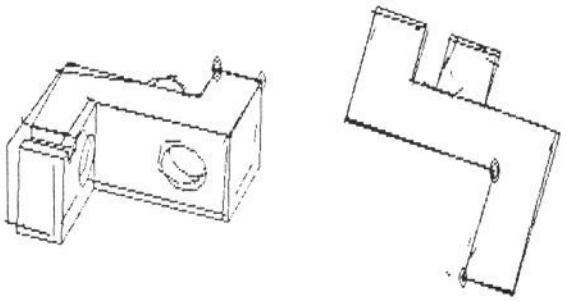

Figure 11. Results of matching, showing model scaled and transformed onto image lines for two examples.

4.0 ROBOT MANIPULATION

The task for the robot arm is to present a sequence of views of the object to the inspection sensor. The choice of gripper and arm is very problem dependent, based on the type of objects that must be grasped and the associated set of movements that the arm must perform. In our case we are using an enhanced SCARA robot manufactured by the UK company UMI Ltd. The arm has 6 degrees of freedom (up/down, shoulder, elbow, wrist pitch, wrist yaw and wrist roll) and simple pincer gripper.

Due to the primitive manipulator of the robot arm, it is necessary to select the required sequence of inspection views by releasing and re-grasping it from a different position. It is possible to rotate the object through several axes (about the wrist), release onto a flat surface and then pick it up again at a new gripping point, thus reorientating the object in the gripper. However, this does require that the object is placed on the surface in one of its stable positions, otherwise it will fall into one of these positions and its exact position may no longer be known. In addition, placing the object onto a surface restricts the range of approaches that the arm can make in grasping the object, and hence to achieve a particular view transformation may take several such moves.

A better alternative is to allow the arm to "pass" the object to another gripper in free space - this could be another arm, or more simply (and cheaply) just a gripper. This arrangement allows much greater access of the arm to the object and also means that the objects' position and orientation (relative to the original grasping point) does not change until the object is re-gripped.

The task of planning the sequence of movements in order to achieve the required sequence of inspection views is complex. Currently we are investigating the planning of the robot movements from within a Prolog environment.

5.0 SUMMARY AND FURTHER WORK

This paper has been concerned with recognition algorithms developed for use in an inspection strategy for inspecting complex 3-d manufactured components. These rely heavily on information obtained from a CAD data base and modelling system used to provide stable views of the components etc. Binary and grey scale based recognition methods have been developed that can recognise the various orientations of the objects. Binary methods have been developed that enable the object to be described structurally as a sequence of line primitives. By matching these to those obtained from the CAD model, the stable position and location of the object is ascertained and valid grasping points determined. Grey scale processing has enabled a primitive structural description of the object to be obtained that is then matched with the model via corners and line segments. This is performed in image space by transforming the model onto the image.

Future work is aimed at improving the recognition methods in terms of performance and speed as well as integrating the methods into a single system. These will need to work in a wide range of environments and be robust and general purpose. The automatic generation of the robot strategy remains essentially unsolved and will require major effort. The inspection tasks to be performed are problem dependent and we are currently investigating the application of previously developed techniques (10) for this. These do not need to be vision based and in some cases e.g. dimensional tolerancing, tactile sensors are necessary to obtain the required accuracy.

ACKNOWLEDGEMENTS

THe authors would like to acknowledge the support of the SERC, who are funding the project as part of a grant concerned with application of image processing to industrial inspection and intelligent instrumentation.

REFERENCES

1. Moukas P, "Computer Integrated Visual Inspection". Report MIC/PM1, Measurement and Instrumentation Centre, The City University, September 1987.
2. "Description of NONAME". Report GMUM:1 Geometric Modelling Project, University of Leeds, December 1983.
3. Lee Y T, "BOXER User Manual". Pafec Ltd, May 1984.
4. Requicha A G A, "Representations of Rigid Solid Objects". In: Encarnacao, J (Ed.): Computer Aided Design Modelling, Systems Engineering, CAD-Systems. Springer-Verlag 1980 pp. 2-78.
5 Preparata F P & Shamos M I, "Computational Geometry: An Introduction". Springer-Verlag 1985.
6. Lieberman L I, "Model Driven Vision for Industrial Automation". IBM Int. Symposium on Advances in Digital Image Processing, Bad Neuenahr, Germany, Sept 1978.
7. Chand D R & Kapur S S, "An Algorithm for Convex Polytopes". JACM vol. 17(1), pp 78-86, Jan. 1970.
8. Moukas P, "A Gift-Wrapping algorithm for Non-Simplicial Polytopes using DCEL's". Report MIC/PM2, Measurement and Instrumentation Centre, The City University, September 1987.
9. Duda R O & Hart P E, "Use of the Hough transform to detect lines and curves in pictures". Commun. ACM, Vol 15, pp 11-15, 1975.
10. Ellis T J, Hill W J & Finkelstein L, "Advances in surface inspection using on-line image processing". ACTA IMEKO 1982, Technological and methodical advancesin measurement, IMEKO 9th World Congress-WestBerlin 1982,North Holland PublishingCompany, Amsterdam 1983.
11. Brook R A & West G A W, "Industrial Applications of Image Processing and Recognition". Proc. Interkarma-congress 1986, Dusseldorf,West Germany, pp147-165, ISBN 0-387-17033-2.
12. Freeman H. 'On the encoding of arbitrary geometric configurations', IRE Trans. Electronic Computers, June,1961.