Robotica (1990) volume 8, pp 47–60

A system for robot manipulation of electrical wires using vision

David Vernon

Department of Computer Science, Trinity College, Dublin, Ireland

(Received in Final Form December 8, 1988)

SUMMARY

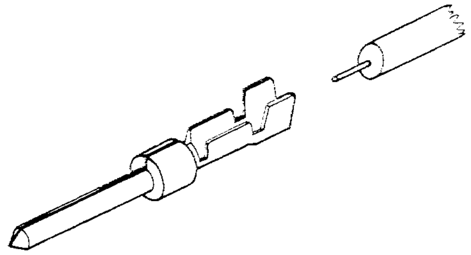
A prototype robot system for automated handling of flexible electrical wires of variable length is described. The handling process involves the selection of a single wire from a tray of many, grasping the wire close to its end with a robot manipulator, and either placing the end in a crimping press or, for tinning applications, dipping the end in a bath of molten solder. This system relies exclusively on the use of vision to identify the position and orientation of the wires prior to their being grasped by the robot end-effector. Two distinct vision algorithms are presented. The first approach utilises binary imaging techniques and involves object segmentation by thresholding followed by thinning and image analysis. An alternative general-purpose approach, based on more robust grey-scale processing techniques, is also described. This approach relies in the analysis of object boundaries generated using a dynamic contour-following algorithm. A simple Robot Control Language (*RCL*) is described which facilitates robot control in a Cartesian frame of reference and object description using frames (homogeneous transformations). The integration of this language with the robot vision system is detailed, and, in particular, a camera model which compensates for both photometric distortion and manipulator inaccuracies is presented. The system has been implemented using conventional computer architectures; average sensing cycle times of two and six seconds have been achieved for the grey-scale and binary vision algorithms, respectively.

KEYWORDS: Robot manipulation; Electrical wires; Automated handling; Robot vision.

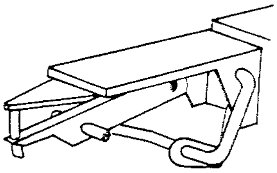
INTRODUCTION

The manufacture of many electrical and electronic sub-assemblies frequently involves the use of insulated electrical wiring. Such wiring varies considerably in both length and gauge and must be properly prepared before being incorporated in the unit being manufactured. This preparation includes cutting the wire to length, stripping the insulation from both ends, and either tinning the ends with solder or attaching crimps (see Figure 1). While high-volume automatic cutters and strippers are in common use, the final tinning or crimping has hitherto involved manual operation of solder baths or crimping presses. This paper describes the development of a robot system which demonstrates the feasibility of automating this process using a five-degree-of freedom manipulator and robot vision. Robot manipulators are increasingly being selected for situations of a batch nature, involving low-volume throughput with frequently changing workpiece characteristics.[1] Conventional first generation robots, which do not incorporate visual sensing, require that the workpieces are all uniformly oriented and uniquely presented to the robot arm.[2] Accomplishing this with wire strips is not a trivial task as the wire will, in general, adopt an arbitrary curvilinear profile, requiring specialised jigs to present the wire correctly to the end-effector. Additionally such machinery would have to be able to adapt to wire-strips of different length and gauge.

The use of robot vision to select and identify the wire strip offers a legitimate alternative solution. Guo et al.[3] have developed a robot vision system which identifies the three-dimensional geometry of electrical wires but their technique assumes the presence of clearly-defined shadows and published results have demonstrated the technique for scenes containing a limited number of wires. Presenting a robot with a tray of wires, from which it must select and remove just one, constitutes a variation of the "bin-picking" problem in which a robot is required to pick a three-dimensional industrial part from a bin containing many randomly-oriented occluding parts. It is simplified in as much as one can conceptualise a wire as a one-dimensional object, randomly oriented in three-dimensional space. This implies that the problems caused by occlusion are less serious. The approach described in this paper endeavours to constrain the environment somewhat so that the scene presented to the camera is not truly random. This is achieved by stipulating that the wires be arranged no more than a few layers deep on the tray and by assuming that the wires are "almost flat", i.e., their spatial variation in the third dimension is minimal. A special purpose end-effector has been constructed to simultaneously grip the wire and push it down onto the tray (see Figures 2 and 3). This mechanism for grasping the wire, together with the above constraints, allows the assumption that the wire actually lies in the plane of the tray. This obviates the need to explicitly extract the Z component of the wire position; only the X and Y coordinates of the grasp and end points are required. The system on which the algorithms described in this paper has been implemented and tested is configured as a development system and not as the final industrial target system, to which it is presently being ported. This development system comprises a vidicon camera, a Vicom image processor, a



. 1. Wire-Crimping.



2. Robot end-effector.

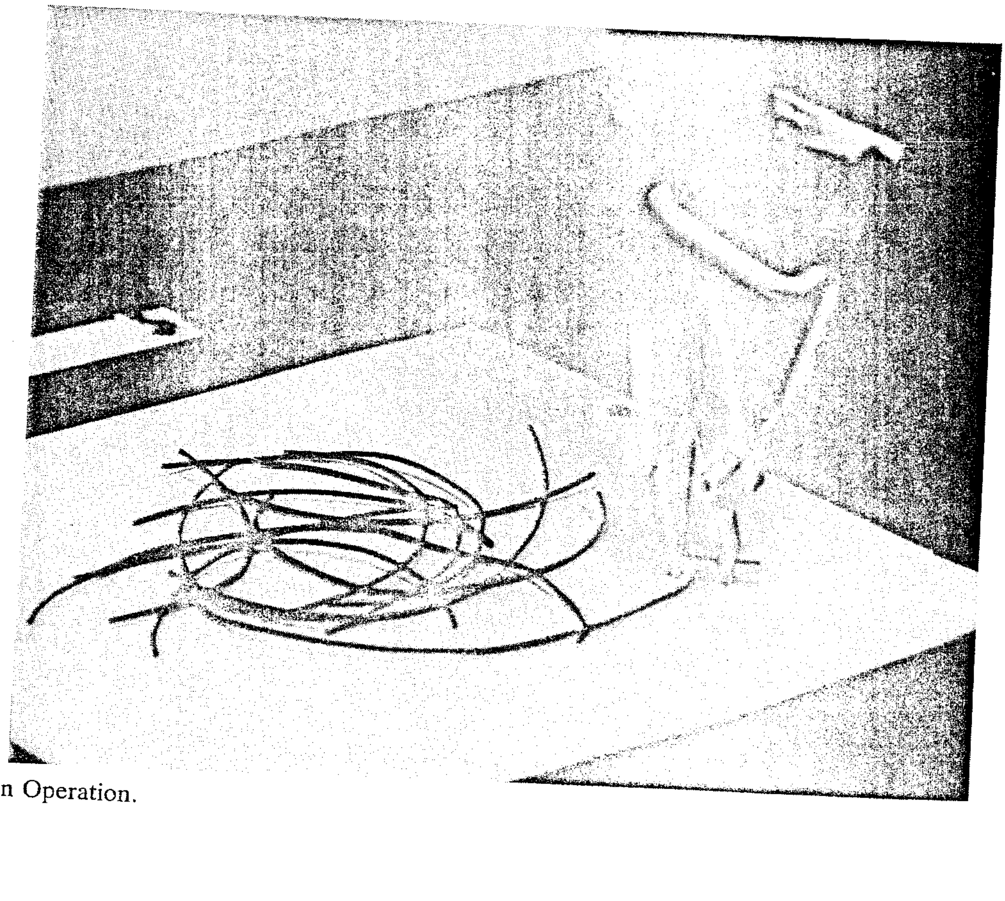
X 11/780, and a SmartArms 6R/600 robot. The m image processor is used solely for image isition, transferring images to the host VAX 11/780 DMA link. The images acquired by the Vicom have olution of 512 × 512 pixels with 128 grey-levels. All essing and analysis software, in particular the Robot rol Language (*RCL*) interpreter with related sensing nes, run on the VAX 11/780. The *RCL* controls the tArms robot via a RS232C serial link, which itself is e-degree-of-freedom D.C. servo-motor anthropohic robot.

VISION ALGORITHMS: A BINARY VISION SYSTEM

The central problem in this application is to identify the position and orienation of both a wire-end and of a suitable grasp point to allow the robot manipulator to pick up the wire and insert it in a crimp. Given the initial assumptions about scene complexity, that the wires are well-scattered and no more than one or two deep, then all the requisite information may be gleaned from the silhouette of the wire. In such circumstances, binary vision techniques are appropriate.[4] A binary image of the wires, in which these are just two levels of grey (black and white), represents a significant reduction of complexity without appreciable reduction in information content. This philosophy of image simplification without significant information loss characterises the overall approach taken in this implementation; the image is reduced to its simplest form before analysis by reduction in resolution, thresholding, and skeletonising. A similar approach has been taken in problems concerned with the analysis of paper pulp fibres[5] and asbestos fibres.[6] It is worth noting that image analysis based on binary images is still prevalent in industrial systems[7] due mainly to the attendant compact representations and simple analysis techniques;[8] even relatively recently published research is based on binary imaging techniques.[7–13]

1. Reduction in resolution and noise removal

The original image acquired with the Vicom image processor is a 512 × 512 pixel image, each pixel representing one of 128 levels of grey (see Figure 4). The first processing stage is a reduction in resolution to



e Robot in Operation.

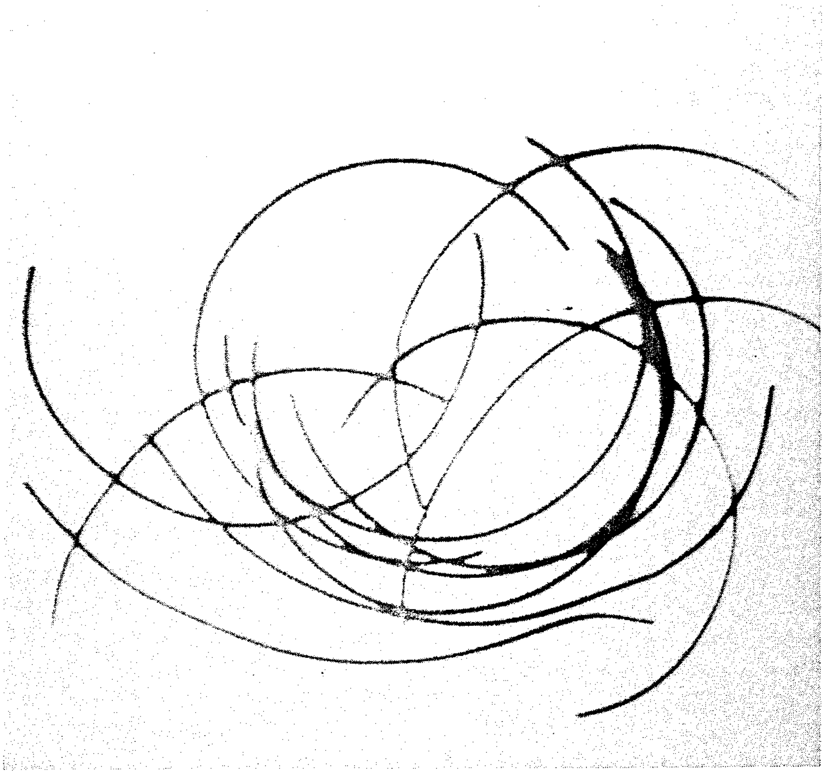


Fig. 4. 512 × 512 Image.

128 × 128 pixels (see Figure 5) resulting on a reduction of the complexity of subsequent operations by a factor of sixteen. This reduction is important as the computational complexity of these operations is significant; indeed the overall execution time for even a 128 × 128 image is quite high. There are essentially two ways in which this reduced resolution image may be generated: by sub-sampling the original image every alternate column and every alternate line or by evaluating the average of pixel values in a 4 × 4 window.[14] Bearing in mind the desirability of removing (or at least attenuating) the noise in the image and recognising that this may be

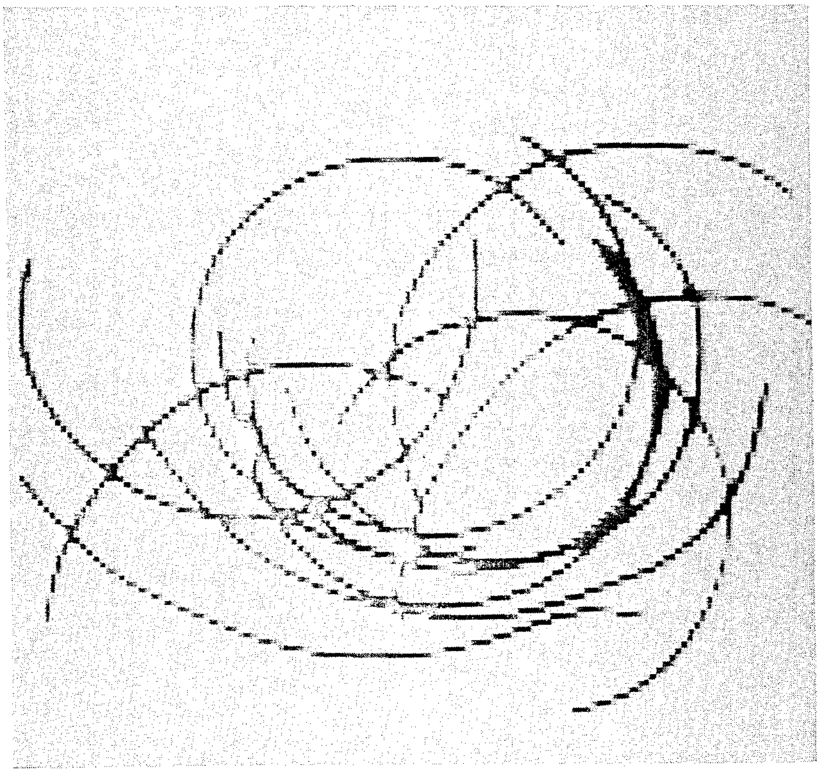


Fig. 5. 128 × 128 Image.

accomplished by local averaging[15] the reduced resolution image in this implementation is generated by evaluating the local average of a 4 × 4 (non-overlapping) region in the 512 × 512 image.

2. Segmenting the image

There are two distinct approaches to the problem of segmenting images and isolating objects: boundary detection and region growing.[16] Boundary-finding approaches accomplish the segmentation by isolating local edges, linking these edges to form short curve segments, and finally generating the object by curve linking, frequently with the use of domain-dependent knowledge.[17] The boundary of the object, once extracted, may easily be used to define the location and shape of the object, effectively completing the isolation. A variation of this approach is used in the second of the algorithms to be described, i.e. the grey-level system. Region-growing, on the other hand, effects the segmentation process by grouping elemental areas (in simple cases, individual image pixels) sharing a common feature into connected two-dimensional areas called regions. Grey-level thresholding is a commonly-used and simple region-based technique. In cases where an object is represented by uniform grey-level and rests against a background of different grey-level, thresholding at an appropriate level will assign a value of 0 to all pixels with a grey-level less than the threshold and a value of 1 to all pixels with a grey-level greater than the threshold. This segments the image into two disjoint regions, one corresponding to the background, and the other to the object. Although three distinct classes of thresholding can be identified,[18] the simplest of these, global thresholding, is utilised here. In this case the threshold test is based exclusively on the grey-level of a test-point, irrespective of its position in the image or of any local context, and upon the global threshold value. The selection of an appropriate threshold is a major problem for reliable segmentation. This topic has received much attention in the literature, and several techniques have been proposed, most of which are based on the analysis of the grey-level histogram: a good survey of thresholding selection techniques may be found in ref. 18. One useful approach to thresholding selection is to use the average grey-level of those pixels having high gradient magnitudes as an estimate of the threshold value.[19,20] Such points normally correspond to edges which are positioned on the boundary between object and background and, as the grey-level of this boundary pixel will typically lie between that of the object and the background, they provide a good indication of the threshold value. The problem lies in deciding what constitutes a "high" gradient magnitude and one is again presented with a (new) threshold selection task. Several variations on this theme have been proposed. For example, the Laplacian operator can be used to identify edges in the image[21] and the grey-level histogram generated using only these edge points is subsequently analysed to yield the threshold point.[22] The technique described here is based on this approach but, instead of using pixels having large gradient values, pixels on the object boundary are explicitly identified using a reliable edge detector which is not dependent on thresholds. Such a detector is derived from the Marr–Hildreth theory of edge detection, a brief summary of which follows. The Marr–Hildreth theory of edge-detection[23] utilises the Laplacian of an image that has been convolved with a two-dimensional Gaussian function. The Laplacian is the sum of the second (unmixed) partial derivatives; the second derivative of a point of high spatial frequency (i.e. a point at which the image grey-level changes very sharply) generates high positive and negative values on either side of the intensity change. Isolating these positive-to-negative, or zero, crossings effectively identifies points of sharp intensity changes, i.e., edges. The Gaussian is used to smooth the image and different standard deviations yield edges detected at different scales within the image. By correlating edges detected at different scales, true or significant edges may be generated. A property of convolution allows the convolution with Gaussian and the evaluation of the Laplacian to be combined as the convolution with the Laplacian of a Gaussian. While Marr's theory requires the correlation of edge segments derived using Gaussians of different standard deviation, empirical studies carried out during this research indicate that the edges detected by one operator alone are sufficiently reliable for this application. The operator implemented here uses a Gaussian with a standard deviation of two.

In summary, the threshold selection procedure first uses a Marr–Hildreth algorithm to locate edges in the image and the mean grey-level of the image pixels at these edge locations is computed. This mean represents the global threshold value. Figure 8 illustrates the result of convolving the Laplacian of a Gaussian with a 128 × 128 resolution image shown in Figure 5. Note that the image has been normalised so that values lie in the range 0–255 and only the absolute value has been used. Thus, white represents large values (either positive or negative) and black represents values of zero. Figure 6 shows the binary image generated by thresholding the original grey-scale image at the automatically determined threshold, and Figure 9 shows the identified zero-crossings.

3. Generating wire skeletons

Once the binary image has been generated, the next step is to model the wires in some simple manner. The skeleton of an object may be thought of as a generalized axis of symmetry of the object[24] and is thus a suitable representation for electrical wires which display obvious axial symmetry. Serra[25] attributes the first formalizations of a skeleton to Motzkin[26] and Blum;[27] indeed the Medial Axis Transform (MAT) proposed by Blum[28] is one of the earliest and most widely studied techniques for generating the skeleton (see a survey in ref. 29). The skeleton is frequently used as a shape descriptor[30] which exhibits three topological properties:[31] connectedness, invariance to scaling and rotation, and information

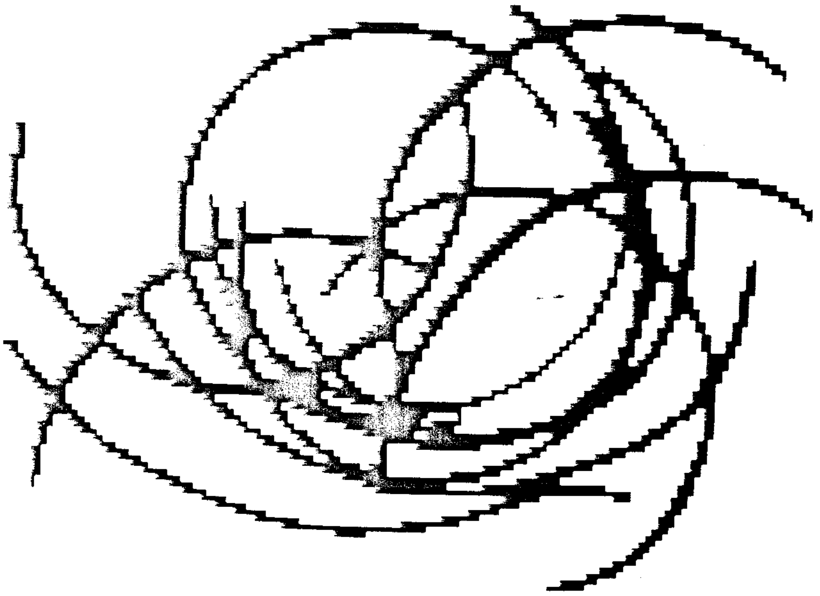


Fig. 6. Binary Image.

preservation in the sense that the object can be reconstructed from the medial axis. The concept of thinning a binary image of an object (Figure 7) is related to such medial axis transformations in that it generates a representation of an approximate axis of symmetry of a shape by successive deletion of pixels from the boundary of the object. In general, such a thinned representation is not formally related to the original object shape and it is not possible to reconstruct the original boundary from the object. Numerous thinning algorithms have been devised and a survey of thinning algorithms based on two-dimensional geometry is given in ref. 32.

If one treats thinning as an operation that removes object pixels from an image according to some constraints then it remains to consider what these constraints must be. The first restriction is that the pixel must be a border pixel. This implies that it has at least one 4-connected neighbouring pixel which is a background pixel. The removal of pixels from all borders simultaneously causes difficulties: for example, an object two pixels thick will vanish if all border pixels are removed simultaneously. A solution to this is to remove pixels of one border orientation only on each pass of the image by the thinning operator. Opposite border orientations are used alternately to ensure that the resultant skeleton is as close to the medial axis as possible.[33]

The second restriction is that the deletion of a pixel should not destroy the objects connectedness, i.e., the number of skeletons after thinning should be same as the number of objects in the image before thinning. This problem depends on the manner in which each pixel in the object is connected to every other pixel. A pixel is said to be connected to, and a component of, an object if it has a grey-scale of zero and at least one adjacent object pixel. Consider now the five pixel object shown in

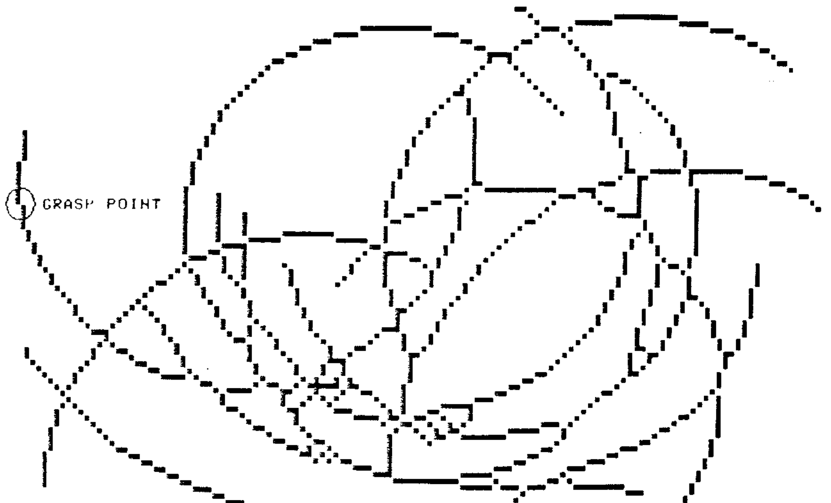


Fig. 7. Thinned Image.

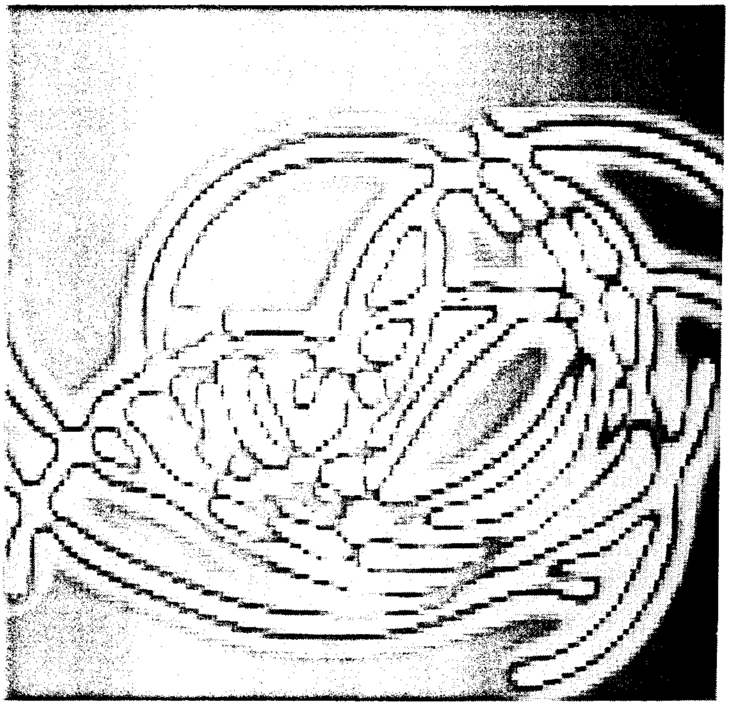


Fig. 8. Laplacian of Gaussian.

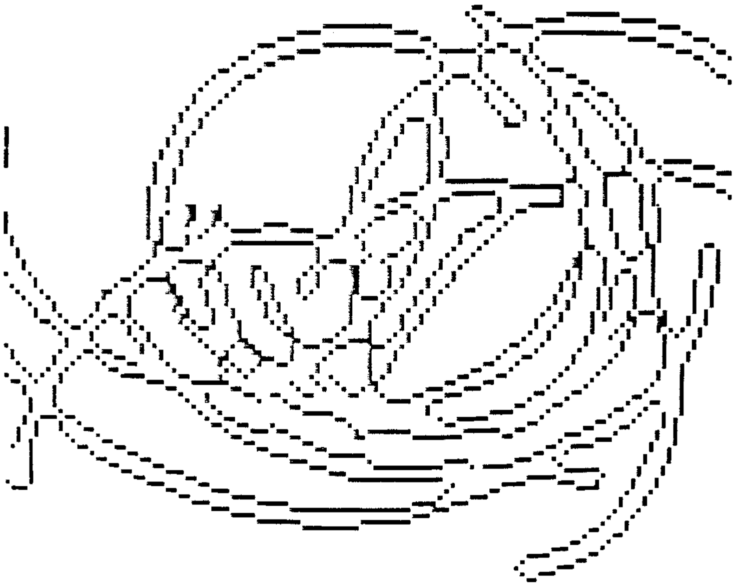


Fig. 9. Zero-Crossings.

Figure 10. The pixel C "connects" the two object segments AB and ED, that is, if C were removed then this would break the object in two; this pixel is "critically-connected". Obviously, this property may occur in many more cases than this, and critical-connectivity may be characterised as follows:

Given a pixel, labelled 9 and its 8 adjacent neighbours, labelled 1–8 (see Figure 11), and assume that writing the pixel number (e.g. 8) indicates presence, i.e. it is an object pixel whereas writing it with an over-bar (e.g. $\bar{8}$) indicates absence, i.e. it is a background pixel. Assume, also, normal Boolean logic sign conventions (+ indicates logical OR, and . indicates logical AND). Then pixel

A		E
B	C	D

Fig. 10. Critically-Connected Object.

1	2	3
8	9	4
7	6	5

Fig. 11. 3 × 3 Neighbourhood.

nine is critically connected if the following expression is true.

$$\begin{aligned}9\cdot\{&[(1+2+3)\cdot(5+6+7)\cdot\bar{4}\cdot\bar{8}]\\ &+[(1+8+7)\cdot(3+4+5)\cdot\bar{2}\cdot\bar{6}]\\ &+[1\cdot(3+4+5+6+7)\cdot\bar{2}\cdot\bar{8}]\\ &+[3\cdot(5+6+7+8+1)\cdot\bar{2}\cdot\bar{4}]\\ &+[5\cdot(7+8+1+2+3]\cdot\bar{4}\cdot\bar{6}]\\ &+[7\cdot(1+2+3+4+5)\cdot\bar{6}\cdot\bar{7}]\}\end{aligned}$$

Hence, the second restriction implies that if a pixel is critically connected then it should not be deleted.

A thinning algorithm should preserve an objects length (approximately, at least). To facilitate this, a third restriction must be imposed such that arc-ends, i.e., object pixels which are adjacent to just one other pixel, must not be deleted. Finally, it is noted that a thinned image should be invariant under the thinning operator. Since the pixels of a fully thinned image are either critically-connected or are arc-ends, imposing restrictions 2 and 3 allow this property to be fulfilled. These three restrictions embody the topological and non-topological considerations required of thinning operations: preservation of object connectivity, accuracy of localization position, and thinness of the resultant line, respectively.[32] The final thinning algorithm, then, is to scan the image in a raster fashion, removing of all object pixels according to these three restrictions, vary border from pass to pass. The image is thinned until four successive passes (corresponding to the four border orientations) producing no changes to the image are made, at which stage thinning ceases. Figure 7 illustrates the application of this thinning algorithm to the binary image shown in Figure 6.

4. Analysing the image

There are essentially two features that need to be extracted from the image:

- The position of a point at which the robot end-effector should grasp the wire and the orientation of this point on the wire.
- The position and orientation of the wire-end in relation to the point at which the wire is to be grasped.

The orientations are required because unless the wire is gripped at right angles to the tangent at the grasp point, the wire will rotate in compliance with the finger grasping force. The orientation of the end-point is important when inserting the wire in the crimping-press as the wire is introduced along a path coincident with the tangent to the wire at the end point. Based on the skeleton model of the wires, a wire segment may be defined as a subsection of a wire bounded at each end by either a wire-crossing or by an arc-end (wire segment end). Thus, a wire segment with two valid-end points, at least one of which is an arc-end, and with a length greater than some predefined system tolerance, contains a feasible grasp point. This is a point some suitable fixed distance (15 mm) from the wire end.

Once the positions of both the grasp-point and the end-point are known, the orientation or tangential angles of these these two points are estimated. The tangent to the wire at the grasp-point is assumed to be parallel to the line joining two skeletal points equally displaced by two pixels on either side of the grasp point. The tangent to the wire end is assumed to be parallel to a line joining the end point and a skeletal point three pixels from the end. Both of these tangential angles are estimated using the world coordinates corresponding to the these pixel positions; these world coordinates are obtained using the camera model and inverse perspective transformation described later.

Table I. Average Sensing Process Times

Process	Average time (seconds)
Image Acquisition	0.10
Transfer to VAX 11/780 Host	3.87
Generation of 128 × 128 Image	4.95
Threshold Selection	45.80
Thinning	5.86
Image-Analysis	0.28

A typical selected grasp point is shown in the thinned image (Figure 7) of the original image of a tray of wires (Figure 4). The average time taken to determine a feasible grasp point is 6.5 seconds; the average sensing process time are summarised in Table I. Note that when calculating the average time taken to determine a grasp point, the preliminary automatic threshold selection time is excluded. In addition, the Vicom-Vax transfer time and 512 to 128 resolution conversion times would not be applicable in a target system and, as such, these times are not included either. It is worth noting also that all the preceding results have been obtained without specially designed lighting; the ambient lighting generated by normal overhead strip-lights has proved adequate.

The average sensing cycle time, with the current implementation, is lengthy. It remains to be seen what speeds will be obtained with a dedicated target system although this approach would almost certainly require the implementation of the thinning process in hardware.

VISION ALGORITHMS: A GREY-SCALE VISION ALGORITHM

Once the organisation of the wires becomes more complex than assumed in the previous section, with many layers of wires occluding both themselves and the background, the required information may no longer be extracted with binary imaging techniques. The grey-scale vision system described in this section addresses these issues and facilitates analysis of poor contrast images and faster sensing times than those achieved with the binary system. It is based on a flexible and efficient processing and analysis architecture and has also been extended to incorporate a general-purpose user-trainable vision facility. It is organised as a three-level hierarchy, comprising a peripheral level, an attentive level, and a supervisory level. All shape identification and analysis is based on boundary descriptors built dynamically by the attentive level using edge information generated at the peripheral level. The supervisory level is responsible for overall scheduling of activity, shape description, and shape matching. The use of an area-of-interest operator facilitates efficient image analysis by confining attention to specific high-interest of sub-areas in the image. Thus, the algorithm described here uses three key ideas: dynamic boundary following, planning based on reduced resolution images, and an organisation based on an peripheral, attentive, and supervisory hierarchical architecture.[34-38] The first two ideas facilitate efficient analysis and compensate for the additional computational complexity of grey-scale techniques, while the third is intended to facilitate future research using more sophisticated visual cues. The system is based on 256 × 256 pixel resolution images; the reduced resolution image is generated by local averaging in every 2 × 2 non-overlapping region in the 512 × 512 image captured by the Vicom image processor. The choice of resolution is based on a consideration of the smallest objects that need to be resolved and the minimum resolution required to represent these objects.

1. The peripheral level

The peripheral level is the lowest in the processing hierarchy and corresponds to conventional low-level visual processing, specifically edge detection and the generation of edge and grey-scale information at several resolutions. The Prewitt gradient-based edge operator was chosen as it provides relatively good quality edges with minimal computational overhead, especially in comparison to other small-kernel edge operators. The implementation of this edge detector facilitates the generation of edge-elements at any arbitrary point (allowing dynamic boundary following based exclusively on local processing) and the detector may operate on both 256 × 256 and 64 × 64 resolution images. High resolution edge detection is used for image segmentation and low resolution edge detection is used by an area-of-interest operator.

The ability of any edge detector to segment an image depends on the size of the objects in the image with respect to the spatial resolution of the imaging system. The system must be capable of explicitly representing the features (edges) that define the objects, in this case, electrical wires. When dealing with long cylinder-like objects, the constraining object dimension is the cylinder diameter. At least three pixels are required to unambiguously represent the wire (across the diameter): one for each edge and one for the wire body. Using wires of diameter 1.5 mm imposes a minimum spatial resolution of 2 pixels/mm or a resolution of 256 × 256 for a field of view of 128 × 128 mm. Using a spatial resolution of 1 pixel/mm will tend to smear the object. Edge detection tests at this resolution show that such smearing does not adversely affect the boundary/feature extraction performance if the wire is isolated (i.e. the background is clearly visible) but in regions of high occlusion where there are many wires in close proximity the edge or boundary, quality does degrade significantly. Tests using a spatial resolution of 0.5 pixels/mm indicate that a detectors ability to segment the image reliably is severely impaired in most situations.

2 The attentive level

The attentive level is concerned with guiding the information process on a local level, specifically to build the object boundaries. There are several approaches which may be taken to boundary building; this system

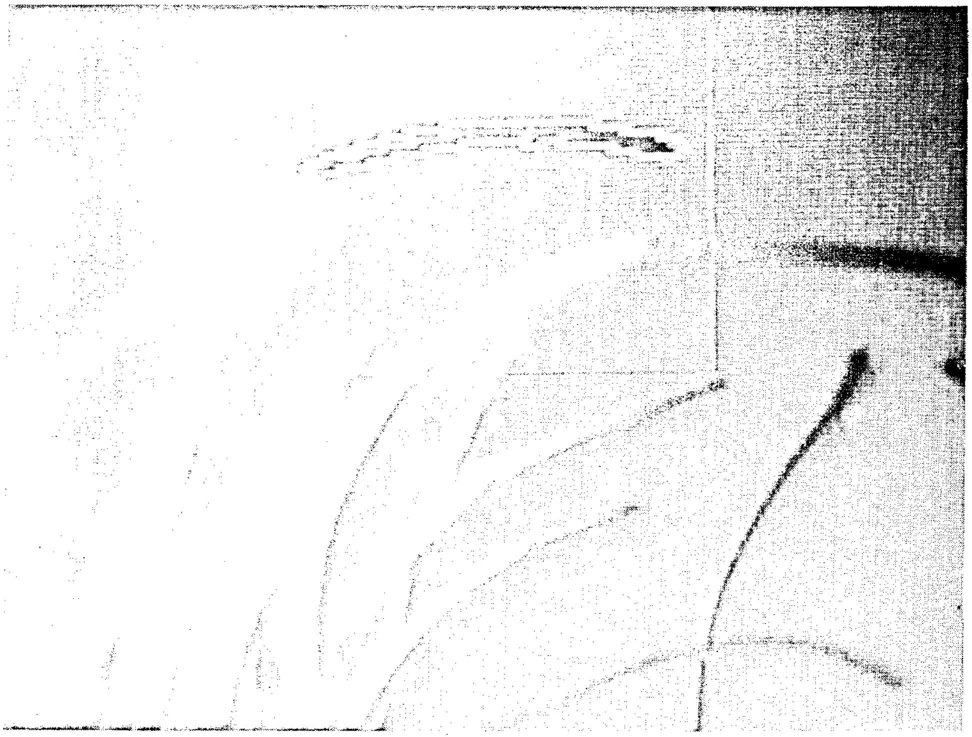


Fig. 12. Boundary Following.

uses a dynamic contour following algorithm which follows the local maximum gradient (derived using the edge detector. i.e. at the peripheral level) and is capable of bridging gaps and linking short edge-segments. The attentive level also interfaces with the supervisory level and passes to it the segmented object represented by a Boundary Chain Code (BCC). Figure 12 illustrates the boundary following process at various points along the wire contours. The disadvantage of the contour following technique is that, because the algorithm operates exclusively on a local basis using no a priori information, the resulting segmentation may not always be reliable and the resulting contour may not correspond to the actual object boundary. In particular, the presence of shading and shadows tend to confuse the algorithm.

While the boundary following algorithm, which effects the objects segmentation, is an attentive level process, it is guided by processes at the supervisory level on two distinct bases. Firstly, the supervisory level defines a subsection of the entire image to which the boundary following process is restricted: this sub-area is effectively a region within the image in which the vision system has high interest. Secondly, the supervisory level supplies the coordinates of a point at which the boundary following procedure should begin. This is typically on the boundary of the object to be segmented.

The boundary following algorithm proceeds on a pixel to pixel basis, tracing the local maximum gradient given by the gradient direction at the point on the boundary. Tracing continues as long as the difference between the current and candidate pixel gradient directions is not too large; this helps avoid following boundaries into noisy areas which are characterised by frequent changes in edge direction. If no acceptable edge is encountered when tracing, a search is made in two zones ahead of the boundary; the first zone separated by a gap of one pixel from the current boundary point, the second by two pixels. If a suitable edge is found, the intervening gap pixels are filled in the trace is restarted from this point. If none are found then the boundary is traced in the reverse direction from the original start point.

As the algorithm traces around the boundary, it builds a Boundary Chain Code (BCC) representation of the contour. A Boundary Chain Code facilitates the description of a contour in an image and comprises an integer pair, denoting the coordinates of an (arbitrary) origin point on the contour. The directional range is quantised and there are just eight possible directions corresponding to the eight pixel neighbours in a 3 × 3 neighbourhood. The boundary following algorithm adheres to the Freeman chain code convention.[39] The complete BCC represents the segmented object boundary and is then passed to the supervisory level for analysis.

To avoid following multiple close parallel boundaries caused by the presence of "thick" edge responses, typical of gradient-based edge detectors, pixels in directions normal to the boundary extracted by the following algorithm are suppressed, i.e. they are labelled, so as to exclude them from later consideration by the boundary following algorithm.

3. The supervisory level

The top level, corresponding to the supervisory phase, is concerned with overall scheduling of activity within the vision system and with the transformation and analysis of the boundaries passed to it by the attentive level.

In guiding the attentive level, its operation is confined to specific areas of high interest and it is supplied, by the supervisory level, with start coordinates for the boundary-following algorithm. An interest operator has been designed which identifies a sequence of sub-areas within the image, ordered in descending levels of interest. This operator is based on the analysis of the edge activity in a reduced resolution image and allows the system to avoid cluttered areas with many (occluding) wires and concentrate on points of low scene population which are more likely to contain isolated and accessible wires. The area of interest is one sixteenth of the size of the original image and is based on a 4 × 4 division of a 64 × 64 pixel resolution image.

The approach taken to the wire-crimping application is to extract a contour, representing the boundary of a group of wires, in a specific area of interest in the image and to analyse this boundary to determine whether or not it contains a boundary segment describing a wire-end. For example, Figure 13 shows a typical contour extracted by the attentive processes, together with a typical boundary segment describing a wire-end. What is required of the supervisory processes is to ascertain which part of the contour, if any, corresponds to the wire-end template and to subsequently determine the position and orientation of both the end of the wire and a suitable grasp-point. An empirical investigation indicated that the use of BCC-based shape descriptors to identify the wire-end are not reliable and, instead, the end is identified by heuristic analysis, formulated as follows.

A boundary segment characterising a wire-end is defined to be a short segment (20 units in length) in which the boundary direction at one end differs by 180 degrees from the direction at the other end, and in which the distance between the end points is less than or equal to 5 units. In addition, the wire-end should be isolated, i.e. there should be no neighbouring wires which might foul the robot end-effector when grasping the wire. This condition is identified by checking that the edge

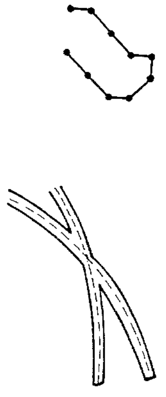


Fig. 13. Representation of Extracted Contour and Required Wire-end Shape

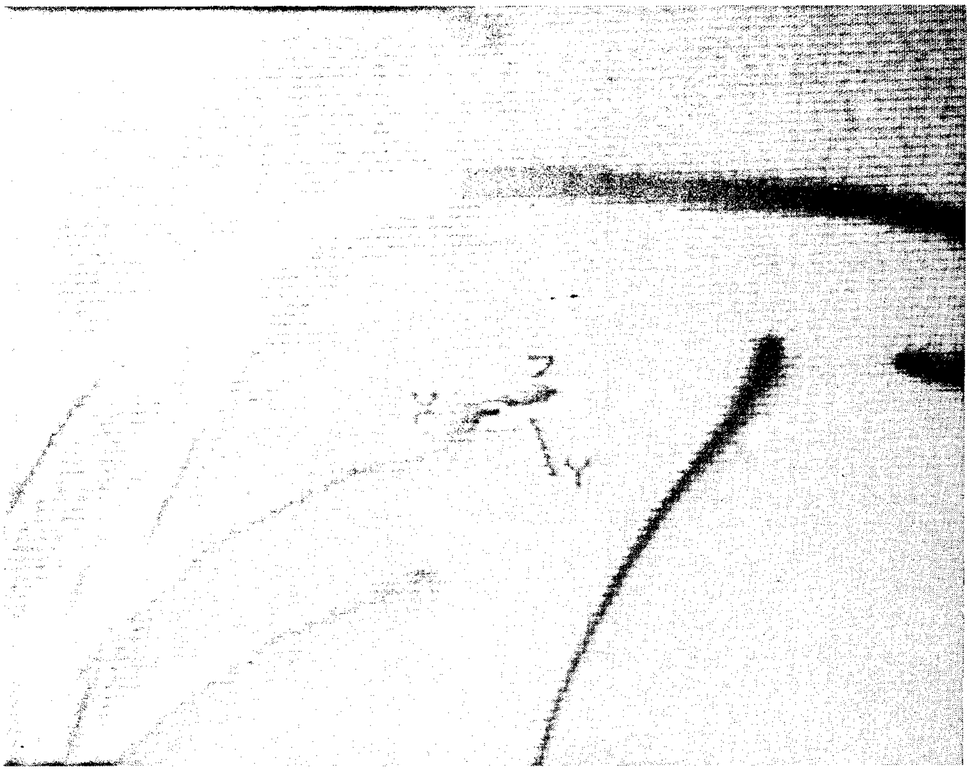


Fig. 14. Identification of wire-end (with attached *XYZ* frame).

magnitude in the low resolution image in a direction normal to the boundary direction is less than the usual threshold used by the edge detection process. Figure 14 illustrates a wire-end extracted from a boundary using this heuristic technique.

ROBOT PROGRAMMING: AN INTERFACE BETWEEN VISION AND MANIPULATION

Once the wire end shape has been identified, it is necessary to describe the wire object in some useful manner, specifically to facilitate manipulation of the object (i.e. the wire) using a robot control language. Homogeneous transformations, first introduced as a data structure for this type of description by Roberts,[40] can be used to describe position and orientation of objects in a manner which is particularly useful for computer vision and robot manipulation.[41] A significant advantage is that if the relative position and orientation between two objects is represented by homogeneous transformations, the operation of matrix multiplication of homogeneous transformations can establish the overall relationship between any two objects.

Two frames, at least, must be associated with the wire to describe it in a manner suitable for robot manipulation. These frames, W and WG, represent the position and orientation of the wire-end, as used in the manipulation task, and the position and orientation of the robot gripper with respect to the wire-end. The frame W is embedded in the wire-end and the frame WG is embedded in the wire a short distance from the end. The actual orientation of the frame axis is defined such that they can be conveniently used by the robot programming task; this specification will be discussed in more detail below and it is sufficient to note here that the origin of the wire frame W is defined to be at the end of the wire, with its *Z* axis aligned with wire's axis of symmetry directed away from the end. The *X* axis of W is defined to be normal to the tray on which the wires lie (and, hence, is normal to image plane) directed vertically upwards. The *Y* axis makes up a right-hand system. The origin of the wire gripper frame WG is defined to be located on the *Z* axis of the W frame, in the negative *Z* direction, and located a short distance from the origin of W. The *Z* axis of W is defined to be normal to the plane of the tray, directed downwards. The *Y* axis is defined to be normal to the axis symmetry of the wire, in plane of the tray. The *X* axis makes up a right hand system. Figure 15 illustrates the definition of these frames.

The position and orientation of an object, represented by homogeneous transformations as discussed above,

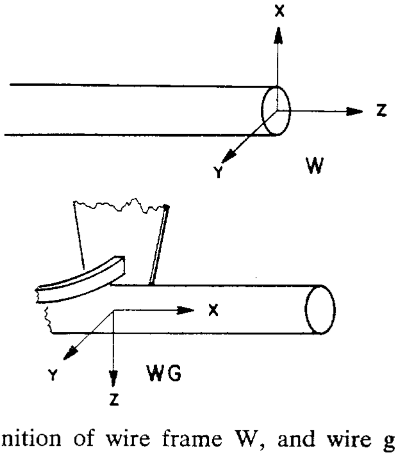


Fig. 15. Definition of wire frame W, and wire gripper frame WG.

may be specified in either the image coordinate reference frame or in the real world reference frame. Since the vision techniques work in the former reference frame and the robot works in the latter the relationship between these two reference frames must be established. The usual approach to this problem is to generate a general transformation from the three-dimensional real-world to the two dimensional image world: such a transformation is called a camera model.[16] The camera model will map a real-world point $[x, y, z, 1]^T$ (in homogeneous coordinates) to an image point $[u, v, t]^T$. The trailing superscript denotes matrix transposition. Hence the form of the camera model C say, is a 3 × 4 matrix with twelve coefficients. These coefficients may be computed by associating the coordinates of six world points with the coordinates of six corresponding image points. This allows twelve equations to be formulated, yielding a solution for the twelve coefficients of the camera model. However, since the camera model is a homogeneous transformation, the scaling factor (element [3,4]) maybe set arbitrarily to 1, leaving eleven unknowns. This implies that the system of twelve equations is over-determined, and a least-square-error solution is normally obtained.

For robotic applications, one is normally more interested in the inverse of this transformation which determines every line in 3-space corresponding to an image point (in 2-space). This is the inverse perspective transformation. Given image coordinates u and v and the z coordinate of a point in 3-space, the inverse perspective transformation allows one to determine the corresponding real-world coordinates x and y. Once the camera model coefficients are known, expressions for such x and y are generated and evaluated.

The aforementioned technique assumes that the image/robot reference frame relationship is linear. However, there is a significant non-linear component in the present TCD camera/robot configuration. This is due to geometric distortion introduced by the imaging system and inaccuracies in the actual robot calibration. Consequently, the above approach to generating a camera model was unsuccessful and, in addition to defining the relationship between image and real-world, a useful solution must also model the spatial or geometric distortion in the camera and the non-linearity of a supposedly rectilinear Cartesian reference frame of the robot. To facilitate to such a solution, the requisite transformation is restricted to be a plane-to-plane non-linear mapping, normally referred to as a spatial warping function. This implies that for any given plane in the real-world one can generate a transformation between image coordinates (u, v) and robot coordinates (x, y), assuming z is constant and known. This relationship is expressed by the following equation

$$(x, y) = (W_x(u, v,), W_y(u, v))$$

Thus, given any image point (u, v) the corresponding world/robot x and y coordinates maybe generated using the warping functions W_x and W_y, respectively. Since analytic expressions for W_x and W_y will rarely be known, a common approach is to model each spatial warping function by an n^{th} order polynomial.[15] Polynomials (in two variables) of the form suggested in [42] have been used for this application. Thus

$$\begin{aligned} W_x(u, v) = a_0 + a_1u + a_2v + a_3u^2 + a_4uv + a_5v^2 + a_6u^3 \\ + a_7u^2v + a_8uv^2 + a_9v^3 + a_{10}u^3v + a_{11}u^2v^2 + a_{12}u^3v \\ + a_{13}u^3v^2 + a_{14}u^2v^3 + a_{15}u^3v^3 \end{aligned} \tag{1}$$

$$\begin{aligned} W_y(u, v) = b_0 + b_1u + b_2v + b_3u^2 + b_4uv + b_5v^2 + b_6u^3 \\ + b_7u^2v + b_8uv^2 + b_9v^3 + b_{10}uv + b_{11}u^2v^2 + b_{12}u^3v \\ + b_{13}u^3v^2 + b_{14}u^2v^3 + b_{15}u^3v^3 \end{aligned} \tag{2}$$

The problem is now to derive the sixteen coefficients of each polynomial in u and v. The solution is facilitated by solving a set of simultaneous equations, of the form of (1) and (2), derived by associating sixteen control points in the real-world with their corresponding sixteen image points. The values of the coordinates of these image and real-world points are determined empirically. In this implementation, we have over-determined the system by using thirty-six points (to better model the entire space) and a least-square-error estimate of the polynomial coefficients is computed using the pseudo-inverse method.

To model the robot non-linearity, the robot has been programmed to identify the thirty-six points itself; the program attempts to map out thirty-six points as a six by six square grid. An image is generated of the resulting (non-square) grid and displayed on the Vicom image processor monitor. The corresponding image point coordinates are then determined by interactively using a cursor. Once the polynomial co-efficients have been computed they are saved on file for later use by the robot vision suite of programs and, in particular, by the supervisory level of the grey-level vision system.

RCL: ROBOT MANIPULATOR TASK SPECIFICATION USING FRAMES

As mentioned above, the most common representation for the description of an object's position and orientation in robotics and graphics is the homogeneous transformation.[43] The use of homogeneous transformations, i.e. coordinate frames, has two drawbacks however. Firstly, the frame does not, in general, specify a robot configuration uniquely. For example, with a six degree of freedom robot, there are usually on the order of eight robot configurations which can place the gripper at a specified frame.[43] Secondly, coordinate frames may over-specify the configuration. Despite these drawbacks, it is held that frames are likely to continue to be the primary representation of positions in robot programs and, hence, a robot programming system should support the representation of coordinate frames and computations on frames using transforms. Furthermore, transforms should be broken into translations and rotations to make them as easy to use as possible.[44] Since robot manipulation is concerned with the relationship between objects and manipulators, and since coordinate frames

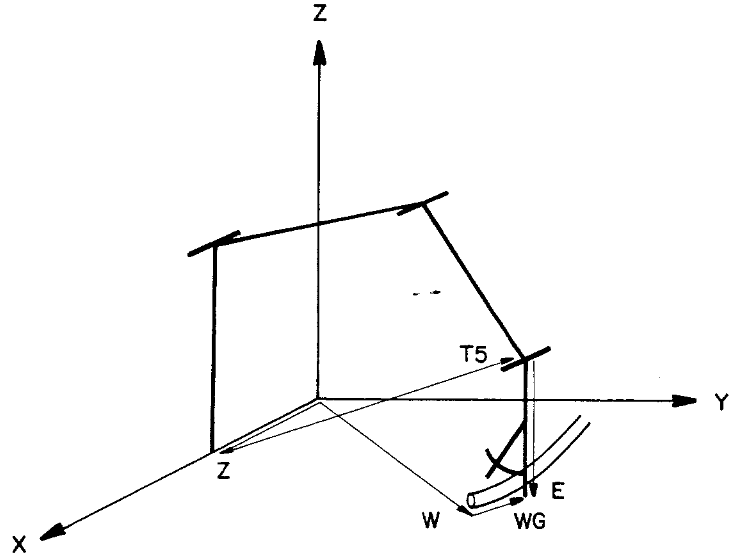


Fig. 16. Robot manipulator grasping a wire.

can conveniently represent such relationships, the homogeneous transformation can be used not only for the description of an isolated object but also for the description of the manipulation task itself. Paul describes an elegant approach to structural task description in terms of homogeneous coordinate transforms.[40] This approach forms the basis of *RCL*.

The actual structure of the task is described by considering the structure of the task's component objects and, in particular, the explicit positional relationships between these objects. Since coordinate frames are to be used to describe object position and orientation, and since it may be required to describe a coordinate frame in two or more ways, a mechanism for representing and manipulating these descriptions is required. Paul suggests the use of transform equations and transform graphs for this purpose.[41]

An example taken from the wire manipulation task will serve to illustrate these techniques. Consider the situation, depicted in Figure 16, of a manipulator grasping a wire. The coordinate frames which describe this situation are as follows

Z is the transform which describes the position of the manipulator with respect to the base coordinate reference frame.

${}^{Z}T5$ describes the end of the manipulator with respect to the base of the manipulator.

${}^{T5}E$ describes the end-effector with respect to the end of the manipulator, i.e., with respect $T5$.

W describes the position of the wire-end, defined with respect to the base coordinates system.

${}^{W}WG$ describes the position of the end-effector holding the wire, defined with respect to the wire-end.

The leading superscript on a frame identifies the coordinate system that the frame is defined with respect to. Observing that in the above example the end-effector is described in two ways, one may generate two equivalent descriptions of the end-effector position by combining these frames. Thus, the end-effector is described by both

$$Z\,{}^{Z}T5\,{}^{T5}E \quad \text{and by} \quad W\,{}^{W}WG.$$

Equating these descriptions, one obtains the following transform equation:

$$Z\,{}^{Z}T5\,{}^{T5}E = W\,{}^{W}WG.$$

Solving for $T5$:

$${}^{Z}T5 = Z^{-1}W\,{}^{W}WG\,{}^{T5}E^{-1}$$

$T5$ is a function of the joint variables of the manipulator and if it is known, then the appropriate joint variables may be computed (using the inverse kinematic solution of the manipulator).

In general, the task movements M_n, say, will be represented in terms of $Z\,{}^{Z}T5\,{}^{T5}E$ and this transform definition can then be equated to other transforms (representing the tasks component objects) which describe the task structure. Ultimately, each transform equation may then be solved in terms of $T5$, which is a computable function of the joint variables, and thus $T5$ is used to determine effective manipulator action. *RCL* is, in essence, a robot programming language to facilitate the direct interpretation of these transform equations using normal structured programming constructs. Thus, a move to the wire grasp position defined above would be written in *RCL* as follows:

```
^T5: = INV(^Z)*^W*^WG*(INV(^E)
MOVE(^T5)
```

The ^ suffix on the variables name is a convention intended to explicitly distinguish frame variables from other variables. Since the position and orientation of the wire is ascertained by visual means, the two frames ^*W* and ^*WG* are returned by an *RCL* vision primitive.

CONCLUSIONS

The robot system described in this paper demonstrates the feasibility of automatically manipulating flexible electrical wires using vision. Both binary and grey-scale image analysis were employed and the binary system proved adequate for scenes exhibiting high contrast. An average sensing time of 6.5 seconds was achieved using this technique; most of this time was taken up by a thinning algorithm used in generating wire skeletons. An industrial production system would necessitate that the thinning be effected by hardware. The grey-scale system represents an attempt to both increase sensing cycle times through the use of locally-confined processing and analysis and to enable the system to deal with low contrast scenes. This approach yielded average sensing cycle times of 2.0 seconds. It is of interest to note that the use of grey-scale techniques has also facilitated the development of general-purpose 2-D object recognition and manipulation. The standard linear approach to modelling the image-to real-world transformation proved inadequate due to non-linearities in the imaging system and in the robot calibration: non-linear spatial warping functions were successfully used instead. Homogeneous transformations were used throughout to describe the (relative) positions of objects in the manipulation task and the task structure itself was specified by considering the relationships between objects using task transform equations. Manipulator control is effected using a simple robot control language which directly interprets these transform equations. The vision subsystem and the language interpreter communicate using homogeneous transformations defining the objects position and orientation. While the feasibility of the manipulation task has certainly been established and manipulation cycle times are adequate, a significant speedup could be achieved through the use of, for example, edge detection hardware.

References

1. M. Mujtaba, *Motion Sequencing of Manipulators* (Report No. STAN-CS-82-917, Stanford University, 1982).
2. G.L. Simons, *Robots in Industry* (NCC Publications, Manchester, England, 1980).
3. H. Guo, M. Yachida, and S. Tsuji, "Three-dimensional Measurement of Many Line-like Objects" *Advanced Robotics* **1,** No. 2, 117–130 (1986).
4. G. Agin, "Computer Vision Systems for Industrial Inspection and Assembly" *Computer* **13,** No. 5, 11–20 (1980).
5. T. Kasvand, "Experiments on Automatic Extraction of Paper Pulp Fibres" *Proc. 4th International Joint Conference on Pattern Recognition* 958–960 (1978).
6. R.N. Dixon and C.J. Taylor, "Automated Asbestos Fibre Counting In: *Machine Aided Image Analysis* (Institute of Physics, Conference Series No. 44), 178–185 (1978).
7. M.J. Chen and D. Milgram, "A Development System for Machine Vision" *IEEE Computer Society Conference on Pattern Recognition and Image Processing* 512–517 (1982).
8. R. Cunningham, "Segmenting Binary Images" *Robotics Age* **3,** No. 4, 4–19 (1981).
9. R.C. Bolles and R.A. Cain, "Recognizing and Locating Partially Visable Objects: The Local-Feature-Focus Method" *Int. J. Robotics Research* **1,** No. 3, 57–82 (1982).
10. R.C. Bolles and R.A. Cain, "Recognising and Locating Partially Visable Workpieces" *Proc. IEEE Computer Society Conference on Pattern Recognition and Image Processing* 498–503 (1982).
11. P.-E. Danielsson, "An Improved Segmentation and Coding Algorithms for Binary and Non-Binary Images" *IBM J. Research and Development* **26,** No. 6, 698–707 (1982).
12. J. Segen, "Locating Randomly Oriented Objects from Partial View" *Proceedings of SPIE* **449,** 676–684 (1983).
13. T.F. Knoll and R.C. Jain, "Recognising Partially Visible Objects Using Feature Indexed Hypotheses" *IEEE J. Robotics and Automation* **RA-2,** No. 1, 3–13 (1986).
14. S.L. Tanimoto, "Image Data Structures" **In:** *Structured Computer Vision* (Academic Press, New York 1980) pp. 31–55.
15. W.K. Pratt, *Digital Image Processing* (Wiley, New York, 1978).
16. D.H. Ballard and C.M. Brown, *Computer Vision* (Prentice-Hall, New Jersey, 1982).
17. D. Cooper and F. Sung, "Multiple-Window Parallel Adaptive Boundary Finding in Computer Vision" *IEEE Transactions on Pattern Analysis and Machine Intelligence* **PAMI-5,** No. 3, 299–316 (1983).
18. J.S. Weszka, "A Survey of Thresholding Selection Techniques" *Computer Graphics and Image Processing* **7,** 259–265 (1978).
19. Y.H. Katz, "Pattern Recognition of Meteorological Satellite Cloud Photography" *Proc. Third Symposium on Remote Sensing of the Environment,* Institute of Science and Technology, University of Michigan 173–214 (1965).
20. W.A. Barrett, "An Iterative Algorithm for Multiple Threshold Detection" *Proc. IEEE Computer Society Conference on Pattern Recognition and Image Processing,* Dallas, 273–278 (1981).
21. G. Gallus and P.W. Neurath, "Improved Computer Chromosome Analysis Incorporating Preprocessing and Boundary Analysis" *Phys. Med. Biol.* **15,** No. 3, 435–445 (1970).
22. J.S. Weszka and A. Rosenfeld, "A Threshold Selection Technique" *IEEE Transactions on Computers* **CC-23,** No. 12, 1322–1327 (1974).
23. D. Marr and E. Hildreth, "Theory of Edge Detection" *Proceedings of the Royal Society of London* **B207,** 187–217 (1980).
24. G. Levi and U. Montanari, "A Gray-Weighted Skeleton" *Information and Control* **17,** 62–91 (1970).
25. J. Serra, "Images et Morphologie Mathematique" *La Recherche* **14,** No. 144, 723–732 (1983).
26. Th. Motzkin, "Sur Quelques Proprietes Caracteristiques des Ensembles Bornes Non Convexes" *Atti. Acad. Naz. Lincel* **21,** 773–779 (1935).
27. H. Blum, "An Associative Machine for dealing with the Visual Field and some of its related Properties" *Biol. Prot. and Synth. Syst.* **1,** 244–260 (1962).
28. H. Blum, "A Transformation for Extracting New Descriptors of Shape" *Models for the Perception of Speech and Visual Form* (MIT Press Cambridge, MA. 1967) pp. 153–171.
29. J.C. Mott-Smith, "Medial Axis Transformations" **In:** *Picture Processing and Psychopictorics* (Academic Press, New York, 1970) pp. 267–283.
30. T. Pavlidis, "A Review of Algorithms for Shape Analysis" *Computer Graphics and Image Processing* **7,** 243–258 (1978).
31. R. Wall, A. Klinger, and S. Harami, "Algorithm for Computing the Medial Axis Transform and its Inverse" *Proc. of the 1977 Workshop on Picture Data Description and Management* (Proceedings 77CH1187-4C, IEEE Computer Society, Picastaway, New Jersey) 121–122 (1977).
32. H. Tamura, "A Comparison of Line-Thinning Algorithms from Digital Geometry Viewpoint" *Proc. 4th International Joint Conference on Pattern Recognition* 715–719 (1978).
33. A. Rosenfeld and A. Kak, *Digital Picture Processing* (Academic Press, New York, 1982).

34. H.U. Lee and K.S. Fu, "The GLGS Image Representation and its Application to Preliminary Segmentation and Pre-attentive Visual Search" *IEEE Computer Society Conference on Pattern Recognition and Image Processing,* 256–261 (1981).
35. A.R. Hanson and E.M. Riseman, "Segmentation of Natural Scenes" **In**: *Computer Vision Systems* (Academic Press, New York, 1978).
36. W.N. Martin and J.K. Aggarwal, "Survey-Dynamic Scene Analysis" *Computer Graphics and Image Processing* **7,** No. 3, 356–374 (1978).
37. R. Jain and S. Haynes, "Imprecision in Computer Vision" *Computer* **15**(8), 39–48 (1982).
38. L.F. Pau, "Approaches to Industrial Image Processing and their Limitations" *Electronics and Power,* February, 135–140 (1984).
39. H. Freeman, "On the Encoding of Arbitrary Geometric Configurations", *IRE Trans. on Electronic Computers* 260–268 (1961).
40. L.G. Roberts, "Machine Perception of Three-Dimensional Solids" **In**: *Optical and Electro-Optical Information Processing* (MIT Press, Cambridge, Massachusetts 1965), p. 159–197.
41. R. Paul, *Robot Manipulators*: *Mathematics, Programming, and Control* (MIT Press, Cambridge, Massachusetts, 1981).
42. E.L. Hall, *Computer Image Processing and Recognition* (Academic Press, New York, 1979).
43. T. Lozano-Perez, "Robot Programming" *MIT AI Lab, AI Memo* 698 (1982).
44. S. Bonner and K.G. Shin, "A Comparative Study of Robot Languages" *Computer* **15,** No. 12, 82–96 (1982).