A Prototype Intelligent Robot that Assembles Objects from Plan Drawings

MASAKAZU EJIRI, TAKESHI UNO, HARUO YODA, TATSUO GOTO, AND KIYOO TAKEYASU

Abstract—An intelligent robot that recognizes and assembles three-dimensional objects by means of vidicon cameras, an articulated mechanical hand, and a digital computer is described. Its problem-solving functions include three essential parts: the recognition of macro-instructions from a human master, the recognition of the objects to be handled, and the decision making for executing the necessary tasks.

The instruction is in the form of a three-view plan of a simple polyhedral assemblage whose overall spatial configuration is recognized together with its component parts. In this process the set of planes of the assemblage is disassembled into open shells; these are then reconstructed into closed shells to find the parts by solving a linear equation where the shell vectors are taken into consideration.

In the object recognition, all the geometric features of polygonal prisms in the field of vision are extracted to find the specific parts required for the assembly. Further computation is made to search for the assembly procedure and is based on the restraint vectors of each object and consequent disassembly tree. Finally, the mechanical hand starts the autonomous manipulation of the parts so as to accomplish the assigned assembly.

Index Terms—Assembly procedure, computer manipulation, decision making, drawing recognition, intelligent robot, object recognition, orthographic drawing, pictorial instruction, polyhedra, problem solving.

I. Introduction

AIMING AT the realization of a future intelligent robot system, several studies have been made on automatic machines which are capable of recognizing three-dimensional objects or the external world and moving their hands or bodies in response to the results of the recognition to perform certain missions. Emphases of these intelligent machines are placed on the problem-solving techniques that will be necessary for the realization of a visual device to take the place of human eyes and a coordinated mechanical system to simulate movements of human hands.

The first attempt to control a mechanical hand by means of a computer was made by Ernst [1], and Roberts [2] was the first to try to make a machine capable of recognizing three-dimensional objects. His computer program was able to input a photographic image of three-dimensional objects through a facsimile scanner and figure out the three-dimensional configuration of the photographed objects. Nilsson and his associates [3], [4] made a mobile automaton that recognized simple objects such as rectangular and triangular prisms by visual data processing, thereby proceeding to form a model of the external world. McCarthy *et al.* [5] constructed a computer system with eyes, hands, and also ears, thus making it possible to instruct the robot by a spoken language with a comparatively simple syntax. Another leading group in the field of artificial intelligence is that of Minsky, which led to the Guzman program [6] capable of performing complex scene analysis to identify individual objects.

These studies have resulted in posing as many new problems as solutions. One such problem is how to give the robot macroscopic instructions. Most existing machines, such as numerical control machine tools, are operated by microscopic instructions which are programmed in advance with one-to-one correspondence to each step of the machine movements. The instructions for the robot systems mentioned above are much more macroscopic, but as they are usually linguistic, it seems that a sequence of instructions related to the machine behavior in some manner is still necessary when a complex mission such as the assembly of objects is commanded.

The robot now being described operates by pictorial macro-instructions that have no relation to the specific movements of the machine. The instructions are in the form of graphical plans on which the objectives of jobs are illustrated, thus making it possible to adapt the machine to new jobs and giving a versatile system. This is a prototype of an integrated intelligent system that has the problem-solving functions for recognition of the drawing and of the three-dimensional objects, and for the decision making needed to manipulate the objects. Its development represents basic research directed toward building the automated visual cognition and handling techniques for an intelligent production system in future assembly processes. This robot consists of three subsystems: eyes, a hand, and a brain. Looking at the given assembly drawing with one of the eyes, the robot understands the overall spatial configuration of the object shown in the drawing, as well as the shapes, the numbers, and the mode and order of assembly of the component parts that make up the object. With the other eye, the robot looks at the real parts placed on a table and understands their geometric features such as their shapes, orientations, positions, and sizes. Furthermore, it identifies the specific parts required for each step of assembly, and makes a decision on how to manipulate these parts to assemble them into the form specified on the drawing. In accordance with the decisions thus reached, the robot automatically starts to move its hand to do the assembly work. The study described in this paper has made it possible for a computer system to handle the in-

Manuscript received May 25, 1971; revised September 1, 1971. Part of this work was presented at the 2nd International Joint Conference on Artificial Intelligence, London, England, September 1–3, 1971.

The authors are with the Central Research Laboratory, Hitachi, Ltd., Kokubunji, Tokyo, Japan.

formation not only on individual objects, but also, to some extent, on the corelationships between them, although the objects to be dealt with are limited to polyhedra so far. In this paper it is the intention of the authors to give a detailed description of this intelligent robot incorporating the three techniques—the drawing-recognition technique, the object-recognition technique, and the decision-making technique for object handling.

II. The System Configuration of the Intelligent Robot

Fig. 1 shows a block diagram of the hardware of this intelligent robot. The visual system of the robot consists of two vidicon cameras operating at a scanning speed of 60 fields/s and their peripheral equipments. One of the two eyes is used for the recognition of the assembly drawing given by human master and the other is used to see the real world for the recognition of three-dimensional objects.

The effective visual field of the eye is divided into image blocks of 60 (high)×80 (wide), and each block is divided into picture elements of 4×4, so that the entire image area consists of 240×320 picture elements. The optical image is converted into a 5-bit (32-level) digital image by means of a sample-and-hold circuit with a 667-ns sampling period and an A–D converter. One picture element from each block is entered in real time into the control computer, which constitutes the brain system of the robot, through a small-capacity buffer register capable of holding the information of 40 picture elements. In this case, the commands from the computer can determine which part of the field of vision and which picture element from each block are transmitted into the computer, thus permitting the video information within a rectangular opening with optional size being fed in from any desired part of the field of vision at the three modes of fineness: fine, medium, and coarse.

The handling system of the robot is an articulated mechanical hand with seven degrees of freedom of motion in all, including a parallel-jaw type grasping mechanism. Each joint of the hand is coordinated by controlling seven independent servomechanisms simultaneously. The computer sends to the seven registers the data corresponding to the desired displacement of the hand, which in turn are converted into 400 Hz phase signals by means of a pulse distributor and seven digital phase modulators. These signals are then compared with the phase signals of synchroresolver type position detectors, and the error signals are amplified and fed into dc servomotors through thyristors to control the motion of each joint of the hand with continuous path mode.

The eyes and the hand of the robot are linked through data channels with a HITAC 7250 digital computer with a 32-kiloword 16-bit word length core memory and a 512-kiloword magnetic drum. The cycle time is 2 μs, and the maximum data transfer rate is 4 Mbits/s.

All the recognition and decision-making functions of this robot are given as the software of this computer. Except for a few assembly language routines, this application software is written in Fortran and comprises three major programs—

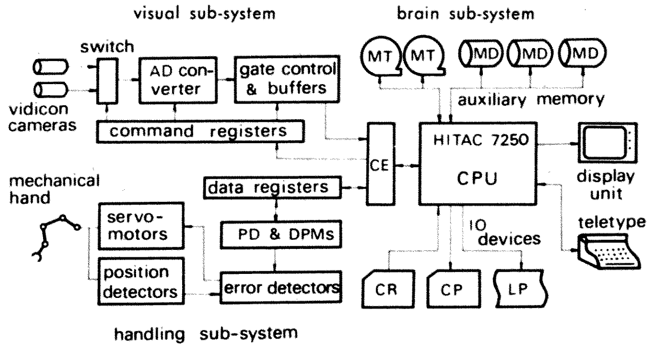


Fig. 1. System configuration.

the drawing recognition program, object recognition program, and decision-making program—and their subprograms.

III. Drawing Recognition Algorithm

The macro-instruction for this intelligent robot is given in the form of an assembly drawing. The drawing is an orthographic projection showing three sides of the required assemblage, and is, at present, composed only of straight solid lines; accordingly, the assemblage must be a polyhedral structure composed only of flat planes. The drawing recognition algorithm converts the drawing into three-dimensional information and distinguishes the individual component parts that make up the assemblage. First, the image of the drawing picked up by the vidicon camera is fed into the magnetic drum through the core memory of the computer; thereafter this digitized information is used to detect all the lines in the top, front, and side views of the drawing, which provides a list of nodes, lines, and loops. The trenching method, which will be discussed in the section on object recognition, is employed in the detection of the straight lines in the drawing.

The drawing recognition algorithm then carries out the conversion of the detected nodes in the two-dimensional image into spatial vertices. Since the three-view drawing is the orthogonal projections of an object on the planes xy, yz, and zx in the xyz Cartesian coordinate system, the x and y coordinates of a spatial vertex must be expressed in top view, the y and z coordinates in front view, and the z and x coordinates in side view. Therefore, the following set of spatial vertices P can be obtained from the coordinates of the ith node in the front view, the jth node in the side view, and the kth node in the top view, examining the corresponding relationships between all the sets of nodes in the respective views.

$$P = \{p \mid p = (i, j, k), \quad \delta(x_j - x_k) \cdot \delta(y_k - y_i) \cdot \delta(z_i - z_j) = 1\}$$

where

$$\delta(a) = \begin{cases} 1, & \text{if } a = 0 \\ 0, & \text{if } a \neq 0. \end{cases} \tag{1}$$

If the given drawing is perfectly described with no ambiguity, that is if the projections of all vertices of the three-dimensional assemblage are sufficiently represented as nodes in each view, the set of spatial points thus obtained only from corresponding relationships will be a perfect one. However, in general the set P contains excess vertices which do not exist in the real assemblage, because (1) is a necessary but not sufficient condition for obtaining the spatial vertices. These kinds of unnecessary information are excluded at the subsequent stage of processing, as will be mentioned later.

However, there are more complex drawings involving some degree of ambiguity in description. For example, there is a case as is seen in Fig. 2, in which nodes 1–5 on the front view have no corresponding nodes on the other views. This is due to the omission of the broken lines for expressing the invisible lines. The assemblage shown in Fig. 2 is composed of a grooved block (an octagonal prism) B_1 and two triangular prisms B_2 and B_3 mounted on top. Node 1 in front view is a false point created by the overlapped view of these two triangular prisms. This can be deduced from the fact that node 1 is T-shaped and has no corresponding nodes in the top view despite of its uppermost position in the front view. Therefore, assuming that the line segment $\overline{6,1}$ is the visible section of $\overline{6,7}$, node 1 is eliminated from the set of nodes in the front view and $\overline{6,7}$ is registered instead of $\overline{6,1}$ in the set of lines in the front view. Regarding the internal nodes with no corresponding nodes in other views, such as nodes 2–5, it is assumed that there are lines, perpendicular to the drawing, that pass through these nodes; thus the edge lines not expressed in other views are taken into consideration. By means of these two processes, the addition and the elimination of the two-dimensional information are performed and the corresponding nodes are located again by (1) to find all spatial vertices. As these two are rather special cases selected among others, they are not sufficient to deduce all kinds of ambiguity. However, these two processes usually give fairly good results for the recognition of the assemblage constituted mainly by polygonal prisms.

Next, the lines are converted into spatial information. Whether or not a line segment exists between any two vertices $p_m=(i, j, k)$ and $p_n=(i', j', k')$ in the set P of spatial vertices can be determined by checking the existence of its projected line segments ii', jj', and kk' in the set of lines in the respective views. Sometimes, a spatial line is projected into a point in one of the views; in this case it is regarded that a projected line segment exists. Thus the set of spatial lines L is obtained and its element l is expressed as the set of two spatial vertices $\{p_m, p_n\}$, which are the endpoints of the line segment.

The spatial planes are then computed. The two-dimensional loops in the drawing are already expressed by the chains of nodes in the respective views. As the assemblage is limited to a group of polyhedra, each loop can be regarded as a projection of a loop of spatial vertices. Therefore the candidates for spatial planes are determined from these loops in the respective views and are investigated to see that all spatial vertices in each candidate lie on the same plane. If this is the case, they are registered as spatial planes. On the other hand, there also is a case in which a spatial loop is projected on a single line segment in some view. Therefore it is possible to obtain from the set of spatial lines, which are projected on a single line in the respective views, spatial loop information as a candidate of a spatial plane. The ultimate set of spatial planes S is the union of the sets obtained by these two methods.

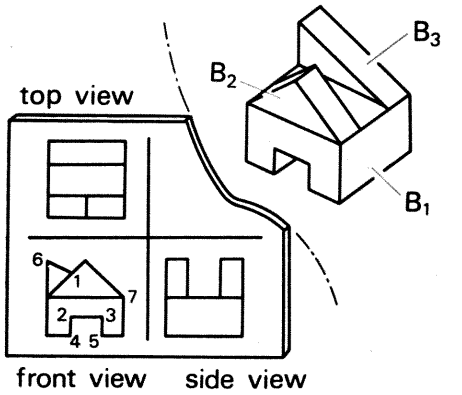


Fig. 2. Example of orthographic projection and its spatial configuration.

Further investigation is made as to whether or not the resulting spatial vertices, spatial lines, and spatial planes give a complete description of three-dimensional assemblage. Since each element s_i in the set S of spatial planes is expressed as the set of spatial lines composing the plane,

$$L = \bigcup s_i. \tag{2}$$

In other words, all the unions of s_i with respect to i give the set of spatial lines L. Now consider the following two sets of lines obtained from all elements in S:

$$L' = \bigcup (s_i \cap s_j), \qquad i \neq j \tag{3}$$

$$L'' = \bigcup (s_i \cap s_j \cap s_k), \qquad i \neq j,\ j \neq k,\ k \neq i. \tag{4}$$

Obviously, L' is the set of edge lines each of which is contained in two or more planes and similarly L'' is the set of lines contained in three or more planes. Also it is evident that $L \supseteq L' \supseteq L''$. Now we consider the set $L-L'$. This is the set of edge lines such that each of the lines belongs to only one plane. Therefore the plane s_i, where $(L-L') \cap s_i \neq \emptyset$, is an isolated plane which does not compose a three-dimensional body. In general, this is the excess information and for this reason it is excluded from the set S of planes. An example of such an isolated plane is given in Fig. 5 which will appear later.

On the other hand, $L'-L''$ becomes the set of edge lines each of which is contained only in two planes. Therefore the two planes s_i and s_j, which have the common l where $l \in (L'-L'')$, always compose the same solid. Let us say that two planes having such l are united together with a tight link. In contrast, when s_i and s_j have in common the edge line contained in L'', they do not necessarily compose the same solid and therefore this case is called the loose link.

Next, the set of planes S, from which all isolated planes have been eliminated, is divided into subsets $S_1, S_2, \cdots, S_n$ by grouping the planes, so that the planes in each subset are united with each other only with tight links. These subsets are usually the open shells whose openings are composed of the edge lines in L''. This process is shown in Fig. 3, where, for instance, S_1 is a shell constituted by planes 1–5, which are united only with tight links, and S_2 is also a shell consisting only of plane 6. An upper diagram in Fig. 3 shows the connections among the planes and the numbers in the circles represent the plane numbers. A solid can be obtained by putting together some of these shells to construct a closed shell with no openings. This procedure corresponds to obtaining the independent solutions of a linear equation using shell vectors.

First, the column vector whose dimension is equal to the number m of elements in L'' is defined as the following equation for each shell s_i:

$$s_i = \text{col}\,(a_{i1}, a_{i2}, \cdots, a_{im}), \qquad i = 1, 2, \cdots, n \tag{5}$$

where

$$a_{ij} = \begin{cases} 1, & \text{if the } j\text{th element in } L'' \text{ is contained in the shell } S_i \\ 0, & \text{if it is not contained.} \end{cases}$$

As a result, "l" elements in s_i correspond to the edge lines of the opening. Therefore the opening of the newly formed shell, obtained by uniting the shell S_i and S_j, has the edge lines corresponding to l elements of the vector $s_i + s_j$ which will be obtained by an algebraic addition with modulo 2. If the sum of several arbitrary shell vectors is zero, the opening is closed. In other words, the combined shell by this sum is a closed shell which forms a solid. Therefore, the independent solutions of the following equation by an algebraic operation with modulo 2 correspond to the three-dimensional objects, where M is an $m \times n$ matrix $(s_i, s_2, \cdots, s_n)$:

$$M \cdot v = 0 \tag{6}$$

where

$$v = \text{col}\,(v_1, v_2, \cdots, v_n).$$

In other words, the union of shell S_i for all i such that the element v_i of v is not 0 is a solid.

The resulting solids $B_1, B_2, \cdots$ are geometrically closed shells, but the same spatial region might still be involved in two or more solids. Therefore these solids must be separated so that each solid has its own spatial region. For this purpose, let q_{ij} be an arbitrary point on an arbitrary plane of the set $B_i - B_j$; that is, q_{ij} is on the plane which composes the solid B_i but does not compose B_j. And let V_i denote the spatial region inside of the solid B_i. Then the flow diagram shown in Fig. 4(a) represents the method of obtaining independent solids having no space region in common. In Fig. 4, if there is a plane which composes the solid B_i but does not compose B_j, that is, if $B_i - B_j \neq \varnothing$, an arbitrary point q_{ij} on the plane is checked to see if it is involved in V_j. When involved, it is assumed that the solid B_i is inside of the solid B_j, and B_j is reorganized by replacing co-owned planes

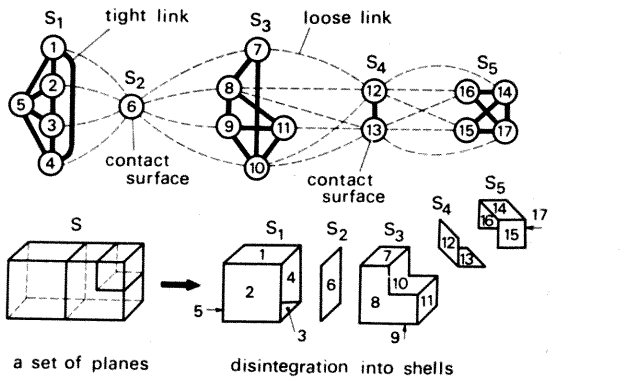


Fig. 3. Disintegration process of set of planes into shells. Succeeding process reveals independent parts as: $B_1 = S_1 \cup S_2$; $B_2 = S_2 \cup S_3 \cup S_4$; and $B_3 = S_4 \cup S_5$.

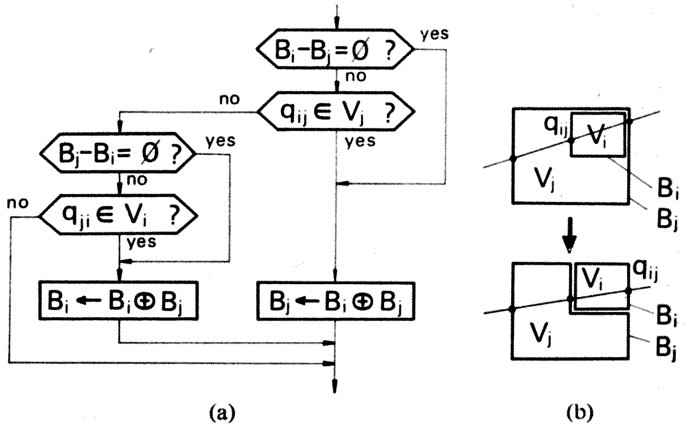


Fig. 4. Process of finding independent three-dimensional parts from closed shells. Symbolic operator ⊕ gives union of exclusive portions between two sets. (a) Flow diagram. (b) Principle of spatial separation.

$B_i \cap B_j$ of B_j with the not co-owned planes $B_i \cap \bar{B}_j$ of B_i. This process is shown in Fig. 4(b). In this case, whether or not q_{ij} is included in V_j can be determined by checking to see if one side of an arbitrary straight line passing through q_{ij} intersects the planes of B_j by an odd number of times or by an even number of times. Similarly, when q_{ij} is not involved in V_j, the same procedure checks whether or not the solid B_j involves the solid B_i. Of course there will possibly be some pairs of objects one of which is partially and not completely involved in the other. Even in this case, this flow will give a set of independent solids having no space region in common by repeating the computation along the flow until no modification is needed for all the pairs of solids, B_i and B_j. The solids which have been isolated by this process are hereafter called the parts.

The shape of a part can be judged by the shapes of planes forming the part and the relative positions of the planes. The corelationships between the parts B_i and B_j can be judged from the set of planes $D_{ij} = B_i \cap B_j$. If $D_{ij} = \varnothing$, B_i and B_j are apart from each other. If $D_{ij} \neq \varnothing$, they are in contact with the surfaces which are the elements of D_{ij}.

Fig. 5 shows some examples of such drawing recognition plotted by means of an XY plotter. Fig. 5(a) shows the three-

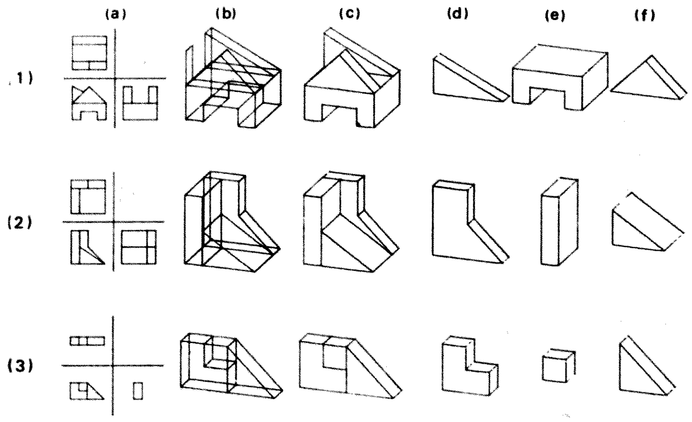


Fig. 5. Examples of drawing recognition. (a) Orthographic projection. (b) Edge structure. (c) Spatial configuration. (d)–(f) Parts.

view plans recognized as two-dimensional information from the images which are fed in the computer through a vidicon camera, and Fig. 5(b) shows the sets of edge lines obtained by calculating the correspondence between the nodes and lines in the drawings. Fig. 5(c) represents the spatial configuration of assemblages obtained by calculating the spatial planes from the information on spatial edge lines in (b). Especially in example (1), it is shown that the excess edge lines are excluded as they constitute an isolated plane. Furthermore, (d)–(f) show the component parts disintegrated from the information in (c). Here for the display purpose, both the hidden lines and the information of contacting surfaces which are equivalent to, so to speak, pasted portions of planes are eliminated in (c)–(f).

IV. Object Recognition Algorithm

The object recognition algorithm is aimed at recognizing, by a vidicon camera, all the geometric features such as shapes, positions, orientations, and sizes of three-dimensional objects placed in the real world. If we use only monocular vision with a fixed field, it is usually impossible to recognize all of the geometric features from the two-dimensional projected image. To allow complete recognition, some means by which a point on an object and a point on its image are related to each other is necessary, as well as some extent of restriction on the shape of the object. Here, the objects to be recognized are restricted to the polygonal prisms placed on a table. The vidicon camera looks down these objects obliquely and the geometric relation between the surfaces of the table and the image plane on the vidicon is calibrated in advance by an equation in which the distance and angle between them are used as parameters. The objects to be recognized are white colored and are arbitrarily arranged on a black table in random positions and orientations within a field of vision of about 30 by 30 cm. In this case the objects in the image must at least be separated from each other for the extraction of all the geometric features.

The digitized image with 5-bit brightness information, which has been fed into the computer, is first thresholded to distinguish the object areas from background. Then the number of objects is found by labeling each object area

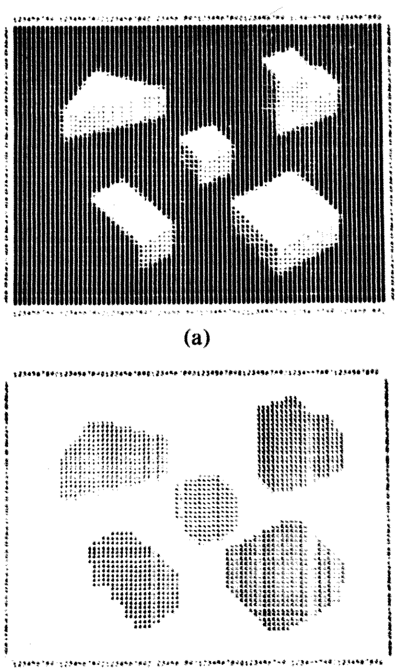


Fig. 6. Labeling process for finding each object area. (a) Digitized camera image. (b) Numbered image.

in the image with a number. Fig. 6 shows an example of the brightness image and the labeled image printed out on a line printer, where the total field of vision is represented by thinning out the picture elements. Here, the same number is given to the picture elements composing the same object. This number is stored in some bit positions of the corresponding word in the same memory area as of the digitized image. Then the digitized image I_{ij} is converted into the spatially differentiated image Z_{ij} by using the adjacent 2×2 picture elements as the following equation:

$$Z_{ij} = \tfrac{1}{2}\{ \, | I_{ij} - I_{i+1,j} + I_{i,j+1} - I_{i+1,j+1} | + | I_{ij} + I_{i+1,j} - I_{i,j+1} - I_{i+1,j+1} | \, \}. \quad (7)$$

Though this unipolar spatial differentiation uses the minimal number of picture elements, it gives a good contour image with the edge lines emphasized, as the differential gain is fairly constant regardless of the contour directions. The spatially differentiated image is also stored in the same area as the original image as a 5-bit digitized image.

From the differential image thus obtained, one of the object parts of the image is selected in the order of numbers given by the above labeling process and the edge lines of the object are detected one by one by the trenching method. In this method, the intersection points between edge lines and two parallel searching lines are found first. Secondly, the position and direction of an edge line are roughly estimated from the combination of the intersection points. Then a slit-like mask with wings is put on the roughly estimated edge line, and its inclination and length are adjusted so that it fits the edge line to the maximum degree. Finally, the converged mask is regarded as a line segment. In this case, the wings are used for finding the direction of the modification. Though

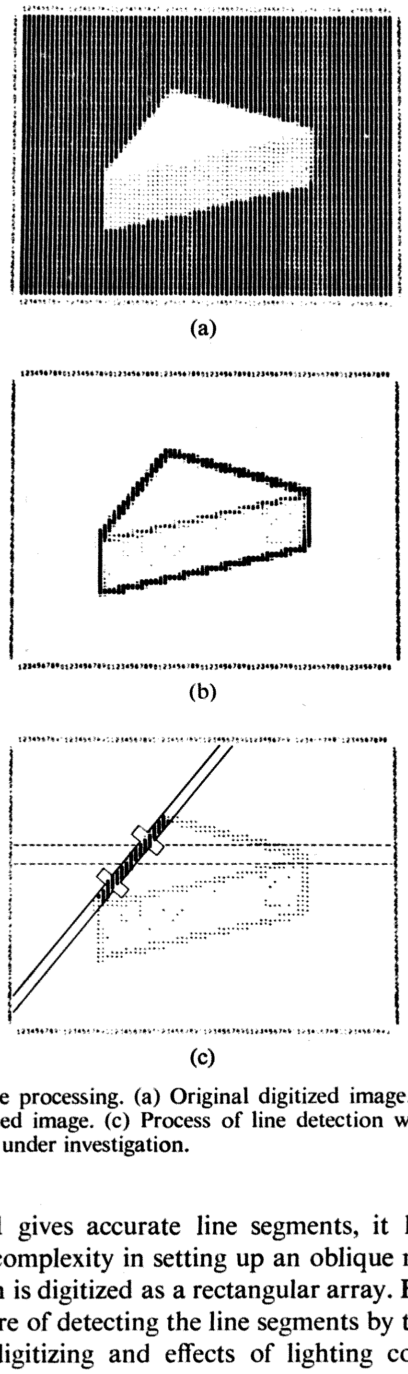


Fig. 7. Image processing. (a) Original digitized image. (b) Spatially differentiated image. (c) Process of line detection where first line segment is under investigation.

this method gives accurate line segments, it has a disadvantage of complexity in setting up an oblique mask on the image which is digitized as a rectangular array. Fig. 7 shows the procedure of detecting the line segments by this method.

Due to digitizing and effects of lighting conditions, it sometimes happens that the endpoints of some line segments do not precisely coincide with others or that a line is expressed as two adjacent lines. In these cases, a finishing process rearranges the lines and makes the necessary corrections, such as eliminating unnecessary twigs, binding together some endpoints as a vertex, and combining two adjacent line segments into a single line. Since the line segments which form a loop usually constitute a plane, the loop information is preferentially reserved in this process. Fig. 8 includes the results of this process on XY plotter. The above algorithms for line detection and correction are also employed in the previously discussed drawing recognition process.

By means of such processing we determine the set P of vertices, the set L of line segments and the set S of planes (or

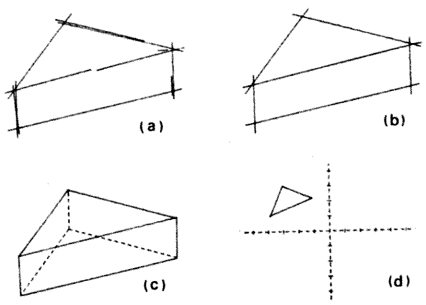


Fig. 8. Process of line rearrangement and structure analysis. (a) Detected lines. (b) Modification of lines. (c) Decision of structure. (d) Determination of position on table surface.

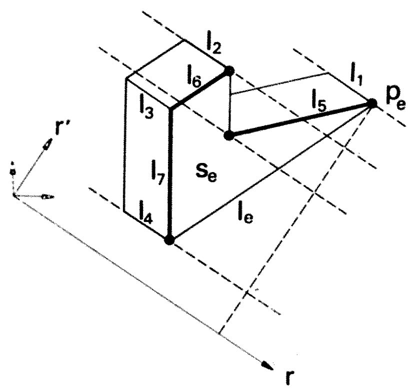


Fig. 9. Set of edge lines in the two-dimensional image of a pentagonal prism, illustrating the process of spatial structure analysis. $L' = \{l_1, l_2, l_3, l_4\}$ and $L'' = \{l_5, l_6, l_7\}$.

loops) in the two-dimensional projected image. Next we proceed to the structure analysis. Since the object to be treated is restricted to a polygonal prism, we first find a subset L' ($\subset L$), composed of more than two parallel line segments with equal length, as illustrated in Fig. 9, and assume it to be the set of edge lines of the column of the polygonal prism. If the direction of this column is denoted by r, it can be proved that one of the extreme points $p_e(\in P)$, whose coordinate position is maximum or minimum in the direction r, constitutes at least the polygonal surface $s_e(\in S)$, which we may call the characteristic surface of a polygonal prism. Moreover, it can be proved that the edge line l_e, which has the largest angle with r among all the lines passing through the extreme point p_e, is included only in a plane s_e without fail. Therefore the candidate of the characteristic surface $s_e \in S$ such that $l_e \in s_e$ is at first obtained, and the vertices with local maximum and minimum in the direction r' (perpendicular to r) are found among the vertices which constitute the loop s_e. At these vertices, the image field is divided into several strips parallel to r and the set of lines $L''(\subset s_e)$ is determined in which the odd-numbered lines in s_e are involved, when viewed in the direction r. Then each $l \in L''$ is moved in parallel in the opposite direction of r as much as the assumed column length of the prism. If the coincident line segment is found in L even partially, it is concluded that $l \in L'$ is the column of the prism and s_e is the characteristic surface, and therefore the structure analysis is completed by adding hidden points, lines, and planes to the sets *P*, *L*, and *S*, respectively. The above algorithm usually yields good recognition results even if some of the column lines are missing.

In the case of a polygonal prism, there are only two kinds of typical orientations; that is, the characteristic surface is parallel or perpendicular to the table surface on which it is placed. Therefore, by assuming the vertices and edge lines at lower positions in the image are in contact with the table surface, it is possible at this stage to calculate the spatial coordinates of the vertices from the previously calibrated relationships between the coordinates of the table surface and those of the image plane of the vidicon camera, thereby determining the position of the object in *xyz* coordinates on the table. This process is illustrated in Fig. 8.

The above process is repeated for every object in order and finally all the objects in the image are recognized. With this method, apart from the limitation of the resolution of the vidicon camera, it is generally possible to recognize any number of any polygonal prisms when they are independently arranged from each other and their characteristic surfaces are visible. In this case, it is possible to recognize even a concave polygonal prism where part of its surfaces is hidden by other parts of itself.

The accuracy of recognition is about ± 2–3 mm in terms of the coordinates on the table, due to the digitizing error of the field of vision and the linearity of the vidicon camera.

V. Handling Algorithm

The completion of drawing and object recognition processes is followed by the decision-making process for handling the objects. The handling algorithm is aimed at deciding on a sequence of manipulation steps in assembling the objects on the table so as to complete the assemblage instructed on the given drawing.

Firstly, the correspondence of the parts is investigated by referring to the list of the disassembled parts from the drawing and the list of the real parts recognized, and by selecting the required parts for the assembly. For this correspondence of the lists of polygonal prisms, the chain information of both the interior angles and the line lengths of the characteristic surfaces are utilized as well as the information of the column lengths of the prisms. When there exist more objects with the same shape than the number specified on the drawing, only the required number of parts are regarded as the necessary parts. In this case, the parts at positions nearer to the origin on the table and the parts with orientation closer to the orientation specified on the drawing are preferentially selected to make handling easier.

Secondly, the assembly order is decided. It is an interesting puzzle to determine a practical order of assembly from the given drawing. For the decision of the assembly order, not only must the corelationships between the parts in the assembled form be considered, but also the restrictions arising from the structure and function of the handling mechanism. The greater the number of required component parts involved, the more difficult the problem-solving method would

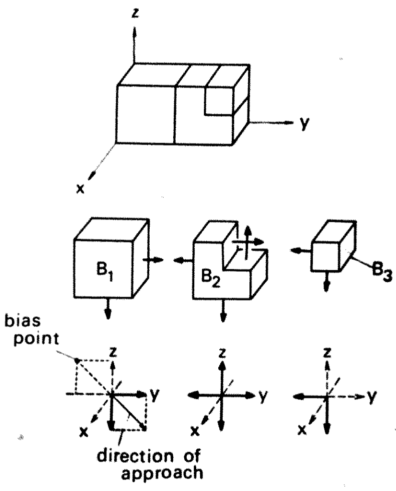


Fig. 10. Restrained planes and their directions. Restraint vectors of each part are: $u_1=(0, 0, 1, 0, 0, 1)$, $u_2=(0, 0, 1, 1, 1, 1)$, and $u_3=(0, 0, 0, 1, 0, 1)$.

be. However it is comparatively easy to solve the problem if we adopt a method of judging how to disintegrate the assemblage from the assembled state to the arranged state. Obviously, the assembly order is the reverse of this disassembly order.

Whether the disassembly is possible or not can be judged on the basis of plane vectors. These are the vectors which are perpendicular to each surface of the object and are directed toward the outside of the object. The direction of the vector is decided easily by checking whether the line perpendicular to the plane in question pierces through other planes of the object by an odd or even number of times. By the drawing recognition, we have already obtained the list of the parts B_1, B_2, $\cdots$ as the sets of planes which compose each part. We also have obtained the list of the planes as the sets of edge lines composing them, the list of the edge lines as the sets of their spatial endpoints, and the list of points with their *x*, *y*, and *z* coordinates described. On the basis of the above information, the contact surfaces between the component parts are first calculated from $B_i \cap B_j$, and a list of the contact surfaces, including the object number being contacted with them, is made for each part B_i. To each list thus obtained is added a plane, if one exists, such that the *z* coordinate is 0. This is one of the bottom planes of the assemblage and comes into contact with the table surface when the part is assembled. The above two kinds of contact planes are hereafter called the restrained planes. Now it is possible to obtain the vectors of the restrained planes in each object. An example is shown in Fig. 10. In a relatively simple assembly like this, the plane vectors point only in the three main directions *x*, *y*, and *z*. If there is a contact plane in an oblique direction, we may consider its component vectors in the *x*, *y*, and *z* directions. Therefore it is sufficient to consider a total of six directions including both positive and negative direction of each coordinate axis.

Now the restraint vector u_i of the part B_i is defined as follows:

$$\boldsymbol{u}_i = (a_{+x}, a_{-x}, a_{+y}, a_{-y}, a_{+z}, a_{-z}) \tag{8}$$

where

$$a_r = \begin{cases} 1, & \text{if the part is restrained in the direction } r \\ 0, & \text{if the part is not restrained.} \end{cases}$$

It is evident that the unrestraint vector of the part can be obtained by substituting each component a_r of $\boldsymbol{u}_i$ by its logical complement $\bar{a}_r$. The logical possibility of disassembling the part from the assemblage can be calculated by the following logical formula:

$$\overline{a_{+x}a_{-x}} \cdot \overline{a_{+y}a_{-y}} \cdot \overline{a_{+z}} = \begin{cases} 1, & \text{possible} \\ 0, & \text{impossible.} \end{cases} \tag{9}$$

This shows that it is possible to disassemble the parts when it is free in the upper direction and both x and y are free in either the positive or the negative direction. However, since

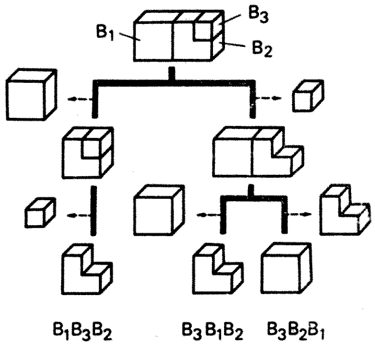
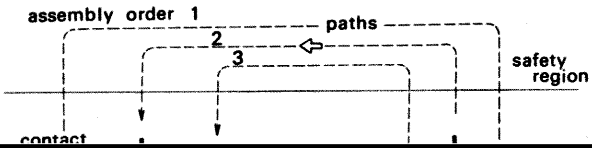


Fig. 11. Example of disassembly tree.



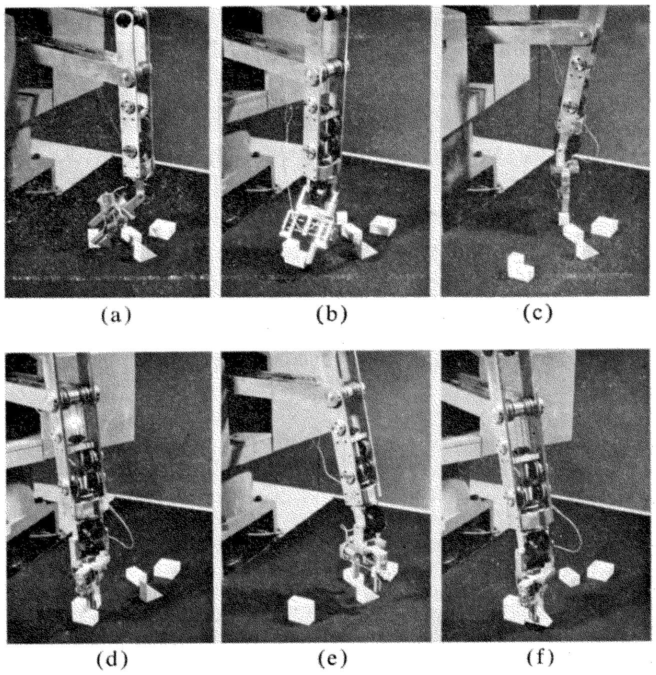


Fig. 13. Some phases of assembly sequence for case shown in part (3) of Fig. 5. (a), (b) Handling of first object. (c), (d) Handling of second object. (e), (f) Handling of third object.

VI. Discussion

The major concern of the present work is the development of the computer algorithms dealing with the information about objects. For this reason, no special efforts have been made so far on the processor hardware, and only a conventional-type control computer HITAC 7250 has been used. The whole software system for this robot requires a memory capacity of 400 kwords, including the operating system and the working area for the visual image. Moreover, more than 90 percent of the programs are written in Fortran for this particular computer whose speed no longer seems to be high enough. These are the main reasons of taking comparatively long periods of time for computations, such as 240 s for image processing, 20 s for drawing recognition, 50 s for object recognition, and 10 s for decision making, on the average. It is probably possible to reduce the working time to 1/10 or even to 1/20 by refining the programs. However the development of a low-cost processor based on a new concept, which has at least the parallel processing capability, seems to be essential for the solution in future industrial application.

With regard to drawing recognition, we have made it possible for the robot to recognize somewhat less than perfect drawings by providing two imaginative functions. It will be necessary in the future to enhance this imaginative function by further analyzing the method by which man uses his imagination to make up for missing information when reading an imperfectly written message and to enhance the capabilities of recognizing a drawing, including curved lines.

In this work the objects to be recognized are restricted to polygonal prisms, but most of the techniques used can be extended to the recognition of ordinary polyhedra. No restriction will be needed if the purpose of object recognition is merely to guess the shape of a given polyhedron. The above restriction is imposed only because of the necessity to detect even the spatial coordinate information on the attitude and position of the object. We think it possible to raise further the level of the recognizable objects by adopting new techniques, such as the binocular stereoscopic viewing technique.

For finding the assembly order in the decision process, the restraint vector is taken into consideration. The judgment of the disassembly possibility by (9) is effective only when the planes are butt-jointed. A proper modification is necessary when the restriction is more severe, such as with fit-jointed surfaces. In actuality there are many instances of assembly such as when $u_i = (1, 0, 1, 1, 1, 1)$, for example, and where there is freedom only in one direction and consequently some restrained planes slide against each other when assembled.

Since the components of the restraint vector are supposed to be binary values, the direction of approach in the assembling operation is restricted to being from above and from diagonally above at an angle of 45°. For some kinds of assemblage, a more harmonious approach will result if we consider, as the plane vector, an analog vector having some relation to the geometric features of the plane such as a vector inversely proportional to the area of the plane. In the case of complicated objects to be assembled, it is possible that none of the components of the restraint vector are 0. Therefore it will be necessary to devise a means in which new coordinates are obtained by rotating the *xyz* coordinate axes so that at least one of the components of the restraint vector is made to be 0 under the new coordinates, thereby obtaining the assembly direction.

Examples given in this paper state the usual capabilities of the robot. However, a quantitative analysis on what extent of complexity of the assemblage would be handled is too difficult, and it has not yet been clarified. In the case of the assemblage where one component is barely supported by mounting another component on it eccentrically from its center of gravity, removal of this component does not restore the original geometric relations of the remaining components. For such unstable assemblages, the handling algorithm is powerless at present because the function for checking stability has not yet been implemented, although drawing recognition is normally performed. To make this possible, the handling algorithm should be provided with a branch for judging stability and with suitable decision-making algorithms for judging unstable structures. Also, it would be necessary to equip two or more manipulating hands or supplementary tools for essential solutions to assembling unstable structures.

VII. Conclusion

We have discussed a prototype of an intelligent robot that is able to understand macroscopic instructions given in the form of three-view plans, make decisions necessary to achieve the assignments indicated on the plans, and perform the assembly work in accordance with the positions and orientations of the given objects. Usually all the human master need do is change the drawing. The robot studies and understands the given drawing even if it has never been exposed to the plans before, and from it learns a new job.

The development efforts for this robot have mainly been focused on the exploration of object information processing techniques capable of handling information not only of individual three-dimensional objects, but also of the corelationships between them. The drawing recognition capability is one of the unique achievements of these efforts and seems to be important not only as an attempt to give a machine the intelligence to understand macroscopic instructions, but also as a basis for providing graphic display techniques of three-dimensional objects which may lead to the development of sophisticated computer-aided designing techniques in the future.

The object recognition techniques previously have been limited to comparatively simple shapes such as rectangular and triangular prisms, as far as the recognition of all the geometric features is concerned. In this paper this technique has been extended to general polygonal prisms, both convex and concave. The autonomous manipulation of the objects by the aid of a computer system has also been made possible by developing the basic decision-making algorithms for assembly.

These techniques must be improved further for the realization of an intelligent system with more intensified problem-solving capability.

Acknowledgment

The authors wish to thank Dr. M. Jotaki and Dr. K. Sato of the Electro-technical Laboratory, MITI, Japan, for their invaluable advice in carrying out this study, Dr. T. Miura and Dr. T. Numakara for their continuing guidance of this project, and S. Ikeda, T. Inoyama, R. Shimomura, and J. Motoike for their extensive contribution toward this study.

References

[1] H. A. Ernst, "MH-1, a computer-operated mechanical hand," in *1962 Spring Joint Comput. Conf., AFIPS Conf. Proc.*, vol. 21. Palo Alto, Calif.: National Press, May 1962, pp. 39–51.

[2] L. G. Roberts, "Machine perception of three-dimensional solids," in *Optical and Electro-Optical Information Processing*, J. T. Tippett *et al.*, Eds. Cambridge, Mass.: M.I.T. Press, 1965, pp. 159–197.

[3] N. J. Nilsson, "A mobile automaton: An application of artificial intelligence techniques," in *Proc. Int. Joint Conf. Artificial Intelligence*, May 1969, pp. 509–520.

[4] G. E. Forsen, "Processing visual data with an automaton eye," in *Pictorial Pattern Recognition*, G. C. Cheng, Ed. Washington, D. C.: Thompson, 1968, pp. 471–502.

[5] J. McCarthy, L. D. Earnest, D. R. Reddy, and P. J. Vicens, "A computer with hands, eyes, and ears," in *1968 Proc. Fall Joint Comput. Conf., AFIPS Conf. Proc.*, vol. 33. Washington, D. C.: Thompson, Dec. 1968, pp. 329–338.

[6] A. Guzman, "Decomposition of a visual scene into three-dimensional bodies," in *1968 Fall Joint Comput. Conf., AFIPS Conf. Proc.*, vol. 33. Washington, D. C.: Thompson, Dec. 1968, pp. 291–304.

Reprinted from IEEE TRANSACTIONS ON *COMPUTERS*
Volume C-21, Number 2, February, 1972
pp. 161-170