Robotica (1998) volume 16, pp. 47–57. Printed in the United Kingdom © 1998 Cambridge University Press

An optimal planning technique for an intelligent robot's part assembly in partially unstructured environments

Changman Son

332060 Georgia Tech Station, Atlanta, Georgia 30332-1425 (USA)

(Received in final form: April 7, 1997)

SUMMARY

An efficient algorithm for generating an optimal plan for part-bringing tasks, using robotic manipulators, is introduced. The task of transporting a micro-part in a partially unstructured environment, that includes obstacles whose locations are not initially known, is introduced with the optimal plan formulated on the basis of the observed environmental conditions. Fuzzy set theory, well-suited to the management of uncertainty, is introduced to address the uncertainty associated with the part-bringing procedure. A part-bringing algorithm for generating the optimal plan related to a part assembly, despite existing obstacles, is presented. It is shown that the machine organizer using a sensor system can intelligently determine an optimal plan, based on explicit performance criteria, to overcome environmental uncertainty. The algorithm utilizes knowledge processing functions such as machine reasoning, planning, memory, and decision-making. Simulation results show the effectiveness of the proposed approach. The proposed algorithm is applicable not only to a wide range of robotic tasks including pick and place operations and maneuvering mobile based robots around obstacles, but also to the control of unmanned aircraft.

KEYWORDS: Optimal planning; Path planning; Machine reasoning; Decision-making; Fuzzy entropy; Part assembly.

I OVERVIEW

A current direction of research in robotic systems focuses on adding to accurate control some measure of intelligence.[1,2] The intelligence is essential for a robust controller capable of operating reliably and efficiently in a partially unknown or unstructured environment. Under such uncertain conditions, a degree of reasoning, planning, learning, and decision-making is required to cope with complex tasks entailing a sequence of logical steps.

The literature on robot obstacle avoidance consists primarily of material handling applications for mobile robots. Both known and unknown obstacles are considered. However, the algorithms for obstacle avoidance are based on geometrical or topological computations, and do not possess and inherent intelligence.[3-6] There has not been much work that applied intelligence functions, such as machine reasoning, to obstacle avoidance for robotic part assembly. Moreover, it is rare that a measure of optimality is concurrently used, such as a cost function or performance criterion, e.g. minimum fuzzy entropy. In the process planning for flexible part assembly, it is necessary to determine several feasible element sequences. Each such sequence lists an order in which the plan for assembly can be executed.[7] The advantage of this element sequence planning, as shown in the sequel, is that it is easy to incorporate machine intelligence.

The proposed algorithm for generating the optimal plan can be summarized as follows: First, the machine organizer with its sensor systems, e.g. a vision system, identified the positions of the obstacles, part, and part destination (target) within the workspace as shown in Figures 1(a) and 2. This is accomplished by superposing on the workspace a rectilinear grid, where the size of the grid blocks is based on the sizes of the obstacles, the part and the target. Each object in the workspace can then be approximated by a contiguous collection of these blocks. This idea will be refined later by representing each object as a collection of vertical and horizontal 'block chains.' In Figure 1(a), the obstacles o_1, o_2, and o_3 have different shapes, and the obstacles o_1 and o_2 are slightly rotated. Figure 1(b) shows the largest portions in the obstacles' areas observed from a top view. It is assumed that the overhead reveals the maximum 'footprint' of the objects in the workspace, and further that only paths some specified percentage wider that the part will be considered feasible paths. In Figure 1(b) max $[V_0]$ is the actual maximum area covered by each object, while in Figure 1(c) min $[V_b]$, represents the minimum number of contiguous blocks that cover the object. Thus, max $[V_o] \leq$ min $[V_b]$. Clearly, by making the grid block smaller makes min $[V_b]$ approach max $[V_b]$, allowing the identification of narrower paths at the cost of more computation.

The position of each block with respect to the world coordinates $(x_w, y_w, z_w, \theta_w)$ can be obtained by the sensor systems, such as a vision system. The information about the positions of all the blocks is stored in the computer memory. Subtracting all blocks in the block chains that cover the obstacles leaves the grid blocks through which the part can be moved. The region division algorithm, described in the sequel, then merges these individual grid blocks into a minimal set of blocks and merged blocks called path segments. All feasible paths not hitting obstacles taken from the permutations of this set of path segments are then generated. Next, the machine reasoning and planning procedures are used to

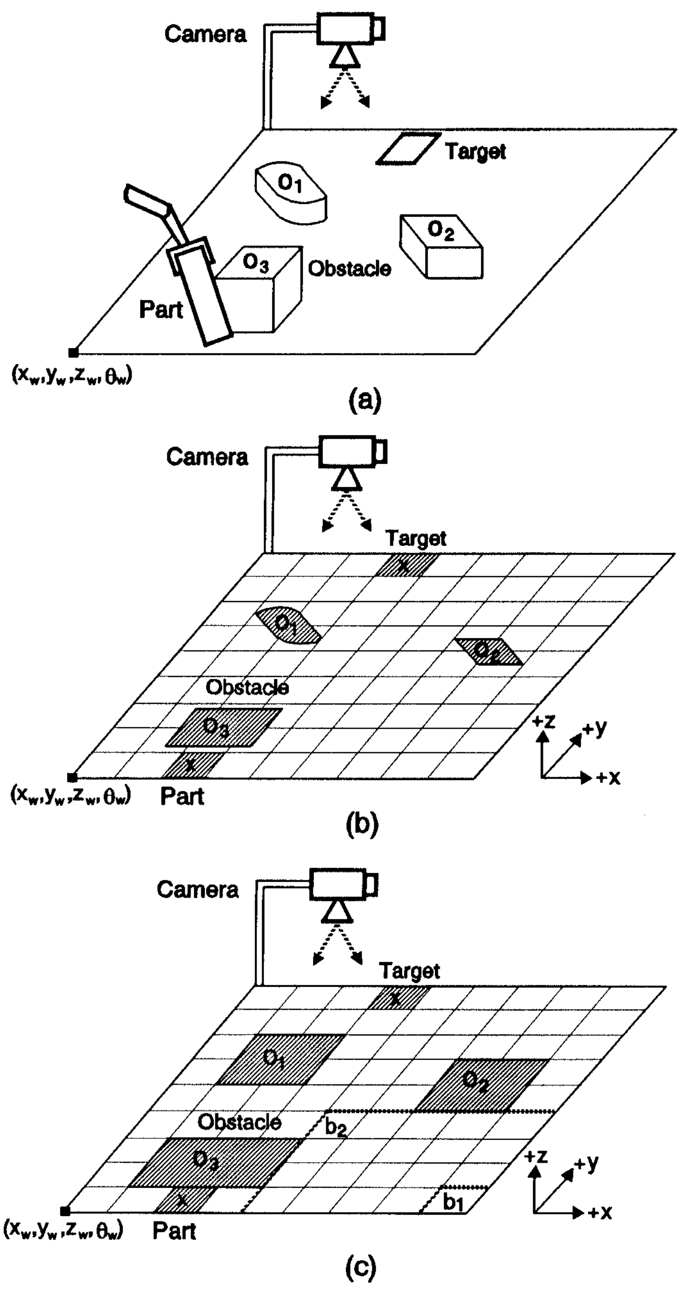


Fig. 1. (a) Partially unstructured workspace (b)(c) max $[V_a] \leq$ min $[V_b]$

find the valid sequences of elements, that is, those sequences of path segments from among all the sequences, that are consistent with the specified rules for the order of task execution. The feasible sequences of path segments are then refined into sequences of events which are the order of movement of the part's pitch angle, and x, y, and z directions; namely events $e_{\varsigma i}$, $(\varsigma = x, y, z, \theta_p)$, $(i = 1, 2, \ldots, n)$. This is accomplished by introducing in Sections II.7 and II.8 a set of nodes within the path segments. These nodes are then linked together to form a feasible path. Finally, fuzzy set theory is introduced to manage the uncertainty problem associated with the above part-bringing procedure. Through the machine decision-making procedure, an optimal plan for a specific task execution that satisfies a certain criterion, or cost function, e.g. minimum fuzzy entropy, is determined.

The information flow of the optimal planning technique for the part assembly task is shown in Figure 2. Based on the sensor information about the observed partially unstructured workspace, the machine organizer generates an optimal plan for a specific task execution by performing knowledge (information) processing tasks. The machine organization level is the most intelligent part of a hierarchical system. It performs general knowledge processing tasks, which may be composed of reasoning, inferencing, planning, decision-making, learning and etc., of varying precision. The knowledge base information, such as heuristics, experience, and expectation, affects the knowledge processing. The machine organizer determines appropriate set points used by the low level controller, e.g. joint angles, lateral distance, and other parameters relevant to the low level controller.[8] Similarly, necessary information from local sensors and the gripper coordinator are fedback to the top level of the hierarchy.

The above proposed methodology can be applied to a variety of problems. For example, a pick and place robot manipulator could pick up a part, at a location chosen by the system, while avoiding collisions with nearby obstacles. Then, using the proposed algorithmic procedures, a part could be placed at a specified destination despite existing obstacles. Alternatively, the technique presented here allows a mobile base robot to travel from a starting position to a target position, avoiding existing obstacles. The part-bringing (path planning) algorithm could be applied to the problem of guiding an unmanned aircraft to its target while avoiding intervening obstacles, such as buildings, trees, etc.

This paper is organized as follows: The algorithmic procedure of the optimal planning technique is described. A case study based on the proposed algorithm, is then presented, followed by simulation results and a closing discussions with conclusion. To conserve space, the explanation of the algorithm draws uses figures from the case study. Therefore the reader may want to review the case study before proceeding.

II ALGORITHM FOR OPTIMAL PLANNING

The objectives of the proposed algorithm are to demonstrate how the machine organizer with its sensor systems can intelligently formulate and determine an optimal plan for a specific task execution based on a particular performance criterion or cost function. The overall algorithm is itself a collection of algorithms, discussed individually below.

II.1. Optimal block size for region division

The following discussion is simplified by making the following definition:

Definition. A block chain is a contiguous horizontal or vertical collection of grid blocks. A vertical block chain consists of two or more grid blocks in a vertical column. A horizontal block chain consists of two or more grid blocks in a horizontal row.

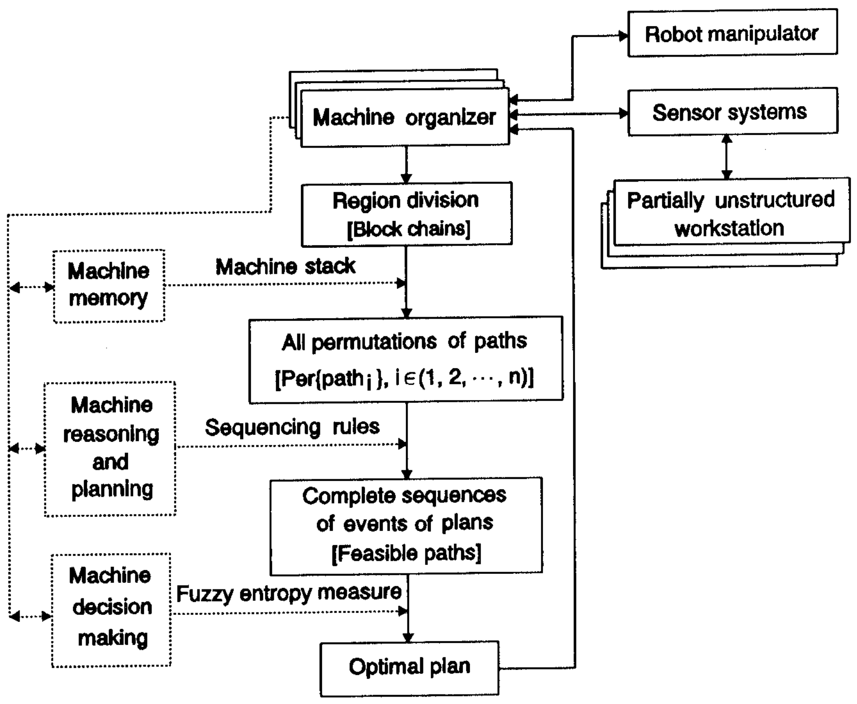

Robot manipulator
Sensor systems
Machine organizer
Partially unstructured workstation
Region division [Block chains]
Machine memory
Machine stack
All permutations of paths [Per{path$_i$}, i∈(1, 2, ···, n)]
Machine reasoning and planning
Sequencing rules
Complete sequences of events of plans [Feasible paths]
Machine decision making
Fuzzy entropy measure
Optimal plan

Fig. 2. Optimal planning technique for part assembly

The workspace consists of block chains representing obstacles, the part, the target, and available paths. The following additional assumptions are also made. One, the obstacles are higher than the part's lowest point. Two, all obstacles have vertical sides so that the footprint of the obstacle in the workspace can be determined from a top view of the obstacle. The proposed algorithm can be applied to more complicated shapes if one or more side views of the workspace are employed. Three, the obstacle can be rotated at an angle to the grid blocks. Four, all obstacles are separated by distances compatible with the size of the basic grid blocks used to define the workspace.

The implications of this last assumption will be clearer once the algorithm for determining the appropriate grid block size is discussed. At this juncture suffice it to say that the basic grid block will be larger than the part, so that the part can traverse the paths composed of the block chains. It is assumed that the size of the part is known and the target is bigger than the part but close to the same size. The algorithm for determining the size of the basic block is now given.

(B1). The rectilinear grid is chosen so that the basic block size is compatible with the part size. Essentially, the smallest possible block size would be the smallest rectangle that covers the part. The basic block size may subsequently be increased by the other steps of the algorithm. One can assume that the block size is initially set to its smallest size and then modified by the following steps.

(B2). Cover the part, the obstacles and the target with the minimum number of block chains, as observed from a top view. For the minimum block size, all feasible paths between the part and the target should be available, and the part, obstacles and target should be represented by nonoverlapping block chains. As the basic block size is increased, the some of block chains covering the part, the obstacles and the target may overlap or merge in the sense that they have common vertical or horizontal blocks chains. When this occurs the previously distinct block chains are merged into a single object.

(B3). Check if all the original feasible paths still exist. If so, increase the block size and return to **B2**. This process is repeated until the largest block size that does not eliminate feasible paths is obtained.

II.2. Region division of feasible paths

The objective of this algorithm is to represent all feasible paths between the part and the target using the smallest number of path 'segments.' A path segment is obtained by collecting into block chains the region not occupied by the objects in the workspace, namely the part, the target and the obstacles. The algorithm, to be described next, will consider all possible permutations of these path segments, which can be thought of a 'coarse' paths. Thus, the goal is to make the total number of path segments as small as possible. However, it is possible to overdo this reduction, resulting in inconsistencies when finding specific paths that traverse the adjoined path segments. This idea will become clearer as the algorithm unfolds. The goal is to reduce the computational complexity by using the minimum number of *useful* path segments. The steps of the algorithm are now given.

(R1). The block chains covering an obstacle can be represented formally as

$$\max\,[V_{o_i}] \leq \min_{k} \left[\bigcup_{j=1}^{k} V_{b_j} \right]$$

where V_o is the largest area of the *i*-th obstacle observed from the top view, and V_{b_j} is the area of the j-th block. This expression can be used to identify path segments that lie between obstacles. That is, if the total or partial areas of adjacent obstacles lie on the same horizontal or vertical block chain, then all empty individual blocks that lie between the adjacent obstacles are merged into a *single* path segment.

(R2). The workspace now contains the following objects: the part, the target, obstacles and path segments between adjacent obstacles. For each of these objects there will be a region between the boundaries of the workspace and the four sides of each object. Each of these regions becomes a single path segments. If one of the sides of the objects abutts the edge of the workspace, there is, of course, no path segment on that side.

(R3). There may remain vertical or horizontal block chains that span the workspace and contain none of the objects from **R2**, namely the part, the target, the obstacles and path segments between adjacent obstacles. If these vertical or horizontal block chains exist each becomes a single path segment.

(R4). Any path segment from the previous step with one or more common vertical or horizontal block chains are merged into a single path segment.

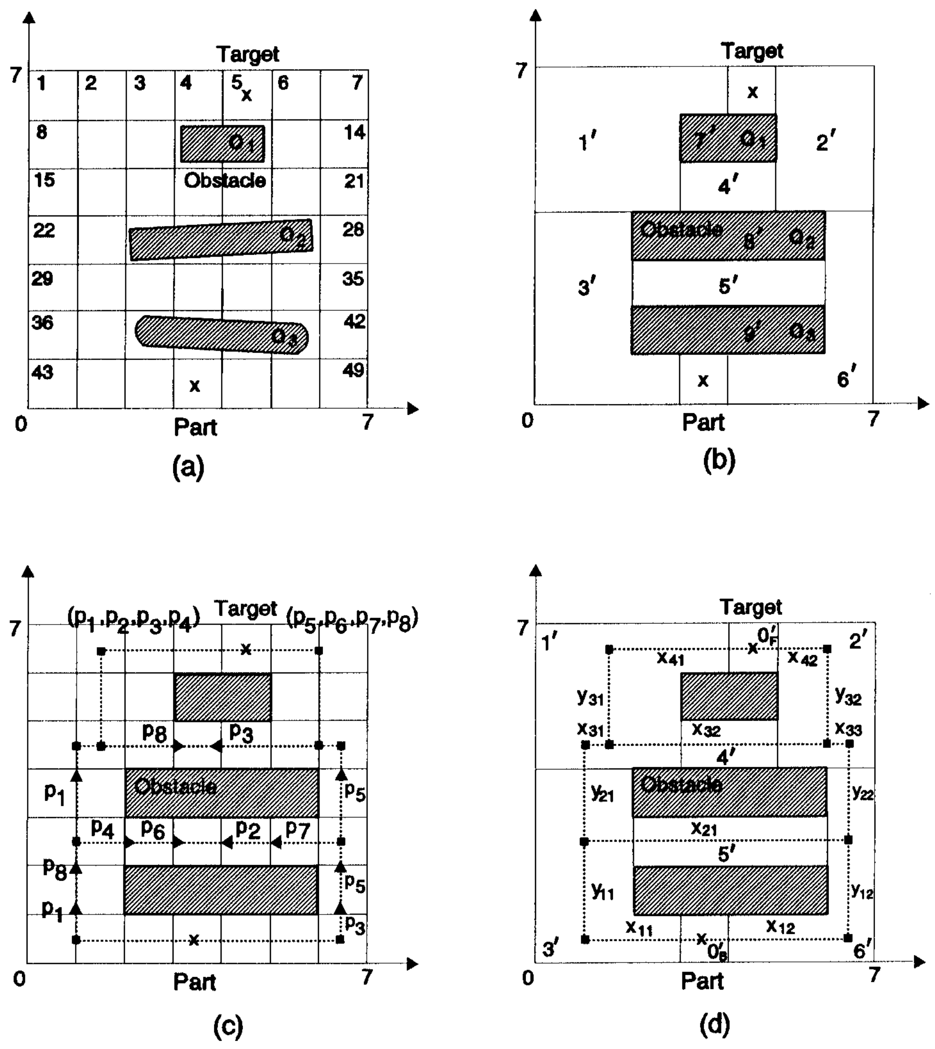
(R5). Contiguous path segments that form a square or a rectangle are merged into a single path segment. This step may fragment existing path segments. This is addressed in the next step of the algorithm.

(R6). If the path segments generated in steps **R4** or **R5** include part of path segments to the right and left of the part or target, then these latter path segments are merged with those generated by **R4** or **R5**.

All feasible paths between the part and the target based on all the permutations of the path segments resulting from the region division are generated. An example of the above procedures is shown in Figure 3(b) of Section 3.

II.3. Machine organizer's sequencing rules

The next order of task is to generate all feasible paths between the part and the target, by combining the paths segments generated by the region division algorithm. These feasible paths can be obtained from the element sequence rules given below. These feasible paths could alternatively be obtained using either the precedence graph method or the face-to-face composition method.[9-11] The element sequence rules provide a means of separating those element sequences which constitute a



Target
Part
Obstacle
O_1
O_2
O_3
1' 2' 3' 4' 5' 6' 7' 8' 9'
(p_1,p_2,p_3,p_4) Target (p_5,p_6,p_7,p_8)
x_{41} x_{42} y_{31} y_{32} x_{31} x_{32} x_{33} y_{21} y_{22} x_{21} y_{11} y_{12} x_{11} x_{12}
O'_F O'_S
(a)
(b)
(c)
(d)

Fig. 3. (a) 49 regions (b) 11 regions (c) Node selection (d) Events sequences feasible path from those that do not. Here, an element is a path segment. Any element sequence inconsistent with the element sequence rules is removed. The sequencing rules of the machine organizer are described next. Let $(p'_{\text{comp}})_r$ and $(p'_{\text{comp}})_{r+1}$ denote the elements e_r and $e_{r+1} \in p'_{\text{comp}}$, respectively, where $p'_{\text{comp}} = p_{\text{comp}} - p_{ob}$.

(S1). The rule $p_{\text{comp}} = \{e_{c_i}\}$, $(i = 1, 2, \ldots, n)$ is a composition rule, stating that a plan for execution of a given task consists of the elements $e_{c_1}, e_{c_2}, \ldots$, and e_{c_n}.

(S2). The rule p_{ob} identifies elements that should not be included in the elements $e_i \in p_{\text{comp}}$, $(i = 1, 2, \ldots, n)$. For example, if the group of elements $\{(e_1^k, e_2^k, \ldots, e_m^k)\} \in e_{ob}^k$, $(k = 1, 2, \ldots, n)$ belongs to the list of elements in p_{ob}, , the sequence of elements in a plan are formulated with only the elements $e_i \in (p_{\text{comp}} - e_{ob}^k)$.

(S3). The rule p_{ind} identifies individual element, and the rule p_{group} a groups of elements. For example $\{(e_1^k, e_2^k, \ldots, e_m^k) \in e_{\text{group}}^k$, $(k = 1, 2, \ldots, n)$ defines a group consisting of individual elements.

(S4). The rule p_{begin} identifies the elements with which a plan should begin. The rule p_{begin}^c is used to identify an element contiguous to an element of p_{begin}. The rule p_{begin}^p identifies elements in p_{begin} that have a higher priority than the remaining elements in p_{begin}.

(S5). The rule p_{end} identifies the elements with which a plan should end. The rule p_{end}^c identifies the elements contiguous to the elements in p_{end}.

(S6). The following set relations apply.

$$p_{\text{group}} \bigcup p_{\text{begin}} \bigcup p_{\text{end}} \bigcup p_{\text{ob}} = p_{\text{comp}}$$

and

$$p_{\text{group}} \bigcup p_{\text{begin}} \bigcup p_{\text{end}} = p'_{\text{comp}}$$

(S7). The rule $p_{n\text{conn}t}$ identifies elements e_r and e_l that should not be contiguous. That is $p_{n\,\text{conn}t} = \{(e_r, e_l)\}$, if invoked, means that any plan having the contiguous pair $(\ldots, e_r, e_l, \ldots)$ or $(\ldots, e_l, e_r, \ldots)$ among its sequence of elements is invalid. **S7** is referred to as a contiguity rule. The following set relations apply.

$$[p_{n\,\text{conn}t} \subset ((p'_{\text{comp}})_r \times (p'_{\text{comp}})_{r+1})] = [p_{n\,\text{conn}t} \subset ((p'_{\text{comp}})_{r+1} \times (p'_{\text{comp}})_r)]$$

(S8). The rules p_{col} and p_{row} are concatenation rules used to define vertical or horizontal sequences of three or more contiguous elements to be tested for feasibility as part of a plan, where number of elements to be considered is N_l. These rules are important to determining the part's vertical and horizontal movements to avoid obstacles. The order of specification of the elements in p_{col} and p_{row} is crucial. For instance, for $p_{\text{col}} = \{(e_{r_1}, e_{r_2}, e_{r_3})\}$, only the sequences $e_{r_1}, e_{r_2}, e_{r_3}$ and $e_{r_3}, e_{r_2}, e_{r_1}$ are valid. That is, if $p_{\text{col}} = \{(e_{r_1}, e_{r_2}, e_{r_3})\}$ is used to determine the valid sequence of elements, then a plan having a sequence of elements $(\ldots, e_{r_1}, e_{r_2}, e_{r_3}, \ldots)$ or $(\ldots, e_{r_3}, e_{r_2}, e_{r_1}, \ldots)$ is valid a plan but a plan with the sequence $(\ldots, e_{r_1}, e_{r_3}, e_{r_2}, \ldots)$ is invalid. That is, for a plan to be valid the elements specified by p_{col} or p_{row} must appear in the plan in the order specified by p_{col} and p_{row} or in the *reverse* order. The following set relations apply:

$$[p_{\text{col}} \subset ((p'_{\text{comp}})_r \times (p'_{\text{comp}})_{r+1} \times \cdots)] = [p_{\text{col}} \subset (\cdots \times (p'_{\text{comp}})_{r+1} \times (p'_{\text{comp}})_r)]$$

$$[p_{\text{row}} \subset ((p'_{\text{comp}})_r \times (p'_{\text{comp}})_{r+1} \times \cdots)] = [p_{\text{row}} \subset (\cdots \times (p'_{\text{comp}})_{r+1} \times (p'_{\text{comp}})_r)]$$

(S9). The rule $p_{co} = \{(e_r, e_l)\}$ is a consistency rule that determines order of execution, namely that element e_r should be executed before element e_l. Note that this rule can be used as a priority rule. For instance, if e_r is *not* to be executed before e_l, then writing $p_{co} = \{(e_l, e_r)\}$ insures this.

(S10). The following set relations apply.

$$[p_{co} \subset ((p'_{\text{comp}})_r \times (p'_{\text{comp}})_{r+1})] \neq [p_{co} \subset ((p'_{\text{comp}})_{r+1} \times (p'_{\text{comp}})_r)]$$

$$[p_{co} \subset ((p'_{\text{comp}})_r \times (p'_{\text{comp}})_{r+1})] = [p_{po} \subset ((p'_{\text{comp}})_{r+1} \times (p'_{\text{comp}})_r)]$$

$$[p_{po} \subset ((p'_{\text{comp}})_r \times (p'_{\text{comp}})_{r+1})] \neq [p_{po} \subset ((p'_{\text{comp}})_{r+1} \times (p'_{\text{comp}})_r)]$$

$$[p_{po} \subset ((p'_{\text{comp}})_r \times (p'_{\text{comp}})_{r+1})] = [p_{co} \subset ((p'_{\text{comp}})_{r+1} \times (p'_{\text{comp}})_r)]$$

(S11). A feasible sequence of elements p_i, $(i = 1, 2, \ldots, n)$ of a plan must be consistent with sequencing rules **S1–S10**. That is,

(a) $\text{Seq}^1\{p'_{\text{comp}}\} \in p_{\text{begin}}$, where $\text{Seq}^1\{p'_{\text{comp}}\}$ is the first element in a plan,

(b) $\text{Seq}^n\{p'_{\text{comp}}\} \in p_{\text{end}}$, where $\text{Seq}^n\{p'_{\text{comp}}\}$ is the last element in a plan, and

(c) all sequences of elements in a plan must satisfy the contiguity, concatenation, consistency, priority and other specified sequencing rules.

II.4. Machine memory

The information about the positions of all the objects in the workspace, codified as block chains, the region division algorithm, and the machine reasoning algorithms must be stored in computer memory. This could constitute a considerable amount of memory allocation. The computation time, required to generate all sequences of elements with all the permutations of elements $e_i \in (p_{\text{comp}} - p_{\text{ob}})$, and satisfying **S1–S11** could also be large. Both memory allocation and computation time can be greatly reduced by considering only those sequences of elements that begin with the elements in p_{begin} and end with the element in p_{end}. This, of course, requires some method of carefully selecting beginning and ending elements.

As an example, Table I shows the change in computational requirement based on the number of merged blocks and the number of beginning and ending elements, where n_b^m represents the number of merged blocks, n_e^{begin} the number of elements in p_{begin}, and n_e^{end} the number of elements in p_{end}. In Table I, for example,

Table I. Computational requirement ratio

n_b''' (Number of merged block)	[Case 1] $n_e^{\text{begin}} = 1(2)$ (Number of element in p_{begin})	[Case 2] $n_e^{\text{begin}} = 1$ and $n_e^{\text{end}} = 1$
$n_b''' \leq 5$	20.0% (40.0%)	5.0%
$5 < n_b''' \leq 10$	14.2% (28.5%)	2.3%
$10 < n_b''' \leq 15$	7.5% (15.3%)	0.6%

when the number of merged blocks is $10 < n_b''' \leq 15$, in Case 1 the number of elements in p_{begin} is, alternatively, one or two with no restrictions on the number of elements in p_{end}. The results is a reduction in computation to 7.5% (15.3%) of that required if all permutations are considered. In Case 2 both p_{begin} and p_{end} are limited to either one or one elements, in which case the savings are even more dramatic.

II.5. Machine reasoning

The sequence of steps that constitute the machine organizer's 'reasoning' to determine an order of task executions can be described as follows.

1. Define the sets p_{begin}, p^c_{begin}, p^p_{begin}, p_{end}, p^c_{end}, p_{comp}, p_{group}, and p_{ob}.

(a) The machine organizer assigns the block or merged block related to the present position of the part and target to the elements of p_{begin} and p_{end}, respectively.

(b) The other sets are obtained similarly.

2. Define the sets p_{col}, p_{row}, p_{po}, and $p_{n\,\text{connt}}$ related to the subportion of the specific task.

(a) Paris of empty block(s) or merged block(s) that are not contiguous to each other are assigned as elements of $p_{n\,\text{connt}}$ for all the elements $e_i \in (p_{\text{comp}} - p_{ob})$, $(i = 1, 2, \ldots, n)$.

(b) Vertical and horizontal, contiguous, empty blocks or merged blocks are assigned as elements of p_{col} and p_{row}, respectively, if the number of the consecutive, contiguous, empty blocks or merged blocks, N_l, is $N_l \geq 3$.

(c) All pairs $\{(e_r, e_l)\}$ described below are assigned as elements of p_{po}.

(i) $\{e_r \in p_{\text{end}} \mid r = 1, 2, \ldots, n_a\}$ and $\{e_l \in p^c_{\text{begin}} \mid l = 1, 2, \ldots, m_a\}$,

(ii) $\{e_r \in (p_{\text{comp}} - p_{ob} - p_{\text{begin}}) \mid r = 1, 2, \ldots, n_b\}$ and $\{e_l \in p_{\text{begin}} \mid l = 1, 2, \ldots, m_b\}$,

(iii) $\{e_r \in p_{\text{end}} \mid r = 1, 2, \ldots, n_c\}$ and $\{e_l \in (p_{\text{comp}} - p_{ob} - p_{\text{end}}) \mid l = 1, 2, \ldots, m_c\}$.

(γ) The other rules are similarly implemented.

II.6. Machine planning

The steps by which the machine organizer plans an order of task execution are as follows.

1. For the n elements in $e_i \in (p_{\text{comp}} - p_{ob})$, $(i = 1, 2, \ldots, n)$, generate all the possible n-element sequences such that $\text{Seq}^1\{p_{\text{comp}} - p_{ob}\} \in p_{\text{begin}}$ and $\text{Seq}^n\{p_{\text{comp}} - p_{ob}\} \in p_{end}$.

2. Eliminate any of the plans generated in step 1 that do not satisfy **S1–S11**, that is the contiguity rule **S7**, the consistency rule **S9**, etc.

II.7. Linkage of subpaths of a feasible path

The path segments of the feasible paths between the part and the target generated by the machine planning algorithm, are linked together by determining nodes within each path segment and then connecting the nodes to establish a definitive path from the part to the target. In short the 'plan' established before the node generation is a collection of path segments of varying heights and widths, within which a specific path has to be established.

A node represents an intersection point of lines that bisect rectangular horizontal and vertical block chains within each subpath as shown in Figure 3 of Section III for the case study presented later. The algorithm for generating the nodes which are then connected to generate a definitive path are as follows:

(N1). Bisect the entire path segments that contain the part and the target both horizontally and vertically unless the bisector is interrupted by obstacle(s). If the horizontal or vertical bisector is interrupted by obstacle(s), bisect the entire path segment between the obstacle and the part or the target.

(N2). Bisect the distances between all neighboring obstacles as follows:

(a) Bisect the distances both horizontally and vertically between all neighboring obstacles that do not partially or wholly belong to the same block chain.

(b) Bisect the distance between neighboring obstacles vertically if they partially or wholly belong to a common horizontal block chain and horizontally if they partially or wholly belong to a common vertical block chain. It is important to note that the horizontal or vertical block chain to which the obstacles partially belong does not have to be a block chain that defines either obstacle, but is simply a block chain that does include part of both obstacles.

(N3). Bisect the entire vertical (horizontal) block chain between each of the four sides of the workspace and the nearest obstacle, part, or target located on the same horizontal (vertical) block chain.

(N4). Extend all the bisected lines to their feasible limits, left, right, up, and down, and assign all intersection points as nodes.

(N5). Let $n_{i,j}$ represent a node in the i-th column (vertical block chain) and the j-th row (horizontal block chain) of the workspace. Then if nodes $n_{(i+k),(j+1)}$ and $n_{(i+1),(j+k)}$, $(k = 0, 1, \ldots)$ occur in the same horizontal or vertical block chain, respectively, choose the node(s) to be connected into a path as follows.

If an obstacle, part or target lies between the nodes $n_{(i+k),(j+1)}$ $(n_{(i+1),(j+k)})$ and nodes $n_{(i+k),j}$ $(n_{i,(j+k)})$ previously selected at part of a path then

(i) Choose the nearest node(s) left and right (above and below) of the obstacle, part, or target.

(ii) Choose additional nodes left and right (above and below) if the obstacle lying between the set of nodes under consideration and the nodes in the next horizontal (vertical) block chain to be considered shares common vertical (horizontal) block chains with the nodes already selected. That is, the nearest nodes left and right that do not share common vertical (horizontal) block chains with this next object.

(N6). For multiple nodes, in the same horizontal (vertical) block chain as the part or the target choose the nearest node(s) left or right (above or below) of the part or target that do not share vertical (horizontal) block chains with any obstacle between the nodes under consideration and nodes selected in the previous step.

An example of the above procedures is shown in Figures 3(c)(d) and 4(a).

II.8. Determination of feasible paths

At this juncture a set of nodes have been selected within each path segment. The next task is to connect these nodes to form a set of feasible paths from the part to the target as described below.

(C1). All the coordinates for the nodes selected in the previous section are stored in machine memory along with the following associated data:

(i) The path segment to which the node belongs.

(ii) The neighboring nodes, that is the nodes to which it is possible to move. An illustration is given in Table II.

(C2). For a given connected order of path segments the nodes from an initial node to a final node are linked as follows.

(a) Find the node in path segment $i+1$ that is in the list of neighboring nodes of the last selected node in path segment i, $(i = 1, 2, \ldots, n-1)$. If there is no such node then go to the next step to choose the next node within path segment i.

(b) When linking nodes within a path segmemt a 'dispersion' node may be encountered, that is a node from which multiple paths can branch off. If one of the neighboring nodes is in path segment $i+1$ choose that node. If none of the neighboring nodes is in path segment $i+1$, then choose a neighboring node from path segment i. This process may have to be repeated until a node within path segment i is found that has a neighboring node in path segment $i+1$.

(c) Repeat steps (a) and (b) until the node in the final block n is linked.

(C3). If two path segments are linked through a path segment that does not have a node, called an intermediate path segment, then two approaches are available. One is simply to define fictitious nodes within each intermediate path segment. The second is to store additional information about the nodes. For instance, in Table II, one of the neighboring nodes for n_4 in path segment $1'$ is $n_7[2', 4']$. This notation means that n_7 belongs to path segment $2'$ and n_7 is a neighboring node to n_4 through the intermediate path segment $4'$. Thus, in Table II, the merged block $1'$ has three nodes; n_3, n_4, and n_5. The node n_4 is linked to the contiguous nodes n_7 in the other merged block $2'$, through the merged block $4'$ having no node. Therefore, $n_7[2', 4']$ and $n_4[1', 4']$ should be included in the list of contiguous nodes of n_4 and n_7 of path segments $1'$ and $2'$, respectively. The choice of using fictitious nodes or the annotation scheme described above is largely a matter of choice. The fictitious node approach essentially eliminates the concept of intermediate path segments, and will not effect the total entropy of a path. The relative computational efficiencies of the two approaches is not much of an issue since intermediate path segments usually are few in number.

Table II. Node linkage according to sequences of blocks and merged blocks

Block or merged block	Node (coordinates)	Neighboring nodes [Block or merged block]
$0'_B$	n_B (3.5, 0.5)	$n_1[3']$, $n_{10}[6']$
$0'_F$	n_F (4.5, 6.5)	$n_5[1']$, $n_6[2']$
$1'$	n_3 (1, 4.5)	$n_2[3']$, $n_4[1']$
	n_4 (1.5, 4.5)	$n_3[1']$, $n_5[1']$, $n_7[2', 4']$
	n_6 (1.5, 6.5)	$n_4[1']$, $n_F(0'_F]$
$2'$	n_6 (6, 6.5)	$n_7[2']$, $n_F[0'_F]$
	n_7 (6, 4.5)	$n_4[1', 4']$, $n_6[2']$, $n_8[2']$
	n_8 (6.5, 4.5)	$n_7[2']$, $n_9[6']$
$4'$	n_4 (1.5, 4.5)	$n_3[1']$, $n_5[1']$, $n_7[2', 4']$
	n_7 (6, 4.5)	$n_4[1', 4']$, $n_6[2']$, $n_8[2']$

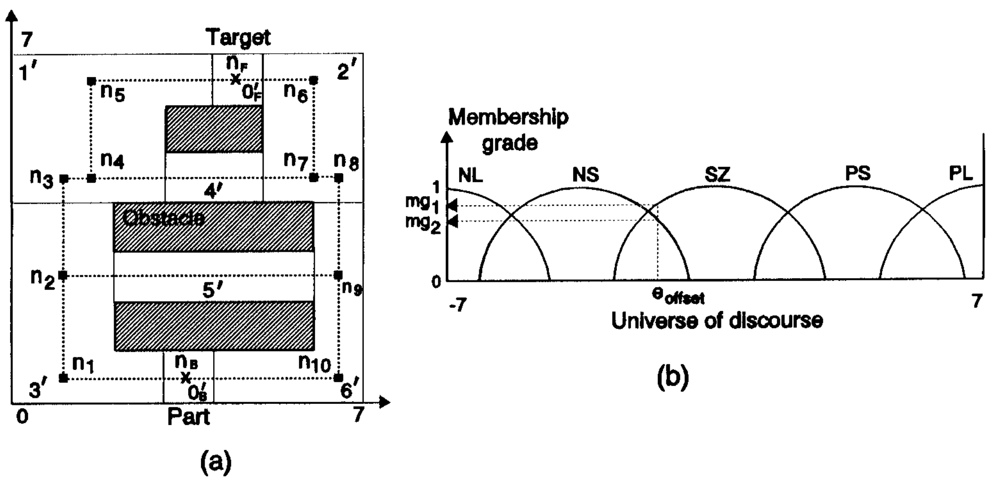


Fig. 4. (a) Node connection (b) Entropy measure of e_{offset}

For a known order of connected path segments, the nodes can be easily linked from an initial position to a desired position as illustrated in Table II.

II.9. Machine decision-making

The degree of uncertainty associated with the sequence of events of a specific feasible plan is used as a criterion for choosing an optimal plan for a specific task execution.[12,13] After finding all the feasible plans, the plan with the lowest degree of uncertainty, based on a fuzzy entropy measure, is chosen as the optimal plan to perform the task.

The uncertainties associated with the part-bringing task arise primarily from the existence of uncertainty, incompleteness, and impreciseness in measurements provided by sensors. For instance, the sizes and positions of obstacles, part, and target in a partially unstructured workspace are obtained by sensor systems, such as a vision system. Also, the size of the block used in the basic rectilinear grid is determined based on the sizes of obstacles, part, and target. Additionally, the size of the block is directly related to the determination of control values associated with the part's movements, namely events e_{ς_i}, $(\varsigma = x, y, z, \theta_p)$, $(i = 1, 2, \dots, n)$, since the feasible paths, which are derived from the connected orders of path segments, are determined based on the block chains. The fact that multiple feasible paths can be generated, each involving a different sequence of movements through the workspace is also a secondary cause of uncertainty.

In fuzzy set theory, the degree of uncertainty of a fuzzy set is called fuzziness. Fuzzy set theory is well suited to addressing uncertainties of the type described above. The membership value used for measuring the uncertainty of a sequence of events that was formulated based on sensor information is obtained through the membership function. The universes of discourse $U_{e_{\varsigma_i}}$ related to the maximum and minimum offset values of the events e_{ς_i} are determined. Finally, two methods are used to measure the degree of uncertainty associated with the sequence of events of a feasible plan.

(M1). The uncertainty associated with the sequence of events of a plan is measured by Shannon's entropy. The entropy f as a measure of fuzziness of a fuzzy set $F = \{x, \mu_F(x) \mid x \in X\}$ is defined as $f(F) = \eta \sum_{i=1}^{n} \varpi[\mu_F(x_i)]$.[14,15] The total entropy of the specific plan p_s is

$$E(p_s) = \eta \sum_{k=1}^{n} \sum_{l=1}^{m} \varpi_{kl}[\mu_F(x_l)] \tag{1}$$

where $\mu_F(x)$ is the membership function of F for the fuzzy element, x, η is a constant, $\varpi_{kl}(x) = -x \ln x - (1-x)\ln(1-x)$, the subscript l in x_l reflects the number of the linguistic terms used for obtaining the membership grade of the offset value, and the plan p_s consists of k offset values of events. Figure 4(b) shows the entropy measure of offset value of event e_{offset} whose entropy is the sum of entropies $\varpi(mg_1)$ and $\varpi(mg_2)$ due to the membership grades mg_1 and mg_2, respectively.

(M2). The Euclidean distance measure,[15] $\beta = 2$, is used for measuring the degree of uncertainty of the sequence of events of plan p_s.

$$E(p_s) = \sum_{k=1}^{n} \left[\sum_{l=1}^{m} |\mu_F(x_l) - \mu_C(x_l)|^\beta \right]^{1/\beta}, \quad x \in X \tag{2}$$

where $\mu_C(x)$ has a specific (crisp) value that characterizes the fuzzy representation $\mu_F(x)$, and $\beta \in [1, \infty]$.

II.10. Machine learning

The movement of the part from one node to the next will be called a fuzzy event because of the uncertainties inherent in this movement, as discussed above. From a learning algorithm based on the probability of a fuzzy event and a modified distance metric measure, the following stochastic approximation learning algorithm[16] is utilized to update the related probability of a fuzzy event

$$P_m^n(k+1) = P_m^n(k) + \gamma(k+1)\{\xi - P_m^n(k)\}$$

where $\gamma(k+1)$ is a sequence for convergence, k is the experiment number, and P_m^n is the probability of the mth fuzzy set of plan composed of n events with each one having an assigned degree of membership. A performance index ξ is 1 if $f(p_m^n) < f(p_q^n)$ for $q \in \{1, 2, \dots\} - m$, and 0 otherwise. $f(p_m^n)$ is a value of modified Hamming distance metric measure (2) with $\beta = 1$ and μ_C, a measure of degree of fuzziness, of the mth fuzzy set of plan composed of n events p_m^n for a specified task. Here, μ_C is 0 if $0 < \mu_{p^f} \leq \frac{1}{2}$, and 1 if $\mu_{p^f} = 0$ or $\mu_{p^f} > \frac{1}{2}$. The fuzzy set, whose membership function is Borel measurable, of a fuzzy event $p^f \in R^n$ can be written as $p^f = \{(x, \mu_{p^f}(x)) \mid x \in R^n\}$. The initial probability of each fuzzy set of plan[15] is then

$$P(p^f) = \sum_{x \in p^f} \mu_p f(x) P(x).$$

III. A case study

A robotic system is intended to bring a part to the vicinity of the target for the purpose of a micro-part insertion, despite existing obstacles, by adjusting the part's pitch angle, and x, y, and z directions which correspond to the events, e_{p_i}, e_{x_i}, e_{y_i}, and e_{z_i}, $(i = 1, 2, \dots, n)$, respectively. Figure 3 shows the positions and shapes of the obstacles, part, and target in a partially unstructured workspace. Figure 3(a) shows the largest portions in the obstacles' areas, observed from a top view, which are covered with the minimum numbers of block chains as shown in Figure 3(b), and also shows that the obstacles o_1, o_2, and o_3 have different shapes, and the obstacles o_2 and o_3 are slightly rotated.

• **Uncertainty associated with the part assembly.** The uncertainties associated with the part-bringing procedure are introduced for the reasons discussed in Section II.9. In Figure 3(c), for example, consider the part's first movement in the plan p_3. The part moves in the right direction, $+x$, as much as the width of three blocks in order to avoid the obstacle o_3. If the block width is

w_b(cm.), the first event of the plan p_3 is $e_{x_1} = 3w_b$(cm.). In order to overcome these uncertainties, fuzzy set theory is introduced.

The universes of discourse realted to the offset values of the events e_{x_i}, e_{y_i}, e_{z_i}, and e_{p_i} are $U_{e_{x_i}}(=U_{e_{y_i}} = U_{e_{z_i}} = U_{e_{p_i}}) = [-7, \ldots, 7]$, $(i = 1, 2, \ldots, n)$. For example, 7 in $U_{e_{x_i}}$ is 100 cm., 7 in $U_{e_{y_i}}$ is 100 cm., -7 in $U_{e_{z_i}}$ is -50 cm., and -7 in $U_{e_{p_i}}$ is $-10°$. The membership function related to the offset values of the events e_{x_i}, e_{y_i}, e_{z_i}, and e_{pi} is

$$\mu_F(x) = e^{-(x-b)^2/a^2}$$

where a and b are adjusted according to the linguistic terms. In Figure 5, the linguistic terms NL, NS, SZ, PS, and PL represent negative large, negative small, small zero, positive small, and positive large, respectively.

• **Region division leading to feasible paths.** Assume that the workspace is divided into 49 regions (Figure 3). All feasible paths are found by applying the procedures described in Sections II.1 and II.2. The result of the region division is shown in Figure 3(b), where the original 49 basic blocks have been merged into eleven regions, composed of eight path segments and the three block chains occupied by the obstacles.

• **Machine reasoning and planning.**
1. Define all the sets associated with the gross movement portion of the specific task.

First, the part is adjusted to be vertically straight, and the height of the part's end point is maintained at a certain level from the floor of the workspace, namely $p^c_{\text{begin}} = \{e_{p_1}, e_{z_1}\}$. Then the feasible sequences with other elements are determined.

$$p_{\text{comp}} = \{e_i\}, \quad (i = 1, 2, \ldots, 49),$$
$$p_{\text{begin}} = \{e_{46}\}, \quad p^c_{\text{begin}} = \{e^3_{\text{group}}, e^6_{\text{group}}\},$$
$$p_{\text{end}} = \{e_5\}, \quad \text{and } p^c_{\text{end}} = \{e^1_{\text{group}}, e^2_{\text{group}}\}.$$

In Figures 3 and 4, i' stands for the group elements $e^i_{\text{group}}, (i = 1, 2, \ldots, 6)$. $p_{\text{group}} = \{e^i_{\text{group}}\}$, where $(e_1, e_2, e_3, e_4, e_8 - e_{10}, e_{15} - e_{17}) \in e^1_{\text{group}}, \ldots, (e_{28}, e_{35}, e_{42}, e_{47}–e_{49}) \in e^6_{\text{group}}$, and $e_8 - e_{10}$ represents the elements from e_8 to e_{10}.

The blocks from 11 (24, and 38) to 12 (27, 41), respectively, are occupied by the obstacles. Therefore, they are assigned to the elements of p_{ob}, $p_{ob} = \{e^i_{ob}\}$, $(i = 7, 8, 9)$, where $e_k \in e^7_{ob}$, $(k = 11, 12)$, $e_l \in e^8_{ob}$, $(l = 24, \ldots, 27)$, and $e_m \in e^9_{ob}$, $(m = 38, \ldots, 41)$.

2. Define all the sets related to the subportion of the specific task.

(a) Assign the relevant elements among e_5, e_{46}, and $\{e^1_{\text{group}}\}$, $(i = 1, 2, \ldots, 6)$ to $p_{n\ \text{connt}}$, p_{col}, and p_{row}.

$$p_{n\ \text{connt}} = \{(e^1_{\text{group}}, e^2_{\text{group}}), (e^1_{\text{group}}, e^5_{\text{group}}), (e^6_{\text{group}}, e_5), \ldots, (e_{46}, e_5)\},$$

$$p_{\text{col}} = \{(e^1_{\text{group}}, e^3_{\text{group}}), (e^3_{\text{group}}, e^1_{\text{group}}), (e^2_{\text{group}}, e^6_{\text{group}}), (e^6_{\text{group}}, e^2_{\text{group}})\},$$

and

$$p_{\text{row}} = \{(e^1_{\text{group}}, e^4_{\text{group}}, e^2_{\text{group}}), (e^3_{\text{group}}, e^5_{\text{group}}, e^6_{\text{group}}), \ldots, (e^6_{\text{group}}, e_{46}, e^3_{\text{group}})\}.$$

(b) Assign the set of all the pairs $\{(e_r, e_l)\}$ described below to p_{po}.

(i) $\{e_r \mid r = 1, 2, \ldots, 11\} \in p^c_{\text{end}}$ and $\{e_l \mid l = 1, 2, \ldots, 21\} \in p^c_{\text{begin}}$,

(ii) $\{e_r \mid r = 1, 2, \ldots, 38\} \in (p_{\text{comp}} - p_{ob} - p_{\text{begin}})$ and $e_1 \in p_{\text{begin}}$,

(iii) $e_1 \in p_{\text{end}}$ and $\{e_l \mid l = 1, 2, \ldots, 38\} \in (p_{\text{comp}} - p_{ob} - p_{\text{end}})$.

Therefore, $p_{po} = \{(e^1_{\text{group}}, e^3_{\text{group}}), (e^1_{\text{group}}, e^6_{\text{group}}), (e^2_{\text{group}}, e^3_{\text{group}}), (e^2_{\text{group}}, e^6_{\text{group}}), (e_5, e_{46}), (e^r_{\text{group}}, e_{46}), (e_5, e^s_{\text{group}})\}$, $(r = 1, 2, \ldots, 6)$, and $(s = 1, 2, \ldots, 6)$.

3. Generate all sequences of elements with all the permutations of elements e_5, e_{46}, and $\{e^i_{\text{group}}\}$, $(i = 1, 2, \ldots, 6)$, and repeat it until the number of element of plan is one element.

4. Remove the following sequences of elements, and retain only valid sequences of elements. (a) Seq$'$ $\{p_{\text{comp}} - p_{ob}\} \notin p_{\text{begin}}$, (b) Seq$''$ $\{p_{\text{comp}} - p_{ob}\} \notin p_{\text{end}}$, (c) does not satisfy the contiguity, the concatenation, the consistency, and the priority rules, and (d) does not satisfy other specified sequencing rules.

• **Determination of feasible paths.** By applying the procedures described in Sections II.7 and II.8, for the connection of the path segments, feasible paths. For this case study a feasible path is really a sequence of events constituting a plan. Figures 3(c)(d) and 4(a) show the result of this procedure. In Figures 3(c) and (d), the intersection point between the bisected lines represents a node, and also shows the eight feasible plans p_i, $(i = 1, 2, \ldots, 8)$. Table II shows the linkage of nodes according to the connected orders of the blocks and merged blocks. In Table II, for example, the merged block 1$'$ has three nodes. The node n_5, whose coordinates are (1.5, 6.5) in the workspace, is linked to

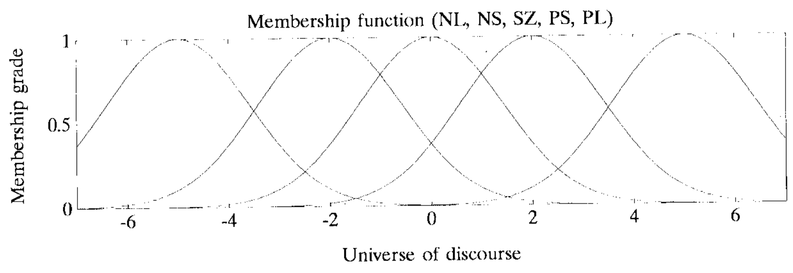


Fig. 5. Membership function related to events e_x, e_y, e_z, and e_p

the neighboring nodes n_4 in the same merged block 1′ and n_F in the block $0'_F$, where the target is located. The coordinates of the chosen nodes, the blocks to which they belong, the neighboring nodes, and the blocks to which they belong, are known. Given this information and the connected order of the block(s) and merged block(s), the nodes can be easily linked from the part positon to the target position.

In Figures 3(c)(d) and 4(a), and Tables II and III, the sequences of events of a plan are formulated as follows. Consider the plan p_8. If the connected order of the blocks and merged blocks is $0'_B \rightarrow 3' \rightarrow 1' \rightarrow 4' \rightarrow 2' \rightarrow 0'_F$, then the sequence of events of plan p_8 is formulated as $e_{p_1} \rightarrow e_{z_1} \rightarrow e_{x_{11}} \rightarrow e_{(y_{11}+y_{21})} \rightarrow e_{(x_{31}+x_{32})} \rightarrow e_{y_{32}} \rightarrow e_{x_{42}}$. Here, e_{ς_i}, $(\varsigma = x, y, z, \theta_p)$ represents the offset values of the events e_{ς_i}, and $e_{(\varsigma_k+\varsigma_l)}$ stands for the sum of the offset values of the events e_{ς_k} and e_{ς_l}.

• **Machine decision-making and learning.** The degree of uncertainty associated with the sequence of events of a plan is measured by (1). Figure 4(b) shows the membership grades of the offset value of the event e_{offset}. Through the above procedures, a plan p_3, which has the lowest degree of uncertainty with entropy measure, among the generated valid plans p_i, $(i = 1, 2, \ldots, 8)$ is chosen as an optimal plan for a specific task execution (Figures 3 and 4, and Table III). Also, it turns out that the plan p_3 has the lowest Euclidean distance measure (2). The degree of uncertainty of plan p_6, using Euclidean distance measure, was 9.33. Here, the entropies of other plans p_1, p_2, p_4, p_5, p_6, and p_7 were measured 12.56, 15.43, 14.65, 12.49, 15.51, and 14.55, respectively.

Since the plan p_3 has the lowest value in the distance metric measure, its probability increases. If it is subsequently used to perform the specified task, equivalently; a high degree of a membership grade is assigned to its fuzzy event and its probability will keep on increasing as the result of reward. On the other hand, the probabilities of other plans will keep on decreasing as a result of punishment. If p_3 satisfies the condition $\xi = 1$ on the first 20 experiments, and in the interval between 60 and 80, this indicates that the plan p_3 is subsequently used 20 times to carry out the given task over the interval. Figure 6 shows the learning curves for p_3 for this condition. It shows that the probability of p_3 increases during the above intervals. It decreases elsewhere.

Table III. Entropy measure for chosen feasible plans

Plan	Sequence of events of plan [e_p (degree), e_z (cm.), e_y (cm.), e_x (cm.)] [Connected order of blocks and merged blocks]	Entropy measure
p_3	$[4, 42, 57, 10] \rightarrow [4, -71, 28, 10] \rightarrow [4, 42, 28, 10]$ $[0'_B \rightarrow 6' \rightarrow 2' \rightarrow 4' \rightarrow 1' \rightarrow 0'_F]$	11.61
p_8	$[4, -35, 57, 10] \rightarrow [4, 71, 28, 10] \rightarrow [4, -21, 28, 10]$ $[0'_B \rightarrow 3' \rightarrow 1' \rightarrow 4' \rightarrow 2' \rightarrow 0'_F]$	11.71

IV RESULTS

Simulation results are shown in Figure 7. For the simulation, the width (x axis) and the length (y axis) of the workspace in which the obstacles, the part, and the target are located is set to 30 cm. × 30 cm.. This simulation displays the path finding process graphically on a computer terminal. On the terminal the workspace consists of an area on the screen 300 pixels wide by 300 pixels high. The center of the part's starting position (in cm.) is (12.5, 27.5) and that of the target (17.5, 2.5). The four edges of the obstacle O_1 are (0, 5), (19.9, 5), (19.9, 9.9), and (0, 9.9); of the obstacle O_2 are (0, 15), (14.9, 15), (14.9, 19.9), and (0, 19.9); of the obstacle O_3 are (20, 20), (29.9, 20), (29.9, 24.9), and (20, 24.9).

In this example, three feasible plans p_1, p_2, and p_3 were generated. The path of p_1 is $(15, 27.5) \rightarrow (17.5, 27.5) \rightarrow (17.5, 12.5) \rightarrow (2.4, 12.5) \rightarrow (2.4, 2.5) \rightarrow (14.9, 2.5)$; of p_2 is $(15, 27.5) \rightarrow (17.5, 27.5) \rightarrow (17.5, 17.5) \rightarrow (24.9, 17.5) \rightarrow (24.9, 12.5) \rightarrow (24.9, 2.5) \rightarrow (20, 2.5)$; of p_3 is $(15, 27.5) \rightarrow (17.5, 27.5) \rightarrow (17.5, 12.5) \rightarrow (24.9, 12.5) \rightarrow (24.9, 2.5) \rightarrow (20, 2.5)$.

The degree of uncertainty associated with the specific path is used as a criterion, or cost function, for choosing an optimal path for a specific task execution. The degree of uncertainty associated with the specific path is measured by (1). In Figure 7, the path p_3 (thick dotted-line), which has the lowest degree of uncertainty with fuzzy entropy measure of 6.269, is chosen as an optimal path for this specific task execution. The degree of uncertainty of the other plans p_1 and p_2 was measured to be 6.831 and 6.973, respectively.

The simulation results shown that the machine organizer with the optimal planning technique copes with each different environmental condition properly.

V DISCUSSIONS AND CONCLUSION

Normally, a robot may be subject to unplanned events and unfamiliar situations. It will be required to respond intelligently in these situations. Under such uncertain circumstances, a certain degree of logical behaviour is needed to cope with the cases.

Previous work related to path planning for parts or material handling shows some fundamental work on obstacle avoidance by detouring, geometric computation, or topological calculation, and do not possess intelligence.

In order to cope with complex tasks or non-deterministic processes partially or fully constrained by the task geometry under environmental uncertainties, the intelligent control that not only provides an effective means of describing the behaviour of systems in highly sensor dependent environments but also can be incorporated into a controller in the form of a knowledge base and an associated reasoning and planning mechanism is indispensable.

In this paper, an algorithm of an optimal planning technique for an intelligent robot's part assembly task under partially unstructured environment conditions has been introduced. A convergence of the proposed algorithm has been demonstrated through a case study

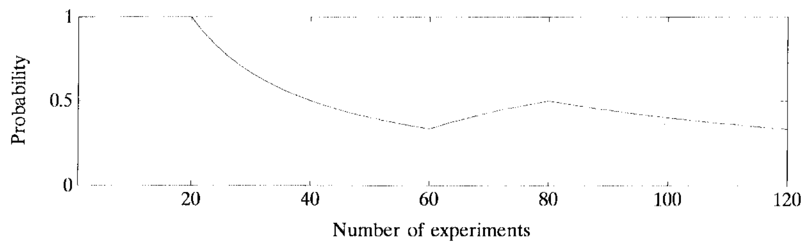
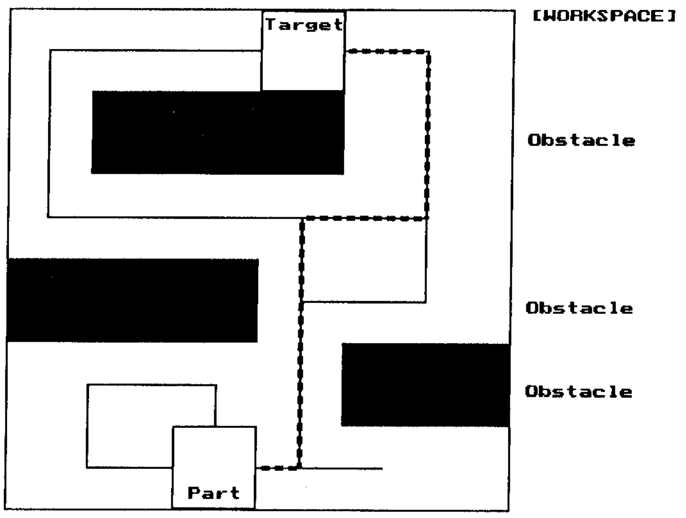


Fig. 6. Learning curve for p_3



PROCEDURES	STEP
Bisection of Path Segments Node Connection Sequencing Rules	×
Feasible Paths	×
Optimal Path (Min Entropy)	×

Fig. 7. Simulation results

and a simulation, and the computational complexity of the algorithm can be considerably reduced by considering the suggested method in Section II.4. The intelligent functions of the algorithm contribute to a more effective control of the system for specific task execution in circumstance of environmental uncertainty. The above methodology is necessary for a robot manipulator to perform complex assembly, manufacturing, or machining tasks in a partially unknown or an unstructured environment.

VI ACKNOWLEDGEMENTS

The author is grateful to Professor Dorsey of the Georgia Institute of Technology for his valuable comments.

References

1. E. Scharf and N. Mandic, "The application of a fuzzy controller to the control of a multi-degree-of-freedom robot arm" *Industrial Applications of Fuzzy Control,* (Elsevier Science Publishers, North-Holland, (1985) pp. 41–61.
2. C. Son, "A fuzzy expert organizer/robust control strategy for a moving base robotic part assembly," *Proc. IEEE International Conference on Robotics and Automation,* (May 1994) pp. 3290–3295.
3. Y. Kanayama and B. Hartman, "Smooth local path planning for autonomous vehicles" *Proc. IEEE International Conference on Robotics and Automation,* (1989) pp. 1265–1270.
4. S. Sundar and Z. Shiller, "Optimal obstacle avoidance based on the Hamilton-Jacobi-Bellman equation" *Proc. IEEE International Conference on Robotics and Automation* (1994) pp. 2424–2429.
5. R. Brooks. "Planning collision-free motions for pick-and-place operations." *Int. J. Robotic Research* **2**(4), 19–44 (1983).
6. T. Lozano-Perez J. Jones, E. Mazer and P. O'Donnell, "Task-level planning of pick-and-place robot motions" *Computer* 21–29 (March 1989).
7. V. Minzu and J. Henrioud, "Systematic method for the design of flexible assembly systems." *Proc. IEEE International Conference on Robotics and Automation* (1993) pp. 56–62.
8. G. Saridis and H. Stephanou "A hierarchical approach to the control of a prosthetic arm" *IEEE Transactions on Systems, Man, and Cybernetics* **SMC7,** No. 6, 407–420 (June, 1977).
9. L. Floriani and G. Nagy, "A graphic model for face-to-face assembly" *Proc. IEEE International Conference on Robotics and Automation* (1989) pp. 75–78.
10. K. Valavanis and S. Carelo, "An efficient planning technique for robotic assemblies and intelligent robotic systems" *J. Intelligent and Robotic Systems,* 321–347 (1990).
11. K. Chen and J. Henrioud, "Systematic generation of assembly precedence graphs" *Proc. IEEE International Conference on Robotics and Automation* (1994) pp. 1476–1482.
12. C. Son, "An optimal planning technique with a fuzzy coordinator for an intelligent robot's part assembly" *IEE Proceedings Control Theory & Applications* (January, 1997) **Vol. 144,** No. 1, pp. 45–52.
13. C. Son, "Sensor fusion techniques used for learning/identifying features of robot's part assembly tasks" *J. Engineering Manufacture* **210,** 429–435 (October, 1996).
14. G. Klir and T. Folger, *Fuzzy Sets, Uncertainty, and Information* (Prentice-Hall, USA, 1988).
15. H. Zimmermann, *Fuzzy Set Theory and Its Application* (Kluwer Academic Publishers, Holland, 1991).
16. Z. Nikolic and K. Fu, "An algorithm for learning without external supervision and its application to learning control systems" *IEEE Transactions on Automatic Control* **AC11,** No. 3, 414–422 (July, 1966).