



University of Pennsylvania  
**ScholarlyCommons**

---

Technical Reports (CIS)

Department of Computer & Information Science

---

June 1988

## Segmentation via Manipulation

Constantine J. Tsikos  
*University of Pennsylvania*

Ruzena K. Bajcsy  
*University of Pennsylvania*

Follow this and additional works at: [https://repository.upenn.edu/cis\\_reports](https://repository.upenn.edu/cis_reports)

---

### Recommended Citation

Constantine J. Tsikos and Ruzena K. Bajcsy, "Segmentation via Manipulation", . June 1988.

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-88-42.

This paper is posted at ScholarlyCommons. [https://repository.upenn.edu/cis\\_reports/694](https://repository.upenn.edu/cis_reports/694)  
For more information, please contact [repository@pobox.upenn.edu](mailto:repository@pobox.upenn.edu).

---

## Segmentation via Manipulation

### Abstract

The motivation for this paper is the observation that a static scene that contains more than one object/part most of the time cannot be segmented only by vision or in general by any non-contact sensing. Exception to this is only the case when the objects/parts are physically separated so that the non-contact sensor can measure this separation or one knows a great deal of *a priori* knowledge about the objects (their geometry, material, etc.). We assume no such knowledge is available. Instead, we assume that the scene is reachable with a manipulator. Hence the problem represents a class of problems of segmentation that occur on an assembly line, bin picking, organizing a desk top, and the like. What are the typical properties of this class of problems?

1. The objects are rigid. Their size and weight is such that they are manipulable with an suitable end effector. Their numbers on the scene is such that in a reasonable time each piece can be examined and manipulated, i.e the complexity of the scene is bounded.
2. The scene is accessible to the sensors, i.e the whole scene is visible, although some parts may be occluded, and reachable by the manipulator.
3. There is a well defined goal which is detectable by the available sensors. Specifically the goal maybe: an empty scene, or an organized/ ordered scene.

### Comments

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-88-42.

# **SEGMENTATION VIA MANIPULATION**

**Constantine J. Tsikos**

**Ruzena K. Bajcsy**

**MS-CIS-88-42**

**GRASP LAB 145**

**Department of Computer and Information Science**

**School of Engineering and Applied Science**

**University of Pennsylvania**

**Philadelphia, PA 19104**

**June 1988**

---

**Acknowledgements:** This research was supported in part by U.S. Postal Service grant 104230-87-M-0195, NSF-CER grant MCS-8219196, U.S. Army grants DAA29-84-K-0061, DAA29-84-9-0027, LORD Corporation and IMB Corporation grants.

# **Segmentation via Manipulation**

**Constantine J. Tsikos**

**and**

**Ruzena K. Bajcsy**

**Computer and Information Science Department**

**University of Pennsylvania**

**200 South 33rd Street**

**Philadelphia, PA 19104**

## **1. Motivation and Background:**

The motivation for this paper is the observation that a static scene that contains more than one object/part most of the time cannot be segmented only by vision or in general by any non-contact sensing. Exception to this is only the case when the objects/parts are physically separated so that the non-contact sensor can measure this separation or one knows a great deal of a priori knowledge about the objects (their geometry, material, etc.). We assume no such knowledge is available. Instead, we assume that the scene is reachable with a manipulator. Hence the problem represents a class of problems of segmentation that occur on an assembly line, bin picking, organizing a desk top, and the like. What are the typical properties of this class of problems?

1. The objects are rigid. Their size and weight is such that they are manipulable with an suitable end effector. Their numbers on the scene is such that in a reasonable time each piece can be examined and manipulated, i.e the complexity of the scene is bounded.
2. The scene is accessible to the sensors, i.e the whole scene is visible, although some parts may be occluded, and reachable by the manipulator.
3. There is a well defined goal which is detectable by the available sensors. Specifically the goal maybe: an empty scene, or an organized/ ordered scene.

The segmentation problem as is specified above is a subclass of a more general problem of disassembly task that we wish to address in the future. The solution to this problem calls for more non-traditional approach, that is Active Sensing, [BAJCSY 85], as opposed to the traditional static analysis of passively sampled data. It should be axiomatic that perception is not passive, but active. Perceptual activity is exploratory, probing, searching.

### 1.1 What is Active Sensing?

In the robotics and computer vision literature, the term “active sensor” generally refers to a sensor that transmits (generally electromagnetic radiation, e.g., radar, sonar, ultrasound, microwaves and collimated light) into the environment and receives and measures the reflected signals. Here we use the term active not to denote a time-of-flight type sensor, but to denote a passive sensor employed in an active fashion, purposefully changing the sensor’s state parameters according to sensing strategies. Hence the problem of Active Sensing can be stated as a problem of intelligent control strategies applied to data acquisition process which will depend on the current state of the data interpretation including recognition.

This approach is gaining more and more recognition in the literature, see [ALOIMONOS 87], [BALLARD 88], [HUANG/AHUJA 87], [POGGIO et. al. 88]. It is in the spirit of active sensing that we formulate the problem of segmentation via manipulation.

In this paper we shall briefly review the literature related to this problem and show possible applications. Subsequently we shall outline the underlined theory or principles and suggest some possible strategies for segmentation which will demonstrate the scope and validity of the theory. Our aim is to cleanly separate the a priori build in knowledge from the poorly data driven parts. Last we will present the experimental setup and the actual experiments.

### 1.2 Background

Reading through the recent proceedings of the IEEE International Conference on Robotics and Automation, the Proceedings of International Robotics Research, the IEEE Transaction on Robotics and Automation, one finds that most of the research on object description makes the following assumptions:

1. Objects are rigid, i.e. made from solid materials.
2. Objects have usually non-flexible parts.
3. If two or more objects are attached to each other, the recognition whether they are one or more objects is guided by a priori information on the shape or possibly size of the object.

People have been concentrating more on the assembly process than the disassembly process, which is a model driven, top down process. Yet if one wants to understand the structural composition of an object unless a priori given, one needs to decompose it. The closest to our thinking has come [YAMADA et. al. 87] in building an expert system which will generate all possible procedures of disassembling the object from the 3D models. This is important for example for repairs! Another application is the Post Office where mail pieces and other packages are thrown on the conveyor belt and in order to sort them one needs to decompose the pile into singulated pieces [KAK et. al, 88], [COWAN et. al. 88], [McCLAIN/KENIG 88], and [TSIKOS/BAJCSY 88].

Our problem however is to recognize how to take apart an object, explore complex scenes, composed of more than one object in arbitrary positions. That is to say, data driven perception which results in discerning solid and rigid separable objects and/or their parts and describing them in terms of their structural and geometric properties. Our working hypothesis is that this cannot be done only by vision, that one needs some possibilities of manipulation and the use of haptic information processing. But by the same argument this cannot be done by haptic exploration alone either!

## **2. Segmentation via Manipulation - The Theory:**

The theory of segmentation has the following components: Models of sensors, models of actions, a task/utility model, and the world/ scene model. The segmentation process is formulated in terms of graph-theoretic operations which are mapped into corresponding manipulatory actions.

Models of sensors: These include the characterization of the non-contact sensor such as the spatial resolution, signal to noise ratio, and their like, the physical parameters of the different end effectors, such as the vacuum suction cup, the size of the spatula for pushing objects, the span of the gripper, the maximum allowable weight and or force. Models of objects are specified in terms of their geometry, size and substance.

The Model of our world for this paper is limited to arrangement of objects thrown at random on a plane, called a heap. Then a scene is a (partial) view of a heap. The objects in the scene are represented as nodes in the digraph and the arcs denote on-top-of relation. It is important to emphasize that this digraph represents relations of only the visible surface segments, i.e. as they appear through the visual sensor which is not always the same as the physical objects and their surfaces segments. The true physical arrangements of objects on the scene as well as the part-whole relations of objects are not known.

The scene can be classified based on the analysis of the digraph into the following categories:

1. EMPTY,                      if there are no vertices in the digraph,  
i.e. an empty digraph;
  
2. DISPERSED,                if there no edges in the digraph, i.e. a  
null digraph;
  
3. AMBIGUOUS,                if there is one or more directed cycles  
in the digraph;
  
4. OVERLAPPED,               if there are at least two vertices con-  
nected with an edge;
  
5. UNSTABLE,                 this category is not tested by the anal-  
ysis of the graph  
but through analysis of the contact  
point/line of the object with  
the support plane. If this contact is  
point or line it is classified as  
unstable.

Task models which includes the final goal of the process. An example of a final goal can be the empty scene and the intermediate goals then can be those scenes that are simpler measured by a cost/benefit function. This cost/benefit function entails the cost of performing the particular manipulation, and the benefit is measured via the estimate of the outcome of the

manipulation with respect to the final goal, i.e. emptying the scene.

6. Models of Action that parametrizes the scene / object / manipulation interaction. In principle there are two kinds of Actions:

1. Sensing Actions, i.e. data acquisition action (look and/or feel),
2. Manipulatory Actions.

The purpose of the manipulatory actions for this paper is to exert a physical disturbance, being either global (as shaking) or local (as pushing/pulling). In view of our formulation of the segmentation problem as a graph generation/decomposition problem we classify the manipulatory action in relationship to the operation that apply on the digraph. There are two such operations: the node removal which means in terms of manipulation, removal of an object from the scene, the arc removal which in turn translates into object displacement in the scene so that the on-top-of relationship does not hold any more between the two objects. Put it in other ways an isomorphism exists between the manipulation actions and graph decomposition operations [TSIKOS 87]. Our approach is to close the loop between sensing and manipulation. The manipulator is used to simplify the scene by decomposing the scene into visually simpler scenes. The manipulator carries the contact sensors to the region of interest and performs the necessary exploratory movements that will determine the nature of the mechanical binding between objects in the region.

### **3. Perception-Action Interaction modeled by a Non-deterministic Finite State Turing Machine.**

The model of sensing, manipulation and control is a Non-deterministic Turing Machine (NDTM) as we show in Figure 1. The physical world (scene) is the “tape” of the machine, the “read\_from\_tape” actions are the sensing actions and the “write to\_tape” actions are the manipulation actions. The model is a Turing machine because the manipulation actions constantly change the physical environment (tape) and hence its own input. The above model is non-deterministic because of the non predictable state of the scene after each manipulatory step. From this of course follows also the non-deterministic control of actions. In addition to the non-determinism of the control strategies, the automaton has finite states, which are determined by

the finite numbers of recognizable scenes and the finite number of available actions.

We believe that this model is quite general providing that one can quantize the scene descriptions and/or the sensory outputs into unique and mutually exclusive states, and of course one has only a finite number of manipulatory actions. There are several advantages to the formalism of the non-deterministic finite state Turing machine.

The first advantage, [ALBUS et. al. 82], is that the sense-compute-act formalism allows the control problem to be partitioned in time and complexity. At any given time, the system deals only with present state and present input, produces an output which is a function of current state and current input and moves to a new state. Current state encodes information about past history of states and actions of the machine and its environment. Current sensory input is not deterministic (noise in sensory data). The next state of the NDTM is not deterministic because the machine modifies its tape via actions whose outcome cannot be known a priori (push and shake actions).

The second advantage is that the theoretical tools needed to prove correctness of the machine's behavior have long been established and tested. Path sensitization and graph de-cyclization algorithms exist, [HARTMANIS/ STEARNS 66], [KOHAVI 70], [DEO 74], to prove: 1) The goal state is reachable. 2) The state transition diagram does not contain deadlock states, or cycles.

The third advantage is that it facilitates error handling. If additional states need to be defined to deal with non-anticipated error conditions, then these states can be simply inserted. The fourth advantage is that is modular and allows insertion of new sensors, actions and feedback conditions. The fifth advantage is that it makes debugging easy. The sixth advantage is that it allows a system to be developed incrementally.

One disadvantage is that the number of states and transitions needed to represent the machine and its environment increases as more sensors are added. Addition of more sensors implies increased complexity.

**Definition:** An NDFSFA, is a quadruple (I, O, S, T) where:

- I: Inputs from a variety of contact and non-contact sensors.
- O: Outputs are Actions, such as: Shake, Push, Pick, Look, Stop, etc.
- S: States
- T: State Transition Function,  $(I \times S_c) \rightarrow S_n$ , where, the next state

$S_n$  is a function of current state  $S_c$  and current input I.

Figure 1. describes the sensing and manipulation interaction for segmentation. Relating this diagram to the NDFSFA, we shall describe in subsections the inputs, outputs, states and the transition function, i.e. the control, respectively.

### 3.1 Inputs

As indicated above, the inputs come from sensors. In our current implementation the sensor is a laser range finder (non-contact sensor) and the subsequent processing of this data. The scene is segmented into what appears to be spatially-connected surface regions. For each region, we compute the position of the center of gravity, the orientation of the surface normal at the center of gravity, an estimate of size of the smallest parallelepiped bounding the region, and an estimate of the maximum curvature . From these measurements, the objects are initially classified into one of three generic shapes such as: flat, box, and tube/roll.

The On-Top-Of relation between all pairs of visible regions in the scene is computed and the directed graph representing this relation is constructed. Vertices represent visible, connected, surface regions. Directed edges represent the spatial relations between the vertices. See Fig. 2, 3, 4, and 5.

Top-most surface segments are important in physical scene segmentation because they may belong to top-most objects in the scene. Top-most objects are important because they usually have more surfaces exposed (more ways to be grasped). The forces required to extract them from the scene are less and therefore the chances of losing positional information after the object

is being grasped are minimized. Furthermore, manipulating the top-most object keeps scene disturbances to a minimum.

A partially dispersed scene corresponds to a disconnected digraph. An efficient algorithm based on “fusion” of adjacent vertices is given in [DEO 74]. A totally dispersed scene, (as well as a singulated scene), correspond to a null graph (a graph with vertices and no edges). Efficient graph theoretic algorithms exist (testing the digraph’s adjacency matrix for all zero entries) for singulation verification. Finding the top-most objects in the scene corresponds to topological sorting of the digraph.

Visual information may be sufficient to accurately describe simple objects and non-overlapping scenes. However, it is not sufficient to distinguish between overlaps caused by two different objects in the scene and overlaps caused by a single, self-occluding object. For example, a thin flat object supported by and totally occluding a smaller box-shaped object can be mistaken as a large box-shaped object. Therefore, machine vision alone is not enough.

### 3.2 Outputs

Are indeed actions both sensing (LOOK) and manipulatory actions (SHAKE, PUSH, PICK, STOP). The sensing action (LOOK) in this implementation is only a command to take data. Naturally, it can be and in the next implementation it will be more complex, i.e choosing its view point, sampling rate and resolution and other data acquisition parameters. The cost associated with the data acquisition process must be included into the overall control schema.

The manipulation actions are composed hierarchically from simpler actions. SHAKE is the simplest action, it provides global disturbance and displacement to the work place. On the other hand PUSH and PICK exert local disturbance and causes local displacement of an object. In fact in our implementation, both PUSH and PICK action have two forms: PUSH-WITH-SPATULA, PUSH-WITH-SUCTION-TOOL, PICK-WITH-GRIPPER, PICK-WITH-SUCTION-TOOL, see Figure 12 for an example of a PICK-WITH-SUCTION-TOOL Action. In addition, each of these manipulatory actions is associated with an ERROR- RECOVERY action.

The hierarchy of actions is in terms of composition of complex actions from simpler actions and does not apply to the execution of these actions. The hierarchy of action composition is given in [TSIKOS 87]. An example of such hierarchy is shown for the action: PICK-WITH-GRIPPER in

Figure 6. Each node in the graph in Figure 6. is an manipulatory action. Some of these actions are modeled as deterministic finite state automata (FSA), while others are modeled as non-deterministic, finite state automata (NDFSA). The lowest level in the hierarchy of actions consists of very simple actions, such as: Robot-Move-To (RMT), Robot-Move-To-while-Sensing (RMTS), Gripper-Move-To (GMT), Gripper-Move-To-while-Sensing (GMTS), and their like.

The advantages of hierarchical construction are modularity, testability, and incremental growth. These actions (as expected) use additional information from contact sensors. Some of the contact sensors are: Two force/torque sensors (mounted on the gripper jaws) are used in closed loop feedback during manipulation. Force feedback is used to provide force servoing to the gripper, to sense collisions, to measure the weight of objects, and to determine if an object or tool is properly grasped. A finger position sensor is used in a closed-loop feedback manner during manipulation. Position feedback is used to provide basic position servoing to a gripper, and to refine size estimates of objects (computed from vision). A vacuum sensor is used to verify proper grasp, to differentiate between small size, non- penetrating cavities, from holes which penetrate an object. Note that all the contact sensory feedback is carried out in a local, reflexive mode rather than in a planned mode with one exception, that is when a pathological state is detected.

### 3.3 The States

In Figure 1., the states of the environment as perceived by the sensors are: Empty, Dispersed (Figure 2.), Overlapped (Figure 3.), Ambiguous (Figure 4.), and Unstable (Figure 5.) This is a finite set of states describing the environment of the Turing machine as perceived by the sensors. If new sensors are added, the set of states is partitioned to describe the scene as perceived by the additional sensors. For example, if a sensor capable of determining the “touch” relations of objects in the scene is added, then the set of the above five states can be partitioned (a finer partition) to describe both the “touch” and “on-top-of” relations.

The goal of scene segmentation is the EMPTY state. This state must be not only reachable but also measurable with the current sensors. In other words, for the machine to halt the system must have sensors to sense that the goal state has been entered. In this work the empty state is both reachable (see section on strategies), and easily measurable (all range values

in the scene are zero, which means that no surface segments and thereby no objects exist in the scene).

A specific place must be given to Error states. They are prioritized in order of severity (most severe first). For more details see [TSIKOS 87]. The pathological states are: Sensor damaged, Unable to get tool, Tool and object lost, Lost tool, Lost object above the workspace, Lost object away from the workspace, Unable reach object, Unable to Pick, and Unable to Push. As more sensors and actions are added into the system, more yet finite pathological states must be defined. These states invoke the ERROR-RECOVERY actions.

### 3.4 The State Transition Function

The control problem is transformed into the problem of topological sorting of object arrangements. The manipulation actions of object acquisition (pick) and local displacement (push) are defined as decomposition operations on digraphs representing the on-top-of relation of objects in the arrangement. The pick action is used to break the vertex connectivity of the digraph by removing vertices. Several tools may be used to implement this action. An object may be picked and removed from the scene using the gripper, or it may be picked by selecting a tool (i.e. suction tool). The push action is used to break the edge connectivity of the digraph representing the on-top-of relation. Several tools may be used to implement this action. An object may be pushed using the gripper, or it may be pushed by selecting a push tool (such as a spatula or the suction tool). Complete planning of the push actions is very complicated, [LOZANO-PEREZ 80], [LOZANO-PEREZ 81], [MASON 82], [MASON 86], and requires knowledge of the friction coefficients of all objects in the scene as well as knowledge of the spatial relations of all objects in the scene to decide where and how far to push.

#### 4. Segmentation Strategies

In the previous section we have described a Non-deterministic finite state Turing machine as a model for relating sensing, manipulation for segmentation purposes. The question is: is this model sufficient? Let us recall that “read-from -tape” are the sensing actions, “write-to-tape” are the manipulatory and error recovery actions, and the states are scene descriptions. Even with the restriction that one can categorize every scene into distinct classes (discrete states) we had to add the following rules:

1. No action is allowed to be repeated on the same object, consecutively, more than  $x$  number of times.
2. If local displacement action is called for, the PICK action has the highest priority.
3. Non-graspable objects are PUSHED.
4. Ambiguous scenes are first displaced globally (shaken) and then locally .
5. Unstable scenes are displaced globally.
6. Partially visible objects must be displaced. Displacement proceeds in an outermost first, order.
7. Flat objects are picked with a suction tool, applied at the center of gravity of the visible surface segment.
8. Box-shaped objects and tubes/rolls are picked either with the gripper or a suction tool. If this process fails then they are pushed at the edges.

With these above rules and the theory described in sections 2 and 3, we can compose several different strategies to examine the validity and generality of our theory for physical segmentation purposes.

#### 4.1 Strategy 1: (Look, Pick, Look, ...)

The control structure is shown in Fig. 7. The strategy does not use local displacement (push). The general idea is to look, pick the top-most object, and look again. If the scene is ambiguous or unstable, it shakes the heap. If shaking fails, it continues with the pick action. This strategy is simple and very effective in dealing with scenes where all objects are graspable with the set of acquisition tools. The strategy eliminates ambiguities via the shake and pick actions. If the shake action fails to remove the ambiguity then non-topmost objects are picked up. This causes objects to be lost during acquisition. For the strategy to succeed the sensor thresholds must be raised to enable the system to tolerate higher torques caused by picking objects off the center of gravity. When the threshold is raised, the probability of tool losses increases as well as the probability of damaging the sensors. Therefore, the probability of entering the fatal error state is increased. If the weight of the objects is low, the probability of damaging the sensors (even if the system picks objects supporting other objects) is low, and the strategy converges, see [TSIKOS 87].

#### 4.2 Strategy 2: (Look, Push-until-Dispersed, Pick, Look, ... )

The control structure is shown in Fig. 8. This strategy allows no interaction between the pick and push actions. The only interaction allowed is when the push action cannot reduce the number of edges in the associated graph any further. The strategy enforces a rather strong partition on the manipulation actions. This shows up as a serial plan where a single action is triggered from one state and the automaton iterates until the “look” action brings the automaton to another state. This strategy is very effective in dealing with heaps of few, small-sized objects relative to the workspace. As object size and number increases so does the number of objects pushed out of the scene and never picked up. For a proof of convergence see [TSIKOS 87].

**4.3 Strategy 3: (Look, Pick/Push, Look, ...)** The control structure is shown in Fig. 9. The central idea is to allow immediate interaction between the two manipulation actions. Since pick is more effective than push, priority is given to pick. Only if an object cannot be picked up after several unsuccessful attempts, then the next immediate action is to push that object, and immediately return to pick the next object (if one exists), or to the look action. This strategy is most effective in dealing with heaps

containing a small number of top-most objects located far away from each other. These types of heaps can be decomposed with the minimum number of collisions.

#### **4.4 Strategy 4: (Look, Push\_Partially\_Visible, Pick, Push, Look, ...)**

The control structure is shown in Fig. 10. The central idea is to interleave the interaction between the pick and push actions. In other words, the strategy is to look, then execute a series of push actions and displace partially visible objects out of the scene, then to pick all topmost objects, then push all objects that the pick action failed to remove after several attempts, and finally, to look again. This strategy segments the heap from both the top and the sides. The partially visible objects are first pushed out of the scene. This creates free space for future push actions and minimizes the likelihood of collisions towards the borders of the scene. By ordering the sequence of actions we achieved minimum interference between the manipulation actions and we sequenced the execution of the look action to occur when it is needed the most, (after a series of local displacements). If all the objects targeted for pick are graspable, they are removed one by one, following the topological ordering of the graph. This strategy is the most effective. It keeps action interference to a minimum. It uses the most expensive action (look) only when it is necessary, (after a shake, or a series of push actions). This grouping and sequencing of actions has performed very well for the majority of heaps and objects. The strategy keeps the number of tool changes to a minimum.

### **5. Implementation and Experimental Results**

The system block diagram is shown in Fig. 11. It consists of a range imaging system, a linear stage, a PUMA 560 robot, a LORD Corp. servoed instrumented gripper, a micro-VAX-II computer, a support structure, several tools, tool fixtures, and accessories.

All experiments run on the real system. No simulation results are reported. The domain was mostly objects found in the mail stream, such as: parcels, flats, tubes and rolls. A number of additional experiments were conducted with objects containing holes, cavities, and some porous objects. The heaps were created by stacking these objects at random to an average of

five object layers per heap. The weight of every object was under one pound. During all experiments the heap was observed to transform and to enter all five states: ambiguous, unstable, overlapped, dispersed, and empty[.

### **5.1 Experiment 1**

The purpose of this group of experiments was to evaluate strategy 1. The strategy performed well on unstable, overlapped, and dispersed heaps. Difficulties were observed with ambiguous configurations. The shake action was not very effective in removing ambiguities. One reason is that the action was implemented using the linear stage in a vibration mode at maximum speed and acceleration. These speeds and accelerations were not enough to produce a significant change in the scene. Using the pick action to remove ambiguities resulted in an increased number of tool and object losses. The shake action failed to eliminate the ambiguities caused by configurations of flats. This is because flats form stable configurations. However, because flats are rather lightweight and flexible, it was possible to use the pick action to break-up cyclic object configurations without many tool or object losses. Strategy 1 failed to converge when the heap contained porous objects.

### **5.2 Experiment 2**

For the strategy to work efficiently, very large work space is required. This is because all overlaps must be removed and the scene must become dispersed before any pick action. In many unstable scenes containing a mix of flats, boxes and tubes, the shake action forced the tubes to fall into the cavities of the heap. This stabilized the heap and created more overlaps. The shake action was very effective in dispersing heaps of cylindrical objects. In scenes containing only tubes/rolls the shake action removed all overlaps. Strategy 2 performed well in heaps of few, small-sized objects. As the object size and number increased, so did the number of objects that were pushed out of the scene and never picked up.

### **5.3 Experiment 3**

The strategy has the tendency to produce additional overlaps when an object being pushed falls on other objects in the next layer of the heap. This was

not catastrophic because this overlap was detected during the next look action. We have observed that if the objects cannot be picked up by any tool, then the strategy degrades into a sequence of unsuccessful pick, followed by push actions. The system picked all objects with cavities and pushed all large objects containing holes, as well as all porous objects. The major drawback of this strategy is that if push actions are interleaved with pick actions then the push action may displace other objects targeted for the pick action. Therefore, the next pick action will most likely fail. Although this is not catastrophic, it makes it necessary to trigger another look action. Better planning of the push actions may help to eliminate some of these problems. However, one has to keep in mind, that even better planning will not solve the problem, unless assumptions about the heap and knowledge of the surface properties of all objects composing the heap is introduced.

#### 5.4 Experiment 4

This strategy is the most effective and capable of handling a variety of objects and heaps. One reason for its success, is the way it manipulates the heap; from the top and sides. Another reason is that the actions are ordered and vision is applied when needed the most, after global displacement (shake), or a series of local displacements (a series of push actions).

### 6. Conclusions

We introduced the paradigm of iterative, interactive scene segmentation and simplification via vision, manipulation, force/torque and other sensory input. The scene simplification is based on graph decomposition operations of vertex and edge removal. These operations are in turn defined isomorphic to the pick and push manipulation actions. We have shown that the sensors can be used as the partial graph generators, and the manipulator as the decomposing mechanism of this partial graph. The actions and strategies are modeled as non-deterministic, finite state Turing Machines that decompose these graphs under sensor supervision. The strategies converge (for theoretical proof, see [TSIKOS 87]). If pathological states are detected then error recovery actions are invoked.

We have integrated a vision system, a manipulator, force/torque and other sensory input into an experimental robot work cell and conducted experiments to test convergence, error recovery and graceful degradation of

four different strategies. We have found that many of these strategies can recover from pathological states, tolerate errors in the sensory data, recover from un- successful actions, and converge. What we have learned during this work is:

1. In an unstructured environment, where there is uncertainty and incomplete information, a detailed, sophisticated plan is not enough. The plan must constantly change and adapt to the sensory input.
2. Redundancy of Actions (in addition to redundant sensors, tools, etc.) is needed for exploration of unstructured environments.
3. In the domain of heap segmentation, it is possible to reach the goal state via iteration and interaction of a few sensors, tools and a few simple, short-range manipulation actions.

## ACKNOWLEDGEMENTS

This research was funded in part by the United States Postal Service, BOA Contract: 104230-87-H-0001/M-0195, DARPA/ONR grant N0014-85-K-0807, NSF grant DCR 8410771, Air Force grant AFOSR F49620-85-K-0018, Army grant DAAG-29-84-K- 0061, by DEC Corporation, IBM Corporation, and LORD Corporation.

## REFERENCES

[ALBUS et. al. 82]

J. Albus, A. Barbera and M. Fitzgerald, "Programming a Hierarchical Robot Control System", 12th International Conference on Industrial Robots, Paris, France, June 1982.

[ALOIMONOS et. al. 87]

J. Aloimonos, A. Basu, and C. M. Brown, "Contour, Orientation and Motion", First ICCV, London, June 1987.

[BAJCSY 85]

R. Bajcsy, "Active Perception vs Passive Perception", IEEE Computer Society Third Workshop on Computer Vision: Representation and Control, Bellaire, MI, October 13-16, 1985.

[BAJCSY/TSIKOS 87]

R. Bajcsy and C. Tsikos, "Perception via Manipulation", 4th International Symposium of Robotics Research, Santa Cruz, CA, August 9-14, 1987, pp. 199-206.

[BALLARD 88]

D. Ballard, "Eye Fixation and Early Vision: Kinetic Depth", (Submitted) to Second ICCV, Florida 1988.

[CAI/ROTH 87]

C. Cai and B. Roth, "On the Spatial Motion of a Rigid Body with Point Contact", Proceedings of the IEEE International Conference on Robotics and Automation, Raleigh, North Carolina, 1987, pp. 686-695.

[COWAN et. al, 88]

C. K. Cowan, J. DeCurtins, P. G. Mulgaonkar, and A. R. de St. Vincent, "Advanced Research in Range Image Interpretation at SRI", Proceedings of the Third Advanced Technology Conference, Washington, DC, May 3-5, 1988, pp. 361-375.

[DEO 74]

N. Deo, "Graph Theory with Applications to Engineering and Computer Science", Prentice-Hall, 1974.

[HARTMANIS/STEARNS 66]

J. Hartmanis and R. Stearns, "Algebraic Structure Theory of Sequential Machines", Prentice-Hall, 1966

[HOFF/AHUJA 87]

W. Hoff and N. Ahuja, "Extracting surfaces from stereo images: An integrated approach", In Proceedings of the International Conference on Computer Vision, 1987, pp. 284-294.

[KAK et. al 88]

A. C. Kak, K. D. Smith, A. J. Vayda, and R. L. Cromwell, "Range Image Interpretation for Postal Object Recognition", Proceedings of the Third Advanced Technology Conference, Washington, DC, May 3-5, 1988, pp. 391-405.

[KOHAVI 70]

Z. Kohavi, "Switching and Finite Automata Theory", McGraw-Hill, 1970.

[KUMAR/ARBIB 87]

V. kumar and M. A. "Problem Decomposition for Assembly Planning", Proceedings of the IEEE International Conference on Robotics and Automation, Raleigh, North Carolina, 1987, pp. 1361-1366.

[LOZANO-PEREZ 80]

T. Lozano-Perez, "Spatial Planning with Polyhedral Models", Ph.D. Dissertation MIT Department of Computer Science and Electrical Engineering, June, 1980.

[LOZANO-PEREZ 81]

noindent T. Lozano-Perez, "Automatic Planning of Manipulator Transfer Movements", IEEE Transactions on Systems, Man, Cybernetics, Vol. SCM-11, No. 10, Oct. 1981.

[MASON 82]

M. Mason, "Manipulator Grasping and Pushing Operations", Ph.D. Dissertation, MIT, Department of Electrical Engineering and Computer Science, June, 1982.

[MASON 86]

M. Mason, "Mechanics and Planning of Manipulator Pushing Operations", International Journal of Robotics Research 5, Spring 1986.

[McCLAIN/KENIG 88]

R. A. McClain and N. Kenig, "Range Image Interpretation for Mailpiece Recognition: An Interim Report", Proceedings of the Third Advanced Technology Conference, Washington, DC, May 3-5, 1988, pp. 943-957.

[MORRIS/HAYNES 87]

G. H. Morris and L. S. Haynes, "Robotic Assembly by Constraints", Proceedings of the IEEE International Conference on Robotics and Automation, Raleigh, North Carolina, 1987, pp. 1507-1515.

[PESHKIN/SANDERSON 87]

M. A. Peshkin and A. C. Sanderson, "Planning Robotic Manipulation Strategies for Sliding Objects", Proceedings of the IEEE International Conference on Robotics and Automation, Raleigh, North Carolina, 1987, pp. 696-701.

[POGGIO et. al. 88]

T. Poggio, E. Gamble, W. Gillett, D. Geiger, D. Weinshall, M. Vilalba, N. Larson, T. Cass, H. Buelthoff, M. Drumheller, P. Oppenheimer, W. Yang, and A. Hurlbert. "The MIT vision machine", Proceedings of Image Understanding Workshop, Boston, MA, April 1988, (Morgan and Kaufman publishers).

[RAJAN et. al. 87]

V. T. Rajan, R. Burridge and J. T. Schwartz, "Dynamics of a Rigid Body in Frictional Contact with Rigid Walls", Proceedings of the IEEE International Conference on Robotics and Automation, Raleigh, North Carolina, 1987 pp. 671-677.

[TSIKOS 87]

C. J. Tsikos, "Segmentation of 3-D Scenes using Multi-Modal Interaction Between Machine Vision and Programmable, Mechanical Scene Manipulation", Ph.D. Dissertation, Department of Computer and Information Science, University of Pennsylvania, December 1987.

[TSIKOS/BAJCSY 88]

C. J. Tsikos and R. K. Bajcsy, "Physical Scene Segmentation via Vision and Manipulation", Proceedings of the Third Advanced Technology Conference, Washington, DC, May 3-5, 1988, pp. 958-974.

[YAMADA et. al. 87]

S. Yamada, N. Abe and S. Tsuji, "Construction of a Consulting System for Structural Description of Mechanical Objects"[T, Proceeding of the IEEE International Conference on Robotics and Automation, Raleigh NC, 1987.

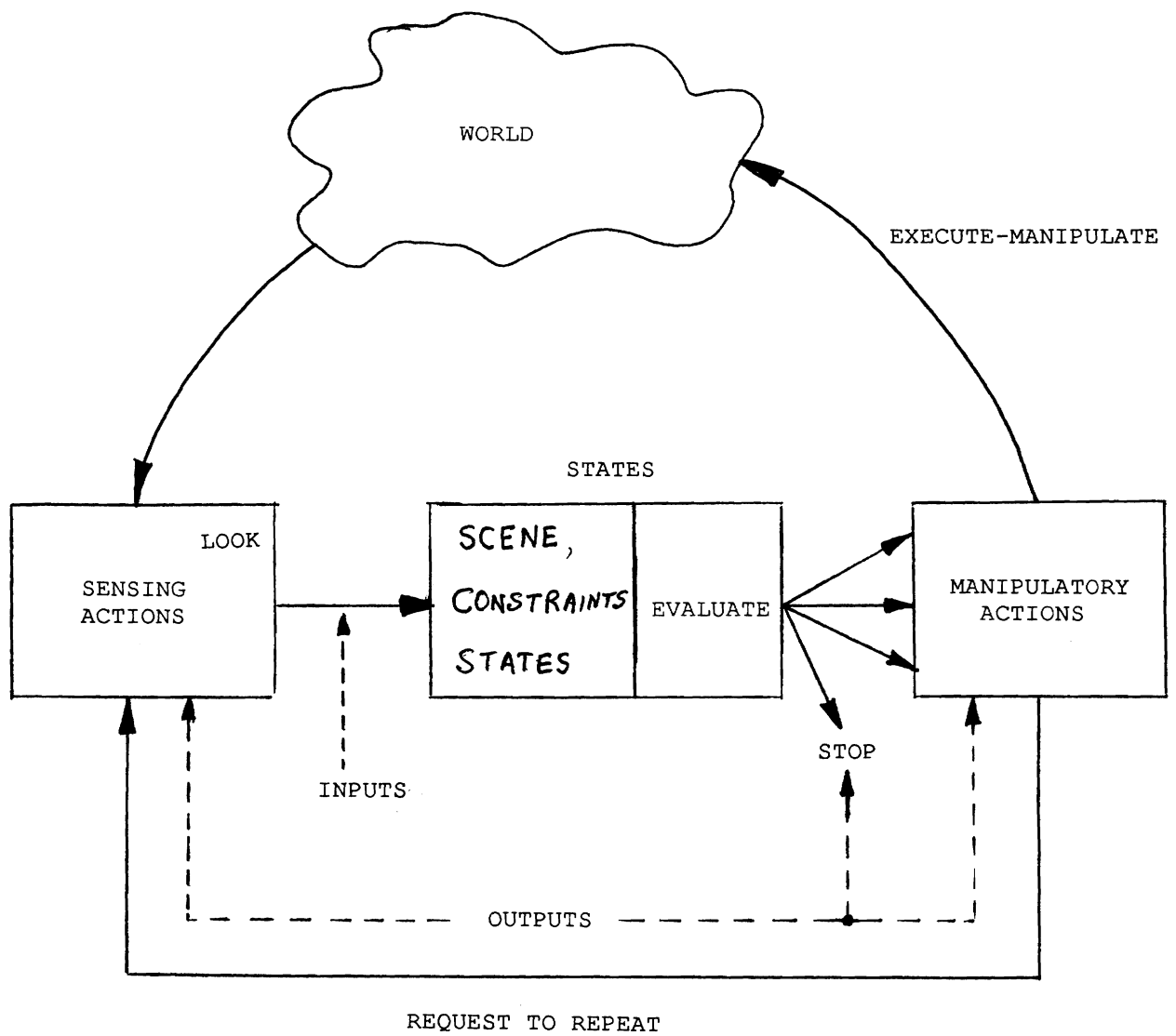


Figure 1. Sensing and Manipulation Interaction for Segmentation

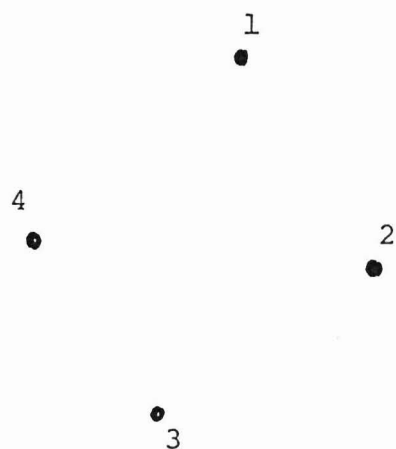
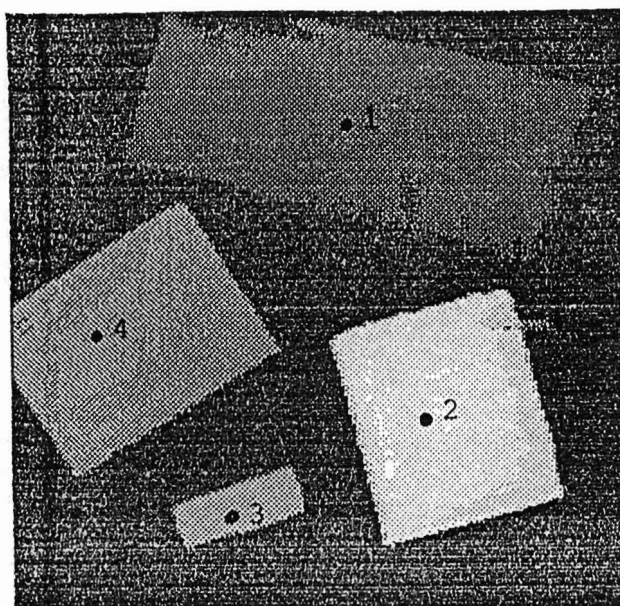


Figure 2. Range Image of a DISPERSED Scene and the corresponding graph.

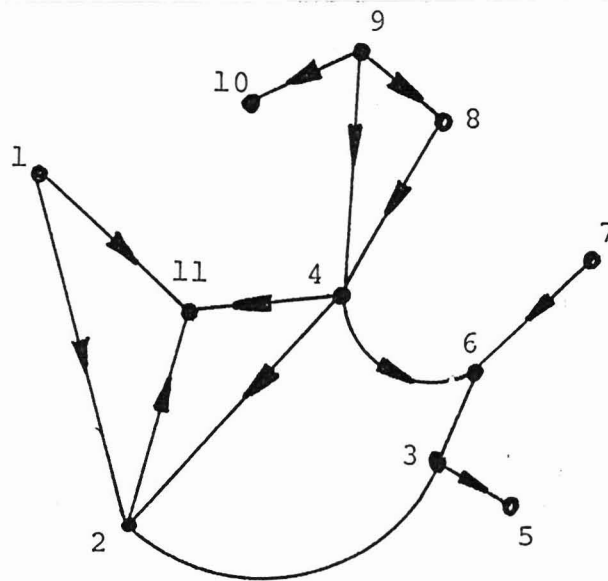
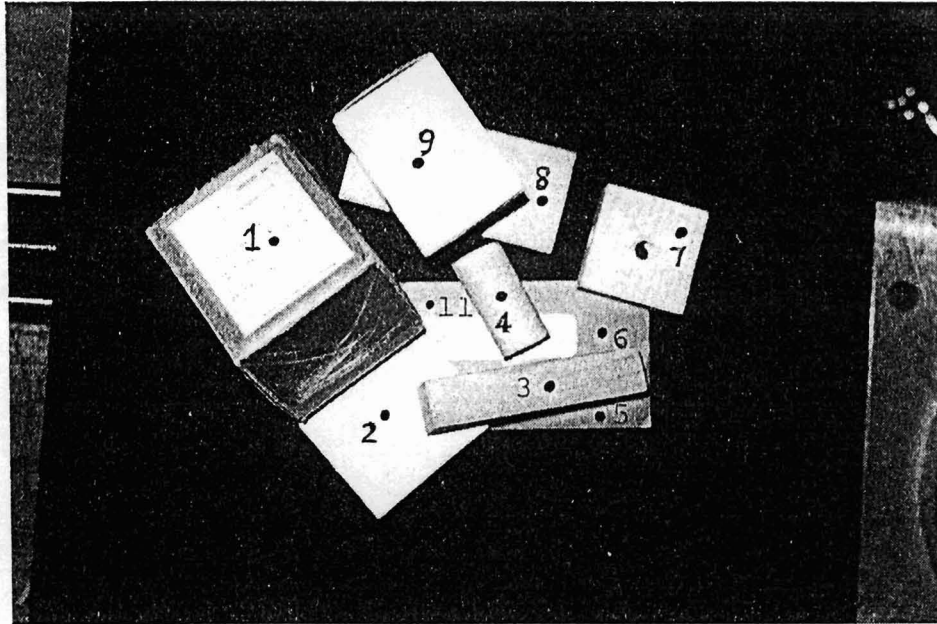


Figure 3. Intensity Image of an OVERLAPPED Scene and the corresponding graph.

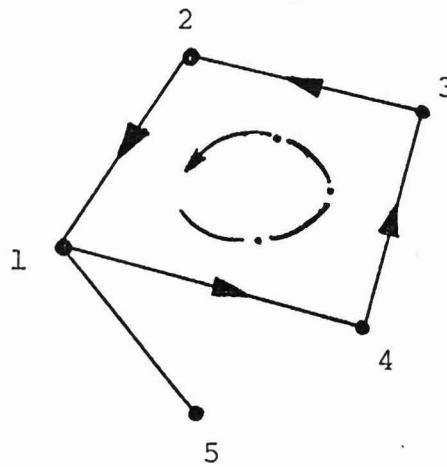
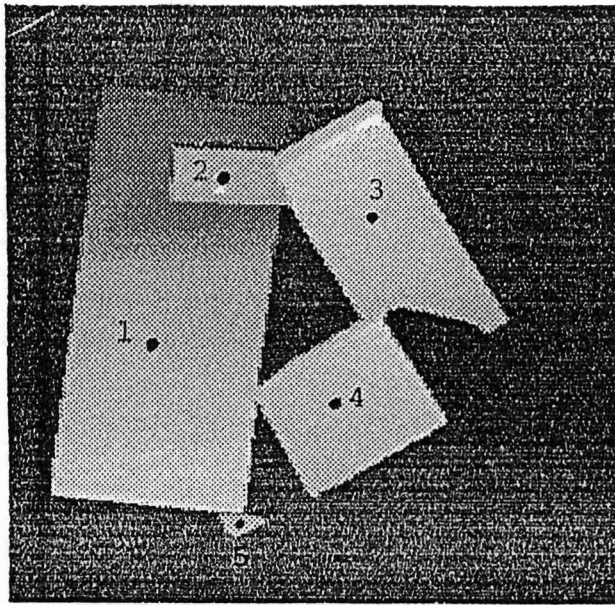


Figure 4. Range Image of an AMBIGUOUS Scene and the corresponding graph.

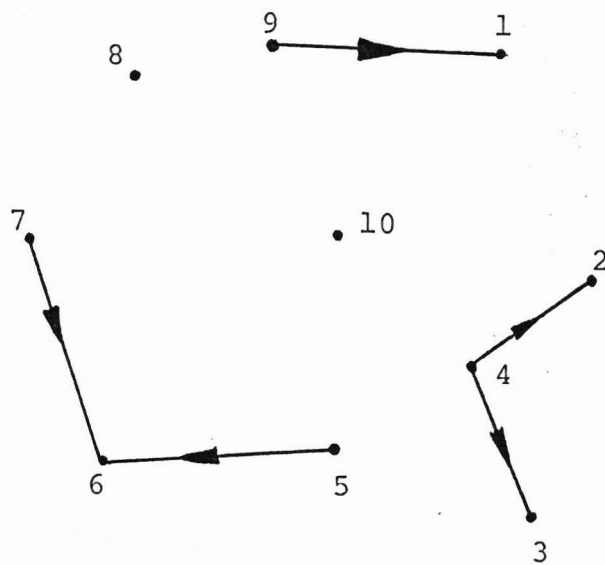
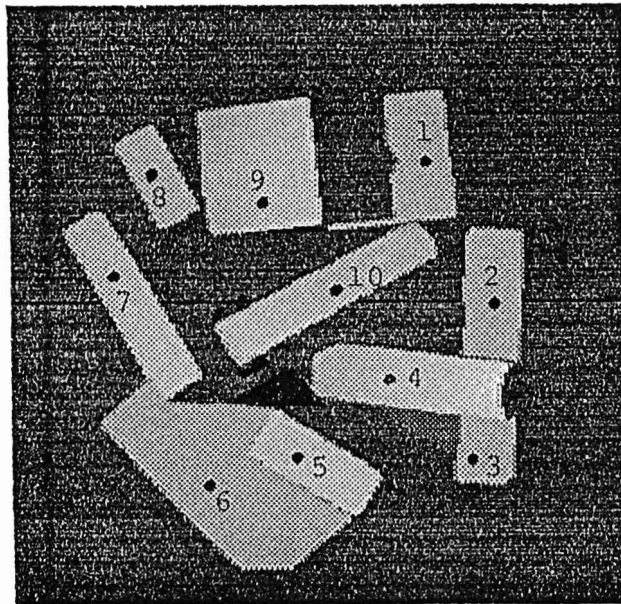
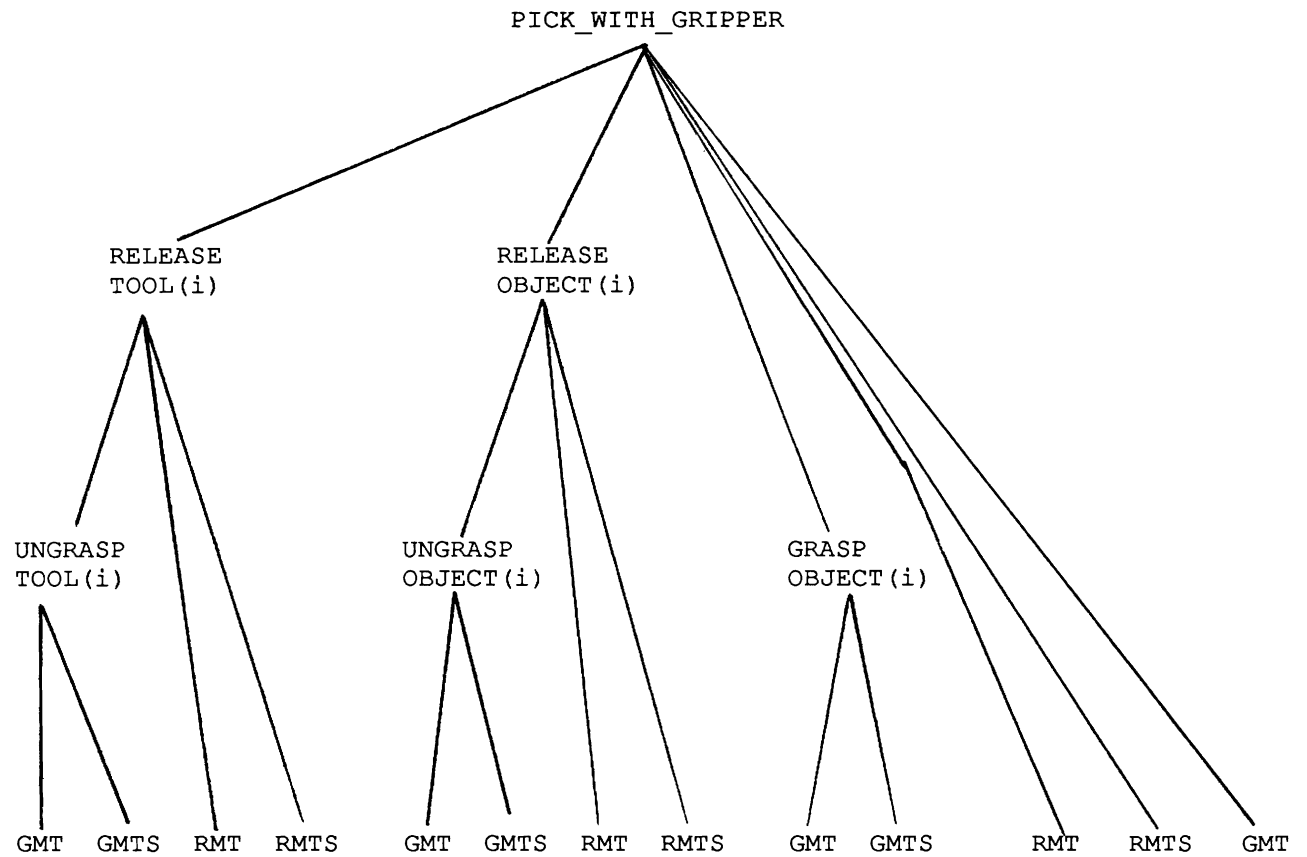


Figure 5. Range Image of an UNSTABLE Scene and the corresponding graph.



Where: RMT = Robot\_Move\_To Action.  
 RMTS = Robot\_Move\_To\_While\_Sensing Action.  
 GMT = Gripper\_Move\_To Action.  
 GMTS = Gripper\_Move\_To\_While\_Sensing Action.

Figure 6. An example of the Action Hierarchy

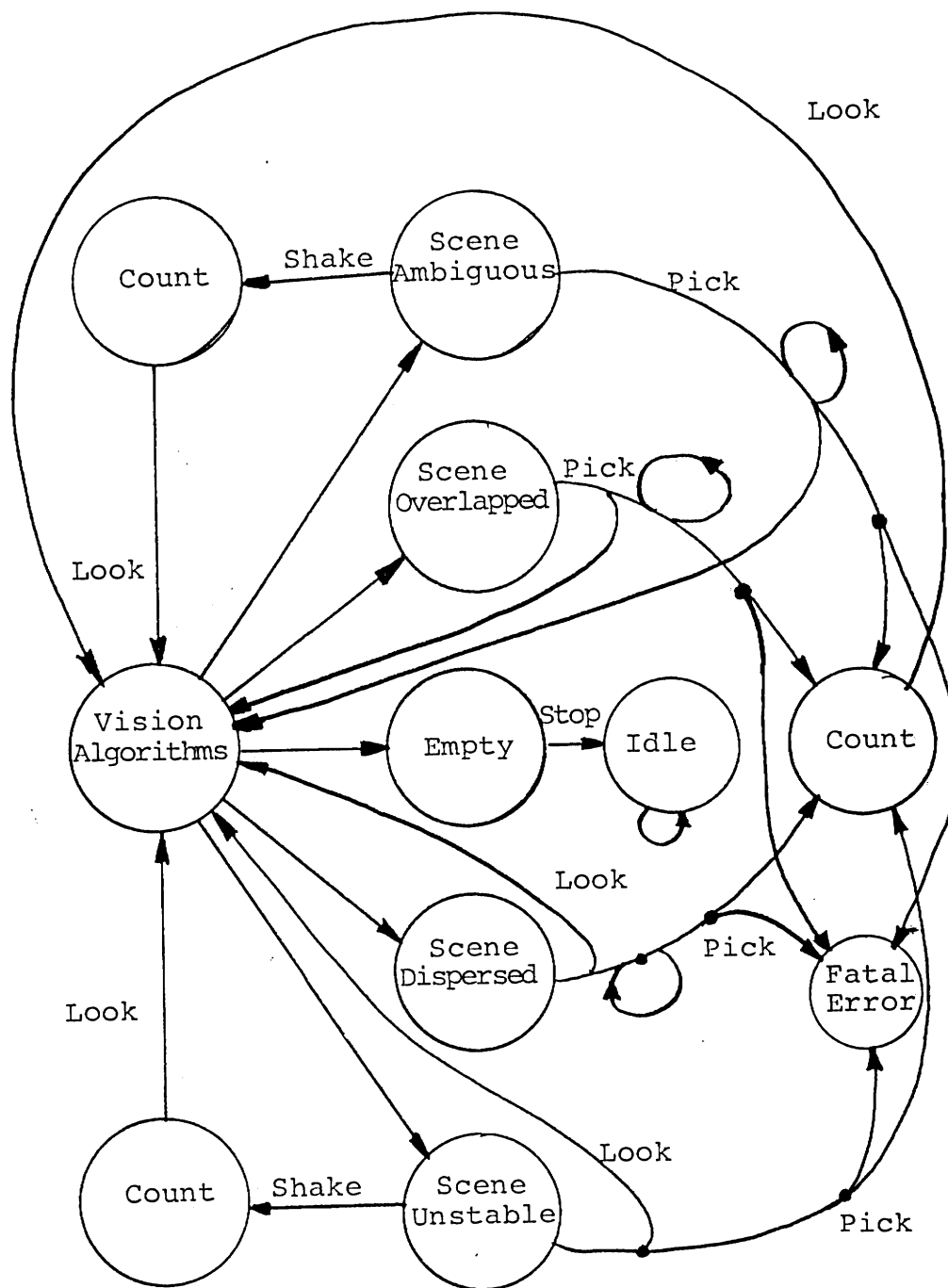


Figure 7. Strategy-1 Action Automaton  
(Look, Pick, Look, ... )

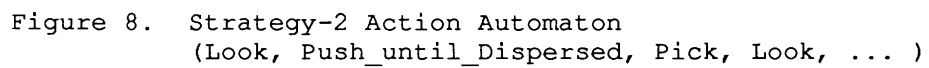
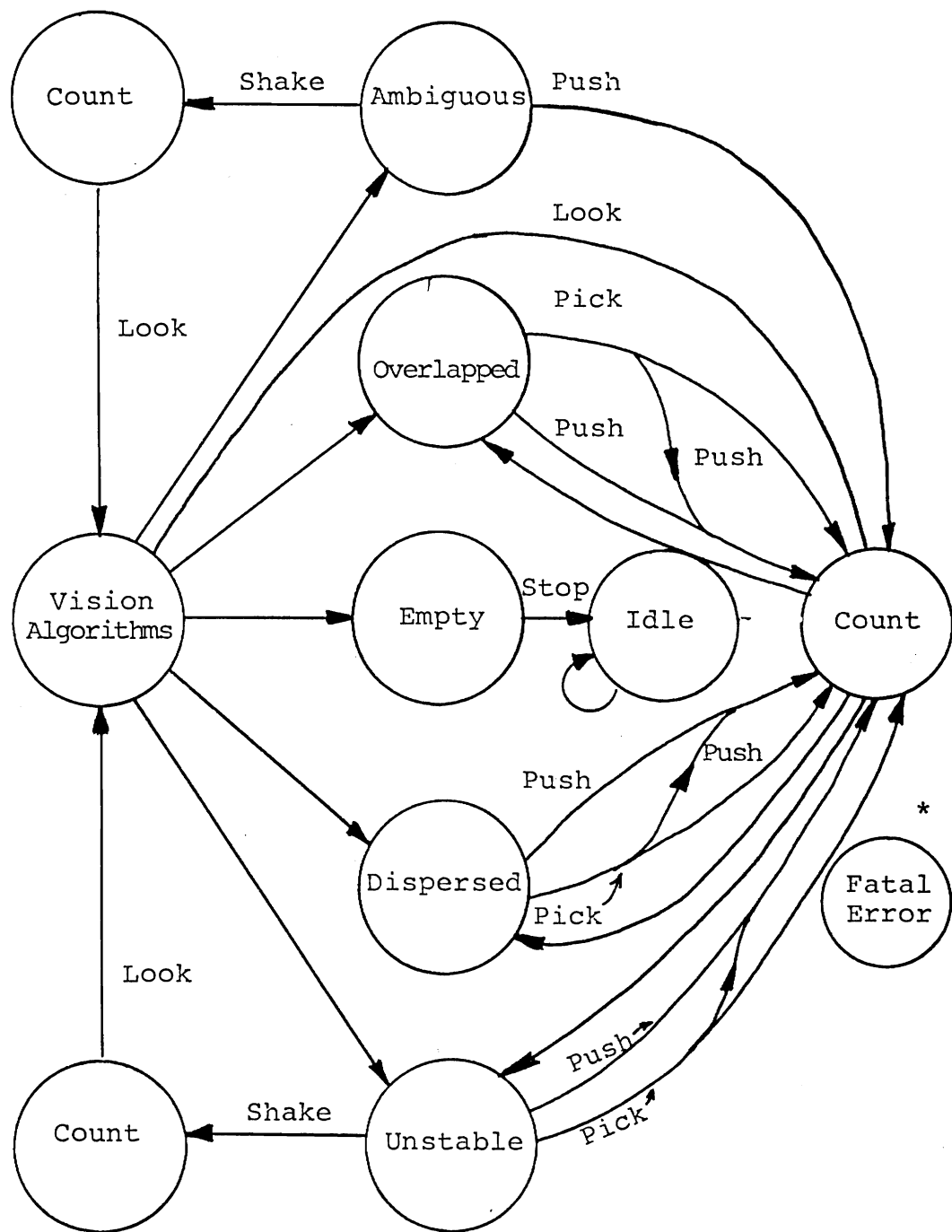
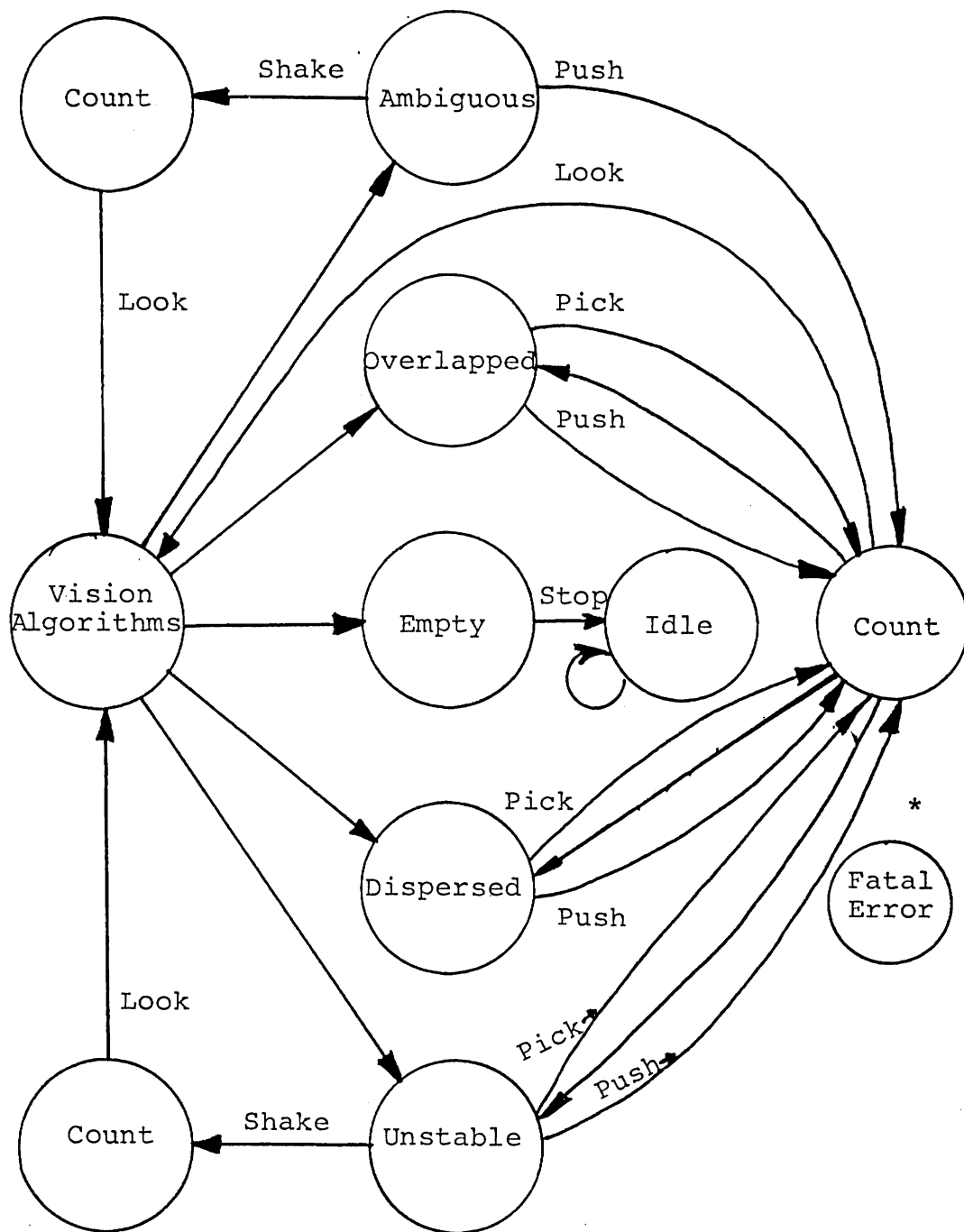


Figure 8. Strategy-2 Action Automaton  
(Look, Push\_until\_Dispersed, Pick, Look, ... )



\* Error Recovery Actions not shown.

Figure 9. Strategy-3 Action Automaton  
(Look, Pick/Push, Look, ... )



\* Error Recovery Actions not Shown.

Figure 10. Strategy-4 Action Automaton  
(Look, Push\_Partially\_Visible, Pick, Push, Look, ... )

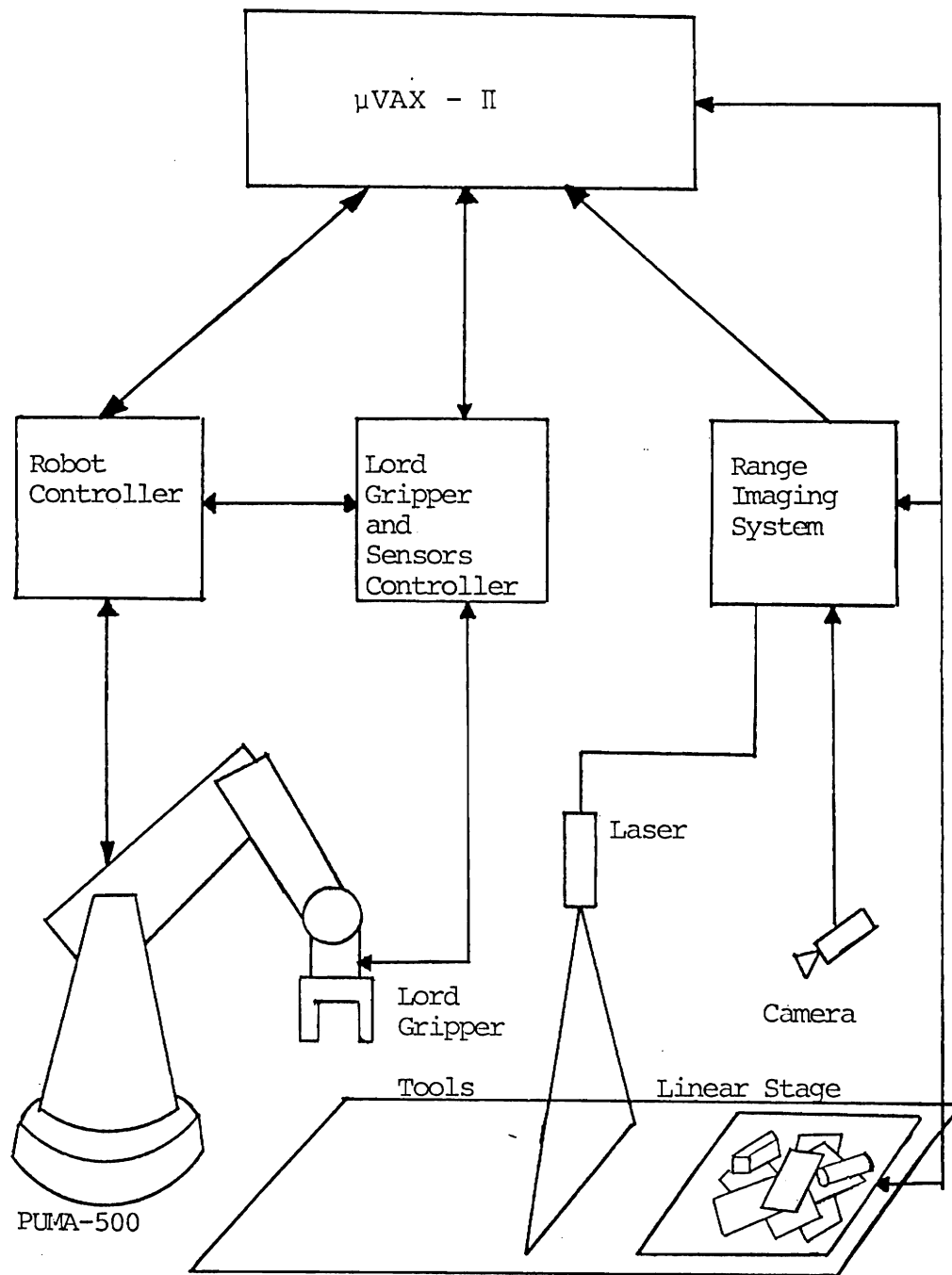


Figure 11. Experimental System Block Diagram.

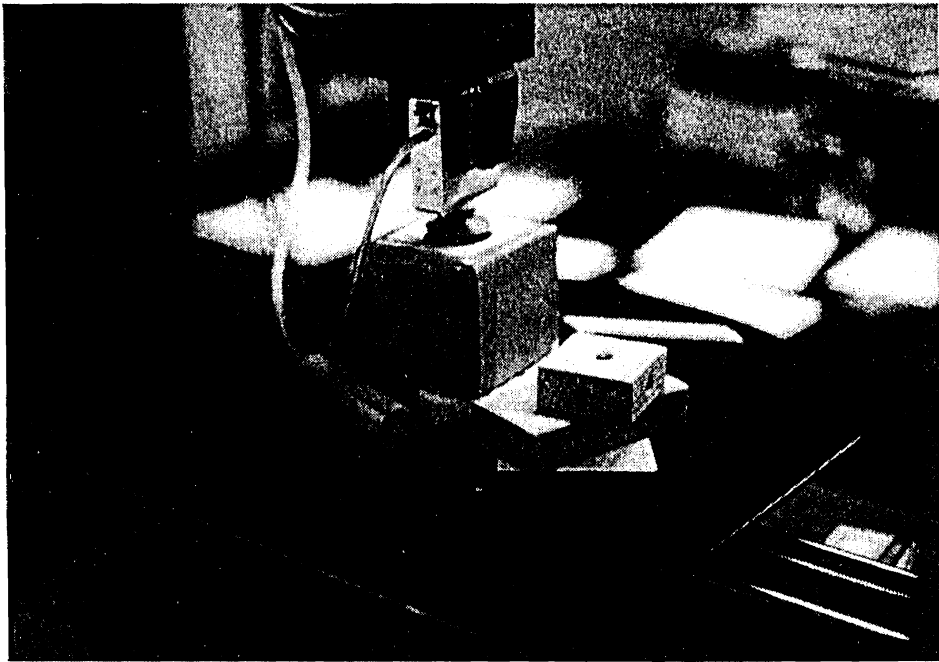


Figure 12. An Example of the "PICK" Action.