pre-prints,
4th Int Symp Robotics Research
August, 1987
Santa Barbara,
Robert Bolles and Bernard Roth
eds. MIT Press

Sensor–based manipulation planning as a game with nature.

Russell H. Taylor
IBM T. J. Watson Research Center
Yorktown Heights, NY, 10598

Matthew T. Mason
Carnegie-Mellon University
Pittsburgh, PA 15213

Kenneth Y. Goldberg
Carnegie-Mellon University
Pittsburgh, PA 15213

Abstract

This paper explores a game-theoretic approach to automatic planning of sensor-based robotic manipulation programs. To win the game, the robot must provably attain a specified task state. The robot moves by choosing a control signal, and nature moves by choosing a sensor signal. Planning is accomplished by searching the game tree. In some task domains, the approach provides a straightforward method of reasoning about uncertainty, non-deterministic actions, and imperfect sensors. We demonstrate the approach in two different task domains: orienting an object using an instrumented tilting tray; and orienting and and grasping an object with an instrumented parallel-jaw gripper.

1 Introduction

This paper approaches robotic manipulation as a game being played between the robot and nature. To win the game, the robot must attain a task state that provably satisfies a specified goal, such as uniquely orienting a polygonal object or achieving a stable grasp. The robot chooses the motor signals, and nature chooses the sensor data. To plan for the worst case, the robot must search the game tree for a winning strategy.

Our approach treats manipulation planning as a tree search. Uncertainty in the robot's world model, error in the robot's actions, and noisy sensors, are all explicitly modeled and accounted for in the construction of the tree. We are implementing planners for two different task domains. First, to extend Erdmann and Mason's (1986) work on orienting objects with tilting trays, we have incorporated the use of sensory feedback. Second, based on Brost's (1986) earlier work on planning parallel-jaw grasping motions, we have implemented a system that plans a sequence of squeezes to orient and ultimately grasp an object, using measurements of finger separation when appropriate.

The rules of the manipulation game are determined by the actions and sensory events that are possible from any given state of the task. To avoid combinatorial explosion, it is imperative that we consider only those actions and sensory events that have different effects. In some cases, it is possible to reduce the number of alternatives from infinity to a small finite number without compromising the planner's scope.

1.1 Previous work

Our approach originates in the desire to construct plans that model uncertainty in the robot's model of the world. Taylor (1976) describes a system that predicts possibile uncertainties, using them to choose among alternative strategies, and to adjust the parameters of the chosen strategy. Brooks' (1982) system verifies and re-works robot plans, based on bounds on uncertainty. Brooks' work can be viewed as complementary to the present paper, since Brooks' plans use sensory information in a continuous fashion, rather than for conditional branching. Our plans use sensory data exclusively for conditional branching.

A formal framework for planning in the presence of uncertainty is developed in Lozano-Pérez, Mason, and Taylor (1984), Mason (1984), Erdmann (1986), Buckley (1987), and Donald (1986). Buckley provides the link to the present paper by his use of AND/OR graphs. Buckley applied the approach to planning of generalized damper and generalized spring motions. The present paper adopts Buckley's notation for AND/OR graphs, explores the general application of the idea, and illustrates the concept with implementations in two task domains.

The present paper describes planners for two different task domains: *tray-tilting* and *squeeze-grasping*. Tray-tilting is the process of orienting a planar object in a tray by moving the tray through a sequence of tilt angles, and squeeze-grasping is the orienting and grasping of a planar object by a sequence of squeezes with a parallel-jaw gripper. The first work on tray-tilting was Grossman and Blasgen (1975), followed by Erdmann and Mason (1986). For earlier work on the mechanics of pushing, and squeeze-grasping in particular, see Mason (1986), Mani and Wilson (1984), Brost (1986), and Peshkin (1985).

Our work is also related to the problem of choosing tactile probe motions to determine the position and orientation of an object (Grossman and Blasgen 1975, Grimson and Lozano-Pérez 1984, Grimson 1986, Ellis 1987). The problem is easily cast as an AND/OR graph search. Ellis' *ambiguity tree* is similar, but describes the interpretation of probe data, with the probe sequence fixed, yielding a pure AND tree.

2 Example

We will begin with a simple example, contrived to illustrate the basic concepts and avoid the many complications. Consider a rectangular object in the plane, constrained to have one of its edges aligned with a fixed reference line. Due to the symmetry of the block, we will distinguish two states, SHORT and TALL. The robot can rotate the block through 90 degrees, which turns a SHORT into a TALL, and vice versa. An optical interrupt type sensor is mounted so as to detect the block's state (see Figure 1).

We can summarize the options of the robot as follows: at any given time it can decide to rotate the block to its other state, an action we

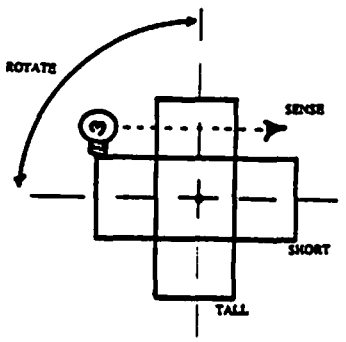

Figure 1: An illustrative example.

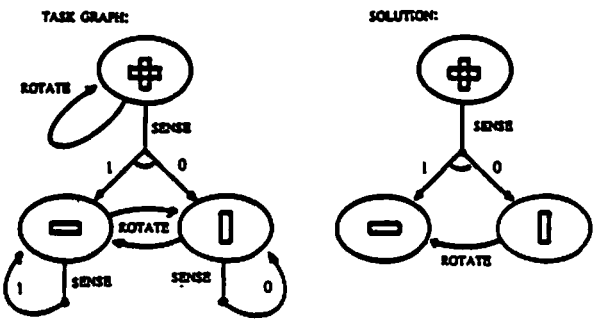

Figure 2: Task graph and solution sub-graph for the example.

will call ROTATE, or it can consult the sensor, obtaining either a 0 or a 1 depending on whether the light beam is broken, or not, respectively. We will assume that the block's state is initially unknown, and the goal is to force the block into the SHORT state. Starting with the rectangle in either of two orientations, we construct a graph (Figure 2) that captures every possible sequence of events in the task. An initial ROTATE changes nothing, the object state is still unknown. An initial SENSE determines the object state, after which it can be meaningfully ROTATEed. Additional SENSEs are redundant. A strategy to accomplish the given task can be expressed as a sub-graph of the task graph, as shown in the figure.

The key features of the task graph are:

- Each node is a set of possible world states.
- Each action, and each sensory event, is a transition from a set of possible initial world states to a set of possible resultant world states.
- The graph has an AND/OR structure. The robot can choose its actions, and can guarantee a win if *any* of its actions lead to a win. But nature chooses the sensory outcomes, so the planner must ensure that *all* sensory outcomes lead to a win.

To represent model uncertainty, each node corresponds to a set of possible world states, not a specific world state. Hence a search node does not represent a state of the world, but rather the state of the robot's model of the world during plan execution (Brooks 1982; Lozano-Pérez, Mason, and Taylor 1984). The distinction is subtle, but important. It is useless to achieve a desired state of the world in ignorance; the robot must know that the goal is achieved. Hence we search not for a specified world state, but for a specified model state. In principle, we can encode the state of the robot's world model, i.e. summarize all information available to the robot, as the set of states consistent with this information. Hence our search nodes are defined by a set of possible states of the world.

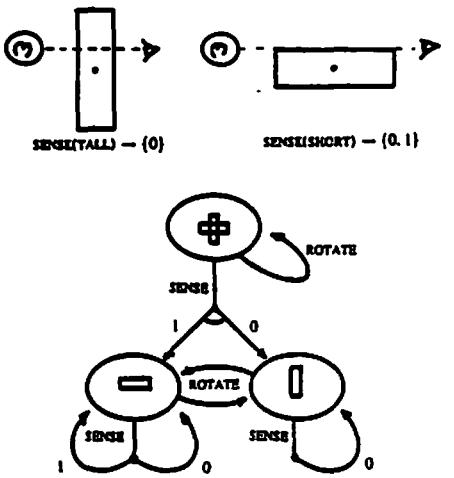

Figure 3: The example with ambiguous sensing.

For many problems, it may be impossible, or inadvisable, to boil the robot's world model down to a set of all possible world states. For example, we might have a system that calibrates the location of a pallet, using touch probe information obtained through a sequence of guarded moves. The sensible thing is to collect the information, then apply a single routine that predicts the pallet location, rather than computing a set of all possible pallet locations after each probe. To express such strategies in a graph, we could augment the state encoded at a node to fully represent the computational state of the robot's world model (Brooks 1982).

Besides representing uncertain knowledge of the state of the world, we have to represent and reason about under-determined actions and noisy sensors. For instance, we can modify our earlier example by introducing noise in our sensory data. Suppose that the optical sensor's location is not precisely known, with the consequence that a broken beam is inconclusive, although an unbroken beam still implies a state of SHORT. The resulting graph is shown in Figure 3. In this case, there is no winning strategy, because the sensor might never give the robot any information. This example shows a shortcoming of our worst-case approach to planning. We might want to consider a strategy that tries to get lucky, i.e. hopes that nature cooperates. In principle, the graphs can be extended using probabilistic models of actions and sensors to construct probabilistic plans.

When we turn our attention from contrived examples to more realistic task domains, planning becomes more complicated. The example above is contrived to have a finite number of world states, a finite number of actions, and a finite number of sensory events. In more realistic problems, we often must deal with continuous spaces of task state, action, and sensory data. Modeling uncertainty, actions, and sensing, are accordingly more complicated. The most striking difference is the combinatorial nature of the search, especially when branching factors of the order of the continuum are contemplated. This paper deals with continuous spaces of choices in two ways. First, we partition a continuous space of actions into classes that have identical effect. Continuous spaces of action were handled thus by Brost (1986) and by Erdmann and Mason (1986). Second, we deal with continuous spaces of sensory events by sampling. There are many variations; we simply sample the space at a fixed, fairly coarse, resolution. Other possibilities are discussed later in the paper.

Taylor -- 3

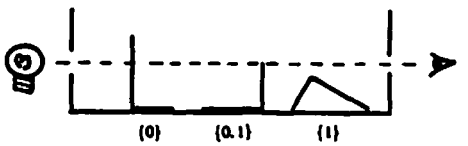

Figure 4: The sensor model.

3 Graphs and searching

A planning problem is defined by the task state space, the sensory and action functions, a set of possible initial states, and a set of goals, with each goal expressed as a set of states. We adopt the following notation:

X	the task state space, i.e. the set of possible states of the physical world.
D	the sensor space, i.e. the set of possible sensor data.
U	the action space, i.e. the set of possible actions.
$I \subset X$	the set of possible initial states.
$G_i \subset X$	the ith goal.
$a : U \times X \rightarrow \mathcal{P}(X)$	the task state-transition function, expressing the effect of actions.
$s : X \rightarrow \mathcal{P}(D)$	the sensor function, expressing the meaning of sensory data.

where $\mathcal{P}(\cdot)$ is the power set, to express the multi-valued character of actions and measurements.

Each planning problem is transformed into a graph search problem. A node in the graph is a set of possible task states, i.e. a subset of X. A node encodes the robot's knowledge of the task state during execution of the plan. Hence the initial node is the set of possible initial states I, and success is achieved by reaching any subset of any goal G_i. Actions and sensory events cause transitions among the nodes of the graph:

$V = \mathcal{P}(X)$	the nodes of the task graph
$\xrightarrow{u}$	a relation on the nodes V, defining the action arcs for action u. $X_1 \xrightarrow{u} X_2$ iff $a : (u, X_1) = X_2$.
$\xrightarrow{d}$	a relation on the nodes V, defining the sensing arcs for sensor datum d. $X_1 \xrightarrow{d} X_2$ iff $s^{-1}(d) \cap X_1 = X_2$.

Here, and throughout the paper, we are treating action and sensing as mutually exclusive concepts. Some problems, though, treat action and sensing as indivisible. For example, the guarded moves described by Lozano-Pérez, Mason, and Taylor (1984) involve choices by the robot and by nature. Buckley's formulation of the AND/OR graph uses sensor data interpreted during the motion to define alternative result nodes for the motion. Another example combining choices by the robot with choices by nature is when the robot has two different sensors to choose between. Our formulation of the graph is easily generalized to deal with such situations, but this paper focuses on the extreme cases: pure sensing and pure action.

A solution to a planning problem can be described by a subgraph of the task graph. The subgraph should lead to a goal in finite time, so cycles and infinite subgraphs are not allowed. Each node (except leaf nodes) should include exactly one outgoing action arc, or *all* of the outgoing sensing arcs, reflecting the AND/OR structure of the problem. We will seek solutions that minimize the worst-case number of arcs, i.e. we seek the solution sub-graph that minimizes the maximum path-length. We presently employ a breadth-first search, hashing the nodes for efficiency. Simple searches converge in a few seconds on a Symbolics 3600.

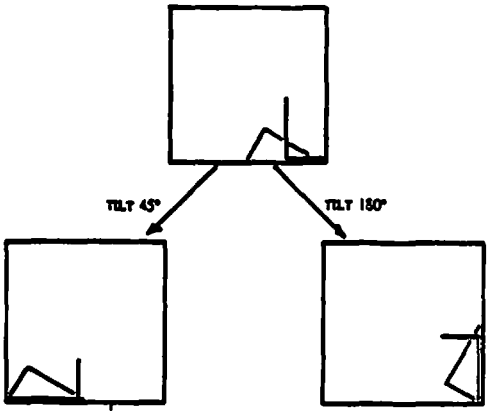

Figure 5: A typical set of tilt actions (Erdmann and Mason 1986).

4 Automatic planning of sensor-based tray-tiltin programs

Our first application is the problem of orienting an object with an instrumented tilting tray. A polygonal object slides freely in a rectangular tray, which is instrumented with simple optical-interrupt sensors. The robot can tilt the tray at any angle desired and query the sensors. The object slides and rolls along the sides and into and out of the corners. Starting from an initially unknown orientation, the problem is to obtain any completely determined orientation and position. In the modeling of the mechanics of tray-tilting we follow Erdmann and Mason (1986), which should be consulted for more details. Briefly:

- A rigid polygonal object, with known shape and center of mass, slides about in a rectangular planar tray.
- The forces acting on the object are gravity, forces of constraint, and friction. Coulomb friction occurs between the object and the tray walls. Friction with the tray bottom is negligible.
- The system is quasi-static: inertial and impact forces are negligible.
- The object is initially in one of a finite number of stable orientations, in the middle of one of the tray walls.
- The attitude of the tray is directly controllable.

To complete the problem definition, we must define the allowed sensory modalities:

- A light beam passes parallel to each wall. The sensor reports whether the beam is broken or not.
- The distance from the beam to the wall lies in some known interval.

For any given orientation of the object against some wall, there are three possibilities, illustrated in Figure 4. If the object's profile is definitely less than the sensor's distance from the wall, a 1 is obtained. If the object's profile is definitely greater than the sensor's distance from the wall, a 0 is obtained. If the object's profile falls inside the interval of possible sensor distances, we cannot predict which sensor datum will occur.

The first problem is to characterize the possible states of the task. For the tray-tilting task domain, we can get by with a finite number of possible configurations. By definition, the initial set of configurations is finite. If

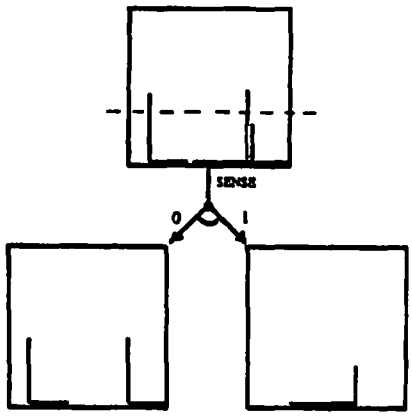

Figure 6: A typical set of sensor events.

we avoid motions that move the object into the center of the tray, and if we wait long enough for the object to complete its motion, then the number of possible configurations remains finite.

A typical tilting action is shown in Figure 5. Initially there are two possible configurations of the block in the tray. Two different choices of tilt angle are shown, with the corresponding result nodes. Searching the graph requires that the effect of a tilt action be predicted automatically, using the methods (in fact, using the actual code) described in Erdmann and Mason (1986) and Erdmann (1984). An interesting type of indeterminacy enters when Coulomb friction and rigid bodies are allowed in Newtonian mechanics. Multiple solutions are possible, and there may even be situations that admit no solutions.[1] Erdmann's system finds multiple solutions. Since the nodes of the task graph are represented by a set of world states, multiple solutions are easily accommodated in the task graph. We have not observed any cases without solutions, and do not know whether they can occur. The result node would be the empty set, i.e. there are no world states consistent with such an action. The meaning of such a situation is that our assumptions of the mechanics of the world are inconsistent, and hence our planner is invalid. This is one case of what we have come to call "confusing nodes": a node which is in some way beyond the scope of the planner, and must perforce be avoided.

Figure 6 shows a typical sensory operation. If the beam is broken, we construct the result node by eliminating every configuration in the initial node that is too short to break the beam. If the beam is not broken, the result node is obtained by eliminating every configuration that is too tall to not break the beam. Some configurations, on the fuzzy edge between short and tall, will appear in both resultant nodes. The most straightforward implementation of sensor interpretation is to begin by computing off-line, for every possible sensor datum, the set of world states consistent with that datum. Now, during the search, we can interpret any hypothetical sensor datum by intersecting the set of possible states with those consistent with the sensor datum, as previously computed. This is only feasible with very simple sensors, such as our optical interrupt sensor. At the opposite extreme, computer vision, for instance, it is not practical to enumerate and interpret every possible image, even if we do it off-line!

It should be apparent that the demonstration system is not limited to the optical interrupt sensors, which were introduced for concreteness. In fact, the available sensors are actually described to the system by their interpretations, i.e. by a mapping from some sensor data space to sets of consistent world states.

[1]The existence of problems admitting no solution consistent with Newton's laws, rigid bodies, and Coulomb friction, has been widely accepted for some time. Lötstedt (1981) for example, describes such a problem, but see Mason and Wang (1987) for a solution to Lötstedt's problem.

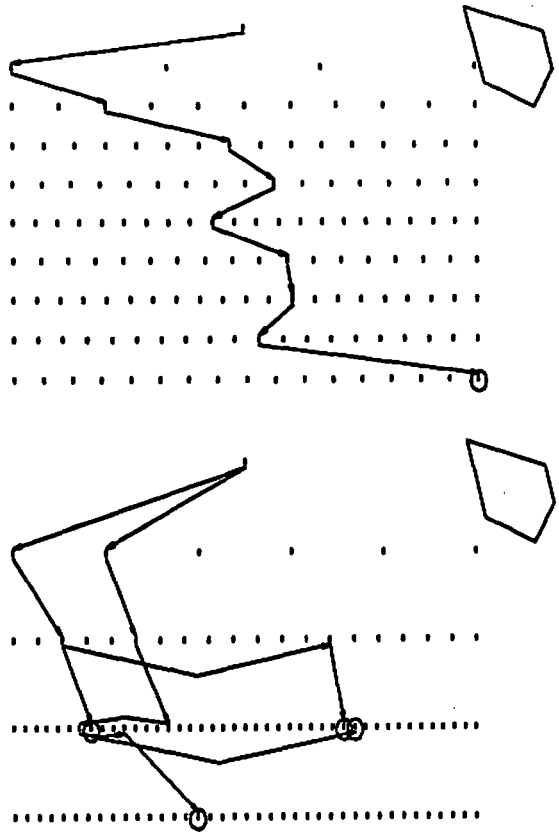

Figure 7: Typical solutions, with sensing (below) and without (above).

At this writing, the planner seems to be running reliably, but we have not tested it systematically, nor have we tested the plans generated by the planner. A typical case is shown in Figure 7, which shows two plans, one with sensing and one without. The goal nodes are circled. All arcs other than those in the solution have been suppressed for readability. For each node, one outgoing arc indicates an action; two outgoing arcs indicates a sensory operation. The planner seems to work very well on randomly-generated polygons, although there are some polygons that are not orientable.

5 Automatic planning of sensor-based squeeze-grasping programs

Our second example application is the problem of planning a sequence of squeezes to completely orient and grasp an object, with sensory data arising from measurement of the finger separation at the completion of a squeeze. The mechanics closely follows the work of Brost (1986), which addresses the problem of planning a single squeeze without sensory feedback. Our assumptions are:

- The object is a rigid planar polygon, in planar motion.
- The two fingers are infinitely broad, rigid, parallel half-planes, approaching from arbitrarily far away.
- The finger motions are symmetric: the tangential components of velocity are equal, and the normal components are opposite, but equal in magnitude. The motions continue until further motion would imply distortion of the object.

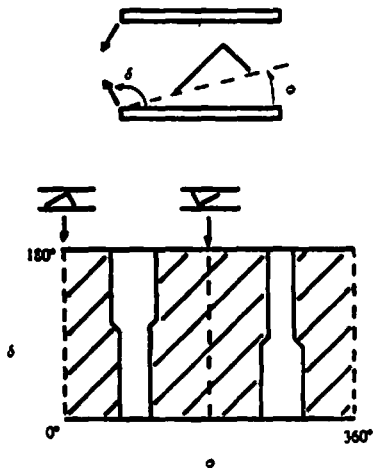

Figure 8: A squeeze-grasp diagram (Brost 1986).

- The forces acting on the object are gravity, normal to the support plane, contact with the support plane, and contacts with the fingers. All contact forces observe Coulomb's law. We assume the static and dynamic coefficients of friction are equal.
- The system is quasi-static: inertial and impact forces are negligible.

Brost assumes that the initial object orientation falls within some known interval of angles, that the coefficient of friction is known to fall in some known interval, and that the angle of finger motion is controlled to within known tolerances. These assumptions result in a two-parameter family of actions—only the finger orientation, relative to the object, and the direction of finger motions, measured as an angle relative to one of the finger's faces, are necessary to predict the object's motion. Brost's analysis leads to simple diagrams, partitioning the two-parameter family of possible actions into equivalence classes that will rotate the object to a completely determined orientation (see Figure 8).

We will extend Brost's results in two ways. First, we have to extend the planning from a single action to sequences of actions. Second, we have to model the sensory operations and incorporate them in the planning. To simplify the constructions we will assume that the coefficient of friction is known exactly, and that the finger motions can be controlled perfectly. To incorporate Brost's methods dealing with these errors would distract us from our main goal, which is to incorporate sensor-based strategies.

To plan sequences of squeezes, we have to represent sets of orientations more general than Brost's intervals. For example, let the initial orientation of a rectangle be completely unconstrained, and consider the orientations possible after one squeeze, represented as a set of points on the unit circle. Typically, there will be four isolated points, corresponding to the four aligned orientations of the rectangle, and four intervals, corresponding to four possible cocked configurations. We will model the set of possible orientations by a finite set of intervals on the unit circle, including singular intervals.

In considering sequences, rather than isolated squeezes, we introduce another complication. Brost's diagrams, such as the diagram of Figure 8, identified regions that would rotate the object to a single determined orientation. However, we have to analyze and identify regions of weaker operations, which do not orient the object by themselves, but which might be useful in a sequence. Figure 9 shows a diagram that is similar to Brost's diagram and is for the most part derived from Brost's analysis, but which identifies all distinct operations, not just those that orient the object completely and immediately.

In order to reduce the number of choices of actions from a continuum to a finite number, we need to identify entire classes that can be represented as a single choice. Both Brost (1986) and Erdmann and Mason (1986) partitioned their action spaces into a finite number of equivalence classes. In Figure 9 regions *A*, *C*, *G* and *E* are equivalence classes, i.e. every action gives the same result and can be represented by a single node apiece. Regions *B*, *D*, *F* and *H* illustrate a refinement on the use of equivalence classes. Under certain conditions discussed later, we can prune nodes that are proper supersets of other nodes. For example, the continuum of nodes arising from actions in region *B* are all supersets of the nodes arising from regions *A* and *E*, and can be pruned.

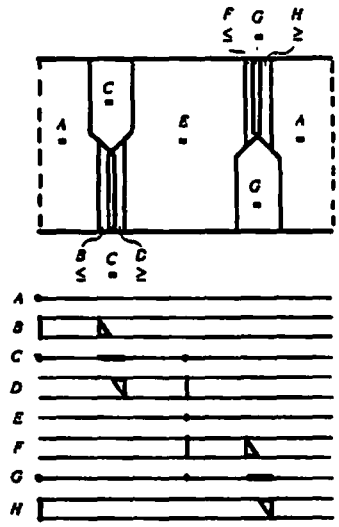

Figure 9: The partitioned space of squeezes.

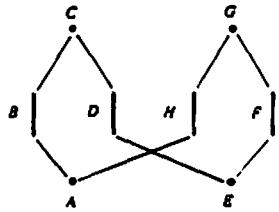

Figure 10: The partial order holding among the squeezes of Figure 9.

The regions of Figure 9 are labeled to indicate whether they are equivalence classes, or whether a subset relation holds. Each region labeled "=" is an equivalence class, and each region labeled "$\leq_i$" or "$\geq_i$" is totally ordered. Both of these types of region can be represented by a single node in the graph. Although we have not encountered one yet, there can also be unordered classes of actions, which in principle should not be pruned. There are several approaches to unordered continua of actions; at present we are simply pruning them. Depending on our results, we may sample them. The ordering among the actions of Figure 9 is shown in Figure 10. It is a partial order, with two minimal elements. If we encountered this situation in a search, we would consider the two minimal actions only.

This leaves us with the problem of modelling sensory operations. We assume that the finger separation sensor measures the distance between the fingers with a tolerance of ± 1 mm.[2] To interpret such data, consider the typical construction of Figure 11, showing finger separation as a function of object orientation. For a measured finger separation of d, the true finger separation lies in an interval $[d-1, d+1]$ mm, and the feasible

[2]The sensor is implemented by a Sony Magnascale position transducer that is part of the Lord hand. The real sensor has micron resolution, and will no doubt have accuracy much better than ± 1 mm.

Taylor -- 6a

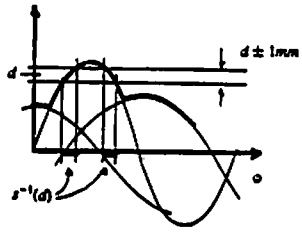

Figure 11: For the triangle, finger separation as a function of object orientation.

object orientations are obtained by inverting the function shown in the diagram.

During the planning stage, this sensory model leads to an infinite branching factor, and needs to be amended. We will divide the sensor range into m equal intervals, and will ignore any information beyond this coarse sampling of the sensory data. Other possible approaches are discussed later in the paper.

At the time of this writing, we have implemented the computation that predicts the outcome of a tilting operation, but not the sensory modeling component.

6 Continua of actions and sensor operations

A key property of task graphs is to have a finite branching factor, and, in practice, this branching factor should be as small as possible. Obviously, a finite branching factor occurs when the sensor data space D and the action space U are finite. A finite branching factor is also obtained when the task state space X is finite, since there are only a finite number of possible search nodes. Even when the sensor space, action space, and state space are all continuous, it may be possible to obtain small finite branching factors. The key method is to define a *dominance* relation among alternative actions. The dominance relation is defined as follows: for a given search node X_i, we say that an action u dominates an action v, $u \leq_i v$, iff $a(u, X_i) \subseteq a(v, X_i)$. If there is a finite set of actions that dominate the entire class of actions U, then we have a finite branching factor at node X_i.

Figure 12 illustrates different types of dominance relations among a continuum of actions, for a given search node X_i. For each action u, we have plotted the corresponding result X_u. In this case, the action space U falls into three continua, one comprised of equivalent actions, one with the actions ordered, and one with the actions unordered. We can choose any action from the class of equivalent actions, and we can choose the minimal action from the ordered actions, but each action in the unordered continuum is different, and potentially useful.

Although we have had no occasion to use it, a similar construction exists for alternative sensory events, with the sense of the relation reversed. From a node X_i, we will say that sensory datum d dominates sensory datum e iff $s^{-1}(d) \cap X_i \supseteq s^{-1}(e) \cap X_i$, written $d \geq_i e$. The reason for the reversed sense is that we are dealing with the opponent's move. In the case of action, we can choose the best, i.e. most specific outcome. In the case of sensing, we must anticipate the worst, i.e. most ambiguous outcome. For action, we seek a finite set of minimal nodes. For sensing, we seek a finite set of maximal nodes.

It is evident that pruning away dominated nodes does not affect the existence of a solution sub-graph. Suppose we are contemplating two different actions u and v from a given node, with results X_u and X_v, and suppose that $u \leq_i v$, i.e. $X_u \subseteq X_v$. The relation between the two nodes is that X_u is *more specific* than X_v, i.e. the robot has *more information* at node X_u. If there is a plan starting from X_v, then there is also a plan, indeed, the same plan, starting from the more specific node X_u. Hence, pruning X_v will not affect the existence of a plan.

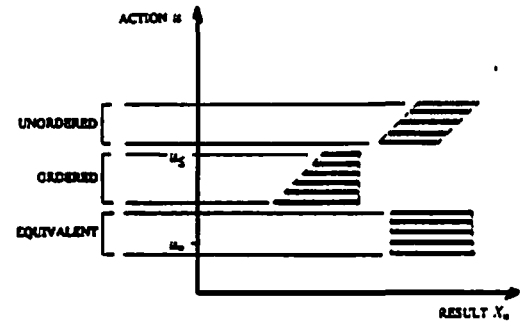

Figure 12: Equivalent, ordered, and unordered continua of actions.

A more difficult question is whether pruning a dominated node affects the search for, rather than the existence of, a solution sub-graph. We require our search procedure to have the following property:

> If a search from node X_v succeeds, and if $X_u \subset X_v$, then a search from node X_u will also succeed.

I.e., we require that our planner cannot be foiled by giving it more specific pre-conditions. We refer to this property as "pre-condition monotonicity"—the scope of the planner is monotonic non-decreasing as the pre-conditions become more specific.

Pre-condition monotonicity may seem to be a property that any respectable planner would have, but it is not as trivial a property as it may seem. Exhaustive breadth-first search, pruning dominated nodes, would seem to have this property, but not if an unordered continuum of nodes occurs. Suppose state A leads to a continuum of result states, and suppose state B leads to the union taken over this continuum, no matter which action is applied. Suppose that a solution is easily found from the union. Starting from node $\{A, B\}$, the search would obtain a single node, the union of the continuum, and would proceed easily to a solution. Starting from node $\{A\}$, however, the search would obtain the continuum, and never converge on the solution.

The searches implemented for the tray-tilting and squeezing domains depart from the ideal exhaustive search, by neglecting certain nodes, which we have been referring to as "confusing nodes". There are four different situations that give rise to confusing nodes:

- Due to the possibility of inconsistencies in Newtonian mechanics with Coulomb friction and rigid bodies, we may someday generate a null node. By definition, the empty set is a subset of all sets, hence the null sequence satisfies any goal, and pre-condition monotonicity holds. There is a paradox, though, because this reasoning suggests that a planner should steer the task state toward inconsistencies in Newton+Coulomb+rigid body mechanics. The philosophical difficulties apply not only to nodes that have links to the null node, but to supersets of nodes (other than the null node) that have links to the null node. Presumably our theory doesn't tell us all possible outcomes, in certain degenerate situations. (Either that, or there are states of the physical world that truly have no possible outcomes, which should surely be avoided at all costs.) We ought to avoid situations where our theory *might* be deficient, not just situations where our theory is *certain* to be deficient.
- Our representation of the tray-tilting task state requires that all possible configurations at a node have contact with a common wall. If

an action would lead to a *multi-wall node* that violates this restriction, the node is labeled "confusing," and pruned. A multi-wall node is more ambiguous than its single-wall subset nodes. If an initial node leads to a multi-wall result, a more ambiguous initial node leads to a more ambiguous multi-wall result, so pre-condition monotonicity seems to hold.

- The tray-tilting planner avoids sliding an object into the center of the tray, losing all contacts. If a sequence of actions leads from an initial node to a no-contact node, the same will occur starting from a more ambiguous initial node. Pre-condition monotonicity seems to hold.
- In the squeezing domain, we prune all the nodes in an unordered continuum of actions. The same example we constructed above, to illustrate the limitations of exhaustive breadth-first search, can also be applied to show that a search that prunes the nodes of an unordered continuum does not satisfy pre-condition monotonicity.

This leaves us in an interesting position. We are justified in pruning dominated nodes, as long as the pruning in every instance produces a finite branching factor. If an un-pruneable continuum arises at any point during a search, our earlier prunings might have prevented us from finding a solution. At this point, variant search strategies might be considered, but we have not pursued them yet.

Action and sensing continua are important in many problems. As an extreme example, consider the *shooting gallery problem*: a point bear, either alive or dead, is at some location on the real line. A sensor tells the exact location of the bear, and an action (shooting), given the exact location of the bear, changes the state from alive to dead. The possible initial states are any live state, and the goal state is any dead state. The solution is obvious: look and shoot, aiming the gun at the location returned by the sensor. Unfortunately, following the initial sensory operation we obtain a completely unordered continuum of nodes. Further, we know that this continuum is not full of superfluous nodes—pruning of any nodes in the continuum would lead to an incorrect plans. A number of approaches suggest themselves.

One approach is to allow nodes in the search that do not correspond to a definite set of states, but rather correspond to some undetermined set of states, presumably using symbolic descriptions depending on one or more parameters. We would have a two-step plan: locate the bear, then shoot at the bear's location. The node between the two steps would represent the state of the execution-time model, but with a parameter, representing the bear's location, that would be resolved at execution time, rather than at planning time. Similarly, the shooting action would be expressed in terms of the parameter.

The introduction of such symbolic descriptions into plan graphs corresponds to the introduction of variables into robot programs, and has been studied by Taylor (1976) and Brooks (1982). In our example, the sensing operation would set a runtime variable modelling the bear's position, which would then be passed as a parameter to the gunlaying function. More generally, we must augment our planning states to encode both the *locus* of the bear, (i.e., the set of places it may be at plan execution time) and the *determination* of the bear (i.e., how accurately the runtime variables will model its true location). Within this paradigm, sensing actions modify the determination attributes of planning states and manipulation actions modify both locus and determination attributes. This interrelationship becomes clearer if we consider the *extended shooting gallery problem* in which "looking" returns a short interval of positions in which the bear might be, and the bear will be killed if and only if the gun's aiming point is accurate within some (other) specified interval. The initial locus of the bear is some (long) interval on the real line. In the absence of other information, the determination is the same interval. Looking leaves the bear locus unchanged but reduces the determination interval to that of the sensor. If this interval is short enough, then a single point-and-shoot strategy will suffice, if we can verify that all points in the bear locus are within range of the gun. Otherwise, it will be necessary to fire several shots to cover the determination interval or to use some other actions ("beating the bush") to drive the bear into range. Further complexities arise if the bear is not sedentary, so that shooting and missing may cause him to run away from the shot, thus restricting the locus but possibly lengthening the determination interval.

A more realistic example is the squeeze-grasping problem. Measurements of finger separation lead to unordered continua, which might be more appropriately addressed with program variables. The straightforward approach is to measure the finger separation and use an inverse trigonometric function to determine the possible orientations. If an unambiguous answer is obtained, the hand can simply be rotated to the correct orientation. An ambiguous answer might still require search to further reduce the possible orientations.

A second approach to the shooting-gallery problem is to redefine our actions. We could wrap up the look and shoot sequence by defining a single action, called "point the gun at the bear." In fact, we have done precisely the same thing in our definitions of the tray-tilting and squeeze-grasping domains. In reality, our actions are implemented using sensory data from joint encoders. The servos hide their sensory data from our planner. In effect, this is equivalent to the idea of using a symbolic description of the entire continuum of nodes, but with the hard part done off-line in the definition of the problem.

A third approach is to pursue a more interactive, less contemplative approach to manipulation. If the robot were to just take a look, and then make a plan, the problem is trivial. Sensing is good during execution, and for the same reasons is good during planning: it eliminates lots of states. What isn't clear is why the robot decides to take a look—unless there is a plan at a more abstract level.

The fourth and simplest approach to continua, besides the option of just pruning them, is to sample. This describes a wide variety of search strategies, corresponding to the large variety of different ways of mapping an infinite set into a finite set. Some examples are:

- Ignore the information, and construct a node that is the union over all nodes in the continuum. For a continuum of actions, this means applying any action, and forgetting which one was applied! For a continuum of sensing, the original node is obtained, and is equivalent to doing nothing.
- Fixed resolution sampling. The sensor range or action space is divided into a finite number of regions. For a sensing continuum, the result nodes must be constructed by taking the union over each region. For an action continuum, any action from the region could be selected.
- Variable resolution sampling. The sensor range or action space is first divided into a finite number of regions, but at a later point of the search may be divided into smaller regions.

A useful view of the variable resolution approach is that we begin by using the first bit of the sensor datum (or action), and, if not successful, we try again with the second bit, and so forth. This still allows many different sampling approaches, corresponding to the infinity of possible binary encodings of the sensor range or action space.

7 Stochastic models

There is one extension of our work which might be very instructive. So far, we represent uncertainty in task state by a set of possible tasks. If we introduce probability distributions, we may be able to plan strategies more effectively. More importantly, we may be able to quantify and compare the effectiveness of different manipulation and sensing techniques in solving a task. We will briefly sketch the extension.

To begin, we require an a priori probability distribution, rather than just an initial set of possible states. The origin of the a priori distribution

is context dependent, but in many cases we would probably assume a uniform distribution over our present initial set.

The difficult part is extending the mechanics so that probability distributions are obtained, rather than the set of possible result states. When the mechanics are deterministic, this is straightforward in principle. When the mechanics are non-deterministic, e.g. when a given tilt angle and a completely determined orientation can lead to two different orientations, it will be necessary to assign probabilities to the alternative events. This might be done empirically, it might be founded on analysis of a more detailed model (such as an elastic model of the object in the tray), or it might be hypothesized with no rationale.

The goal criteria could be unchanged, requiring that a goal be achieved with complete uncertainty. Or, if desired, goal satisfaction might be modified to allow specified confidence levels, e.g. a strategy represents a solution if it will orient the object with probability 0.99. Different objective criteria are possible for choosing among alternative solution strategies. Instead of our present minimum worst-case number of operations, we might consider minimizing the expected number of operations.

The chief attraction of this approach is that it allows us to quantify the uncertainty at a node, and to quantify the effect of an action or sensory operation. For each node we would compute an entropy to measure the uncertainty, and for each link we would compute the change in entropy, measuring in bits the reduction (if any) of uncertainty. (See Sanderson (1984) for the use of entropy in robotic assembly.) We would then be in a position to answer such questions as, "how much information does a tilt operation give me?" and "how many bits am I *really* getting from my sensor?"

8 Extensions to identification and shape uncertainty

Our approach of planning is readily extended to problems in object identification and shape uncertainty. We simply extend the task state space X to incorporate the additional variables. A successful search of the graph would result in a sequence of actions and sensor operations terminating at some some singleton node, implying a completely determined object in a completely determined location. For instance, Erdmann and Mason (1986) considered the problem of an Allen wrench in the tilting tray, with its reflection unknown. The two reflections correspond to two different shapes in the plane: a J wrench and an L wrench. If we augment the task state variable a binary state variable, taking on either of the values J or L to indicate the wrench's reflection state, our planner can immediately be applied to determine which type of wrench is present, as well as its location in the tray. The construction is equivalent to Donald's (1986) definition of "generalized configuration-space" to include variables describing possible shape variations.

We can also deal with problems in "configurable" sensors, which are in principle no different from the problems we have already considered. For example, suppose that our tilting-tray's light beam can be moved closer to or further from the wall on command. The task state space X would be augmented with a variable describing the light beam position, which could then be selected to provide the most information.

There are some interesting variations on this. If we hypothesize that sensor re-configurations are applicable only as the first motion, we have a sensor design problem. For any given object, the graph search will choose a sensor location and the orientation program. Similarly, we can pose sensor calibration problems, asking the robot to resolve uncertainty in the sensor model. Suppose the light beam's position is fixed, but unknown. We augment the task state with a variable that reflects the sensor's position, and proceed as before. During the course of execution of the plan, the robot would simultaneously deduce the sensor location and the object orientation.

9 Conclusion

The paradigm of treating robot planning as a game with nature generalizes readily and provides a very powerful framework for reasoning about robot task planning. Our experience with the example problems shows that this approach can be applied fruitfully to real task domains, not just toy problems. For more complex domains the introduction of symbolic relationships and explicit representation of uncertainty may be necessary. We are also interested in exploring extensions to stochastic models, identification, and calibration problems.

Acknowledgements

We would like to thank Randy Brost and Mike Erdmann, who allowed us to use the code from their earlier work, and contributed to the present work through numerous discussions. The ongoing experimental work depends heavily on contributions from Tom Wood and Dan Christian. We would also like to thank the Lord Corporation, who donated the instrumented gripper system, and the System Development Foundation and National Science Foundation, who provided support for Goldberg and Mason.

References

R. A. Brooks. 1982. Symbolic error analysis and robot planning. *Int J Robotics Research*, v5 n1, pp 29–68.

R. C. Brost. 1986. Automatic grasp planning in the presence of uncertainty. *Proceedings, 1986 IEEE Int Conf on Robotics and Automation*, San Francisco.

S. J. Buckley. 1987. Planning and teaching compliant motion strategies. Ph.D. thesis, MIT Department of Electrical Engineering and Computer Science.

B. R. Donald. 1986. Robot motion planning with uncertainty in the geometric models of the robot environment: a formal framework for error detection and recovery. *Proceedings, 1986 IEEE Int Conf on Robotics and Automation*, San Francisco.

R. E. Ellis. 1987. Acquiring tactile data for the recognition of planar objects. *Proceedings, 1987 IEEE Int Conf Robotics and Automation*. Raleigh, NC.

M. A. Erdmann. 1984. On motion planning with uncertainty. MIT Artificial Intelligence Laboratory, AI-TR-810.

M. A. Erdmann. 1986. Using backprojections for fine motion planning with uncertainty. *Int J Robotics Research*, v5 n1, pp. 19–45.

M. A. Erdmann and M. T. Mason. 1986. An exploration of sensorless manipulation. *Proceedings, 1986 IEEE Int Conf on Robotics and Automation*, San Francisco.

W. E. L. Grimson and T. Lozano-Pérez. 1984. Model-based recognition and localization from sparse range or tactile data. *Int J Robotics Research*, v3 n3, pp. 3–35.

W. E. L. Grimson. 1986. Sensing strategies for disambiguating among multiple objects in known poses. *IEEE J Robotics and Automation*, v2 n4, pp. 197–213.

D. D. Grossman and M. W. Blasgen. 1975. Orienting mechanical parts by computer-controlled manipulator. *IEEE Transactions on Systems, Man, and Cybernetics*, vSMC-5 n5.

P. Lötstedt. 1981. Coulomb friction in two-dimensional rigid body systems. *Zeitschrift fur Angewandte Mathematik un Mechanik*, v61 n12 pp. 605–615.

T. Lozano-Pérez, M. T. Mason, and R. H. Taylor. 1984. Automatic synthesis of fine-motion strategies for robots. *Int J Robotics Research*, v3 n1, pp 3–24.

M. Mani and W. R. D. Wilson. A programmable orienting system for flat parts. *Proceedings, NAMRI XIII*. Berkeley, CA.

M. T. Mason. 1984. Automatic planning of fine-motions: correctness and completeness. *Proceedings, IEEE Int Conf Robotics*. Atlanta.

M. T. Mason. 1986. Mechanics and planning of manipulator pushing operations. *Int J Robotics Research*. v5 n3, pp. 53–71.

M. T. Mason and Y. Wang. 1987. On the inconsistency of rigid-body frictional planar mechanics. CMU-CS-87-130. Computer Science Department, Carnegie-Mellon University.

M. A. Peshkin. 1985. The motion of a pushed, sliding object (part 1: sliding friction and part 2: contact friction). CMU-RI-TR-85-18, Robotics Institute, Carnegie-Mellon University.

A. C. Sanderson. 1984. Parts entropy methods for robotic assembly system design. *Proceedings, Int Conf Robotics*. IEEE Computer Society, Atlanta, GA.

R. H. Taylor. 1976. A synthesis of manipulator control programs from task-level specifications. Stanford Artificial Intelligence Laboratory AIM-282.