Learning Reliable Manipulation Strategies without Initial Physical Models

Alan D. Christiansen, Matthew T. Mason, and Tom M. Mitchell

School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213

Abstract

This paper describes a robot, possessing limited sensory and effectory capabilities but no initial model of the effects of its actions on the world, that acquires such a model through exploration, practice, and observation. By acquiring an increasingly correct model of its actions, it generates increasingly successful plans to achieve its goals. In an apparently non-deterministic world, achieving reliability requires the identification of reliable actions and a preference for using such actions. Furthermore, by selecting its training actions carefully, the robot can significantly improve its learning rate.

1. Introduction

We describe a simple robot system that acquires a model of the effects of its actions by constructing plans to solve specific tasks and then observing the results of these plans. The particular task is to manipulate rigid planar objects by tilting a tray containing the objects. Beginning with no prior knowledge of the effects of various tilting actions, the system exhibits very poor initial performance in achieving its goals. With experience it is able to acquire a discrete model of its actions that leads in some cases to 95% success in moving an object to a randomly selected goal configuration. We view this work as a first step in exploring the potential role of robot learning in extending the flexibility and scope of robot manipulation systems.

The motivation behind this research is straightforward. Current robotic systems are most successful in carefully engineered environments for which the physical characteristics of the world and the effects of robot actions on this world are well-behaved and well-understood. In such cases, these characteristics can be taken into account by a programmer who manually develops manipulation strategies for the robot. We seek to extend the ability of robotic systems to deal with worlds that are poorly characterized and poorly behaved, by allowing the robot to automatically acquire and utilize models describing the effects of its actions.

The system described here is the most recent and the most successful in a sequence of systems we have developed while studying this manipulation learning task. (See [Mason *et al.*, 1989] for an earlier version of the system.) The improved learning competence of the present system appears to be due primarily to two design characteristics:

- *Representing and reasoning about non-deterministic effects of actions.* Certain actions in the tray environment have non-deterministic results; that is, the same action executed twice from the same starting state (as perceived by the robot) may give different results. Allowing the robot to explicitly represent non-deterministic outcomes of its actions, using a learning algorithm that models actions as non-deterministic, and using a planning algorithm that reasons about the reliabilities of plans involving non-deterministic actions, have an important effect on learning competence and on asymptotic robot performance.
- *Use of information-seeking experimentation strategies for automatically collecting training data.* The strategy for collecting training data has a significant effect on the robot's learning rate. An experimentation strategy of randomly generating and observing actions is insufficiently focused. Alternatively, the strategy of attempting to achieve the current object-positioning goal by executing the most reliable known action sequence, is overly focused and fails to provide a sufficiently varied set of data to achieve rapid convergence. A hybrid experimentation strategy that combines components of these two extremes has produced the best results.

There have been other efforts to build robots that learn, and some have adopted methods that are quite different from ours. The adaptive control community uses a servo-loop model of manipulation tasks, with error values being determined and fed back at each loop iteration [Aboaf *et al.*, 1988; Aboaf *et al.*, 1989; Slotine and Craig, 1989]. Connectionist approaches also have been used to learn manipulator dynamics [Miller, 1987; Goldberg and Pearlmutter, 1988]. Mobile robot map-making shares some similarities with our idea of equating theories of the world with maps of the state-action space for a task. Examples of the mobile robot map-making work include [Laumond, 1983], [Crowley, 1984], and [Kuipers and Byun, 1987]. And, learning approaches for manipulation problems different from ours were presented in [Dunn and Segen, 1988] and [Zrimic and Mowforth, 1988].

The tray-tilting problem has been addressed before from an analytical perspective, in which knowledge of physical properties of objects and the tray is used to plan action sequences. Erdmann and Mason [1988] described a system for computing a sequence of tilts that would bring an object of known physical properties into a goal configuration from *any* initial configuration. No sensing was used, and the predictions of object motions were based on Newton's and Coulomb's Laws. Tray-tilting was further addressed in [Taylor *et al.*, 1987], where sensing was added. Sensing operations as well as physical actions were used to reduce uncertainty and achieve desired object configurations. Our paper formulates this problem in yet another way. Our system has no initial knowledge of physical laws or of physical properties of the objects, but does have sensory capabilities that it can use to observe the results of its actions. We demonstrate that these capabilities, along with a simple learning method, are sufficient to attain good task performance.

The remainder of this paper describes the robot system, the learning and planning strategies, experimental results, and limitations of the current system. Section 2 discusses characteristics of our task domain. Section 3 describes the design of our learning robot. Section 4 presents experimental results obtained from this system. Finally, Section 5 concludes the paper and lists possible future directions.

2. The Task Domain

The tray-tilting task was chosen as the testbed for this work for a number of reasons: its analysis is well understood, it is representative of robot manipulation problems, and it allows a range of task difficulty. Task difficulty depends on issues such as the number of objects in the tray (which may collide as the tray is tilted), object shapes (e.g., highly angular objects tend to bounce unpredictably against tray walls), and physical properties such as coefficients of friction and elasticity of the tray and the objects. The results presented here cover a variety of tray environments in which the number, shape, and elasticity of objects are varied. With no change to its initial knowledge or parameters, the learning system is able to acquire effective manipulation strategies across a variety of such tray environments.

In the experiments reported here, an 11 inch square tray is held by a programmable manipulator, which can tilt the tray in any desired direction. A camera mounted above the tray senses the configuration (position and orientation) of any objects in the tray. A typical task uses a 1 by 3 inch rectangle, which slides freely in the tray. The task is to move the object from its initial configuration to a goal configuration, by choosing an appropriate sequence of tilting motions.

From the tray's point of view, and assuming gentle motions, tilting the tray is equivalent to changing the direction of gravity, so that some component of gravity acts in the plane of the tray. Depending on the object's shape, the object's location, and the direction of the gravity vector, the object might remain at rest, might slide or tumble along a wall into a corner, or might slide across the tray and into another wall or into a corner of the tray. Some typical tray-tilt operations are shown in Figure 1.

We have further simplified the task domain as follows:

- Tilting motions start with the tray horizontal, tilt the tray in a particular direction, then return the tray to horizontal. Hence the gravity vector in the plane of the tray has a constant direction during a single tilt motion. This constant direction is called the tilt *azimuth*, and is a direction measured in the plane of the tray floor as indicated in Figure 2. In addition to the tilt azimuth, tilts are characterized by the maximum deviation of the tray floor from horizontal during the tilting operation. For our experiments, this steepness parameter was 30 degrees for all tilts. This value was sufficient in all cases to overcome the static friction between the tray floor and the manipulated object. Lastly, all tilts were performed at a slow constant speed, and the tray tilt was held for one second before the tray was returned to horizontal.

- States and actions are represented discretely. The tray is divided into nine sectors when describing the location of the object centroid, as shown in Figure 2. Object orientation is described as either of two values: horizontal or vertical. The control parameter—the tilt azimuth—is also discretized, but at a resolution of one degree.

The methods explored in this paper appear well-suited to the tray-tilting domain, and the generality of these methods may be judged in terms of the the essential characteristics of this domain:

- The physics governing the task is time-invariant.

- The cost of actions and sensing is negligible.

- The dynamics of the task are finite. The task can be modeled by a finite number of actions, with each action described as a mapping on a finite partition on the state space.

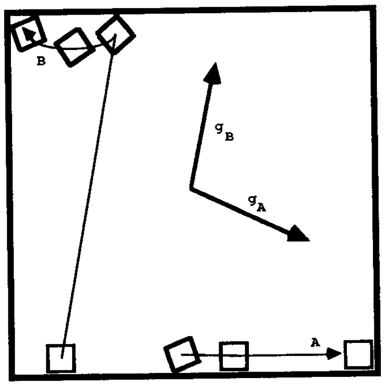

Figure 1: Two examples of tray-tilting operations. Gravity vector g_A causes the object to slide along a wall and into a corner. Gravity vector g_B causes the object to slide across the tray and bounce into a corner.

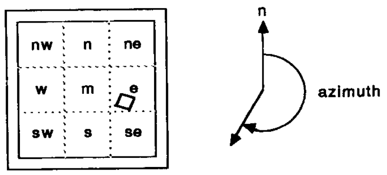

Figure 2: Tray Regions and Azimuths

- Sensing and effecting are imperfect.

If a robot has imperfect sensors and effectors, its task environment may appear (to the robot) to respond non-deterministically to the robot's actions. As such, the robot may observe differing outcomes for an action when applied to (what the robot views as) equivalent situations. This *apparent non-determinism* is a potential problem for any agent operating in the real world, and our robot is not immune to it. Apparent non-determinism may be caused by any of a number of factors:

- *Misclassified State.* It is impossible for a robot to represent the true world state using its limited resources. Therefore, "states" sensed by the robot correspond to sets of world states that are equivalent with respect to its sensors. However, there is in general no guarantee that a robot action will map world states within a single sensory equivalence class into world states that are within a common outcome sensory equivalence class. Furthermore, sensor errors will make the tracking of task features difficult.

- *Misclassified Actions.* A robot's internal description of an action may incompletely specify the action that the robot actually performs. For example, if the robot's effectors are incapable of perfect repeatability of motions, then the same commanded action may result in two different true actions.

- *Violated Assumptions about the World.* A robot might assume that its actions are time-invariant, but if its actions do indeed depend on the time at which they are executed, then the world will appear to be non-deterministic. If the robot assumes that it is the only agent that can change the world and this turns out to be false, then a second agent's actions may confuse the robot. And if a robot

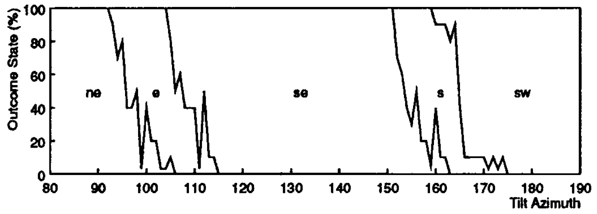

Figure 3: Reliability graph. For a given prior state (a square tile in the northwest corner of the tray) we recorded the result of ten tilts at each azimuth. This figure shows a portion of the tilt azimuths, and the frequency with which each of five different outcome states (northeast corner, east side, etc.) was attained.

assumes that the world behaves deterministically and it turns out that the real world is in actuality non-deterministic (an interesting philosophical question), then the robot will again be confused.

In the tray-tilting domain, apparent non-determinism can be partly attributed to underlying mechanical processes that are unmodelled. The motion of the object is determined primarily by the effects of friction and impact, which are difficult to model. When an object accelerates across the tray and bounces against the wall, the rebound of the object may appear to be random, even to a human observer. The existence of hidden state is also significant. For example, the coarse sampling of object position and orientation make it impossible for the robot to learn actions that depend on an initial contact between the object and the wall.

Figure 3 provides a measure of the apparent non-determinism of tray-tilt actions, obtained by recording the result of ten tilts at each of the tilt azimuths, for a square tile starting in the northwest corner of the tray. The data suggests that for some actions the outcome is quite repeatable, while for others it is not. For example, the graph indicates that tilt azimuth 135 is very reliable when used to move the square from *nw* to *se*, because all of the outcomes observed when that tilt was applied produced a result of *se*. (A vertical line at 135 would intersect only the *se* region.) Alternatively, tilt azimuth 109 is not highly reliable in producing an outcome of *se* because the plot shows that executing such a tilt produced *se* only six times out of ten, and produced *e* four times out of ten. Some tilts, such as 162, produced three different outcomes in the test. Other such studies have shown that some state transitions are inherently unreliable, and should be avoided completely. Because of the lack of repeatability for many actions, we should not expect a learning robot to converge to perfect performance.

3. Design of a Learning Agent

Our robot learns a theory of how to do tray-tilting. An explicit model is constructed that predicts the result state given a current object state and a proposed tilting action. This model, or theory, is not specifically tied to a single goal location for an object, but is rather a model that allows many potential goals to be achieved.

Before discussing the learning mechanism, we will discuss the problem-solving behavior of our robot. The control flow of our robot in performance mode is outlined in Figure 4. Note that there are two loops, one inside the other. The outermost loop is executed once for each goal in the robot's set of goals. Given a particular current goal, the robot constructs a plan to transform the environment to its goal state, based on its current theory of its environment. Each plan is a sequence of actions, and each action corresponds to one tilting operation. The actions comprising the plan are executed in sequence without error monitoring or recovery.

```
percept ← sense_world();
while not null?(goals) do
    goal ← first(goals);
    plan ← planner(goal, percept, theory);
    while not null?(plan) do
        action ← first(plan);
        perform(action);
        plan ← rest(plan);
    end while;
    percept ← sense_world();
    record_success_or_failure(goal, percept);
    goals ← rest(goals);
end while.
```

Figure 4: Performance-Mode Behavior of the Robot.

```
percept ← sense_world();
loop
    action ← choose_random_action();
    perform(action);
    prev_percept ← percept;
    percept ← sense_world();
    theory ←
        update_theory(theory, prev_percept, action, percept);
end loop.
```

Figure 5: Learning from Random Training.

To specify this control flow fully for the tray-tilting task, several variable names and procedures must be explained. The names *goals, goal, plan, action, percept, prev_percept*, and *theory* in Figure 4 correspond to state variables within the robotic agent. In particular, the value of the variable *theory* initially has the value *nil*, corresponding to the robot having no initial predictive theory for its environment. Over time, through updating the *theory* state variable, the robot attains some measure of predictive ability. For our tray-tilting problem, *percepts* correspond to object configurations (position, and in some cases, orientation), which are returned by the *sense_world* procedure. *Actions* correspond to tilts of the tray, which are defined by tilt azimuths. *Goals* are desired object configurations, and therefore of the same "type" as percepts. *Plans* are sequences of tilts that are computed by consulting the robot's current theory of the task.

There are many ways that a robot might gather training examples to update its theory. One of the simplest such ways, which we call the *random training* method, consists of generating random tilt azimuths and performing the corresponding tilts. The results of the tilts are used to update the robot's theory, as indicated in Figure 5. (The robot behaviors of Figures 4 and 5 are instances of a larger class of behaviors defined in [Christiansen, 1989].)

The learning method for our robot involves simple memorization of observed tilting operation outcomes. After each action is executed, the robot makes an update to its theory, reflecting the outcome of that action. Over time, these updates allow the robot to model the consequences of its actions more accurately, and this in turn allows the robot to construct better plans, and to achieve its goals more frequently.

The robot has available to it a set of possible actions. In the manipulation problem of this paper, the actions correspond to tilting operations that the robot performs on the tray. The robot's theory, then, is a relation on percepts, actions, and percepts. That is, a relation *predict(from,act,to)* is maintained by the robot, where *from* and *to* are percepts corresponding to perceived environment states, and *act* is one of the robot's actions. The robot interprets the relation as: "If my sensors tell me the world is *from*, and I do *act*, then I may achieve *to*." Note the lack of certainty about the outcome of the operation. A difficulty that the robot must face is to determine which of the many actions available to it are, in the current context, both *appropriate* and *reliable*.

It is useful to view the robot's theory, the *predict* relation, as a graph structure. The graph has percepts (states) as vertices and actions as edges. The theory updating operation may then be viewed as making changes to the graph, such as adding new edges between vertices and possibly adding new vertices. Once the robot's theory is described as a graph, it is easy to envision what the planner must do when presented with a goal, a current percept, and the current theory of the environment (the graph itself). The planner must find a path in the graph, linking the current percept to a goal percept. This path corresponds to a sequence of actions, or plan. In general, there will be many such paths. But the planner should return a path which is "better than" all other possibilities.

Our planner selects among alternative plans based on their *reliability*. The robot's theory maintains an estimated probability for each predicted outcome, based on the observed frequency of the outcome. The reliability of a *plan* is defined as the product of the reliabilities of the *actions* that comprise the plan. The planner uses a standard best-first search algorithm to find a plan of maximal reliability.

As the robot observes outcomes of tilts, it notes the percept corresponding to the state before the tilt, the percept corresponding to the outcome state, and the commanded tilt (that is, the azimuth angle that defines the tilt direction). The robot uses this triple of values to update its theory. Observations are organized by *from* state and tilt azimuth. For each *from* state and tilt azimuth, all the outcome states ever observed are recorded. If the tilt is perfectly reliable when performed from this particular *from* state, then the theory will record several occurrences of some unique outcome state. When a new observation is used to update the theory, a check is made to see whether the theory has any observations for the current *from* state and tilt. If so, then the current outcome state is added to the list of outcomes already stored. If there is not yet any observation for this *from* state and tilt, then a new observation record is begun. In sum, the learning mechanism is rote memorization of outcomes of tilts. It is the combination of this simple learning mechanism, the experimentation method, and the planning mechanism outlined above that leads to sophisticated behavior of the robot.

The *reliability* of a proposed tilting action, at some instant in time, is based on the observation record associated with the current state and the proposed tilt azimuth. The predicted reliability of a particular outcome state is a number between 0 and 1. Zero reliability corresponds to an outcome being impossible, and a reliability of one corresponds to an outcome being absolutely certain. The reliability of obtaining outcome o_i is computed according to the following formula:

$$r = \frac{x + w}{n + kw}$$

where x is the number of occurrences of o_i in n trials, w is a probability weighting factor, and k is the number of possible outcomes. It should be noted that this formula, except for the w and kw terms, is just the fraction of times that o_i occurred when executing this action from this state. However, the simpler frequency formula would assign the same reliability to a tilt that has been performed once, as it would to a tilt that has achieved the same state in each of 500 trials. The w and kw terms address this shortcoming, assigning a higher reliability to clearly repeatable state transitions. The weighting factor w is a measure of the robot's degree of disbelief in its experiments. If w is zero, then the robot always places complete faith in its experimental results, and the formula reduces to the frequency formula mentioned above. If w is large, then the robot is quite suspicious of its experiments, and it requires many experiments to produce a reliability near unity. For all of the work reported in this paper, w had the value 0.01, which meant that the robot tended to believe its experiments without requiring many repetitions.

The above formula for the reliability of achieving a particular outcome state, given an initial state and action, was developed as a heuristic. However, the formula can be related to a Bayes Estimate of the probability of occurrence of the outcome state. Finding an action of maximum reliability corresponds to finding an action that maximizes the probability of producing the desired outcome. If we assume that the distribution of the outcomes of a particular state and action is multinomial, and we further assume a uniform a priori distribution for the probabilities of the outcome states, then the Bayes Estimate of the probability of outcome o_i is $(x + 1)/(n + k)$, where x is the number of outcomes o_i observed in n trials, and k is the number of possible outcome states. The proper framework for the estimation of this probability involves the use of a conjugate family of distributions for multinomial observations. If instead of using a uniform a priori distribution, we use a Dirichlet distribution [DeGroot, 1970] with parameters all equal to 0.01, then the Bayes estimate of the probability becomes exactly the same as the heuristic formula given above. In this case, the prior distribution suggests that the true probability of any particular outcome is either very near 0 or very near 1, but is unlikely to have any other intermediate value [Goldberg, 1989]. Note that in this Bayesian interpretation, the heuristic regarding the willingness to accept the results of experiments without many repetitions is encoded within the prior distribution.

The structure of our learning robot bears some similarity to a stochastic learning automaton [Narendra and Thathachar, 1974]. It is possible to view the learning procedure described above as an ensemble of k^2 learning automata, where k is the number of possible percepts (or state categories). This learning mechanism, along with multi-step planning and an experimentation strategy that attends to multiple goals during training, comprise the essence of our robot's architecture. (See [Simons *et al.*, 1982] for an example of a robotic manipulation task to which the learning automata approach has been applied.)

4. Empirical Studies

We performed several performance tests on our learning robot. Each of these tests used the learning and planning mechanisms described above. The plans that were formed were both linear and complete, and no error monitoring or error recovery was done.

The first test used a 1 by 3 inch rectangular object. Figure 6 shows the result of this test. The robot trained itself by executing randomly chosen tilts. The robot observed the results of these tilts, and made appropriate updates to its theory, which was initially empty. The problem-solving performance of the robot was sampled at points during the training by giving it a set of 100 goals to solve. For each such goal,

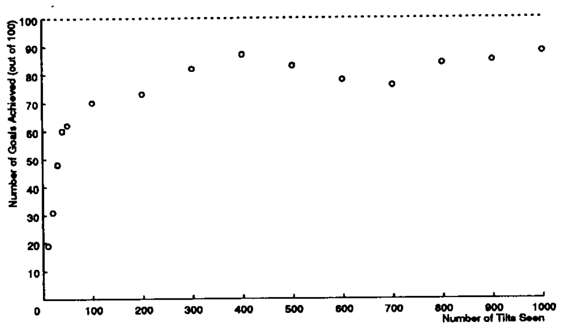

Figure 6: A typical learning curve. The object used was rectangular, and goals specified both a position and orientation of the object within the tray. The training was performed on randomly-generated tilts.

the robot created a plan, if it could, and then executed it. If the robot's sensors perceived that the resulting object configuration was the desired configuration, then the goal was considered to be achieved. In cases where the theory was not sufficiently developed to allow a plan to be computed, the robot performed a random tilt. The graph plots the percentage of goals successfully achieved by the robot, as a function of training experience. In this experiment, the goals specified a discretized position and orientation. For position, the tray was divided into nine regions, as shown in Figure 2. Orientations were divided into two categories, corresponding to the rectangle's major axis being "horizontal" or "vertical". Goals specified both a position region and an orientation category. (There are 9 times 2, or 18 different possible goals. Only twelve of these were actually given to the robot as goals. The other six correspond to "hard" goals. Two of these six are in the middle of the tray, which is an unachievable position. The other four are positions along a wall of the tray where the major axis of the object is perpendicular to the wall. This configuration is nearly unachievable.)

The graph shows a very high learning rate for the robot early in its training, and a lesser rate for the remainder of its history. During the early learning, the robot is finding ways to get to states that it couldn't get to before. After a theory is identified that includes all the goal states in the test set, the robot refines its notion of what actions are most reliable in achieving the goals. Ultimately, the robot appears to achieve a performance level better than 80% after seeing 1000 tilts. The plans that the robot created were typically less than four steps in length.

A second test used a 2.5 inch diameter circular disk instead of the rectangle. Since this object is symmetric, it has no observable orientation. The goals that were given to the robot were therefore positions only, drawn from all but the middle tray position. Thus, there were eight distinct goals, instead of the twelve of the previous test.

Figure 7 shows the performance of two runs of the robot, each starting with an empty theory. Note that the performance after 1000 tilts is about 90% in one case, and about 98% in the other. Why did the robot do better learning to manipulate the circle than learning to manipulate the rectangle? First, the problem is easier. Positioning is easier than positioning and orienting. Second, the state space of this problem is two-thirds the size of that in the previous experiment. Fewer tilts need be observed in order to identify appropriate actions for moving among the state in this smaller space. Third, the circular disk naturally behaves in a predictable manner. There are no corners, and

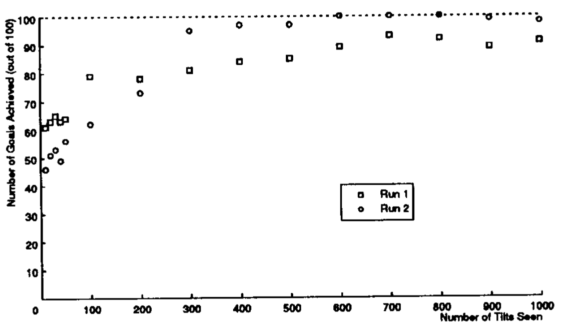

Figure 7: A test of positioning only. Goals specified only positions, and the object used was circular. The graph shows two distinct runs of the robot.

```
percept ← sense_world();
loop
    goal ← first(goals);
    plan ← choose_experiment(goal, percept, theory);
    while not null?(plan) do
        action ← first(plan);
        perform(action);
        prev_percept ← percept
        percept ← sense_world();
        theory ←
            update_theory(theory, prev_percept, action, percept);
        plan ← rest(plan);
    end while;
    goals ← rest(goals);
end loop.
```

Figure 8: Learning from Strategic Self-Training.

thus no aberrant behavior due to impact of the corners of the object with walls of the tray. It is easier to identify reliable manipulation strategies for a task domain that contains inherently more reliable actions.

Note that the effectiveness of the random training method can vary between runs. In one case, the robot found a near perfect theory after 600 tilts, but in the other, its imperfect theory causes a 10% failure rate to remain after seeing 1000 tilts. The difference is due to the particular tilts that are randomly generated during the training. Eventually, the "missing" actions that the robot requires will be generated, but it could take a very large number of tilts before this occurs. Differences like these between runs led us to consider increasing learning rate by giving the robot a non-random experimentation strategy.

Although the random training method is useful for sampling over the entire state-action space of the task, in order to successfully achieve its goals the robot needs only to learn some portion of the task dynamics: it requires only a practical sub-theory that allows it to achieve its goals. If the robot concentrates on identifying such a sub-theory, then its training effort should lead more quickly to good problem-solving performance. Figure 8 defines the control flow of this new training method.

The *strategic self-training* method, instead of generating a random tilt, computes an appropriate tilt for the purpose of acquiring useful

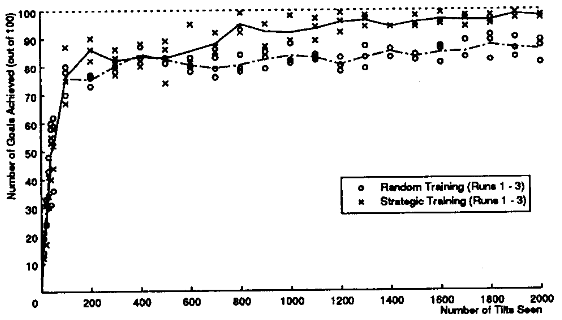

Figure 9: The impact of strategic self-training. The graph shows the results of six distinct runs of the robot—three with random training tilts, and three with a more deliberate strategy. The solid and dashed lines indicate the average performance of each training method over three runs. The object used was rectangular.

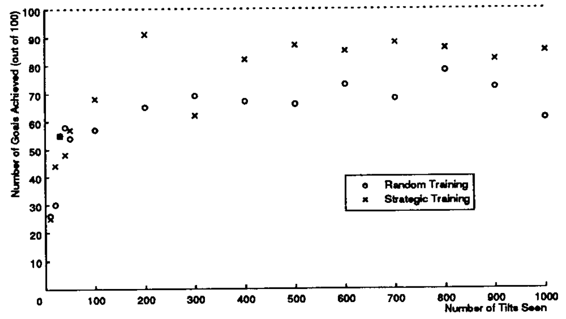

Figure 10: A more difficult problem, with large apparent non-determinism. The tray contained two circular disks, but only one was visible to the robot. Strategic self-training is still superior to random training, but achieves only slightly more than 80% performance.

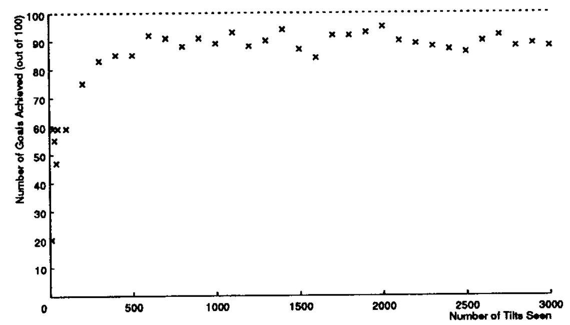

Figure 11: Another difficult problem. A rubber band is placed around the perimeter of the rectangle used in the earlier experiment. The increased friction between the rectangle and the walls of the tray leads to increased apparent non-determinism. However, the robot still achieves about 90% performance on this problem using strategic self-training.

information. This training strategy works as follows:

> Given the current goal, compute the plan of maximal reliability to achieve the goal. If the reliability of this plan is above some threshold (the *promising* threshold) and below another threshold (the *acceptable* threshold), then execute it. If no plan can be found, or if the plan reliability is above the *acceptable* threshold, then execute a randomly generated tilt instead. If the plan reliability is below the *promising* threshold, then modify the plan by replacing its least reliable step with a "nearby" tilt, then execute the plan.

For our experiments, a plan was considered to be of acceptable reliability if the plan reliability was greater than 0.95 . A plan was considered to be promising if its reliability was greater than 0.8 . And, choosing a tilt "nearby" a tilt of azimuth θ consisted of selecting a direction at random from the interval $[\theta - 10, \theta + 10]$. The motivation for using a random tilt when the plan is above the *acceptable* threshold is that in such cases it is unlikely that executing the plan will lead to unexpected consequences, and hence to significant improvements in the robot's theory. The motivation for modifying plans that are below the *promising* threshold is that by testing plausible changes to the weakest step of the plan, the robot gains the opportunity to explore alternatives that may lead to more reliable operations.

Figure 9 shows a comparison of the random training method with the strategic self-training method. The object used was the same rectangle as in the first experiment. This strategic experimentation method leads to acquiring a better theory, with the robot achieving about 98% average performance after seeing 2000 tilts. The random training method yields only about 83% average performance after seeing the same number of tilts. Note that although there is significant variation in performance from run to run, the strategic training method consistently outperforms random training. The main reason the strategic training method works better is that it does a better job of determining the reliabilities of tilts, which allows the planner to produce more reliable plans.

In order to test the robustness of this strategic training method, we devised two variations of the task that were "harder" than those so far described. In the first of these, we placed two circular disks in the tray. One of these disks was white, and thus visible to the robot, while the other was black and thus invisible. This task variation has a large amount of state information hidden from the robot, and was therefore expected to produce a large amount of apparent non-determinism. Figure 10 shows that this task was indeed more difficult to learn. However, the strategic training method is still superior to random training.

The second variation of the task used a single rectangular object as in the first experiment. However, the rectangle was altered by wrapping a rubber band around its perimeter, thus increasing the contact friction between the rectangle and the walls of the tray. In this task variation, the increased apparent non-determinism is not due to an invisible object, but rather due to a physical interaction that is seemingly more complex than before. The object now has a tendency to *roll* along the walls, where in previous tests it would slide along the walls without rolling. Of course, since the robot sees only initial and final states of the object, it bases its theory solely on the repeatability of transitions between states. If there are reliable rolling operations on the object, then it finds them and uses them. In fact, comparing Figures 9 and 11 shows that the increased friction does indeed make the problem more difficult for the learning robot. Without the rubber band, 98% performance is achieved, but with the rubber band only about 90% performance is achieved, even after 3000 training tilts. However, this task is still easier for the robot than the two-disk test, for which the robot achieves only about 80% performance using strategic training. (See Figure 10.)

5. Conclusions and Future Directions

We have shown that a robot, lacking an initial theory of its task domain, can develop a useful theory based on practice and observation. In order that the theory model the environment accurately enough for the robot to achieve its goals, reliable actions must be identified and the robot must prefer to use those reliable actions in its plans. We presented a design for a learning robot applied to a physical manipulation task, and demonstrated the effectiveness of its learning procedure. We further explored robot self-training strategies, and presented a self-training strategy for the tray-tilting task that allowed the robot to learn more quickly than with random training.

The research reported makes several assumptions about the world, the task space, and the agent working on the task. There are several changes to these assumptions that would be more realistic, or might increase task performance:

- *Continuous state and action description.* In all the experiments reported here, a discretized model of world state was used. A nearly continuous description of the world state is available from the camera image, and should be used. However, the simple graph-based model described here will have to be extended to handle such a modification. There are simply too many states and too many actions to explore in a continuous world.
- *Sensor resolution and effector accuracy.* It would be interesting to study the impact of sensor noise on the ability of a learning robot to identify an accurate model of its actions. Clearly, increased sensor noise or decreased sensor resolution will increase apparent non-determinism and therefore make the learner's job more difficult. Decreased effector accuracy will have a similar effect.
- *Dimensionality of States and Actions.* The dimensionality of the state-action space has a major effect on the "learnability" of a task. If a robot has many parameter values to choose for its actions, it may search a very long time before finding a correct association between its actions and states. An appropriate generalization mechanism may provide a solution to this problem.
- *A Priori Knowledge.* This paper explored the case where almost no initial knowledge of the task was made available to the robot. It would be very interesting to determine what knowledge is most useful for guiding learning.

Acknowledgements

This research was supported by NASA Ames under Grant NCC 2-463, and by a Bell Laboratories Ph.D. Scholarship supporting the first author. Thanks to Randy Brost and Mike Erdmann for their comments on this research, and to David Simon for a useful discussion on learning automata. Thanks also to Ken Goldberg and Larry Matthies for their help with the literature on Bayesian Estimation. Ken was primarily responsible for helping us to realize the relationship of our reliability measure to probabilities in a Bayesian framework.

References

Aboaf, E. W., Atkeson, C. G., and Reinkensmeyer, D. J., 1988. Task-level robot learning. In *IEEE International Conference on Robotics and Automation*, pages 1309–1310.

Aboaf, E. W., Drucker, S. M., and Atkeson, C. G., 1989. Task-level robot learning: juggling a tennis ball more accurately. In *IEEE International Conference on Robotics and Automation*, pages 1290–1295.

Christiansen, A. D., 1989. *A Framework for Specifying Robotic Agents.* Technical Report CMU-CS-89-155, Carnegie Mellon University.

Crowley, J. L., 1984. Dynamic world modeling for an intelligent mobile robot. In *IEEE International Conference on Pattern Recognition*, pages 207–210.

DeGroot, M. H., 1970. *Optimal Statistical Decisions.* McGraw-Hill.

Dunn, G. B. and Segen, J., 1988. Automatic discovery of robotic grasp configurations. In *IEEE International Conference on Robotics and Automation*, pages 396–401.

Erdmann, M. A. and Mason, M. T., 1988. An exploration of sensorless manipulation. *IEEE Journal of Robotics and Automation, 4(4)*, 369–379. Originally appeared in 1986 IEEE International Conference on Robotics and Automation.

Goldberg, K. and Pearlmutter, B., 1988. *Using a Neural Network to Learn the Dynamics of the CMU Direct-Drive Arm II.* Technical Report CMU-CS-88-160, Carnegie Mellon University.

Goldberg, K. Y. 1989. Personal Communication.

Kuipers, B. J. and Byun, Y. T., 1987. A qualitative approach to robot exploration and map-learning. In *AAAI Workshop on Spatial Reasoning and Multi-Sensor Fusion.*

Laumond, J., 1983. Model structuring and concept recognition: two aspects of learning for a mobile robot. In *International Joint Conference on Artificial Intelligence*, pages 839–841.

Mason, M. T., Christiansen, A. D., and Mitchell, T. M., 1989. Experiments in robot learning. In *International Workshop on Machine Learning*, pages 141–145.

Miller, W. T., 1987. Sensor-based control of robot manipulators using a general learning algorithm. *IEEE Journal of Robotics and Automation, RA-3(2)*, 157–165.

Narendra, K. S. and Thathachar, M. A. L., 1974. Learning automata—a survey. *IEEE Transactions on Systems, Man, and Cybernetics, SMC-4(4)*, 323–334.

Simons, J., Van Brussel, H., De Schutter, J., and Verhaert, J., 1982. A self-learning automaton with variable resolution for high precision assembly by industrial robots. *IEEE Transactions on Automatic Control, AC-27(5)*, 1109–1113.

Slotine, J. E. and Craig, J. J., 1989. Adaptive trajectory control of manipulators. In Khatib, O., Craig, J. J., and Lozano-Pérez, T., editors, *The Robotics Review 1*, pages 369–377, MIT Press.

Taylor, R. H., Mason, M. T., and Goldberg, K. Y., 1987. Sensor-based manipulation planning as a game with nature. In *International Symposium on Robotics Research*, pages 421–429.

Zrimic, T. and Mowforth, P. H., 1988. An experiment in generating deep knowledge for robots. In *Conference on Representation and Reasoning in an Autonomous Agent.*