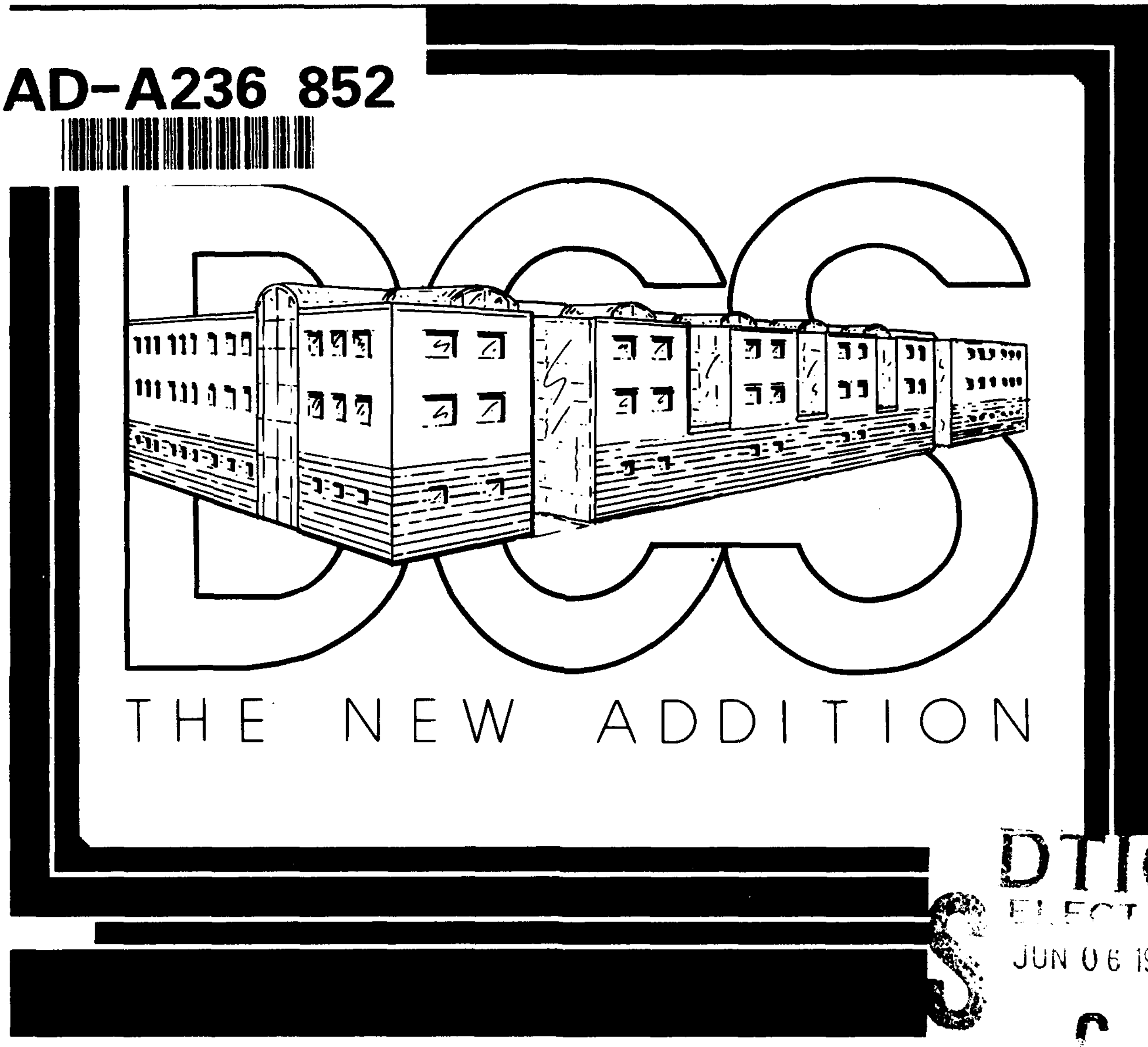
DEPARTMENT OF COMPUTER SCIENCE

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN



REPORT NO. UIUCDCS-R-91-1685

UILU-ENG-91-1731

INCREMENTALLY INCREASING THE UNCERTAINTY-TOLERANCE OF ROBOTIC MANIPULATION PLANS

by

Scott Bennett
Gerald DeJong

April 1991

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS None
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; Distribution unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) UIUCDCS-R-91-1685	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Dept. of Computer Science University of Illinois 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION Office of Naval Research
6c. ADDRESS (City, State, and ZIP Code) 1304 W. Springfield Urbana, Illinois 61801	7b. ADDRESS (City, State, and ZIP Code) 800 N. Quincy Street Arlington, VA 22217-5000
8a. NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N00014-86-K-0309
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. | PROJECT NO. | TASK NO. | WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)

Incrementally Increasing the Uncertainty-Tolerance of Robotic Manipulation Plans

12. PERSONAL AUTHOR(S)

Scott Bennett and Gerald DeJong

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Technical	FROM ______ TO ______	1991 April	22

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Robotics, Planning in Complex Uncertain Domains
05		10	

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Control of a robotic manipulator is subject to control and sensing errors. This is particularly troublesome in fine motion planning. A number of approaches have been taken including propagation of numeric or symbolic uncertainties and the generation of plans which use compliant motion to achieve goals in spite of uncertainties. These approaches have generally sought to guarantee success and thus plan generation costs are *high*. We present a unique approach to incrementally acquiring uncertainty-tolerance in robotic manipulation plans through experience. During plan construction and execution, no reasoning about uncertainty takes place. Consequently, plan generation and execution is very fast. However, in response to failures, plans are refined to increase their uncertainty-tolerance so as to reduce the future possibility of the encountered failure. The incremental refinement approach has several advantages over guaranteed plans. First, resulting plans are general and have explicit applicability conditions. Second, plans achieve a savings because they do not explicitly consider uncertainties. Third, savings is obtained over the guaranteed case since often only a subset of all uncertainties lead to failures in practice. Last, unguaranteed but practical plans can be generated by the incremental approach when they lie outside the scope of the guaranteed planner. To demonstrate our approach we describe an implemented system called GRASPER which learns to grasp novel objects given only imprecise television camera input. No prior model of the objects is assumed, nor are the objects required to satisfy *a priori* constraints on their shapes. Robustness of the system's grasping improves with experience.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL Alan Meyrowitz	22b. TELEPHONE (Include Area Code) (202) 696-4302 | 22c. OFFICE SYMBOL ONR

DD Form 1473, JUN 86 Previous editions are obsolete. SECURITY CLASSIFICATION OF THIS PAGE

Incrementally Increasing the Uncertainty–Tolerance of Robotic Manipulation Plans

Scott W. Bennett

Beckman Institute, University of Illinois, 405 N. Mathews, Urbana, IL 61801,

Email: bennett@cs.uiuc.edu Phone: (217) 244–1503

Gerald F. DeJong

Beckman Institute, University of Illinois, 405 N. Mathews, Urbana, IL 61801

Email: dejong@cs.uiuc.edu Phone: (217) 333–0491

Content Area: ROBOTICS

Keywords: Robotics, Planning in Complex Uncertain Domains

Abstract

Control of a robotic manipulator is subject to control and sensing errors. This is particularly troublesome in fine motion planning. A number of approaches have been taken including propagation of numeric or symbolic uncertainties and the generation of plans which use compliant motion to achieve goals in spite of uncertainties. These approaches have generally sought to guarantee success and thus plan generation costs are high. We present a unique approach to incrementally acquiring uncertainty–tolerance in robotic manipulation plans through experience. During plan construction and execution, no reasoning about uncertainty takes place. Consequently, plan generation and execution is very fast. However, in response to failures, plans are refined to increase their uncertainty–tolerance so as to reduce the future possibility of the encountered failure. The incremental refinement approach has several advantages over guaranteed plans. First, resulting plans are general and have explicit applicability conditions. Second, plans achieve a savings because they do not explicitly consider uncertainties. Third, savings is obtained over the guaranteed case since often only a subset of all uncertainties lead to failures in practice. Last, unguaranteed but practical plans can be generated by the incremental approach when they lie outside the scope of the guaranteed planner. To demonstrate our approach we describe an implemented system called GRASPER which learns to grasp novel objects given only imprecise television camera input. No prior model of the objects is assumed, nor are the objects required to satisfy *a priori* constraints on their shapes. Robustness of the system's grasping improves with experience.

This research was supported by the Office of Naval Research under grant ONR N00014–86–K–0309.

1 Introduction

Dealing with uncertainty is an important problem in robotic planning. In executing a plan on a manipulator both control and sensing errors occur in addition to errors in modelling the environment. Although one could invest in mechanically better robots and higher resolution sensors, the incremental improvement diminishes as the cost escalates.

Problems with uncertainties are particularly pronounced in fine motion planning where small discrepancies in manipulator control or piece modelling can easily result in failures. Some of the first approaches to this problem involved the propagation of error estimates throughout the steps of a task numerically [Lozano–Perez76, Taylor76] and symbolically [Brooks82]. The error estimate could be used to make decisions about what strategy to apply at different stages of the plan. Later, approaches were introduced using *preimages* in configuration space [Erdmann84, Erdmann86, Lozano–Perez84]. A preimage of a goal gives a set of configurations from which compliant motion can be initiated and guaranteed to succeed despite control and sensing uncertainties. Where the previous systems sought guaranteed plans for goal achievement under specified uncertainty bounds, Donald's EDR system relaxes this requirement permitting execution of unguaranteed plans which fail in recognizable ways and constructing plans which succeed in spite of model error [Donald90].

When a fine motion plan has been constructed that takes advantage of compliant motion, results can be very good. For instance, experimental results obtained by Gottschlich and Kak in parts mating operations under uncertainty are encouraging [Gottschlich89]. Unfortunately, systems which represent and reason about uncertainties explicitly to construct guaranteed plans or to guarantee that only recognizable failure states occur, incur a high performance penalty in planning. For example, Canny shows that to construct an r–step motion plan under uncertainty in an m–dimensional configuration space with n polynomial constraints defining the bounds of obstacles in the space takes an algorithm of complexity $2^{2^{O(r\ n\ m)}}$ [Canny89].[1] Results like these have triggered much interest in reducing the complexity of such planning by approximating the real problem by simpler but still realistic ones.

A number of techniques have been introduced for improving the complexity of robot planning. These include slicing configuration space (forbidding motion along certain dimensions) and simplifying the shapes of objects [Latombe90]. Generally, these techniques are conservative such that if the planner succeeds with the simplified model it is guaranteed to succeed with the real problem. Unfortunately, this rules out many situations where fine control is necessary with cluttered spaces in the presence of uncertainty. A general robot manipulation system should be able to take advantage of different simplifications in different situations. For instance, approximating the robot gripper with a large rectangular prism might only be indicated when the spacing between objects exceeds some threshold.

Another approach explored recently involves open–loop planning. If methods can be found to predict how actions behave under uncertainty, less sensing is required. The need to make decisions based on sensor readings at execution time is a significant source of complexity [Canny89]. Mason gives a detailed analysis of pushing and grasping techniques which lead to success in the presence of uncertainty without requiring sensing [Mason85, Mason86].

Yet another approach to improving robot planning efficiency is via machine learning. Segre explores the use of explanation–based learning (EBL) [DeJong86, Mitchell86] for automatic synthesis of manipulation plans [Segre85, Segre88]. Because inference rules and operators are used to construct a proof of why the plan should succeed, a set of generalized conditions can be produced which describe the applicability of the plan. Consequently, the plan can be saved and used later for similar tasks. Segre's ARMs system demonstrated the viability of the approach for learn-

1. Canny points out that this extreme complexity is due to the need to use sensor readings to make decisions at execution time. A sensorless plan could be run in singly exponential time [Canny89].

ing generalized robot plans. It did not, however, deal with the problems of uncertainty which our work addresses.

Simple machine learning techniques have been applied to the problem of uncertainty in robotics planning. Dufay and LaTombe employed an inductive approach which uses a production rule system to drive execution [Dufay84]. Multiple traces are then combined to produce the plan. This can be viewed as a system which uses experience with the world to gain in efficiency, including inducing loops and limiting branch points to those possibilities observed in the traces. The uncertainty–tolerance of plans is limited to those cases represented by the traces but new cases can be added as generated by the production system when the plan fails. The generated plans explicitly consider uncertainties and use sensing operations during execution. Where Dufay and LaTombe's approach is limited to the set of operators described in the production rules, our approach can learn to adjust parameters of those operators. Their plans apply to a specific task and context where our planner derives general preconditions for plan application during plan construction. For example, without these applicability conditions for a plan, Dufay and LaTombe's system can induce loops which don't terminate in certain cases [Dufay84].

Another approach to learning robot plans in the face of uncertainty is Christiansen and Mason's approach [Christiansen90a]. Here, actions are tried and their effects observed over many trials compiling a set of conditional probabilities. This information can then be used in constructing unguaranteed open–loop plans by plotting a course of maximum probability of goal achievement. Variations on this technique are explored in [Christiansen90b]. This approach uses no explicit inference rules or operators about the way the world behaves but makes use of probabilities tabulated over a large number of trials. The world must be discretized into a relatively small number of states to make such a tabulation practical. Because there is no notion of what may effect these probabilities, in a slightly different context, the experiments would have to be performed again.

In this paper we present a general technique for introducing uncertainty– tolerance into robot manipulation plans based on experienced failures with those plans. Uncertainties are only reasoned about in response to failures, not when the plan is originally built and executed. This makes plan construction and execution very fast. A tradeoff exists between the cost of generating robot plans and the chance of their failure. Refinement occurs when plan operators violate expected sensor conditions during execution. The refinement process increases the uncertainty–tolerance of specific portions of the plan so as to decrease the likelihood of a similar failure.

The incremental refinement technique to planning under uncertainty employs machine learning techniques and offers several advantages over guaranteed plans:

1) **generality** The technique learns general manipulation plans from specific problems. Consequently, planning time can be saved if an applicable plan has already been generated and can be instantiated for the task at hand. For instance, a general plan might embody a technique for grasping an object when two near–parallel faces are not occluded by nearby objects. The system must, however, guarantee the conditions for that plan hold. One of these involves checking that no nearby objects occlude the grasping faces. Not only do such general plans save planning time if formerly learned plans can be applied to new situations, but the presence of general conditions for plans is particularly powerful. Suppose a motion planning task involves moving first through a relatively sparse part of the workspace and then through a more cluttered part of the workspace. It becomes possible to derive an operator sequence first using one plan that is inexpensive and applies in sparsely occupied spaces followed by another more expensive plan which applies in cluttered spaces.

2) **complexity** In constructing a guaranteed plan under uncertainty all possible potential errors and their interactions must be considered. Given some distribution of tasks a manipulator must perform, not all those errors and interactions are likely to arise. Consequently, an incremental learning approach can adapt plans to the level of uncertainty–tolerance required without expending the additional effort to guarantee success in unlikely situations. Furthermore, uncertainty–tolerance gets built into

the plan implicitly such that no explicit reasoning about uncertainty takes place when the plan is constructed and executed, only when it is refined.

3) **scope** Guaranteed planning approaches and those which seek to guarantee recognizable failures states can never be applied when such guarantees cannot be proven. The incremental approach has the potential to learn techniques which succeed in practice with a particular set of tasks but which can't be guaranteed.

It is worthwhile to note that any incremental approach which relies on failures makes the assumption that some amount of failures is tolerable. While the rate of failures tends to decrease as experience is gained, one cannot guarantee that future failures are not possible without constructing a guaranteed plan.[2] There are certainly domains where guaranteed plans are much more crucial and worth the additional effort to generate them. One would not want a remote manipulator learning from failure in moving bottles of nitro–glycerin.

In the next sections, we introduce data approximations which are explicit representations for uncertain and/or simplified sets of data, plan parameters which can be tuned to affect uncertainty–tolerance of a plan, and the plan refinement procedure. A detailed robotic grasping example is then presented from an implemented system.

2 Data Approximations

Data approximations are representations for approximate continuously valued data about the state of the world. They can either be *external* or *internal*. External data approximations are used to represent the uncertainty of data in the world. Internal data approximations are used to simplify complex sets of data to make planning more tractable. First, let us consider external data approximations.

2.1 External Data Approximations

An external data approximation involves a set of quantities for which the system is given approximate values typically via imperfect sensors. Let Q_E be a vector

2. Strictly a guaranteed plan is only guaranteed under fixed assumptions about the magnitude of the uncertainties.

$\{q_1, q_2, q_3, \ldots, q_n\}$ of quantity variables. Every q_i exists along a continuous dimension $D(q_i)$ and has a measured value knowable to the system denoted $M_V(q_i)$. There is also an actual world value not knowable to the system denoted $A_V(q_i)$. For a sensor to yield a valid approximation we require that the conditional probability $P(\ M_V(q_i) = a_i \mid A_V(q_i) = r_i\ )$ is monotonic in $|\ r_i - a_i\ |$. Given the actual value r_i for a sensed quantity $(A_V(q_i) = r_i)$, the likelihood of measuring value m_1 is greater than measuring value m_2 in just those cases that m_1 is closer to r_i than is m_2. More precisely:

$$P(M_V(q_i) = m_1 | A_V(q_i) = r_i) > P(M_V(q_i) = m_2 | A_V(q_i) = r_i) \text{ iff } |m_1 - r_i| < |m_2 - r_i| \quad (2.1)$$

In the case of external data approximations, the values $\{M_V(q_i)\ | i = 1, 2, \ldots, n\}$ are the best information the system has about the quantity variables . The only possible way to improve this information would be to interact with the world. For purposes of planning with the data represented by the approximations, the system behaves as if $Q_E = \{M_V(q_i)\ | i = 1, 2, \ldots, n\}$. The qualitative definition of a data approximation is never employed during planning, only when analyzing failures.

2.2 Internal Data Approximations

With an internal data approximation, the system chooses the values for the quantity variables Q_I with a data approximation procedure. Using a simplified internal description of the real world typically results in more efficient, though less accurate, planning. Internal data approximations can be adjusted through the system's reasoning alone. By expending more or less resources reasoning about a plan the system may bias its planning towards accuracy or towards efficiency. The costs associated with planning failures, execution failures, and other features of the task domain may be thought of as specifying a utility function.[3] For a given problem distribution in a domain there is an *optimal* faithfulness of representation that maximizes the sys-

3. For a model of the different aspects of utility for plan to be executed in uncertain, complex domains see [Bennett89].

tem's performance according to the utility costs. One can think of internal data approximations as being like external data approximations except that the ideal unknowable value of the variables represents the optimal setting, while the actual represented value for a quantity corresponds to the value chosen by the system in an attempt to maximize utility.

3 Plan Parameters

Plans employ continuous numeric parameters which can be tuned to affect uncertainty–tolerance. It is important that these parameters depend on the situation to which the plan is applied (the plan's *context*). A simple example is a parameter that specifies the height which a manipulator must be above the workspace to safely navigate without collisions. The possible settings for this parameter are a function of the highest object in the workspace. Therefore, it depends on context. This means a generalized plan must choose its parameter values at the time the plan is applied.

Let Q_P be the set of plan parameters and A_P be the set of their respective values. Every $q_i \in Q_P$ is defined along a continuous dimension $D(q_i)$. Let there be a low bound $L_{low}(D(q_i), C)$ and a high bound $L_{high}(D(q_i), C)$ on the values which q_i may assume along dimension $D(q_i)$ in context C. A *context* is a partial world state specification. Figure 1 gives a pictorial representation for the dimension $D(q_i)$. The

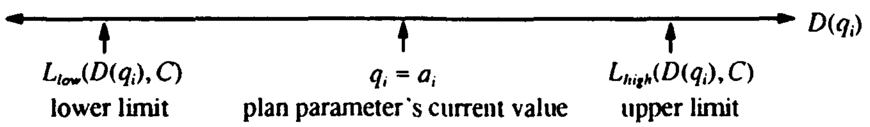


Figure 1. A Graphic Representation of Dimension $D(q_i)$

plan's parameter values A_P are in general dependent on context. The system chooses A_P based on a set of *preferences* $P(q_i, C)$ for each q_i in context C.

A preference $p \in P(q_i, C)$ is a 3–tuple $< V_{low}(C)\ , V_{high}(C)\ , r >$ where $V_{low}(C)$ and $V_{high}(C)$ form an interval dependent on context, and $r \in \{\ increasing,\ decreasing,\ constant\ \}$ is a relation describing the behavior of a *quality* function $F_{Q,q_i,C}$ in that interval. The greater the value of the quality function $F_{Q,q_i,C}(X)$, the better the value for X as the parameter setting in that context. Additionally, a set of preferences are consistent if for the current context their intervals lie within the bounds ($L_{low}(D(q_i),C) \leq V_{low}(C) \leq V_{high}(C) \leq L_{high}(D(q_i),C)$) and no two overlap. Figure 2 shows two consistent preferences and one possible

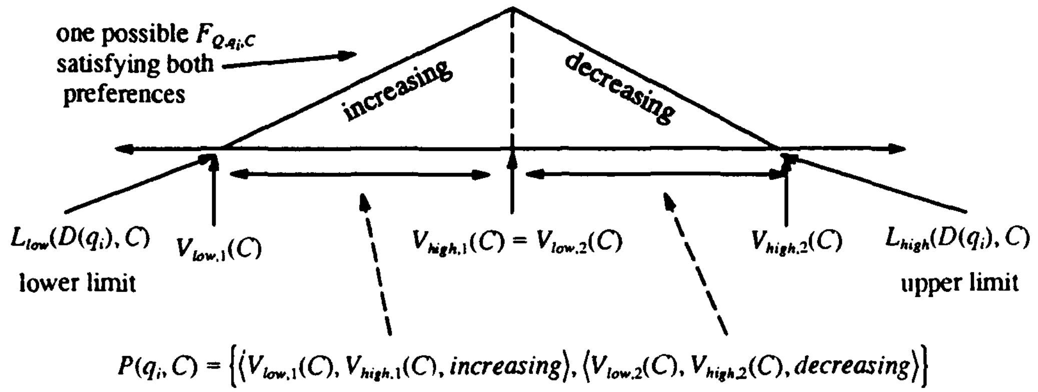


Figure 2. A Graphic Representation Of Two Consistent Preferences and a Quality Function which Satisfies Them

quality function which satisfies them. We assume limited interactions between parameters in that the preferences must maintain their relative ordering under any possible valuation of the other plan parameters.

When a general plan is applied in a specific situation (context), the current quality function for each parameter determines its value. A value for the parameter is chosen so as to maximize its quality value. The choice of parameter value may be constrained in a given context. For instance, in grasping a polygonal object, one parameter involves the angle between grasping faces. Since the object has only a discrete number of faces, there are only a discrete number of choices for the parameter value. The choice giving the best quality value is chosen.

4 Refining Failing Plans for Increased Uncertainty–Tolerance

When constructing a plan, any actions to be carried out in the plan must include a specification of sensor readings expected during execution of the action. The proof which justifies why a plan will succeed (also referred to as the *explanation*) in the system's model of the world also justifies the expected sensor readings. Failures are thus defined by expectation violations. These occur when the supporting proof for an expectation is valid in the model but is contradicted by real–world experience. That difference triggers the first phase of failure recovery: generating a proof of how plan parameters can be tuned to reduce the chance of the failure in the future.

In order to diminish the chance of uncertainty–related failures, it is necessary to decide which plan parameters to tune and how to tune them. In our model, failures can always be attributed to poor data approximations. In order to devise a strategy for tuning parameters so as to decrease the likelihood of a failure, it is necessary to reason about the relationships which exist between data approximate quantities, the failing expectations, and tunable plan parameters.

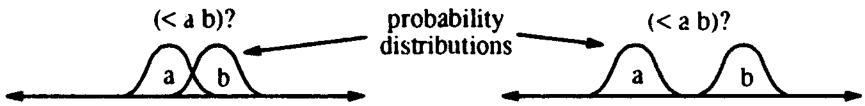
We employ a *qualitative* model of the relationship between continuous quantities. It is important to preserve generality in learning preferences for parameters. Otherwise, we could not transfer experience gained about parameter settings to similar situations. A qualitative model allows us to reason about the important relationships between quantities without tieing us to the specific values for the current situation. Let $Q_{+}(a, b)$ signify that the magnitude of quantity b positively influences the magnitude of quantity a. Similarly, $Q_{-}(a, b)$ means the magnitude of quantity b inversely influences the magnitude of quantity a. That is, if $a = f(b, c_1, c_2, \ldots, c_n)$ the minimum we must know to create such a relation (without ambiguity) is the sign of $\frac{\partial f}{\partial b}$.

If $\frac{\partial f}{\partial b} > 0$ then $Q_{+}(a, b)$ holds. If $\frac{\partial f}{\partial b} < 0$ then $Q_{-}(a, b)$ holds.[4]

Quantitative predicates employed by the system have one of two basic intents. Either they are *calculation predicates*, whose purpose is to compute some value (e.g. a predicate for subtraction), or they are *test predicates*, which are designed to fail for certain sets of inputs (e.g. a predicate for performing a less–than comparison). There is no

4. Q_{+}, Q_{-}, and the associated inference rules about increasing and decreasing quantities are used as in qualitative process theory [Forbus84].

way to vary the probability of success of a calculation predicate since they always succeed. A test predicate's probability of success is sensitive to the probability distribution of its argument quantities. In the diagram below, the less–than test on the right



has a higher probability of succeeding given the illustrated probability distributions for its arguments than the one on the left. While probability distributions are difficult to define and work with, recall the simpler qualitative view of the probability distribution defined for data approximations in section 2.1: probability density decreases monotonically as one moves either higher or lower away from the central value. Therefore, if we can tune one of the arguments to a *test predicate* we can affect its chance of success.

Let DA(q) signify that q is a data approximate quantity. Now let us introduce the notation $PQ_+(p, q)$ to express that the magnitude of the quantity q directly influences the magnitude of the probability of success of test predicate p. Similarly, $PQ_-(p, q)$ indicates that the magnitude of quantity q negatively influences the magnitude of the probability of success of test predicate p. This provides a mechanism for connecting the probability of a predicate being satisfied with the magnitude of a quantity. The inference rule which follows is one of several which follow from our definition for data approximations: $PQ_-(a < b, a) \Leftarrow DA(q), Q_+(b, q), \neg Q_+(b, a)$. [5]

The rule states: if q is a data approximate quantity and hence uncertain and directly influences the magnitude of a quantity b, the likelihood of $a<b$ succeeding is inversely proportional to the magnitude of a provided a does not directly influence b.[6] Let *PP(q)* indicate that the quantity q is a plan parameter and hence is tunable. Let

5. Inference rules here will be shown in the form of Horn clauses with their consequent at the left and a list of the conjunctive antecedents on the right.

6. This rule is stronger than it need be but otherwise rates of change would also have to be considered. We may relax this in a later implementation.

$PS_{\uparrow}(p)$ signify that the probability of success of predicate p is increasing. Let $Q_{\uparrow}(q)$ signify that quantity q is increasing. Therefore we could use the following rules to increase the probability of a predicate p give than q is a plan parameter:

$PS_{\uparrow}(p) \Leftarrow PQ_{+}(p,q), Q_{\uparrow}(q).$ and $Q_{\uparrow}(q) \Leftarrow PP(q).$

The second rule above asserts that we can increase plan parameters to achieve goals (because they are tunable).

It must also be possible to propagate the qualitative probabilities of predicates. Let $ANT(p_1,p_2)$ indicate that p_2 is an antecedent of an inference rule for which p_1 is a consequent. One sound rule for propagation of qualitative probabilities across rules can then be expressed:

$$PQ_{+}(p_1,q) \Leftarrow ANT(p_1,p_2), PQ_{+}(p_2,q), \forall x[[ANT(p_1,x) \wedge x \neq p_2] \Rightarrow \neg PQ_{-}(x,q)] \quad (2.2)$$

The general rules required to construct a qualitative tuning proof fall into four categories:

general qualitative inference rules — inference rules necessary to reason about increasing and decreasing quantities

Example: $Q_{\uparrow}(x) \Leftarrow Q_{+}(x,y), Q_{\uparrow}(y).$

qualitative predicate definitions — rules providing qualitative definitions for system predicates relating quantities

Example:

$$Q_{+}(x,y) \Leftarrow [x = y + z]. \quad \left[because \quad (x = f(y)) \wedge \left(\frac{\partial f}{\partial y} = 1 > 0 \right) \right]$$

approximation definition rules — rules defining the behavior of test predicates using data–approximate quantities

Example: $PQ_{+}(a < b, b) \Leftarrow DA(q), Q_{+}(a,q), \neg Q_{+}(a,b).$

qualitative probability rules — rules about the propagation of qualitative probabilities (Example: Rule 2.2 above)

The qualitative tuning explanation is a sound proof of how to positively influence the probability of success of the predicate which supported the failing expectations. The procedure for constructing the tuning proof and tuning the plan parameters as a result is as follows:

1) Compute the set P of generalized preconditions and effects for the plan justification structure[7]

2) Take all generalized variables which are quantitative arguments to every predicate $p \in P$ as quantity variables for the qualitative reasoning process.

3) Find all qualitative influences among these quantity variables. This is possible since, if two plan quantities are related, we know the exact functional relationship (in the model).

4) Construct a proof based on the qualitative inference rules discussed above for how to unambiguously qualitatively increase the probability of success of the predicate supporting the expectations to the failed action.

5) Collect the set of quantity increases and decreases justified by the fact that plan parameters PP(q) are tunable. This amounts to finding applications of the rules $Q_{\uparrow}(q) \Leftarrow PP(q)$ and $Q_{\downarrow}(q) \Leftarrow PP(q)$ in the proof.

6) For each of the tunable plan parameters in the set collected above, add a new preference to the set of preferences for that parameter. This preference will specify as one of it bounds a general expression for the point at which the failure occurred and as its relation increasing or decreasing as given in step 5. The other bound will be realized in conjunction with the neighboring preferences.

Next, we introduce a fully implemented system and demonstrate the algorithm.

5 The GRASPER System

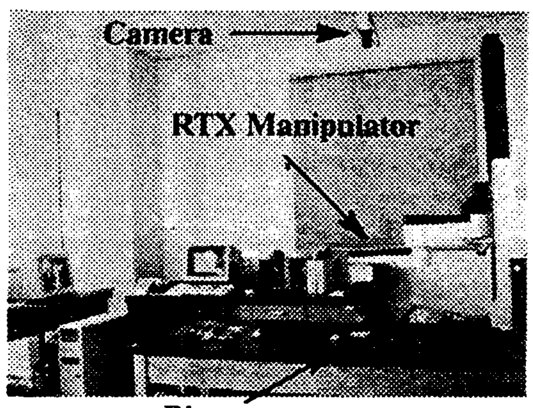


Figure 3. GRASPER Experimental Setup

Figure 3 shows the laboratory setup. The current implementation of the architecture is called GRASPER and is written in Common Lisp running on an IBM RT125. GRASPER is interfaced with a frame grabber connected to a camera mounted over the workspace. The camera produces bitmaps from which object contours are extracted by the system. The system also controls an RTX scara–type robotic manipulator. The RTX has encoders on all of its joint motors and the capability to control many parameters of the motor controllers including motor current. This gives the system a rudimentary capability of detecting collisions with

7. This is accomplished through application of the EGGS [Mooney86] or EBG [Mitchell86] explanation–based generalization algorithms.

the RTX gripper.[8] This type of sensing gives feedback during execution of a plan when the camera's view of the workspace would otherwise be obscured.

Our current goal for the GRASPER system applied to grasping is to successfully grasp the plastic pieces from puzzles designed for young children. Since the pieces are laminar, an overhead camera is used to sense piece contours. These pieces have interesting shapes and are large enough, yet challenging, to grasp. The goal is to demonstrate improving performance at the grasping task over time in response to failures. Some of the failures the current implementation learns to overcome, when using isolated grasp targets, include learning to open wider to avoid stubbing the fingers on an objects, and learning to prefer more parallel grasping faces to prevent unstable grasps. We are also exploring grasping in cluttered workspaces where trade-offs exist between plan parameters.

Figure 4 shows part of the system's status display during a grasping task. First, the

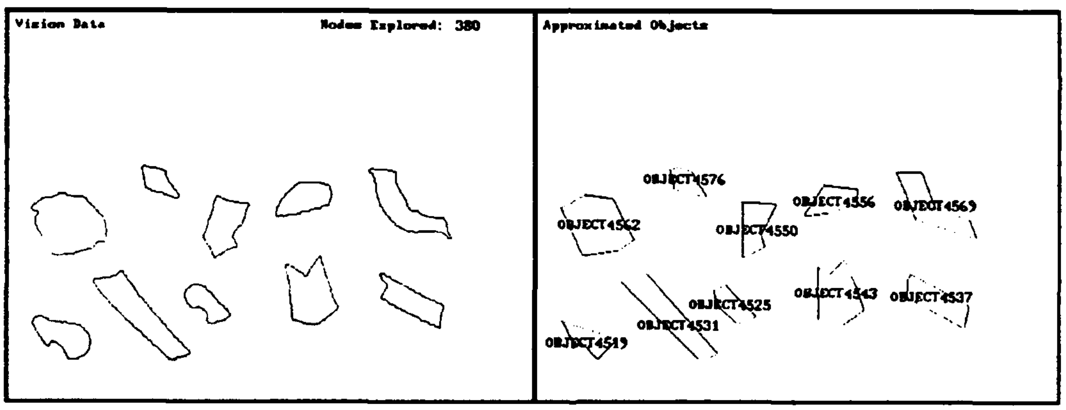


Figure 4. Portion of System Status Display During Grasp of Object4543

system uses the camera to acquire contour information about objects in the workspace. These contours are shown on the left in the figure. Next, the contours are approximated with n–gons (internal data approximations) which result in $(n^2-n)/2$ possible unique grasping face pairs. These approximated object contours appear on the right in the Figure 4. The algorithm chooses the value of n such that an approxima-

8. A wrist force sensor would be more desirable. The method currently used involves applying enough current to the motor to overcome friction of the arm mechanism and interpreting non–changing joint encoders as evidence of a contact.

tion to the object is possible within a certain error threshold. The data–approximated object representations as well as the current information about the state of the robot manipulator are entered into the system's model of the initial situation. The target object is then selected and an explanation is generated for how to achieve a grasp of the target. Figure 5 highlights the selected target object. The light lines indicate the data approximation to the object contour while the heavy outline shows the actual sensed object contour points. The arrows indicate the positions of the leading edges of the fingers for the grasp position given by the produced explanation. The proof tree for achieving *grasp–object* involves a total of about 300 nodes with a maximum depth of 10 levels.

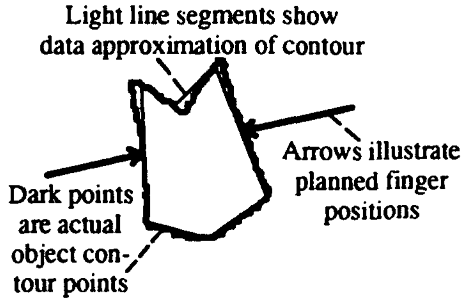


Figure 5. Grasp Target and Planned Finger Positions

Parameters are implemented by inference rules which choose their value dependent on the situation in which they are applied. When new preferences are added for a parameter, the associated rules are updated so as to choose potential maxima of the new quality function. Generalized plans refer to the consequents of the rules which choose parameter values. In this case, initially there were no preferences for the plan's opening width parameter other than it be a legal value between the width of the target object and the minimum of the maximum gripper opening and the distance to the nearest object. One of the initial rules for the opening width parameter is shown in Figure 6. This rule pertains to the case where the gripper is currently open less than the minimum opening to satisfy the surround goal in the plan. It then chooses the minimum opening which satisfies the goal because this is the closest potential maxima of the (initially flat) quality function. The rule therefore affects the separation of the arrows shown in Figure 5. After the explanation was generated, and its associated operator sequence executed, the monitored action shown in Figure 7 encounters a violation of the expected sensor readings.

```
INTRA-RULE: R4513 <-- one of three rules which are initially defined by
  FORM:                the opening width rule approximation
    (CHOSEN-OPENING-WIDTH ?GRIPPER ?X ?Y ?ANGLE ?OBJECT ?RETURN)
  ANTS:                                              find minimum required
    (GRIPPER-OPENING ?GRIPPER ?LOP4510)       opening so fingers don't collide with
    (GRIPPER-PERP-WIDTH ?GRIPPER ?SPAN)    <-- object in approximate model
    (MIN-SPAN-FOR-OBJECT ?OBJECT ?X ?Y ?ANGLE ?SPAN ?LEFT ?RIGHT)
    (SUM ?LEFT ?RIGHT ?RETURN)
    (MAX-GRIPPER-OPENING ?GRIPPER ?MAX-OPEN)
    (<= ?RETURN ?MAX-OPEN) <-- can't achieve it even in approximate model if too
    !                          wide for gripper
    (< ?LOP4510 ?RETURN)
  PARAMETER: CHOSEN-OPENING-WIDTH <-- pointer to the parameter this rule is
                                      associated with
```

Figure 6. One of the Initial Parameter Rules For Opening Width

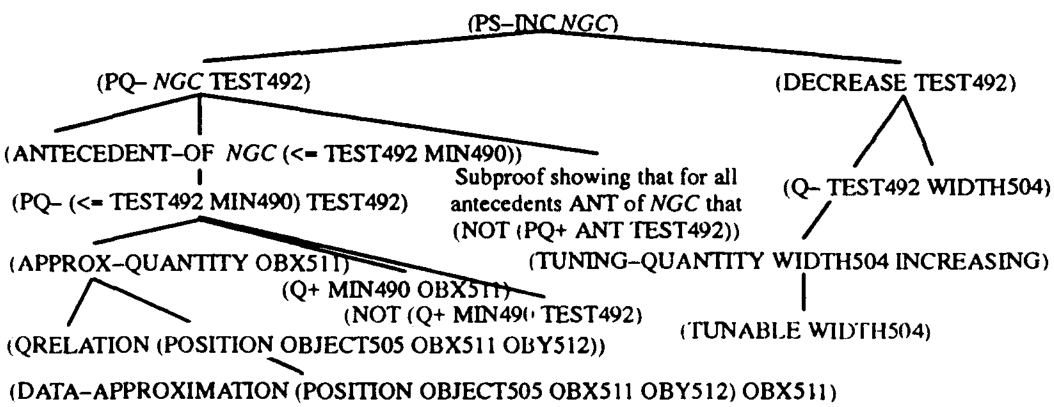
```
(MONITOR  (MOVE-ZED ?GRIPPER199764 DOWN 5 64 20 POSITION) <-- move down
          (AND (POSITION ZED ?ZPOS199309) (FORCE ZED ?ZFOR199310)
          (< ?ZFOR199310 30))        <-- force position to be recorded and all sensed
          (POSITION ZED ?LEVEL199311)    forces on this joint must be less than 30 units
          NIL ?DOC199312
          (NO-GRIPPER-COLLISION-OBJECT ?GRIPPER199764 ?X199501 ?Y199502
          ?ANGLE199503 ?WIDTH199504 ?OBJECT199756))

  terminate when position is 0 (at the table)   justification for sensor expectations (variables
                                                    bound by plan preconditions)
```

Figure 7. The Failing Monitored Action

The original explanation for the *no–gripper–collision–object* goal indicated in the above monitored action is now suspect due to the violated expectations. A sketch of the specific explanation is shown in Figure 8. This explanation for why no external force should have been sensed during the downward move of the gripper is the starting point for developing the qualitative tuning explanation. Data approximate quantities and tunable parameters employed in the plan support proof are identified and asserted as such. A proof is then constructed for increasing the probability of success of the *no–gripper–collision–object* goal. Figure 9 shows the qualitative explanation for how opening the gripper (increasing the opening–width tunable parameter) positively influences the probability that there will be no collision between the first gripper and the object. The topmost left–hand subtree establishes that decreasing the quantity TEST492 can positively influence the probability of success of the the probability of the *no–gripper–collision* predicate. This is because decreasing the quantity TEST492 can increase the probability of the predicate (<= TEST492 MIN490)

- (NO–GRIPPER–COLLISION–OBJECT GRIPPER1 289.62 267.53 –12.70 38.89 OBJECT4543
  - (LEFT–FINGER–OF GRIPPER1 FINGER1)
  - (NON–INTERSECTING–GRIPPER–FINGER–OBJECT GRIPPER1 FINGER1 289.62 267.53 –12.70 38.89 OBJECT4543)
    - *Subproof for translating finger to appropriate opening width (6 facts, 8 built–ins)*
    - *Subproof for counter–rotating object center for clipping against finger (8 built–ins)*
    - *Subproof for calculating extents and checking for overlap (7 built–ins)*
  - (RIGHT–FINGER–OF GRIPPER1 FINGER2)
  - (NON–INTERSECTING–GRIPPER–FINGER–OBJECT GRIPPER1 FINGER2 289.62 267.53 –12.70 38.89 OBJECT4543)
    - *Subproof for translating finger to appropriate opening width (6 facts, 8 built–ins)*
    - *SHARED Subproof for counter–rotating object center for clipping against finger (8 built–ins)*
    - *Subproof for calculating extents and checking for overlap (7 built–ins)*

Figure 8. Explanation Specific to Failure



Where *NGC* represents the failing predicate:

(NO–GRIPPER–COLLISION–OBJECT GRIPPER499 X501 Y502 ANGLE503 WIDTH504 OBJECT505)

Figure 9. A Qualitative Tuning Explanation

which is an antecedent of a rule supporting the *no–gripper–collision* predicate and all of the other antecedents to that rule can be shown as non–decreasing with respect to decreasing TEST492. The probability of success of the predicate (<= TEST492 MIN490) increases when TEST492 is decreased because MIN490 is influenced by a data approximate quantity. The right subtree of the proof establishes that the quantity TEST492 can be decreased because it is influenced inversely by a tunable parameter WIDTH504 which can be increased. The parameter WIDTH504 is the opening width parameter for the gripper.

The qualitative tuning explanation indicates that the chosen–opening–width parameter should be tuned. Namely, that an increasing preference be posted at the minimum opening width, which was chosen in the failure. Figure 10 illustrates the shape

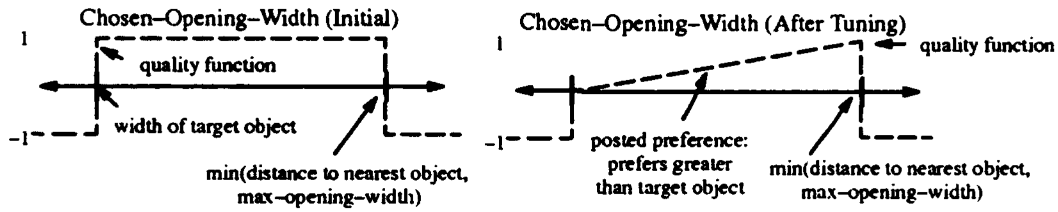


Figure 10. The Chosen–Opening–Width Parameter Quality Function After Learning New Preference

of the chosen–opening–width parameter's quality function before (left) and after (right) tuning has occurred. After parameter tuning, the rules associated with parameter are updated. Afterwards, the rule associated with this parameter reads as shown in Figure 11. This rule prefers selection of the peak of the newly re–calculated quality

```
INTRA-RULE: R4611
   FORM:
      (CHOSEN-OPENING-WIDTH ?GRIPPER ?X ?Y ?ANGLE ?OBJECT ?RETURN)
   ANTS:
      (GRIPPER-PERP-WIDTH ?GRIPPER ?SPAN)
      (DISTANCE-TO-CLOSEST-OBJECT ?OBJECT ?X ?Y ?ANGLE ?SPAN ?RADIUS)
      (GRIPPER-FINGER-PARALLEL-WIDTH ?GRIPPER ?PSPAN)
      (DIF ?RADIUS ?PSPAN ?NRADIUS)
      (MAX-GRIPPER-OPENING ?GRIPPER ?MAX-OPEN)
      (MIN ?NRADIUS ?MAX-OPEN ?RETURN)
      (MIN-SPAN-FOR-OBJECT ?OBJECT ?X ?Y ?ANGLE ?SPAN ?LEFT ?RIGHT)
      (SUM ?LEFT ?RIGHT ?MIN)
      (<= ?MIN ?RETURN)
   CONS:
   PARAMETER: CHOSEN-OPENING-WIDTH
```

Figure 11. Rule Supporting Opening–Width Parameter After Tuning

function which corresponds to opening as wide as the current situation permits. When the new more uncertainty–tolerant plan is applied, the resulting gripper finger positions are as illustrated in Figure 12 and the grasp succeeds. This is only the first tuning of the opening width parameter. The system will likely discover another potential problem: opening too wide is not tolerant of uncertainties with respect to nearby objects. The process continues with this and other parameters to be tuned.

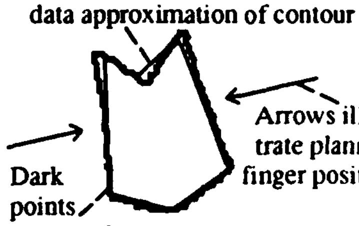


Figure 12. A Successful Wide Grasp

6 Conclusions

To construct manipulation plans for use in the real world requires one to manage a set of tradeoffs. One important tradeoff exists between the tractability of generating the plans and the extent to which success can be guaranteed. This tradeoff is particularly pronounced in dealing with uncertainty where guaranteed plans become much more expensive. The incremental plan refinement approach offers a mechanism for managing the tradeoff. Our approach makes use of explicit representations of inference rules and operators for the task to generate and refine general plans which include explicit applicability conditions. Savings is gained because uncertainty–tolerance of plans improves through refinement without the need to reason about uncertainties during plan application.

We are currently pursuing extensions to this work in several areas. One of these areas involves developing an incremental technique for learning tradeoffs which may exist among plan parameters. For instance, in grasping an object in a cluttered workspace a tradeoff exists between how wide the gripper can be opened to surround an object and which faces were selected for the grasp (because of the relationships of nearby objects). Another area of work involves an empirical comparison of the numeric probabilistic approach to stochastic actions [Christiansen90a] with an approach utilizing a simple explicit domain theory in conjunction with the approach described here.

7 Acknowledgements

We would like to thank Seth Hutchinson and Brian Falkenhainer for their comments on this paper. This research was supported by the Office of Naval Research under grant ONR N00014–86–K–0309.

References

[Bennett89] S. W. Bennett, "Learning Uncertainty Tolerant Plans Through Approximation in Complex Domains," M.S. Thesis, ECE, University of Illinois, Urbana, Il., January 1989. (Also appears as Technical Report UILU-ENG-89-2204, AI Research Group, Coordinated Science Laboratory, University of Illinois at Urbana–Champaign)

[Brooks82] R. A. Brooks, "Symbolic Error Analysis and Robot Planning," Memo 685, MIT AI Lab, Cambridge, MA, September 1982.

[Canny89] J. Canny, "On Computability of Fine Motion Plans," *Proceedings of the 1989 Institute of Electrical and Electronics Engineers International Conference on Robotics and Automation*, 1989, pp. 177–182.

[Christiansen90a] A. Christiansen, M. Mason and T. Mitchell, "Learning Reliable Manipulation Strategies without Initial Physical Models," *Proceedings of the 1990 Institute of Electrical and Electronics Engineers International Conference on Robotics and Automation*, 1990, pp. 1224–1230.

[Christiansen90b] A. D. Christiansen and K. Y. Goldberg, "Robotic Manipulation Planning with Stochastic Action," *Proceedings of the 1990 Workshop on Innovative Approaches to Planning, Scheduling, and Control*, San Diego, CA, November 1990, pp. 3–8.

[DeJong86] G. F. DeJong and R. J. Mooney, "Explanation–Based Learning: An Alternative View," *Machine Learning 1*, 2 (1986), pp. 145–176.

[Donald90] B. R. Donald, "Planning Multi–Step Error Detection and Recovery Strategies," *International Journal of Robotics Research 9*, 1 (February 1990), pp. 3–60.

[Dufay84] B. Dufay and J. Latombe, "An Approach to Automatic Robot Programming Based on Inductive Learning," *International Journal of Robotics Research 3*, 4 (Winter 1984), pp. 3–20.

[Erdmann84] M. Erdmann, "On Motion Planning with Uncertainty," Technical Report 810, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1984.

[Erdmann86] M. Erdmann, "Using Backprojections for Fine Motion Planning with Uncertainty," *International Journal of Robotics Research 5*, 1 (1986), pp. 19–45.

[Forbus84] K. D. Forbus, "Qualitative Process Theory," *Artificial Intelligence 24*, (1984), pp. 85–168.

[Gottschlich89] S. N. Gottschlich and A. C. Kak, "A Dynamic Approach to High–Precision Parts Mating," *IEEE Transactions on Systems, Man and Cybernetics 19*, 4 (July/August 1989), pp. 797–810.

[Latombe90] J. Latombe, *Robot Motion Planning*, Kluwer Academic Publishers, Hingham, MA, 1990.

[Lozano–Perez76] T. Lozano–Perez, "The Design of a Mechanical Assembly System," M.S. Thesis, Technical Report 397, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1976.

[Lozano–Perez84] T. Lozano–Perez, M. T. Mason and R. H. Taylor, "Automatic Synthesis of Fine–Motion Strategies for Robots," *International Journal of Robotics Research 3*, 1 (Spring 1984), pp. 3–24.

[Mason85] M. T. Mason and J. K. S. Jr., *Robot Hands and the Mechanics of Manipulation*, MIT Press, Cambridge, MA, 1985.

[Mason86] M. T. Mason, "Mechanics and Planning of Manipulator Pushing Operations," *International Journal of Robotics Research 5*, 3 (Fall 1986), pp. 53–71.

[Mitchell86] T. M. Mitchell, R. Keller and S. Kedar–Cabelli, "Explanation–Based Generalization: A Unifying View," *Machine Learning 1*, 1 (January 1986), pp. 47–80.

[Mooney86] R. J. Mooney and S. W. Bennett, "A Domain Independent Explanation–Based Generalizer," *Proceedings of the National Conference on Artificial Intelligence*, Philadelphia, PA, August 1986, pp. 551–555.

[Segre85] A. M. Segre and G. DeJong, "Explanation–Based Manipulator Learning: Acquisition of Planning Ability Through Observation," *Proceedings of the IEEE International Conference on Robotics and Automation*, St. Louis, MO, March 1985, pp. 555–560.

[Segre88] A. M. Segre, *Machine Learning of Robot Assembly Plans*, Kluwer Academic Publishers, Norwell, MA, March 1988.

[Taylor76] R. H. Taylor, "The Synthesis of Manipulator Control Programs from Task–level Specifications," AIM–282, Stanford Artificial Intelligence Laboratory, 1976.