Optimizing Robot Motion Strategies for Assembly with Stochastic Models of the Assembly Process

Rajeev Sharma **Steven M. LaValle** **Seth A. Hutchinson**

The Beckman Institute, University of Illinois at Urbana-Champaign

405 N. Mathews Avenue, Urbana, IL 61801

Abstract *Gross-motion planning for assembly is commonly considered as a distinct, isolated step between task sequencing/scheduling and fine-motion planning. In this paper we formulate the problem of gross-motion planning for assembly in a manner that integrates it with both the manufacturing process and the fine motions involved in the final assembly stages. One distinct characteristic of gross-motion planning for assembly is the prevalence of uncertainty involving time — in parts arrival, in request arrival, etc. We propose a stochastic representation of the assembly process that improves the robot performance in the uncertain assembly environment by optimizing an appropriate criterion in the expected sense.*

1 Introduction

Modern manufacturing systems are confronted with planning problems at many scales, ranging from long-term production control, which deals with entire factories and time scales on the order of weeks, or even years, to fine-motion planning, which deals with individual robot assembly operations. Figure 1 illustrates a typical manufacturing system, where **P**'s are the parts, **B**'s are the subassemblies, **A**'s are the assemblies and **RC**'s are the assembly robot workcells.

At the highest level, planning problems (usually called scheduling problems at this level) address the flow of parts through the assembly system. Production control determines what the assembly plant should be producing from, for example, month to month. Given a set of production goals, the shop scheduler is given the task of determining how each robot cell will respond as parts arrive at its input buffers [8]. For example, in Figure 1, workcell **RC5** receives parts from the three workcells **RC2**, **RC3**, and **RC6**, as well as some parts that are directly fed to **RC5** as input to the assembly system (i.e., parts **P4, P7, P8, P9, P10** and **P11**). It is the task of the shop scheduler to determine the order in which workcell **RC5** will process these parts.

At a lower level, each workcell confronts a number of more specialized planning problems [2]. A sequence planner determines constraints on the order in which the robot will perform assembly operations [3]. A gross-motion planner constructs the trajectories that the robot will execute in performing the tasks [5]. And, finally, a fine-motion planner determines robust, local strategies for the assembly operations, that are guaranteed to succeed, even in the presence of significant uncertainty [1]. Most often, as illustrated by the work cited above, each of these planning problems is treated in isolation.

In this paper, we take a first step toward integrating several levels of planning within a unified framework. We consider the problem of optimal gross-motion planning for a robot in an individual assembly cell, within the larger context of a full manufacturing environment. In the past, gross-motion planning has been treated as either a purely geometric problem (e.g., plan motion from point a to point b, avoiding collision with obstacles), or as an optimal control problem (e.g., find the time-optimal, or minimum energy path between point a and point b). In either case, the

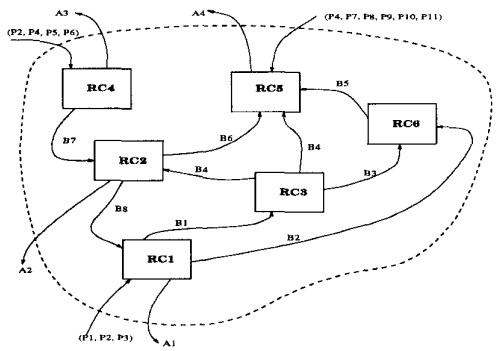

Figure 1: An assembly plant with multiple robot cells.

context in which the gross motion commands will be executed is ignored. Motions are initiated at the request of a higher-level scheduling system, and at the end of the gross motion, a fine motion assembly operation is performed.

If all aspects of the manufacturing system behaved deterministically, we could, in principal, treat gross-motion planning as a path optimization problem: derive the optimal path to move the parts from their initial positions to their destinations, on a schedule given *a priori* by the scheduler. However, real manufacturing systems are not deterministic. There is uncertainty in parts arrival time, position and orientation of parts to be manipulated, robot control, dimensions of manufactured parts, etc. Therefore, a reasonable choice is to optimize the *expected* performance of the gross-motion planner.

We characterize the problem of gross-motion planning for assembly as follows. A scheduler issues requests to the robot to grasp a particular part from a specified source, and to deliver the part to a specified destination. *A priori*, the only information regarding how these requests will be issued is in the form of a probability distribution on the set of possible part/source/destination requests. Because a fine-motion plan will often follow the execution of the gross motion, a source or destination is typically not specified as a single configuration, but is specified as a subset of the configuration space (which could in general be disconnected). The gross-motion planning problem is to derive a set of motion strategies that will produce optimal throughput of the assembly cell, in an expected sense.

Our gross-motion planning technique handles the uncertainty due to the stochastic nature of the assembly system by exploiting the concept of a Markov chain of assembly modes. In the present context, each part/source/destination request corresponds to a distinct *assembly mode*. In general each distinct manipulation that the robot performs (e.g., grasping a part, moving a part across the workcell) potentially changes the motion model or geometric model for the robot in its workcell. By using these concepts, we are able to optimize over a discrete set of possible state spaces, each corresponding to a unique

0-8186-6995-0/95 $04.00

combination of configuration space and assembly mode.

2 Problem Description

There are some aspects of the gross-motion planning that are specific to the assembly situation. These are:

Changing geometry and robot motion model. We can describe motion planning in terms of the configuration space, $\mathcal{C}$, of the robot, as the problem of finding a path that lies in space of collision-free configurations, $\mathcal{C}_{free}$. At a given point in time, the robot position is characterized by a point, $\mathbf{q}$ in $\mathcal{C}_{free}$. Thus one important step in motion planning is to establish the mapping from the workcell to the configuration space, C, of the robot, which depends on the geometry of the robot. When the robot is carrying a part or subassembly, the effective shape of the robot and load ensemble changes. We will assume that when a robot is carrying a load there is rigid relationship between the robot (gripper) and the load so that for each load, the "robot" has a different geometry. This concept of "changing" configuration space is an important aspect of the formulation of the motion planning problem as part of the assembly process. In addition to the geometric changes, the dynamics of the robot could change during the different stages of the assembly process due to the variation in loads and would need to be incorporated into the motion planner.

Preconditions for fine-motion planning. The initial and final stages of moving a part for assembly involve *fine-motion planning*. During these stages the clearances between parts becomes significant relative to the uncertainties involved; hence sensing (e.g., force or torque sensing) becomes an important part of the motion strategy. For gross-motion planning the usual approach is to ignore the fine-motion plan and consider the task of moving the robot between two points in its configuration space. Instead of defining a point-to-point motion goal, we allow the goals to be regions in the configuration space for both the grasp and ungrasp operations, based on relationships between the

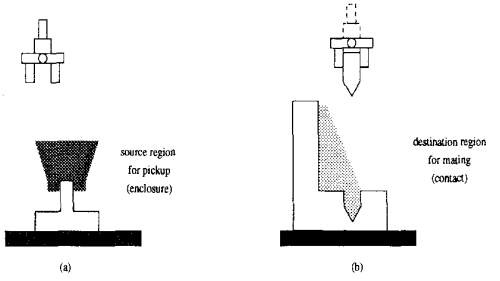

Figure 2: Example source and destination regions.

robot, the part, and the subassembly. Figure 2 shows an example of a source region and a destination region with respect to which appropriate initial and goal conditions for the gross-motion planning will be defined.

The concept of an assembly mode. Gross-motion planning for assembly can benefit by considering more time-varying elements besides the ones considered so far. This includes, for example, the priorities and costs involved in the individual assembly motion subgoals. The priorities of a given operation in turn will be tied to the scheduling of the entire manufacturing facility as we will discuss later in this section. For the gross-motion planner, we define *assembly modes* that correspond to requests for the robot to deliver a part from a source region to a destinations region.

Scheduling and the assembly process. From the viewpoint of an individual workcell, the assembly process consists of a sequence of part/source/destination requests issued by a scheduler. These requests must be serviced in the order in which they are received by the robot in that workcell, thus inducing a sequence of assembly modes. Further, in each assembly mode a particular fine-motion strategy might be used to initially grasp the specified part, or to place the part in its goal position. In order for our new gross-motion planning approach to effectively optimize trajectories in an expected sense, the gross-motion planner must have some characterization of the anticipated behavior of the scheduling system in terms of the sequencing of requests and the fine-motion plans that must be executed for each request.

There have been many approaches to scheduling in large scale manufacturing systems. These range from heuristic methods for global optimization, to relaxation-based global methods to distributed, real-time scheduling policies (e.g., [8]). In this paper, we will assume that a distributed, real-time scheduling approach is used, since such approaches scale to arbitrarily complex manufacturing systems, including job shops, flow shops, and re-entrant lines.

The behavior of the overall manufacturing system can be characterized by its underlying stochastic process. In particular, the behavior of each individual workcell can be characterized as a random process that is conditioned on the behavior of a finite set of neighboring workcells. For example, in Figure 1, the behavior of **RC3** depends only on the output of **RC1**, while the behavior of **RC6** depends only on the output of **RC1** and **RC3**. This stochastic behavior of the overall assembly system induces a stochastic behavior for each individual scheduler in the system. We assume that a scheduler has been chosen that will lead to stability of the manufacturing system. We model the resulting behavior of an individual scheduler as a Markov chain of assembly modes, since this representation is powerful enough to encode many important stochastic processes, such as a Wiener process or a Poisson process.

3 Modeling and Algorithm

In this section we develop the mathematical concepts that model the gross-motion planning for assembly as introduced so far and provide a computational method that determines the optimal motion plans under stochastic uncertainty.

Basic definitions. In addition to static obstacles, let the workcell contain a set of S *source regions*, denoted by $\{\mathcal{S}_1, \ldots, \mathcal{S}_S\}$, and D *destination regions*, denoted by $\{\mathcal{D}_1, \ldots, \mathcal{D}_D\}$. We generally allow a source or destination region to have multiple connected components. Let $\{\mathcal{P}_1, \ldots, \mathcal{P}_P\}$ denote a collection of P rigid parts. A *request* can be issued to the robot that requires a part to be picked up from a source, and delivered to a destination. In general there are PSD different requests that can be issued. It is assumed that at a given time, the robot has complete knowledge of its configuration and all parts, sources, destinations, and requests.

To characterize requests and the status of the robot with respect to requests, we introduce a finite set, M, of *assembly modes*. An assembly mode is represented by four components, $\langle p, s, d, C/W \rangle$. The first three represent the part, source, and destination respectively. The fourth component is W to represent a mode in which a request has been given, but the robot has not yet picked up the part, or is C to represent a mode in which the request has been given and the robot is carrying the part. In addition, we have a special mode, $NR \in M$, which represents the condition in which

no requests are to be processed. Under the model where we assume that the scheduling is done separately (in [10] we discuss how we can relax this constraint), at a given time only one request can be waiting. Hence we have the number of assembly modes, $|M| = 2PSD + 1$. To uniquely identify all of the possible situations that can occur in our problem, we define a *state space* as a subset of the cartesian product, $X \subseteq \mathcal{C} \times M$.

We next define the free configuration space for different modes. Let $M_w \subset M$ be the set of all modes such that a part is waiting to be picked up, and $M_c \subset M$ be the set of all modes such that the robot is carrying a part. If $m = \langle p, s, d, W \rangle \in M_w$, then

$$\mathcal{C}^m_{free} = \{\mathbf{q} \in \mathcal{C} | \ \mathcal{A}(\mathbf{q}) \cap (\mathcal{B} \cup \mathcal{S}_1 \cup \cdots \mathcal{S}_{s-1} \cup \mathcal{S}_{s+1} \cdots \mathcal{S}_S) \neq \emptyset\}, \tag{1}$$

in which $\mathcal{A}(\mathbf{q})$ denotes the robot at configuration $\mathbf{q}$, and $\mathcal{B}$ denotes the static obstacle region (see [5]). In addition to avoiding collision with static obstacles, we also require that the robot avoid collision with other source regions. We also prohibit contact with other source regions.

Suppose $m = \langle p, s, d, C \rangle \in M_c$, which implies that the robot is carrying some part, $\mathcal{P}_p$. We use the notation $\mathcal{P}_p(\mathbf{q})$ to denote the transformed part, when grasped by the robot, which is at configuration $\mathbf{q}$. When a part is being carried rigidly by the robot, the effect is that of a "new robot" described as $\mathcal{A}(\mathbf{q}) \cup \mathcal{P}_p(\mathbf{q})$. The free configuration space becomes
$\mathcal{C}^m_{free} =$

$$\{\mathbf{q} \in \mathcal{C} | \ (\mathcal{A}(\mathbf{q}) \cup \mathcal{P}_p(\mathbf{q})) \cap (\mathcal{B} \cup \mathcal{S}_1 \cup \cdots \mathcal{S}_{s-1} \cup \mathcal{S}_{s+1} \cdots \mathcal{S}_S) \neq \emptyset\}. \tag{2}$$

For $m = NR$, we have

$$\mathcal{C}^m_{free} = \{\mathbf{q} \in \mathcal{C} | \ \mathcal{A}(\mathbf{q}) \cap (\mathcal{B} \cup \mathcal{S}_1 \cup \cdots \cup \mathcal{S}_S) \neq \emptyset\}. \tag{3}$$

Controlling the robot and the assembly process. We use discrete-time representations in which k represents a *stage* (or time index). Let x_k represent the state at stage k, which simulatneously specifies an assembly mode, m_k, and the configuration $\mathbf{q}_k$. An action (or control input), u_k can be issued to the robot at any stage, and U denotes the set of possible actions. These actions can cause the configuration to change, or might influence the next assembly mode.

The *assembly process* is the finite-state Markov process that models the transitions between assembly modes. Probabilities of the form $P(m_{k+1}|x_k, u_k)$ are specified to define the mode transitions. This implies that the probability of the next mode is conditioned on the current configuration and action, in addition to the current mode. The probabilities can be chosen to model a wide variety of stochastic processes, but we have derived the probabilities from a few basic transition types: 1) the probability of receiving a p, s, d request while in mode NR; 2) the probability that the destination will change while $\langle p, s, d, C \rangle \in M_c$; 3) the probability that the source will change while $\langle p, s, d, C \rangle \in M_w$.

The first transition type is the most fundamental, and can be generally expressed as $P(m_{k+1}|m_k = NR) \geq 0$ if $m_{k+1} \in M_w$, and $P(m_{k+1}|m_k = NR) = 0$, otherwise. The second transition type can be expressed as $P(m_{k+1}|m_k \in M_c)$, which we allow to be nonzero only if m_k and m_{k+1} correspond to the same part and source. Ideally, the destination remains fixed, and $P(m_{k+1}|m_k \in M_c) = 1$ if $m_{k+1} = m_k$, and 0 otherwise. Similarly, the third type can be expressed as $P(m_{k+1}|m_k \in M_w)$, which we allow to be nonzero for any value of $m_{k+1} \in M$. Ideally, $P(m_{k+1}|m_k \in M_w) = 1$ if $m_{k+1} = m_k$, and 0 otherwise. Any of these transition probabilities can be derived from an underlying Poisson process, which has been used extensively in the modeling of scheduling systems.

Suppose that the robot has an action, $FMP \in U$, that represents fine-motion planning. We assume that this action can only be applied at appropriate regions in the state space. To grasp or ungrasp a part, the robot can choose this action from state x_k (causing a fine-motion operation to be performed), which results in some state, x_{k+1}. At a source region, $m_k = \langle p, s, d, W \rangle$ deterministically changes to $m_{k+1} = \langle p, s, d, C \rangle$, and at a destination region, $m_k = \langle p, s, d, C \rangle$ deterministically changes to $m_{k+1} = NR$. These transitions could alternatively be defined probabilistically. The condition of reaching a source or goal region can be formally defined in one of two ways: 1) the robot is in contact with the source (or destination) region; or 2) the robot is enclosed in the source (or destination) region.

We next define a *state transition distribution* as $P(x_{k+1}|x_k, u_k)$, and provide an example in which $\mathcal{C} \subseteq \Re^2$, and the robot is limited to translational motion. More complicated motions will be considered in the next section, including modeling of a redundant manipulator. We define the action space as $U = [0, 2\pi) \cup \{\emptyset, FMP\}$. If $u_k \in [0, 2\pi)$, then $\mathcal{A}$ attempts to move a fixed distance toward a direction in $\mathcal{C}$. If $u_k = \emptyset$, then the robot remains motionless.

We define a *strategy at stage k* as a function $\gamma_k : X \to U$. For the examples that we present in this paper, γ_k will be the same for all k (i.e., each robot action depends only on the current state, and not the particular stage). In [10] we discuss how assembly situations that require time dependency can also be handled.

Evaluating robot performance. We define a nonnegative, real-valued *loss functional* of the form

$$L(x_1, ...x_K, u_1, ..., u_K) = \sum_{k=1}^{K} l_k(x_k, u_k). \tag{4}$$

Above, $l_k(x_k, u_k) = 0$ if $m_k = NR$. Otherwise, $l_k(x_k, u_k)$ is the expected time to grasp or ungrasp the part from state x_k if $u_k = FMP$, and $l_k(x_k, u_k) = \Delta t$ for all other actions.

The cost that is minimized for our problem thus becomes the aggregate of times that parts wait before being delivered. If there are no requests (i.e., $m_k = NR$), then no penalty is received. To reduce the loss over a long period of time, the robot will prefer actions that bring the assembly mode back to NR as quickly as possible. The goal of the planner is to determine a strategy that minimizes the expected loss.

Determining optimal strategies. Suppose that for some k, the optimal strategy is known for each stage $i \in \{k, \ldots, K\}$. Let $\bar{L}^*_k(x_k)$ denote the expected loss as a function of state, obtained by starting from stage k, and implementing the portion of the optimal strategy, $\{\gamma^*_k, \ldots, \gamma^*_K\}$.

The principle of optimality states that $\bar{L}^*_k(x_k)$ can be obtained from $\bar{L}^*_{k+1}(x_{k+1})$ by the following recurrence:
$\bar{L}^*_k(x_k) =$

$$\min_{u_k} \left\{ l_k(x_k, u_k) + \sum_{x_{k+1}} \bar{L}^*_{k+1}(x_{k+1}) P(x_{k+1}|x_k, u_k) \right\}. \tag{5}$$

The goal is to determine the optimal action, u_k, for every value of x_k, and every stage $k \in \{1, \ldots, K\}$. One can begin with stage $K + 1$, and repeatedly apply (5) to obtain the optimal actions. At stage $K + 1$, we declare that $\bar{L}^*_{K+1}(x_{K+1}) = 0$. The function $\bar{L}^*_K$, can be determined

from $\bar{L}^*_{K+1}$ through (5). Using the $u_K \in U$ that minimizes (5) at x_K, we define $\gamma^*_K(x_K) = u_K$. We then apply (5) again, using $\bar{L}^*_K$ to obtain $\bar{L}^*_{K-1}$ and γ^*_{K-1}.

In our implementation, we determine optimal strategies numerically, by successively building approximate representations of $\bar{L}^*_k$. This offers flexibility, since analytical solutions are very difficult to obtain for gross-motion planning problems with this form of uncertainty, and have only been previously obtained by considering very specific cases [9]. Each dynamic programming iteration can be considered as the construction of an approximate representation of $\bar{L}^*_k$. We decompose the state space into cells of uniform size to construct a good approximation of $\bar{L}^*_k$. We obtain the value for $\bar{L}^*_k(x_k)$ by computing the right side of equation (5) for various values of u_k, including $u_k = \emptyset$. The value for $\bar{L}^*_k(x_k)$ is obtained by linear interpolation.

After some finite number of iterations, for every state, the optimal actions stabilize, and the representation can be utilized as a state-feedback controller. The stabilization occurs due to stationarity of the assembly process, control of the robot, and other modeling components. Therefore a specific choice of K is not needed. Also, at each iteration of the dynamic programming algorithm, we only retain the representation of $\bar{L}^*_{k+1}$ while constructing $\bar{L}^*_k$. To execute a strategy, the robot uses the resulting representation, which we designate as $\bar{L}^*_1$). The optimal action can be obtained from any real-valued location $x \in X$ though the use of (5), linear interpolation, and the approximate representation of $\bar{L}^*_1$.

The time complexity of the algorithm increases exponentially in the dimension of the state space (as is the case with most algorithms for the basic motion planning problem even without uncertainty [5]), but for a given dimension the algorithm is quite reasonable. Computational complexity of the algorithm increases linearly with the number of parts, destinations, and sources. Significant performance improvement is possible through parallelization.

4 Specific Assembly Situations

In this section we present computed solutions for several problems that involve the transfer of parts in a workcell for assembly. The problems are chosen to illustrate the flexibility and generality of our approach, and contain particular versions of the mathematical model presented in the last section.

The first example is designed to illustrate many of the basic concepts. It involves a rigid robot that translates in a planar workcell cluttered with obstacles (see Figure 3). There are two different parts that can be moved from either of two sources to either of two destinations. There are consequently 17 possible assembly modes. The probability that a request will appear at stage $k+1$ while $m_k = NR$ is given to be 0.05. In addition, we declare that all p, s, d combinations are equally likely to occur. We assume that once a p, s, d combination is given to the robot, it will not change or be retracted until part p is delivered to destination d. The robot moves at velocity $\|v\|\Delta t = 3.0$ with workcell being 100 units square.

Figures 4.a and 4.b depict the level-set contours of $L^*_1(x_1)$ for assembly modes $\langle 1,1,1,W\rangle$ and $\langle 1,1,1,C\rangle$, respectively. In Figure 4.a there is a minimum at the first source region, and in Figure 4.b the minimum appears at the destination region. For translational motion, the negative gradient of $L^*_1(x_1)$ represents the direction of motion for the robot. Hence, $L^*_1(x_1)$ is similar to a numerical navigation function [4, 5]; however, in our work, we obtain the representation of $L^*_1(x_1)$ as a by-product of determining the optimal strategy.

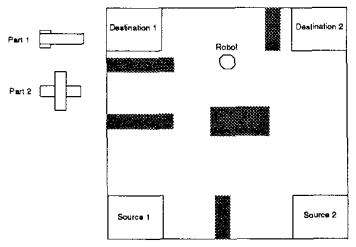

Figure 3: A translating robot problem.

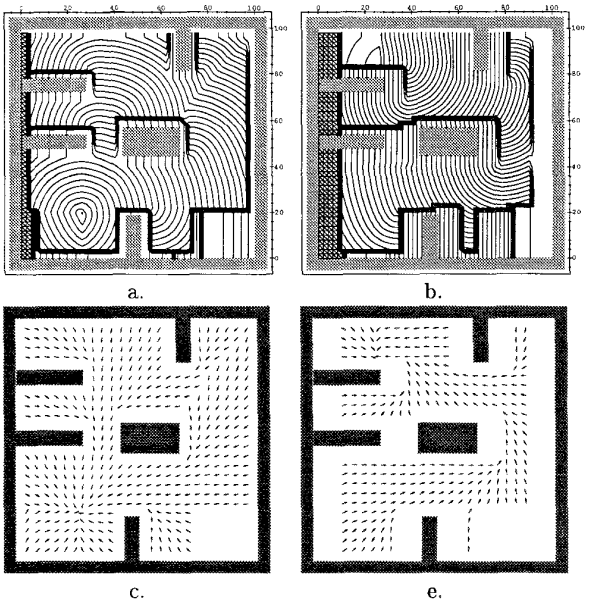

Figure 4: a) Level-set contours of $L^*_1(x_1)$ for $e = \langle 1,1,1,W\rangle$; b) the contours for $e = \langle 1,1,1,C\rangle$; c) the optimal actions as a vector field for $e = \langle 1,1,1,W\rangle$; d) the optimal actions for $e = \langle 1,1,1,C\rangle$

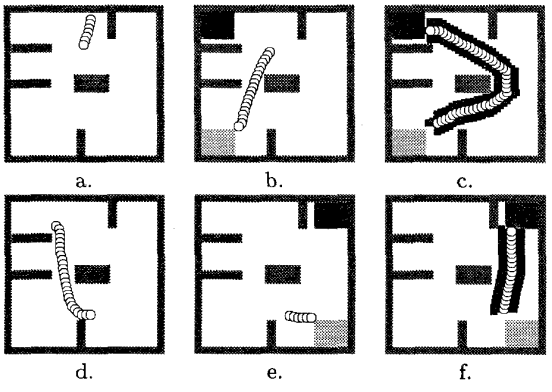

Figure 5: A simulation result of γ^*.

Figures 4.c and 4.d depict the optimal strategy γ^* for assembly modes $\langle 1,1,1,W\rangle$ and $\langle 1,1,1,C\rangle$, resectively. Each arrow indicates the direction of motion (specified as $u_k = \gamma^*(x_k)$) for the robot, from that particular state location. Figure 5 presents a simulation of the robot in the workcell

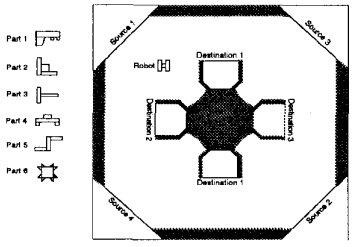

Figure 6: An assembly problem with 145 modes.

over a period of time, under the implementation of γ^*. By sampling an assembly mode sequence, $\mathbf{m}$, from the Markov process, a state trajectory is obtained. The beginning of the trajectory is depicted in Figure 5.a, and it concludes in Figure 5.f. To save space in the figure, many frames are superimposed, and a new picture is shown each time the assembly mode changes. The first column of Figure 5 corresponds to execution during the NR mode. The second column corresponds to modes in M_w, and the final column corresponds to modes in M_c. In the last two columns, the source and destination regions that correspond to the issued request are shaded. In the final column, the part that is carried by the robot is shaded in black.

There are at least two interesting behaviors to note in this solution. When the assembly mode is NR, the robot moves to a location in the lower portion of the workcell to reduce the expected time to deliver a part that might appear. This corresponds to reducing the setup time in a scheduling system, and is hence a preferred behavior for the robot. Another behavior to note is how the changing geometry affects the trajectory of the robot. In Figures 5.b and 5.d the robot does not carry a part, and hence is able to move through a narrow opening. However, in Figure 5.c the robot carries a part, and consequently must take a longer route to reach the destination.

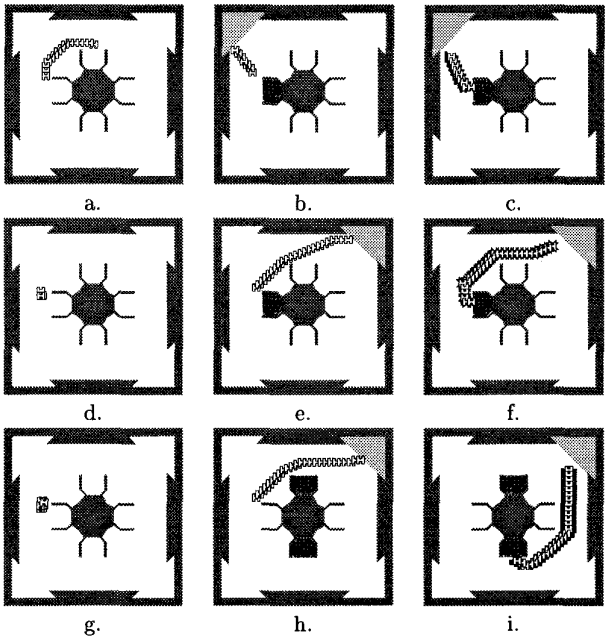

Figure 7: A simulation result of γ^*.

Figure 6 involves a translating robot problem in which there are 6 parts, 4 sources, and 3 destinations. In addition, Destination 1 has two connected components, in which the robot must choose the best delivery region in terms of loss. For this problem there are 72 different kinds of requests (which are equally likely to occur), which results in 145 assembly modes. Figure 7 shows a sample of the execution under γ^*.

Note the behavior of the robot with respect to the connected components of Destination 1. At the start of the time period captured in Figure 7.h, the robot receives a request to move Part 6 from Source 3 to Destination 1. The robot picks up the part from Source 3, and chooses to deliver it to the lower component of Destination 1 (Figure 7.i). This behavior was based on the computation of the optimal strategy for that particular position of the robot in the NR mode.

We show next how the principles in this paper also apply to motion planning for manipulators performing assembly operations. Each manipulator is described by a set of links that are connected by rotating joints, and the final link contains an end-effector that can grasp or ungrasp an object. The configuration space is generated by taking the cartesian product of the real-valued intervals that correspond to joint angles. To apply our techniques, we must consider obstacles, source regions, and destination regions in the configuration space of the manipulator. We require that the entire manipulator avoids collision with static obstacles. To define the source and destination regions in the configuration space, we only consider the end-effector as the robot. This is a reasonable choice since fine-motion planning essentially would involve only the end-effector along with the part that it could be carrying. Each joint of the manipulator can be independently controlled with bounded angular velocity.

For the first manipulator problem, there are two parts, two sources, and two destinations (see Figure 8.a). There are three links that move in the plane, and an end-effector that always maintains the same orientation. There are joint limits that prevent the joints from executing a circular motion. Figure 9 shows a sample of the execution under different requests and under some of the 17 possible assembly modes. The third column shows the part being "carried" to the destination region.

For the second manipulator problem, there are two sources, four destinations, and one part (see Figure 8.b). There are fixed limits for each joint, and the end-effector is rigidly attached to the last joint. Figure 10 shows a sample of the execution. One of the sources has two connected components, which causes the particular component to be chosen during execution depending on the current state.

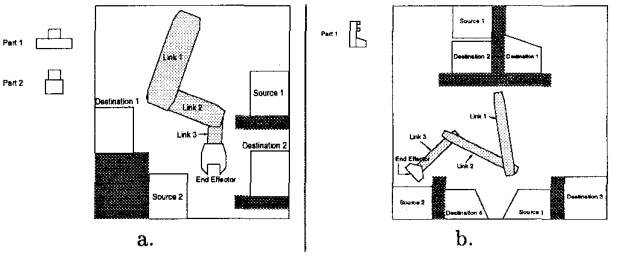

Figure 8: 3-DOF manipulator examples.

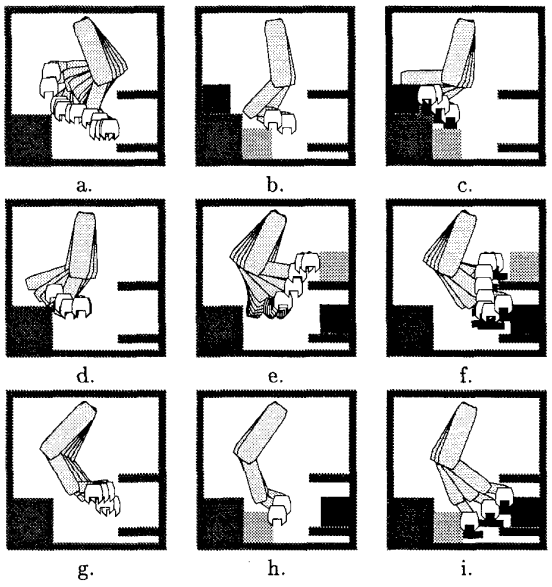

Figure 9: A simulation result of γ^*.

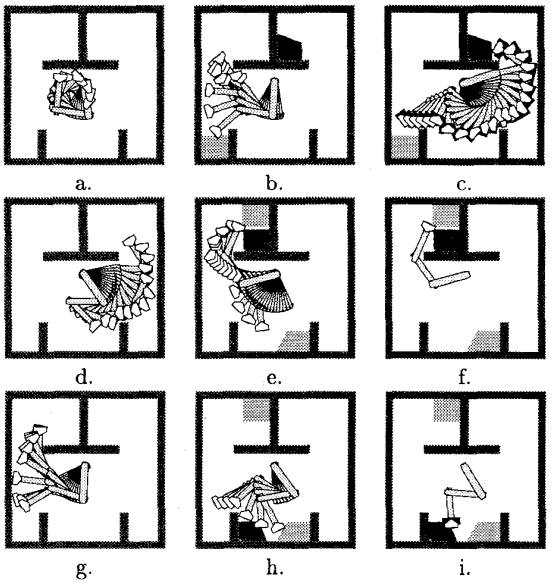

Figure 10: A simulation result of γ^*.

5 Discussion and Conclusions

Apart from the specific assembly situations presented, our framework is capable of representing a larger class of motion planning problems, making it applicable to more assembly situations. This includes a situation in which the model of the assembly process varies over time, e.g., demand arrival varies according to the stage of some other process in the assembly plant. Other situations include moving obstacles, a robot carrying multiple parts, etc. ; these are discussed in [10].

In this paper we have focused primarily on the form of uncertainty in assembly that involves time. This is part of a broader class of uncertainty involving "environment predictability" as categorized in [7]. The positional and control uncertainty associated with the robot are more relevant in the final stages of the assembly, i.e., for fine-motion planning. These forms of uncertainty can be factored into the same probabilistic framework. A treatment of additional forms of uncertainty in fine-motion planning (position and control uncertainty) that is compatible with our treatment of the uncertainty is reported in [6].

Robot motion optimization over time is important because of the repetitive nature of the assembly tasks, since efficient motion planning eventually translates into an increased throughput. While traditional gross-motion planning is not considered part of assembly planning, the results in this paper show how the assembly performance can be improved by considering gross-motion planning as part of the assembly process. At one level, this helps in developing interplay between gross-motion planning and scheduling; at another level it develops better interface between gross-motion planning and fine-motion planning in the presence of uncertainty.

The use of the stochastic assembly process provides a flexible way of capturing the time-varying element of assembly operation at different levels. As demonstrated in the specific models that were developed and discussed, many different approaches are possible for representing the uncertainty and the right choice would depend on the particular assembly situation.

References

[1] M. Erdmann. Using backprojections for fine motion planning with uncertainty. *Int. Journal of Robotics Research*, 5(1):19–45, 1986.

[2] S. Gottschlich, C. Ramos, and D. Lyons. Assembly and Task Planning: A Taxonomy. *IEEE Robotics and Automation Magazine*, 1(3):4–12, 1994.

[3] L. S. Homem de Mello and S. Lee. *Computer-Aided Mechanical Assembly Planning.* Kluwer, 1991.

[4] E. Rimon and D. E. Koditschek. Exact Robot Navigation Using Artificial Potential Fields. *IEEE Trans. Robotics and Automation*, 8(5):501–518, 1992.

[5] J. C. Latombe. *Robot Motion Planning.* Kluwer, 1991.

[6] S. M. LaValle and S. A. Hutchinson. An Objective-Based Stochastic Framework for Manipulation Planning. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 1994.

[7] S. M. LaValle and R. Sharma. A framework for motion planning in stochastic environments: Modeling and analysis. In *Proc. IEEE Int. Conf. Robotics and Automation*, 1995.

[8] J. R. Perkins, C. Humes Jr., and P. R. Kumar. Distributed Scheduling of Flexible Manufacturing Systems: Stability and Performance. *IEEE Trans. Robotics and Automation*, 10(2):133–141, 1994.

[9] R. Sharma. Locally Efficient Path Planning in an Uncertain, Dynamic Environment using a Probabilistic Model. *IEEE Trans. Robotics and Automation*, 8(1):105–110, February 1992.

[10] R. Sharma, S. M. LaValle, and S. Hutchinson. Optimizing Robot Motion Strategies for Assembly with Stochastic Models of the Assembly Process. Tech Report UIUC-BI-AI-RCV-94-11, Univ. of Illinois, 1994.