Assembly and Task Planning: A Taxonomy and Annotated Bibliography

Susan Gottschlich†, Carlos Ramos‡, and Damian Lyons¶

December 27, 1993

Abstract

After over two decades of research in the field of assembly and task planning, an IEEE Technical Committee on Assembly and Task Planning has been created. One of the first activities of this committee has been to develop a taxonomy of the field. We will now present this taxonomy, which should be viewed as a dynamic description of the field that will change shape as the field progresses. The reasons for developing such a taxonomy are many: to aid in building a coherent picture of the field which is particularly useful to newcomers to assembly and task planning, to help active researchers better understand how their work relates to the rest of the field, and finally such a taxonomy is useful for administrative purposes such as as an aid to program committee members and editors for selecting reviewers and as an aid in selecting keywords to be associated with various articles. To further aid the reader of this taxonomy, we have incorporated into it an extensive, but by no means complete, bibliography.

†Susan Gottschlich is with the Center for Advanced Technology and the Electrical, Computer and Systems Engineering Dept. at Rensselaer Polytechnic Institute, Troy, NY 12180-3590
‡Carlos Ramos is with the Department of Electrical Engineering and Computing Faculty of Engineering, Oporto University, Rua dos Bragas, 4699 Porto CODEX, Portugal
¶Damian Lyons is with Philips Laboratories, North American Philips Corporation, Briarcliff Manor, NY 10510
1 Introduction

The age of robotics was issued in with much excitement as engineers and scientists predicted that soon a factory could be entirely run by these flexible machines. Companies rushed to incorporate robots into their assembly lines as a means to save on direct labor costs in their manufacturing processes while producing consistent high quality products. Unfortunately, almost as fast as the interest in robotics surged into a frenzy it ebbed away, leaving many people disillusioned with their newly acquired robots.

What went wrong? Why didn’t these robots meet up to expectations? There are certainly many factors that led to this early downfall of robotics, but perhaps the most important factor was that while these robots were capable of doing almost any task, the effort that was required to program these robots to do the desired tasks had been vastly underestimated. In a robot program, every move of the robot had to be completely spelled out. If there was any uncertainty in the motions of the robot, for instance if the parts needed to perform a particular task were not presented to the robot in very precise locations each time the task was to be performed, these hard coded robot programs would fail miserably. The robot needed to be endowed with sensors for perceiving its environment, and the readings from these sensors had to be processed and used to guide these robot programs. Unfortunately, developing such complicated programs was exceedingly labor intensive, thus merely shifting the burden of the labor effort from manually performing the desired task to manually programming the robot to carry out the desired task.

It soon became clear that there needed to be an easier way to program a robot. Ideally a programmer would be able to tell the robot how to perform a task in the same way he or she would tell a human operator. For instance, the programmer might tell the robot to assemble a flashlight using the parts contained on the input tray. Given this high level instruction the robot system should then be able to synthesize a plan of the motions and sensing operations needed to actually put the flashlight together. Hence the robot needs to be capable of taking a high level command and converting it into a series of low level motion, sensing, and analysis operations.

The field of assembly and task planning arose to address the issue of how a detailed operation plan could be automatically synthesized given a high level description of a task to be performed. Task planning is aimed at planning general robot tasks while assembly planning is aimed at specifically planning assembly tasks. The reason for studying assembly planning is twofold: a large number of the tasks robots are required to perform are assembly tasks or assembly-like tasks, and the assembly domain presents many complex and yet feasible tasks to be performed. Much of what is learned in studying the specific field of assembly planning can be generalized into non-assembly oriented tasks.

Today, two decades after the first realization that assembly and task planning capabilities would be needed for the effective utilization of robotics, the field of assembly and task planning has blossomed into a very broad and active area of research. In the early development stages of this field, there was much enthusiasm that the problems of assembly and task planning could be solved in a straightforward manner, and languages such as IBM’s Autopass [56] were developed that were supposedly capable of “compiling a plan” from a high level program description. As the difficulties in actually implementing such a language
were exposed, a much more healthy respect for the difficulties encountered in assembly and task planning research emerged. This realization paved the way for a sizable assembly and task planning research community still very active many years later.

As a measure of the interest in this field, it has given rise to numerous workshops[36], books[61, 19], special issues[49], and now a technical committee within the IEEE Robotics and Automation Society. To provide a glimpse of the issues and problems studied in this field, a taxonomy of the field of assembly and task planning will now be presented. According to this taxonomy, the field has been broken down into three major areas: the integration of design and manufacturing as it pertains to assembly planning, general off-line assembly and task planning, and on-line planning, execution, and reaction. The first area essentially concerns how to develop plans detailing the procedure that will be involved in manufacturing a product. It is envisioned that this type of planning will be done during the design phase so that it may influence the actual product design. Further details are given in Section 2. Off-line assembly and task planning is a large area encompassing a diverse set of planning methodologies that, taken together, are capable of producing a detailed robot plan. In the case of assembly plans, these planners usually rely on the output of sequence planners discussed in Section 2. The various planning methodologies that make up this area will be further explored in Section 3. The final area addresses on-line execution issues such as how to develop plans on-line, how to execute and monitor a plan that was developed off-line, and how to react to various situations that arise during plan execution. These issues will be further discussed in Section 4.

2 Planning and Integration of Design and Manufacture

Traditionally, the design and manufacturing phases of product development have been completely segmented and carried out sequentially. More and more, it is necessary to consider the manufacturing process during the design phase of product development. Thus, by planning certain components of the assembly process during the design phase, assembly planning results can be fed back into the design process so that the design may be adjusted accordingly. This concurrent engineering methodology is needed to insure a high quality end product; it is unimportant how good a product design may be if it cannot be manufactured efficiently and reliably. Areas of research that are particularly relevant here are as follows.

2.1 Assembly Representations

Before any assembly planning can be done, a representation of the assembly parts and the assembly operations involved must be developed. As for any representation, a good assembly representation will allow the planning system to automatically make deductions from the information contained in the representation in a straightforward and efficient manner. Thus the work on assembly representation focuses on how to create representations that facilitate the planning activities. For work on assembly representations see [82, 40, 88, 81, 77, 58].
2.2 Workcell planning

A robot will require resources to carry out a task, for instance in an assembly task a robot may require parts feeders, fixtures, tools, and special purpose end effectors. Workcell planning involves planning the resources that will be necessary for completing a given task. See [45] for an example of work in this area.

2.3 Sequence planning

Given a task which requires several operations to be performed, it is important to be able to develop a feasible and optimal order in which these operations are to be performed. For instance, in an assembly task, the various operations may involve putting a gasket on an engine block, putting the head on the gasket and tightening the head bolts that hold the head, gasket, and block together. While the order in which these particular operations must be performed may be obvious, this is often not the case with more complicated assembly tasks. That is why it is important to be able to automatically synthesize the sequence of operations required and what resources will be necessary for each operation. An assembly sequence planner considers the order in which the component parts will be assembled. Note that sequence planning often relies on being able to develop and analyze other types of plans, such as a motion plan or a grasp plan. Additionally, sequence planning often takes into account other non-assembly-like manufacturing operations so that they are appropriately incorporated into the overall sequence. Sequence planning has been studied by many researchers, for instance [4, 5, 6, 22, 20, 45, 54, 93, 94, 64, 95, 89, 96, 19, 30, 97, 57, 76].

2.4 Determination of Mating Pose

At some point, if the robot is to put a head gasket on an engine block, it must know the pose transformation relating the position and orientation of the gasket at the end of this operation. Although pose transformations may be specified in many different forms, they require that a minimum of six numbers be specified. Since these are very cumbersome for a human programmer to measure and specify, it is often desirable to instead have the programmer specify a symbolic description of the desired position of the head gasket relative to the block. For instance the desired position may be specified as head_gasket_hole_1 aligned with block_hole_1 and head_gasket_hole_2 aligned with block_hole_2. Given this symbolic description and geometric models of the parts involved, it is possible to be able to synthesize the pose transformation automatically. See [81, 40, 77] for discussions concerning automatic pose determination.

3 Off-line Planning under Uncertainty

Once the object has been designed, the assembly process decided upon, and the sequence of operations specified, it is necessary to be able to synthesize a plan for carrying out each specific operation. An operation such as stack block_a on block_b may require that the robot first locate block_a and block_b, then move to block_a, pick up block_a, move block_a over
to block $b$, stack block $a$ on block $b$, and finally release block $a$. Such a sequence of tasks, when augmented with numerical parameters, can be considered to be a task plan. Thus, task planning is a fundamental planning issue that must be addressed. Task planners often deal with high level issues and thus rely on lower level planners to concentrate on lower level issues. In particular, gross motion planning for assembly operations, and fine motion planning in general are two other technologies usually utilized in assembly and task planning situations. As indicated by the title, one thing that sets assembly and task off-line planning apart from more general off-line planning applications is the fact that it is often necessary to be able to model and account for uncertainty in the planning, as can be seen in the following discussion.

### 3.1 Task Planning

Task planning focuses on deciding what actions will be needed to bring about a particular operation and in what order the actions should be performed. Actions are considered at an abstract level: sensory actions, gross motion actions, fine motion actions, grasping actions, etc. More specific planners such as gross motion planners, fine motion planners, and grasp planners are invoked by the task planner to specify the details of the individual actions. The task planner is also dependent upon a sequence planner to determine the order of the operations that it plans; sequence planning may be performed prior to or during task planning. Many people have considered task planning, for instance see [47, 62, 61, 66, 48, 44, 78, 71, 14, 13]. The field of task planning is in many ways very closely related to the broader field of classical AI planning as is described in [1, 28, 90].

#### 3.1.1 Planning sensory actions

Sensory actions are usually required to locate the various objects in the work cell. Sensory actions may also be needed for other reasons, for instance to measure the force exerted on the robot by its environment. It is necessary to plan what types of sensory actions will be required and at what point in the operation they will be required. See [46, 38, 79] for further work.

#### 3.1.2 Planning manipulation actions

In order for a robot to manipulate an object, several “phases” of a robot motion are required. For instance, the robot usually approaches the object with a gross motion, locates itself directly over the object using a fine motion, and finally obtains the object with a grasp motion or grasp. Thus it is necessary to plan the individual manipulations that will be required and when: gross motions, fine motions, grasps, regraps, and ungrasps. See [91] for a specific treatment of the planning of manipulator actions, additional references given below.
3.1.3 Uncertainty propagation

There will be uncertainty in every parameter of any real robot system. Often times a planner must be able to quantify this uncertainty to decide if the various actions to be performed will succeed in accomplishing their goals. However, every action has an effect on the system uncertainties, sometimes reducing and sometimes increasing the amount of uncertainty. Therefore the planner must be able to characterize the change in the pertinent uncertainties afforded by each action, and to be able to propagate these changes across a sequence of actions. Uncertainty propagation is discussed in [9, 73, 86, 70].

3.2 Planning to Reduce Uncertainty: Grasping and Parts Feedings

Certain manipulations, such as grasping and parts feeding, can actually be used to reduce the uncertainty in the position and orientation of objects to be assembled or otherwise manipulated. There is a large body of work on grasp and regrasp planning, for instance see [75, 10, 50, 55, 69, 17, 44], which does not necessarily consider the uncertainty in the position and orientation of objects. Of particular interest in assembly and task planning, however, is recent work focussing on the use of grasping and parts feeding to actually reduce uncertainties. Work in grasping to reduce uncertainties includes [80, 37]. Related work in parts-feeding is discussed in [7, 11, 35].

3.3 Gross Motion Planning

Sometimes referred to as path planning, gross motion planning is necessary to plan a path for bringing the end-effector to an assembly component so that it may be grasped, for bringing the assembly component to a fixture, bringing a different assembly component to the first so the two may be assembled, etc. Uncertainties are generally ignored in gross motion planning because they are considered to be small relative to the clearances allowed between the objects in the work cell. If necessary an overly conservative motion plan can almost always be developed to account for the uncertainties. Gross motion planning, similar to grasp planning with no uncertainty, is not itself considered an assembly and task planning issue, but a technology on which sequence planners, task planners, and fine motion planners depend heavily. For a general treatment of gross motion planning and collision avoidance, see [9, 53, 60]. The work presented in [42, 51] treats gross motion planning for assembly in particular.

3.4 Fine Motion Planning

When the clearances allowed between parts in an assembly are small relative to the uncertainties, which is often the case, it is necessary to develop a fine motion plan for assembling the parts. Fine motion planning relies on motion with sensory feedback, in particular force and torque feedback, to overcome the uncertainties as the assembly operation proceeds. Many researchers have studied fine motion planning. For a sampling of recent work in this area see [41, 63, 12, 26, 24, 18, 85, 84, 74, 3, 43, 92, 59].
3.4.1 Representing uncertainties in world model

In order to explicitly consider uncertainties, a means for representing these uncertainties must first be developed. Typically, the various uncertainties are categorized into three classes: initial position uncertainty, control uncertainty, and model uncertainty. Initial position uncertainty is the uncertainty in the position of the grasped part relative to the fixed part at the beginning of the fine motion plan. Control uncertainty is the additional amount of uncertainty accrued per unit distance as the grasped part is moved by a non-ideal robot. Finally, model uncertainty is the uncertainty in the models of the parts used in planning. Lozano-Perez, Mason and Taylor first proposed a representation for initial pose (sensor) and control uncertainty in [63] that is used in many planning systems. Model uncertainty is treated in [25, 83, 41, 24].

3.4.2 Error detection and recovery in fine motion planning

If it is not possible to develop a plan that is guaranteed to succeed, a plan that may succeed might instead be developed as long as it is possible to detect when the plan has failed during the execution of the plan. Donald discusses error detection and recovery at length in [23].

4 On-line Planning, Execution and Reaction

Having developed a plan, the next step, clearly, is to carry out the plan with a robot or robots. Early work in this area considered the execution phase to be similar to executing a computer program: a simple instruction by instruction “program-like” execution. However, run-time uncertainty in resource availability, incomplete or erroneous models of physical processes and objects, and unforeseen events, all conspire to ensure the inadequacy of “program-like” execution of task plans. There are a number of approaches to addressing this problem: Explicit monitoring steps can be embedded in plans, plan actions be reactively scheduled to suit the execution environment, plans can be dynamically “patched” or improved to counter run-time problems, and systems can be built that can act usefully independent of any explicit plans.

4.1 Plan monitoring and error recovery

Grasp planning and fine motion planning depend on a model of an ideal grasping action and a fine motion which is somewhat simplified from that of an actual grasping action or fine motion. Hence, it is quite possible that the assembly will not go according to plan because, for instance, a fine motion exhibits some dynamic response that was not predicted by its model. It is necessary to monitor plan execution, detect such errors, and recover from them. Recovery sometimes involves replanning (or rescheduling), which may require that the plans (resource, sequence, task, gross motion, grasping, fine motion, etc) be resynthesized given an updated representation for planning. See [39, 34, 52, 87] for work on plan monitoring and error recovery.
4.2 Reactive Scheduling

Small run-time disturbances in resource availability and action failures are almost inevitable in any normal manufacturing environment. Fox and Kempf [31] were among the first to address these issues and to comment that this uncertainty could be even exploited to reduce complexity in assembly sequencing. The basic idea is to have robot action controlled by a run-time scheduler. The assembly task plan input to the scheduler should be minimally constrained (as in e.g. a hypergraph [20]). The scheduler then uses information from its sensors about current resource availability, the progress of the assembly task, and so on, to determine which action to execute next. Work in this area include [15, 31, 98, 66].

4.3 On-line planning

Reactive scheduling is based on the notion that a complete (in the sense of containing all possible legal sequences) working plan is known — a valid assumption to make in manufacturing environments most of the time. In situations where either a complete plan is impossible (due to uncertainty or combinatorial complexity), or the task has so changed that the current plan has become useless (due to unexpected changes in goals or in the class of environment in which the task must be carried out), the ‘working plan’ assumption is invalid. In such cases it is necessary to have an on-line planning capability. Although some of the objectives of on-line planning are similar to those of off-line planning, there are sufficient differences to merit this separate section. On-line planning typically involves time-constrained algorithms such as anytime algorithms [21], and is oriented towards either local improvement [32] or iterative improvement [68]. For further work in this area see [65, 72, 29].

4.4 Behavior-Based Actions

The key advantage of the behavior-based approach (e.g. [8]) to robot control is that it allows for fast, robust action in the absence of an explicit plan. Typically this approach has been used in application domains such as autonomous or exploratory robotics, where unexpected events and the absence of an explicit plan are more typical than in manufacturing applications. Some researchers have concentrated on behavior-based systems with no explicit plan or model component (typically in mobile robot applications), e.g., [8, 33] where others have considered where a behavior-based approach complements or can be used in conjunction with the standard ‘plan-then-execute’ paradigm, e.g., [68, 67, 27, 2, 16].

Acknowledgements

Many people within the IEEE Assembly and Task Planning Technical Committee have leant a hand in developing this taxonomy, providing references, and putting together this article. We would especially like to thank Randy Wilson, Ken Goldberg, and Sukhan Lee for their help in this process.
References


