September 1993

Characterization of Functionality in a Dynamic Environment

Luca Bogoni
University of Pennsylvania

Ruzena Bajcsy
University of Pennsylvania

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-93-76.

Characterization of Functionality in a Dynamic Environment

Abstract

Identifying the functionality in objects means to be able to associate a purpose with them in a specific environment. The purpose depends on the intention of the agent and on the applicability of the object in a particular task. In our investigation of functionality we focus on functionalities which involve changes of physical relation and properties between objects in the environment. A formal model, based on Discrete Event Dynamic System Theory (DEDS), is introduced to define an interactive task for recovering and describing functionality. To observe and control the recovery process we introduce the notion of *piecewise observability* of a task by different sensors. This allows the description of a dynamic system in which neither all events nor the time of their occurrence may be predicted in advance. We have developed an experimental system consisting of actuators and both force and position sensors, for carrying out the interactive recovery of functionality. In particular, we demonstrate how this approach can be used by carrying out some experiments investigating the functionality of piercing. Furthermore, we discuss the importance of a multisensory approach for the observation and interpretation of functionality.

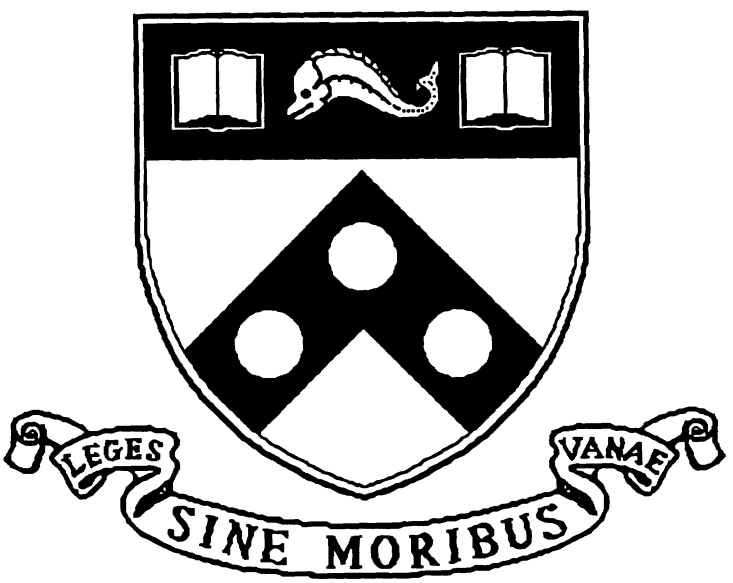
Comments

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-93-76.

Characterization of Functionality In A Dynamic Environment

MS-CIS-93-76
GRASP LAB 358

Luca Bogoni
Ruzena Bajcsy



University of Pennsylvania
School of Engineering and Applied Science
Computer and Information Science Department
Philadelphia, PA 19104-6389

September 1993

Characterization of Functionality in a Dynamic Environment

Luca Bogoni
and
Ruzena Bajcsy

GRASP Laboratory
Department of Computer and Information Science
University of Pennsylvania, Philadephia, PA 19104

September 4, 1993

Characterization of Functionality in a Dynamic Environment

Luca Bogoni

and

Ruzena Bajcsy

Abstract

Identifying the functionality in objects means to be able to associate a purpose with them in a specific environment. The purpose depends on the intention of the agent and on the applicability of the object in a particular task. In our investigation of functionality we focus on functionalities which involve changes of physical relation and properties between objects in the environment. A formal model, based on Discrete Event Dynamic System Theory (DEDS), is introduced to define an interactive task for recovering and describing functionality. To observe and control the recovery process we introduce the notion of piecewise observability of a task by different sensors. This allows the description of a dynamic system in which neither all events nor the time of their occurrence may be predicted in advance. We have developed an experimental system consisting of actuators and both force and position sensors, for carrying out the interactive recovery of functionality. In particular, we demonstrate how this approach can be used by carrying out some experiments investigating the functionality of piercing. Furthermore, we discuss the importance of a multisensory approach for the observation and interpretation of functionality.

Contents

1 Introduction

Functionality of an object can be identified with its purpose and utility in a specific environment. Its purpose depends on the intention of an agent and the utility denotes its applicability in a particular task. Although functionality can be defined abstractly, to be identified in a specific object it needs to be explored in the context of an environment. Furthermore, such an environment must be dynamic since functionality manifests itself in the interaction between objects. We distinguish two types of interactions. The first one expresses some degree of constancy of physical relations and properties of a single object in time. The purpose of containment and support, for instance, characterize such type of constancy. The second type, addresses the changes of the properties of objects and physical relations over time as an object interacts with another (a knife piercing) or with the environment (a sponge absorbing).

In this paper we investigate the representation and recovery of functionality in the context of a dynamic environment. We believe that functionality recognition increases our comprehension of the environment we operate in. Furthermore, it provides us with means for attributing a purpose to an object in a context and hence improves our ability to recognize them.

Object recognition systems involving multisensory modalities are focusing more and more on being adaptive and capable of learning. Hence it is essential that a system supporting this flexibility be able to investigate its environment and determine not only the physical properties of an object but also its applicability in a task.

Functionality is not a characteristic unique to a single object, and a particular object may have more than one specific functionality. For example, a fork could be used for cutting as well as for piercing. Many artifacts do, in fact, possess more than one functionality and do so in different degrees of performance. Furthermore, the functional attribution of an object is context– and application–dependent. Thus, a knife can be defined as a tool suitable for cutting another object, but it is the applicability of a particular object for cutting which allows us to identify it as a knife.

Differently from all other object properties and attributes, such as color, shape, size, material or kinematic properties functionality addresses the interaction between the agent and the world, modulo the task and context. Determining the functionality of an object provides for means of categorizing things based on perceptual information. The seminal work of Rosch, [Rosch, 1973; Rosch, 1978], points out the distinction between a basic category, a sub–category, and a super–category. Rosch argued that while basic categories are primarily discriminated by their shape,

subcategories by details of shape and texture, and the supercategories are characterized by and large by the function of the object. For instance, a vessel is the supercatergory, glass is the basic category and goblet is an example of sub-category.

In our investigation of functionality we focus on functionalities which involve changes of physical relation between an object in the environment by interaction. In particular, we emphasize and develop an **interactive** and **performatory** approach to functionality recovery from sensor data in the context of robotic manipulatory tasks. This interaction does not only provide means to verify the hypothesized presence of functionality in objects but also a way to actively and purposively recognize the object. The representation of functionality allows us to extend the recovery process to a hierarchy of functionalities, allowing complex ones to be composed from simpler ones.

The formal model introduced here, based on Discrete Event Dynamic System Theory (DEDS), allows to define an interactive task for recovering and describing functionality. To observe and control the recovery process we introduce the notion of **piecewise observability** of a task by different sensors. As an example, we address the functionality of **piercing**. We demonstrate the experimental system being developed, with both force and position sensors, for carrying out the interactive recovery of functionality. Furthermore, we carry out some experiments to show how the sensors employed can be used to observe and interpret the interaction. At a later stage we will introduce a mechanism for addressing issues of classification based on functionality, issues of performance, and issues of learning.

1.1 Overview

This paper is organized as follows: in section 2 we outline some of the related work in the area. Section 3 introduces a characterization of functionality. We also investigate the importance and limitation of the information that is provided by shape in the definition of functionality, especially when seen in the context of manufactured objects. Section 4 outlines the formalism based on DEDS to describe a task model that can both represent and interactively recover functionality. In section 5 we examine how planning could be used to construct tasks based on DEDS primitives. As an example of the recovery process, we present, in section 6, a detailed analysis of the task of piercing using the formalism developed in the preceding section. Furthermore, we show how the instantiation of the task may proceed from defining a plan to the actual detailed description using DEDS. Section 7 shows how the task described is mapped to the different actuators and sensors. In section 8 we present three experiments using an implementation of our system. Finally, in section 9, we conclude by pointing out what we have accomplished so far and outline further developments

of the recovery process for functionality.

1.2 Acknowledgments

This work was supported by Navy Grant N00014-92-J-1647, AFOSR Grants 88-0244, AFOSR 88-0296; Army/DAAL 03-89-C-0031PRI; NSF Grants CISE/CDA 88-22719, IRI 89-06770, and ASC 91 0813.

Additionally I would like to thank Visa Koivunen (skii-meister) for the interesting discussions, Matthew Stein (matteo) for introducing me to the realm of the compliant wrist and providing some of the code for the classification of the contacts, Jana Košecká (bella bionda) who initially brought to my attention some of the issues concerning DEDS, Ulf Cahn von Seelen (ulfino) who helped me in the making of the video documenting the experiments, Marcos Salganicoff (elvis) for encouragement and previous discussions and Jim Gee (giacomo) who kept me alert and aware with many a cup of hot java.

2 Related Work

Through the past three decades functionality has received only limited attention, only recently researchers have begun to address its definition, representation and recovery. Yet there is little consensus as to what these should be. In this section we present a brief survey of the research on this topic with emphasis in the area Computer Vision.

Freeman and Newell [Freeman and Newell, 1971] were amongst the first who addressed functionality in objects as means of "devising artifacts for accomplishing goals". In ACRONYM [Brooks, 1980] one of the first attempts to bring functionality to object recognition is presented. In [Lowry, 1982], the author points out that functionality should be represented as a hierarchy of kinematic primitives, functional primitives, and causal networks. In [Winston *et al.*, 1984], the authors use natural language descriptions to provide identification of object physical and show how physical models can be learned using functional definitions.

Brady et al., [Brady *et al.*, 1985], present a system, "Mechanic Mate", intended to assist a handyman in a generic construction and assembly operation. The paper addresses the interplay of planning and reasoning, and the functional significance of higher order structures in the organization of the recovered information. Connell and Brady, [Connell and Brady, 1987], describe a system, based on a modified version of Winston's *Analogy* program, [Winston, 1980], which uses semantic nets to investigate the relation between form and function.

More recent investigations of functionality were carried out by Stark and Bowyer, [Stark and Bowyer, 1991]. They focused on the classification of CAD models of chairs. The work addresses the shape of the object and of its components as means of detecting functionality. In a subsequent development [Stark and Bowyer, 1993], they extend their system to begin acquiring data from a real environment. Brand [Brand, 1993] introduces a vision system for investigating interactively how machines work. This work brings together causal and functional knowledge for interpreting incorrect data from lower level primitives and for directing lower level vision routines to area of interest. Bogoni and Bajcsy, [Bogoni and Bajcsy, 1993], present a framework, based on discrete event dynamic systems, to investigate manipulatory functionalities. Rivlin et al., [Rivlin *et al.*, 1993], present a view of functionality recognition in objects as a goal oriented task in the context of robotics. The functionality of objects is investigated with respect to its shape, using Pentland's Thingworld [Pentland, 1986]. Tsikos and Bajcsy, [Tsikos, 1987], carried out some experiments addressing movability and removability of objects in a scene. Their work focused on part-assembly and disassembly. Campos and Bajcsy, [Campos, 1992], investigated material and kinematic properties

of object and in particular considered joints mobility. Salganicoff, [Salganicoff, 1992], investigates visually guided grasping with focus on learning procedures. These last three works are significant to our research for they focus on the interaction with the environment for the acquisition of objects properties.

In the field of Artificial Intelligence functionality has been considered in the context of Causal Reasoning, Planning, Qualitative Reasoning, etc. The literature addressing these investigations is considerable and will not be addressed in detail here.

In psychology, Jordan, [Jordan, 1991], addresses the importance of physical properties in understanding object functionality. Smith and Medin, [Smith and Medin, 1981], point out how functional features should actually be considered part of the core features appearing in concepts description. By using exploratory procedures (EP), [Klatzky *et al.*, 1987; Lederman and Klatzky, 1987], focus the investigation of object properties and in particular focus on haptic properties.

In robotics the work of [Cutkosky, 1989; Iberall *et al.*, 1988] addresses the importance of understanding functionality when manipulating and interacting with an object. Stein and Paul, [Stein and Paul, 1993], present a system, in the context of telerobotic, for local force control to allow the operation of cutting.

Finally, we find some additional sources on the importance of functionality in the studies carried out by anthropologist Goodall, [Goodall, 1986]. She investigated the functional usage of tools by Chimpanzees. The importance of functionality and the extraction of the functional properties of objects is also found in the study of the function of stone tools carried out by anthropologist Grace, [Grace, 1989].

Having considered the related work, we note that the investigation in the area is just beginning. Issues of definition and representation are still debated and the recovery of functionality is, for the most part, addressed either abstractly or inferred from shape. In most of these cases, the object properties are assumed and not recovered from the object. This is because the recovery of properties ushers research in areas in which uncertainty and noise from sensor measurements must be addressed. Specifically, recovering objects properties involves interactions with the object using different sensor modalities. At times the data obtained might be partial and an active approach must taken to acquire other data. Furthermore, even when some of the properties are known a priori, some testing must be carried out to verify or to establish other conditions. On the overall, this type of investigation requires a cross-disciplinary approach ranging from Control Theory to Vision and Robotics, from Psychology to Artificial Intelligence. The problem of properties recovery suddenly spans too many areas becoming rather complex.

The approach we present here does not sidestep these issues, rather focuses on defining the problem clearly and constraining it so that the investigation of functionality may become tractable. To that aim we have developed a system whose goal is that of interacting with an object recovering the properties which characterize a specific functionality and at the same time investigates the object applicability in a context. In the sections to follow we will present a formalism for expressing a functional task for dealing with the complexity of a multi-sensory interaction in a dynamic environment. Furthermore, we will carry out some experiments to investigate the functionality of piercing.

3 Defining and Representing Functionality

We have defined **functionality** as the association of a purpose with an object in a specific environment. Now, we examine what are the components which define an object and how these concur to the definition of functionality.

3.1 Object Definition

The properties that objects possess can be classified as:

- **Geometrical** properties identify quantifiable parameters defining shape, dimensions, volume, etc.
- **Material** properties are also quantifiable measures. Their attributes are defined in terms of density, coefficient of friction on the surface, thermal properties, etc.
- **Kinematic** properties identify the mobility of parts in an object, such as in a pair of scissors.
- **Dynamic** properties describe how the object responds to forces applied to it, such as the behavior of a compressed spring (stiffness).
- **Functional** properties identify the set of physical, (material and geometrical), kinematic and dynamic properties which characterize the functionality of an object.

Considering the properties listed, it becomes clear that different sensor modalities need to be employed to recover them. Global and local geometrical properties, such as volume, may be recovered from visual observations using stereo, shape from X for monocular vision, or laser-ranging sensors [Shirai, 1987]. Material properties may be recoverable visually by looking at reflectance and texture qualities of the surface. By using exploratory procedures (EP), [Klatzky *et al.*, 1987; Lederman and Klatzky, 1987], however, compliance and surface texture may be "felt" by using contact type sensors. Temperature probes may also be employed for actively determining constituent materials [Campos, 1992]. Kinematic [Campos, 1992] and dynamic properties [Sinha, 1992], however, require more complex EPs.

The physical properties of an object are intrinsic and its functional properties are part of the role it plays in an environment. Hence its functional representation, beside its intrinsic properties, must take into account:

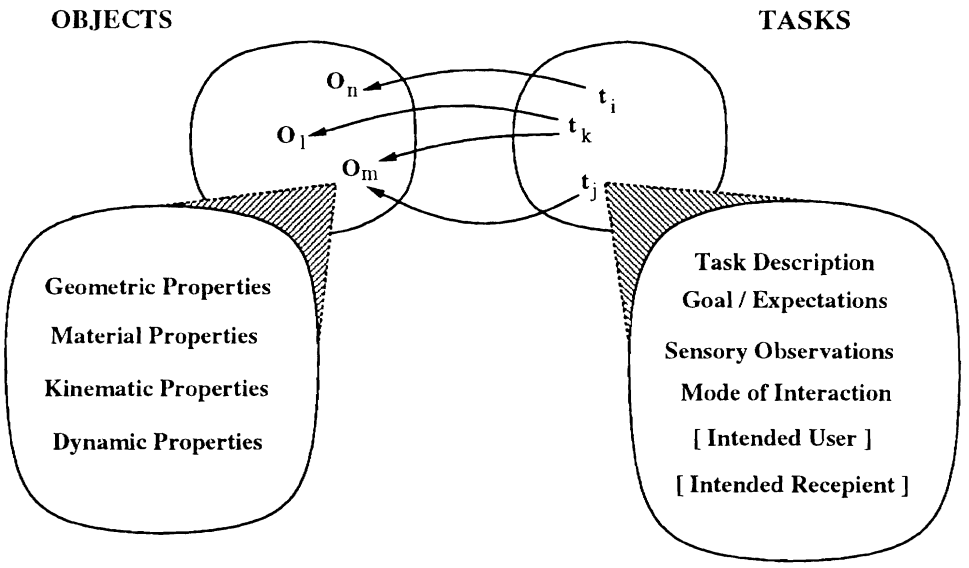


Figure 1: Object properties and functional description of an interactive task. Also shown are many–to–many mappings between tasks defining a functionality and the object employed to accomplish the particular functionality.

- the purpose that an object is to fulfill as expressed in a task description;
- the goal and expectation of the interaction involved in the functional task;
- the sensory modalities responsible for observing the interaction;
- the mode in which the interaction takes place;
- a possible intended agent and recipient of the task.

In Figure 1 we show the relation between an object and a task and emphasize that there is no unique mapping between an object and function that the object may fulfill. For instance, one may use a fork for cutting and piercing. Likewise, a knife and a fork may be used for piercing. The former instance is shown in Figure 1 as the mapping of t_k and t_j to object O_m, while the latter is portrayed as the mapping of t_k to objects O_m and O_l.

3.2 Types of Functionality

Focusing on whether the properties and relations of the object are **changing** or **constant** allows us to distinguish different types of interactions and observations. In particular, the functionality of support or that of containment focuses on whether some of the spatial relation between objects

remain unaltered over time. The functionality of cutting, on the other hand, concentrates on changes which take place as a result of the interaction.

Furthermore functionality can be characterized as intended, imposed, intrinsic, or inherited.

- **Intended** functionality identifies functional properties defined in an artifact at the time of its design.
- **Imposed** functionality defines the ability of using an object for a function for which it is not necessarily intended.
- **Intrinsic** functionality denotes functional properties which either characterize an intended functionality, in the case of an artifact, or define a functionality in virtue of physical properties of the object.
- **Inherited** functionality denotes a property which is either a specialization from an object or constitutes a new object in which functional properties are combined from different objects to fulfill one or more functionalities.

This distinction allows us to better understand the functional roles of artifacts and of natural objects but does not define an ordering of intended, imposed, intrinsic, and inherited functionalities. This is because while such ordering might be possible for some classes of objects, it is not definable in general. In particular, as we have seen above, context must always be be taken into account.

To clarify the distinction between intended and imposed functionality we note that a fork is constructed with the intended functionality of piercing and carrying, yet one may impose on it the functional property of cutting. Artifacts in general possess both intended and imposed functionalities. In artifacts intrinsic properties are defined when the object is designed. Natural objects, such as rocks, on the other hand, have imposed and intrinsic functionality.

The distinction between intrinsic and other types of functionalities becomes clear when seen in the context of natural objects. A natural object, such as a stick, has properties and in a perhaps larger view was "created" to fulfill a function. On the other hand, once we take two sticks we may use them for chopsticks, hence imposing a functionality on a natural object.

Intended and intrinsic functionalities are characterized by necessary functional properties while imposed functionalities require the object to possess properties which are sufficient for it to be applicable in the context. The "rigidity" of a table's surface is a necessary material property of the object to afford the function of support. On the other hand, "thinness" in a penny is just a

sufficient property for applying it as a screwdriver. In the case that the functionality imposed on an object coincides with the intended object's functionality, then the functional properties are both necessary and sufficient. This last case identifies the application of the proper tool for a specific task.

The characterization of functionality as inherited is useful for classifying an object in terms of its functionality with respect to others. This type of specification relates, for instance, the functionality of containment fulfilled by a tea cup to that of a glass. The process of specialization of the functionality of an object or that of combining functionalities from different objects constitute a designation of one or more functionalities into an artifact. However, the designing of some functional properties in an artifact is what we had identified as intended functionality. Hence while this distinction is useful for classification, we will not dwell on it further.

As we can see there are many components playing a role in the definition of functionality of an object. We now investigate the importance played by components other than shape. In particular, this consideration becomes quite easy to notice when considering the functionality of an artifact designed in a manufacturing environment.

3.3 Functionality, Manufacturing and the Role of Shape

We have pointed out that many properties contribute to defining the function of an artifact, see Section 3.1. When a product is developed there are many interactions which take place Computer Aided Design (CAD), Computer Aided Engineering (CAE), Computer Aided Manufacturing (CAM), and Service, to name a few. Each of the participants focuses on different issues concurring in the construction of the product (see Figure 2). The design for a product has several objectives that may be conflicting. Concurrent Engineering provides a systematic way for handling the interaction between the different goals of the various components participating the the design. This process is known as *Designing for X*, [Asfahl, 1992]. Hence shape is often not enough for determining the functionality. Yet being able to construct a geometric model is only a part of the process. Some part properties (Figure 3, 4, 5 and reffig:manuf-assembly), are not "functionally significant" but have the purpose of making manufacturing, assembly, maintaining, etc., easier. In certain cases asymmetry may be introduced for the purpose of making part orientation easily detectable. In other cases, symmetry may be introduced to make assembly simpler and not require special efforts to reorient a pin in order to be assembled. In other situations still, the object shape may be altered as a way to eliminate possible entanglement of the parts.

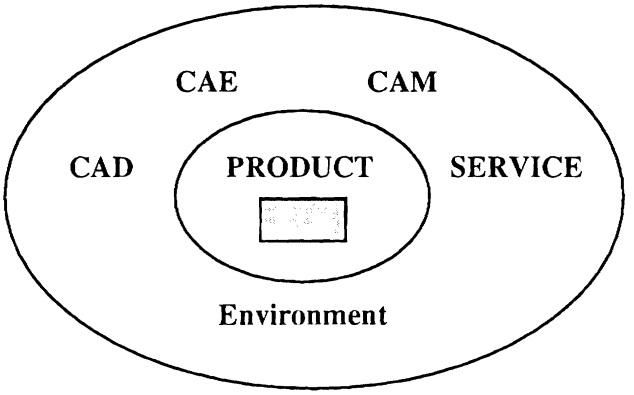


Figure 2: Components in concurrent engineering.

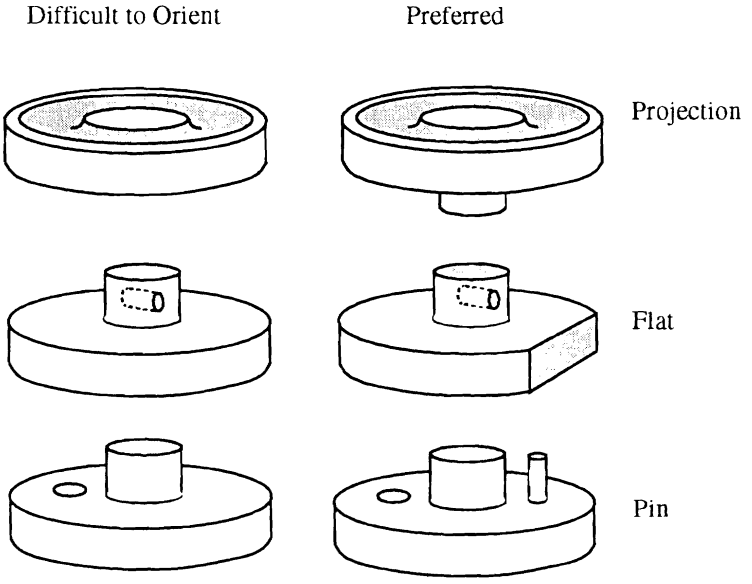


Figure 3: Examples in which parts asymmetry facilitates orientation for automatic assembly but in which no additional functionality is added to the components with the modification.

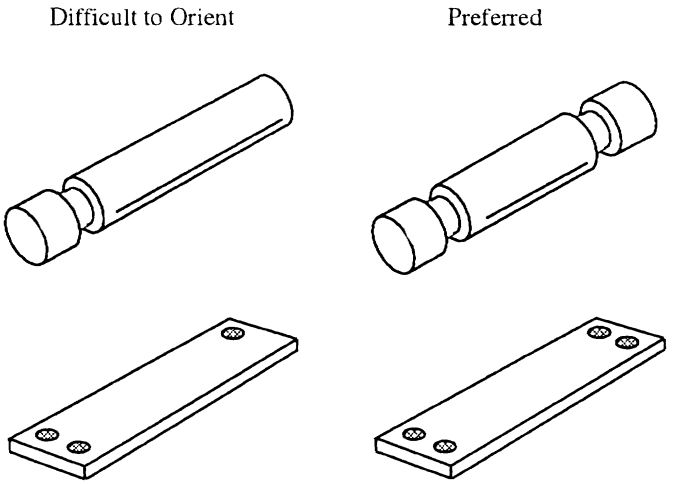


Figure 4: Examples in which part symmetry facilitates orientation.

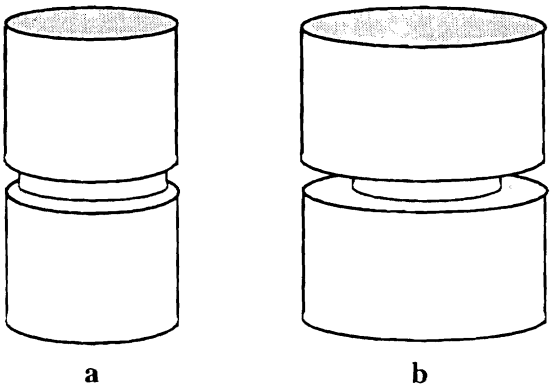


Figure 5: Pin. A change in area can be attributed to CAE requirements addressing perhaps thermal dispersion of the pin.

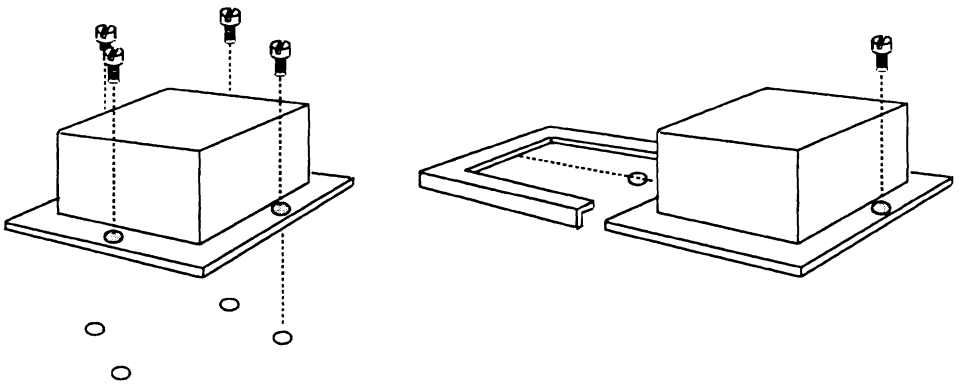


Figure 6: An example of how assembly requirements may influence the design. Left: as designed, the product consists of several components. Right: redesigned product.

We can further notice the relevance of other aspects when considering the work done in the area of reverse engineering. In [Koivunen and Bajcsy, 1993], the authors present techniques for reconstructing geometric models from range data. The reverse engineering process allows to recover the description of the object and to express the model data in a procedural language as well as in a product data exchange format (IGES). Yet other considerations will need to be brought in to understand the functionality of the object. Hence it is clear that while shape is an important descriptor for an object, there are many other aspects which need to be represented as well as recovered if the description is to reflect the functionality of the object.

4 Formalism for a Functional Task

The description of a task must provide for addressing its observability through different sensor modalities. It must also handle an environment in which not all interactions and exact time occurrences might be defined and predictable in advance. To describe an interactive process we adopt the formalism provided by Discrete Event Dynamic System theory, (DEDS) [Ramadge and Wonham, 1989]. This formalism allow us to model the behavior of a system in which uncertainty, external observability, and non–determinism can be addressed. As we shall see, however, this formalism does need to be extended to be able to incorporate probability measures and realtime controls. [Sobh, 1991; Košecká and Bajcsy, 1993] present examples of the application of DEDS as a means for expressing visually guided behavior of systems.

According to DEDS theory the behavior of a dynamic system can be modeled as a non-deterministic finite automaton (NDFA). In such a NDFA arcs identify **events** and states identify fragments of operational behaviors or logical states of the system. Thus a state can be defined in terms of state variables. Transitions to other states may occur when these variables reach specified values. For example in the motion of a robotic arm the set of state variables might include those needed to specify the position of the end-effector. The transitions to a new state could be represented by the state variables having obtained a particular value identifying, for instance, contact. In this example we would have two states, the first one identifying the motion of the end-effector and the second one identifying the contact state.

Events which allow the transition from a state to another may be disabled or enabled as a means of guaranteeing controllability of the system. In [Sobh, 1991] transitions between states are also assigned probability functions. These functions determine the probability that a given event has been asserted.

Any task can be described as a simple action or as a sequence of actions or subtasks. Then we can identify some of these actions to represent states of the system. While events identify changes in the variables describing the system, we distinguish the following sets of events:

- A *change in the state variables*, Δ, in which the value of one or more variables describing the event has reached a specified value.
- The *assertion of logical expressions*, Λ, possibly denoting groups of events.
- The *reaching of a guarded value*, $\mathcal{G}$, for one or more state variables to which a particular meaning has been attributed, such as safety conditions. (where $\mathcal{G}$ is actually a subset of Δ)

This partitioning is done for convenience of expression. Logical expressions, in particular, are here intended as means of clustering events and attributing a meaningful interpretation.

4.1 Automata Model for the DEDS

The set of labels of the events is given by $\Sigma = \Delta \cup \Lambda \cup \mathcal{G}$. A **string** $s = \sigma_1, \sigma_2, \ldots, \sigma_k$ from Σ^+ describes a sequence of events, also known as event trajectories. The **admissible** subset of strings from Σ^+ defines physically possible sequences of events which constitute a task. A **recognizer**, M, can be described as a NDFA consisting of a set of states, Q, an initial state, q_0, a transition function $\delta : \Sigma \times Q \rightarrow Q$, and a set of final states, Q_m (marked states). The set $\Sigma(q_i)$ designates the collection of events which are associated with state q_i. The set $\Sigma(q_i)$ is defined as $\Sigma(q_i) = \Delta(q_i) \cup \Lambda(q_i) \cup \mathcal{G}(q_i)$. A recognizer M_{t_i} will accepts the strings from Σ^+ describing a sequence of events denoting a task, t_i. In particular, M_{t_i} characterizes the task's procedural description.

4.2 Controllability

The set of events which we have identified above may include some which are *controllable* (that can be disabled) and some which are *uncontrollable*. Thus, we can partition Σ into $\Sigma_c \cup \Sigma_u$. Enabling and disabling certain events can be described by the control pattern for the specific state. Let $\Gamma = \{0,1\}^{\Sigma_c}$ define the set of binary patterns assignable to the elements from Σ_c. Then the function $\gamma : \Sigma_c \rightarrow \{0,1\}$ defines whether they are enabled or disabled.

The transition function δ above can now be defined as $\delta_c : \Gamma \times \Sigma \times Q \rightarrow Q$

$$\delta_c(\gamma, \sigma, q) = \begin{cases} \delta(\sigma, q) & \text{if } \delta(\sigma, q) \text{ defined and } \gamma(\sigma) = 1 \\ \textit{undefined} & \text{otherwise} \end{cases}$$

Then the generator $\mathcal{G}_c = (Q, \Gamma \times \Sigma, \delta_c, q_0, Q_m)$ is called the **Controlled Discrete Event System**. Such a controller is called a *Supervisor*. Further details can be found in [Ramadge and Wonham, 1989].

4.3 Observability of the Interaction

A task t_i is **observable** if the sequences of events which define it are observable. Figure 7 portrays an instance in which some of the events from a string from Σ^*, $\sigma = a_1\, a_2\, a_3$, are not observable

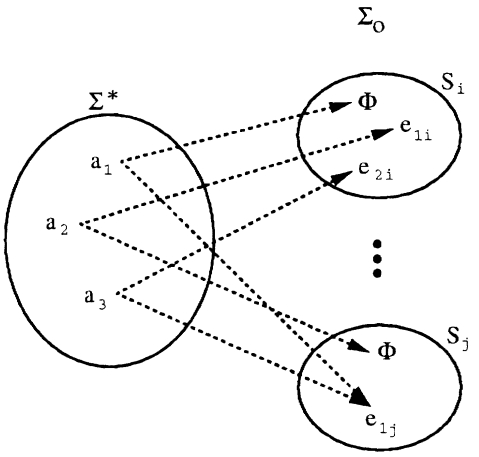


Figure 7: Observable events, Σ_o, mapped to Sensors.

and mapped to the empty set, Φ. We can define a projection function mapping events from Σ^* to the individual sensors S_j's from $\mathcal{S}$ (set of sensors).

Let $\mathcal{S}$ be the set of available sensors, S_j. Then an event σ_i from Σ can be mapped to some event $e_{ji} \in S_j$ if the given event may be observable by the sensor in question and to Φ otherwise. This can be stated as

$$P(\sigma_i, S_j) = \begin{cases} e_{ji} & \textit{if event } \sigma_i \textit{ is observable by sensor } S_j \\ \Phi & \textit{otherwise} \end{cases}$$

Observability is contingent on the ability of monitoring the different events. The observability of the individual events in the task must be guaranteed by the different sensor modalities if the overall task is to be observable.

We now elaborate on the implications of these definitions. In particular we address the following issues: *full*, *partial*, and *piecewise* observability.

4.3.1 Full Observability

Full observability can be stated as follows. Let W, defined as above, describe all possible strings of events accepted by a recognizer M_{t_i}, which describes the procedural behavior of task t_i. Then task t_i is **fully observable** if all of the events in strings from W are observable.

We can express this condition by considering the following. Let $\mathcal{V}$ define an indicator function which will assign a 1 if the projection of some event σ_i in string w_k from W maps to some sensor

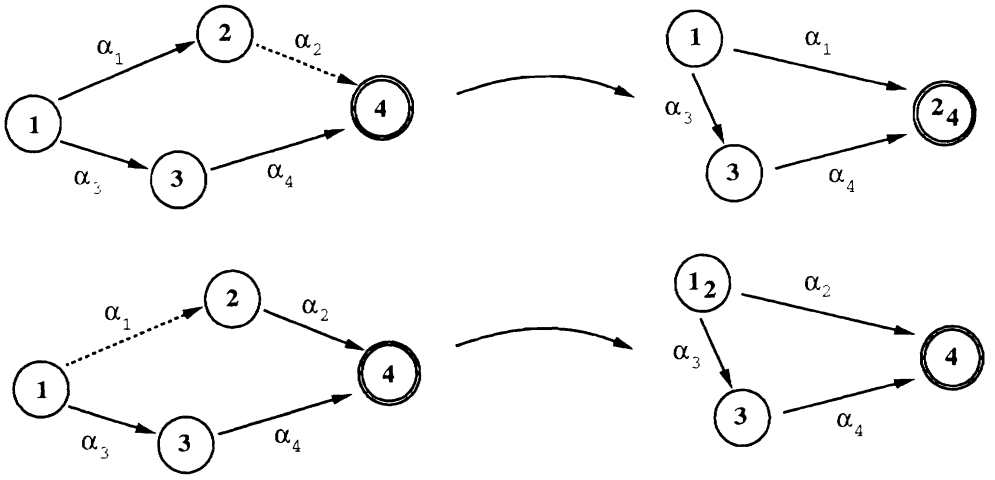


Figure 8: This example illustrates two instances of the effects of partial observability of two actions. The dotted lines represent non observable events.

from $\mathcal{S}$:

$$\mathcal{V}(\sigma_i) = \begin{cases} 1 & \textit{iff } P(\sigma_i, S_j) \neq \Phi \quad \forall\, S_j \in \mathcal{S} \\ 0 & \textit{otherwise} \end{cases}$$

Then we can express full observability of task t_i by the following function describing the boolean product.

$$\mathcal{O}_f(W) = \prod_{w_i \in W} \left(\prod_{\sigma_j \in w_i} \mathcal{V}(\sigma_j) \right)$$

$\mathcal{O}_f(W) = 1$ then indicates that all the paths describing the task t_i are observable. The subscript f in $\mathcal{O}$ stands for the full observability.

4.3.2 Partial Observability

The projection function, $P(\sigma_i, S_j)$, allowed the description of observability of an event by some sensor S_j. What happens, however, if some of the events which characterize the task are not observable, i.e. $\mathcal{V}_f(\sigma_i) = 0$? In this case some of the states become indistinguishable and we have situations as in Figure 8. We call this effect of projecting one event to the null event and collapsing two states into one **aliasing**. (In the mapping transformation illustrated later we will mark aliased states by shading them (see Figure 10).

It is also possible that, as exhibited in figure 9, partial observability may give rise to ambiguity. In fact, in this case while the original task description exhibits a clear procedural flow from the the initial state to the final state, the partial observation transformation introduces ambiguity.

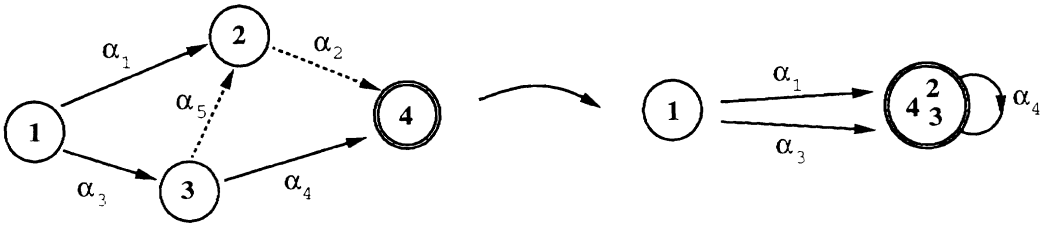


Figure 9: This example illustrates an instance of the adverse effects of partial observability. The dotted lines represent non observable events.

Considering the possible paths in the left automaton which could be taken during the evolution of the task, in figure 9, it is not clear how important it is that event α_3 should take place. In the automaton on the right it is not clear that the action may be at all observable.

4.3.3 Piecewise Observability

If a task is partially observable by different sensors then it is **piecewise** observable as a whole. Redundancy in observing an event using more than one sensor can be employed to corroborate the evaluation of the observations. However, the application of more than one sensor modality goes beyond the issue of corroboration it provides for means of guaranteeing that task is observable. In general, non-trivial tasks will require observations from different sensors.

Sensors are not always faithful and reliable informers and uncertainty has to be introduced in the system. In particular, it is important to be able to identify the uncertainty originating from sensor noise, from the environment, and from the detection of an event denoting a transition to a different state in the system. Thus, as described in [Sobh, 1991], we introduce a probability measure associated with the occurrence of the events.

In Figure 10 we illustrate a DEDS description of a task involving an action leading to contact. In the supervisor description of the task we have distinguished controllable and uncontrollable events. In the observers we have associated with certain events some measure of probability $P(.)$. In the case of the force sensor, some of the states are indistinguishable since the events between them are unobservable. It is, in fact, only upon reaching contact that the force sensor will be able to assert that an event has occurred. As we can see in Figure 10, the overall task is piecewise observable by a vision sensor and a force/tactile sensor. It is, in fact, the combination of the two sensor modalities which make the whole task observable.

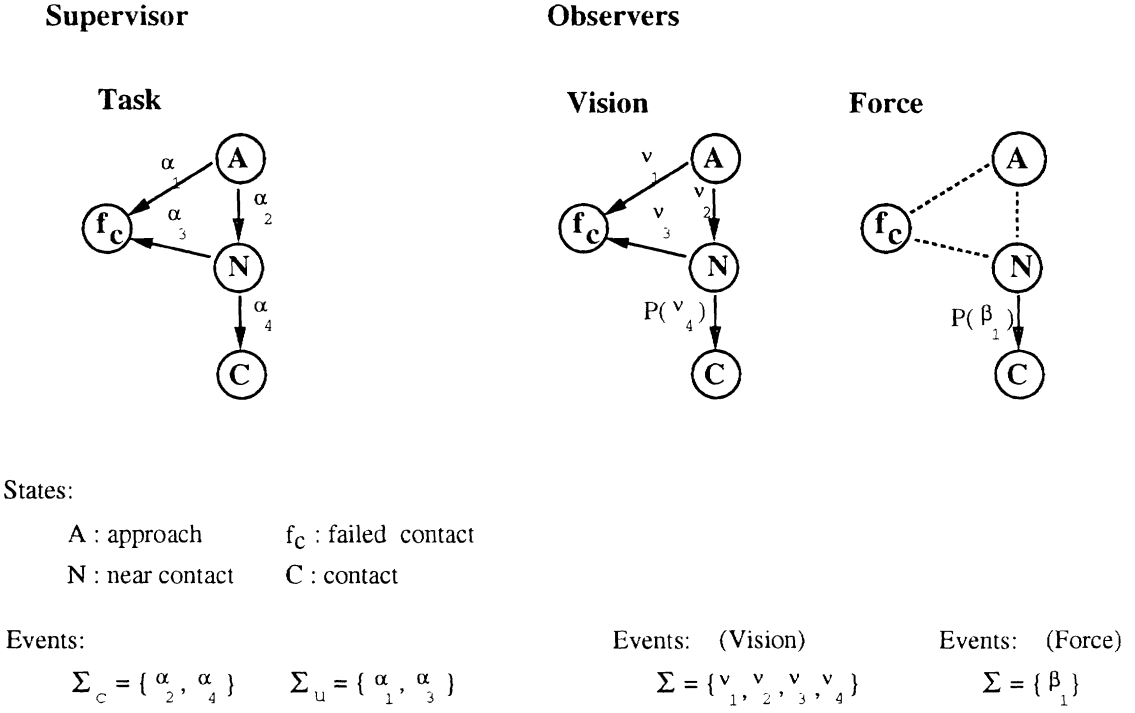


Figure 10: DEDS description of a task involving an action leading to contact. Left: the supervisor description of the task. Right: the task as observed by the different sensors. The nodes in the shaded area identify those that can not be distinguished by the sensor.

4.3.4 Guaranteeing Observability

It is clear from the above examples(Figures 8 9) that not all partial observable mapping are desirable. Thus our notion of observability must include some strong conditions on the observability of certain events. This is equivalently expressed by considering the distinguishability of the states that these events transfer to. This can be accomplished by requiring that the critical paths include those states.

Therefore, we define a set of **distinguished states** to describe those states which, if they appear along a critical path, must not be aliased with some other distinguishable state. The distinguished states described here define a concept similar to what in DEDS terminology is known as **marked states**.

This definition does not require that all the distinguishable states be visited, only that they be unambiguously marked. This distinguishable states do not include the initial and final state, nor the set of the dead states.

Procedurally the observability of events can be verified by checking that after the mapping the events in the supervisor are covered, allowing for those which may be only logical transitions and do not necessarily reflect physical changes of state variables. One such transition could be defined as a transition between one subtask and another. We will discuss some examples in Section 6.2.

4.4 Sensor Selection for Observability

The selection of the sensors which should be involved clearly depends on the task and the environment in which they are going to be applied. Furthermore, they determine the type of observations which can be carried out and the type of models which can be recovered.

We distinguish two levels at which this selection is to take place, a *task* and an *implementation* level.

The task level description identifies the type of interaction and hence the type of sensors which are required to render the overall task observable. When a group of events is observable by more than one sensor it might be preferable to focus more on one of the sensors rather than others. Thus, any event mapped to more than one sensor should have a measure of importance associated with it. [Sakaguchi and Nakano, 1992] present a framework, based on information theory, for selectively choosing the sensors yielding the most informative type of observation, *intentional observation.*

The implementation level is responsible for associating the adequate routines and models with the sensors available, selecting those which best suit the constraints of both task and environment. We find examples of this association between task and sensors in the area of computer vision.

[Ikeuchi and Hebert, 1990] present a task-oriented approach for selecting visual routines which fit the task requirements. The authors given two examples: a bin picking and a rock sampling system in which representation models, feature detection, and sensor selections were task driven. The systematic analysis of the tasks exemplified in the article provides the motivation for the choice of sensors, models, and feature segmentations routines. However, a mechanism to provide for the selection is not specified. One would like, eventually, to be able to device a system for automatically performing this implementation level, but this is a wide and open area of research.

Having outlined the issue of sensor selection we return to investigate the composition of functionalities.

4.5 Primitive and Complex Functional Tasks

The interaction domain $\mathcal{A}$ can be constructed from an initial set of actions denoting functional tasks, $Ker(\mathcal{A})$, which we define to be the *primitive* set. Complex tasks can be composed from a set of primitive tasks which have been fully explored. Since the individual components are piecewise observable the resulting new action will be observable. We can define an algebra of tasks with the operations:

- *Composition* as the sequencing of a list of actions, written as $C(a_1, \ldots, a_k)$.
- *Repetition* as the composition of a given action $a_i \in \mathcal{A}$ with itself.

Sawing, for instance, could be easily seen as an operation in which composition and repetition occur. In Figure 11, in the following section, we show a case in which two simpler functionalities have been composed into the task for piercing. Further discussion on the composition of primitives are given in more detail in [Bogoni and Bajcsy, 1993]. We will now focus on a particular task identifying a functionality.

5 Constructing a Task

In section 4.5 we outlined how complex tasks can be composed from simpler ones, yet the generation of a task must be driven by some top–down structure rather than simply generated bottom–up. A planner may provide such top–down structure to the selection of simple actions, described in the DEDS formalism, which are woven in a structure, possibly a linear path, defining a task.

The AI literature in planning is quite substantial and the criteria underlying their operation are quite varied. [Tate *et al.*, 1990] give a classification of different planners by considering their behavior within the following criteria.

- How is the *planning search* carried out?
- What kind of *abstraction levels and hierarchy* is considered?
- How are the *goals ordering* and *interaction detection and Corrections* carried out?
- Does planning allow for *conditional and iterators*?
- How is the *domain represented*?
- How *time and resources* are handled?
- What is the style of *planning and execution*;
- Are *learning and memory* of the interactions considered?

By considering the different criteria, we observe that a planner, in order to handle a dynamic environment should have:

- a hierarchical abstraction type structure so that plans are only partially developed and can account for changes in the environment – ABSTRIPS ([Sacerdoti, 1973]), NOAH ([Sacerdoti, 1977]), NONLIN ([Tate, 1977])
- be able to allow several events to occur at one time as it is in fact in real environment – SNLP ([McAllewster and Rosenblitt, 1991]), NOAH ([Sacerdoti, 1977]).
- handle alternatives in the form of conditionals and be able to incorporate domain specific information – IPEM ([Ambros-Ingerson and Steel, 1990]),TWEAK ([Chapman and Agre, 1987]), NOAH [Sacerdoti, 1977].

- be able to deal with environmental changes (initially addressed in STRIPS [Fikes and Nilsson, 1971]). This is often considered part of reactive planning techniques which are combined with sensing and actions, [Chapman and Agre, 1987].
- addresses uncertainty. Yet most of the planners dealing with this issue assume a probability model of events, [Kanazawa and Dean, 1989]. However, often such type of model is not available.
- be able to base the planning on previous experiences (either learned or provided by the operator) – CHEF ([Hammond, 1986])

At the basis of the planning schemes presented above lies the assumption that the primitive actions constitute the lowest level of interaction with the environment. When a plan is generated, real-world issues are only summarily addressed. Only rarely are there instantiations of plans into real domains where all the issues are considered.

[Kaebling, 1990] describes an architecture for a reactive system in which a planner is working incrementally in conjunction with information obtained from sensor for navigating. While the underlying philosophy of incremental and reactive planning is investigated, issues of communication with sensors, noise in measurements, and modeling are not really addressed. Because in the experiment outlined, the authors are interested in obstacle avoidance. Interactions require a higher degree of context detail and monitoring. The type of constraints which are required to be incorporated are presented in [Rosenschein and Kaebling, 1988]. The designed system, GAPPS, attempts to bridge the implementation-level with the high-level description and the required context needs of a robotic control. It does, in fact, act as a compiler providing a translation of constraint expressions into an executable circuit for the control of a robotic system.

[Ambros-Ingerson and Steel, 1990] present a framework, IPEM, for integrating planning, execution and monitoring of the operations. It introduces execution and monitoring into [Chapman and Agre, 1987] TWEAK's partial plan representation. The control is carried out using a production system architecture with conflicts resolved by a scheduler. This framework seems to address most of the concerns stated, yet it is unclear how it would handle uncertainty in the environment.

An example of the type of interactions we consider are presented in the HANDEY system [Lozano-Perez *et al.*, 1987; Jones and Lozano-Perez, 1990]. The authors describe interactions carried out in a robot workcell for pick-and-place type problems. The planning presented, however, deals with only some of the issues mentioned above.

By applying the formalism for tasks descriptions defined here it would seem possible to reconcile

and take advantage of planning strategies studied in the area of AI. Namely, one can think of the primitive actions to be defined in terms of small discrete event dynamic system interactions which are combined by the planner. A task defined in terms of a DEDS provides the ability of handle and describe situations which can not be addressed at the high level carried out by planners. Furthermore, it allows the possibility of maintaining a degree of abstraction between the planned action and the instantiated interaction.

The interaction between planners and DEDS has another benefit. When DEDS are combined to compose a new behavior, one has to be concerned about the combination of the events. In [Ramadge and Wonham, 1989] two DEDS are combined using a *shuffle* product. The authors point out that this approach generates an exponential number of states. They propose to limit the proliferation of states by considering a hierarchical approach in which events relative to a specific DEDS remain constrained to it. Thus only events which allow transitions in and out of the event space of that specific DEDS would be combined with others. On the other hand, if the combination of the events were integrated with a top–down knowledge of the events interaction the set of events resulting from the interaction, would be greatly curbed. In particular, by employing a planner, the number of the states to be considered can be reduced by analyzing the possibility of certain events to take place and remove states which are unfeasible and impossible.

6 Defining the Task of Piercing

Piercing involves grasping an object (tool) at one end with the intention of bringing it to contact with a target object. Once the tool has been brought to contact, force must be applied to enable the tool to break the surface and penetrate the target object. A successful piercing operation can be defined in terms of some parameter of depth – *through* defines a penetration of thickness equal to that of the object. However, many different variations are possible depending on the type of material, tool, degree of penetration, and the manner in which the tool is applied (position, force, rate of change of force, etc.).

The type of contacts possible (point, line, surface) vary with the geometry of the tool and that of the target. The selection of the type of contact desired has to be determined by the constraints of the objects but also by the subtask which is to follow contact. In the case of polishing, for instance, a surface contact may be selected over a point contact.

In the DEDS description of the piercing task examined in this paper, we assume that the type of DEDS routine identifying the subtask of bringing the tool to point contact was selected by a planner. Hence, we will focus on the issues dealing with events observability, confidence measures of events, and sensor integration in the context of a complete task identifying piercing.

6.1 DEDS Task Description

As we have seen in section 4, the description of a task is accomplished by defining events and states as monitored and controlled by a supervisor in terms of observers which can report on the changes of the system. In this section we will describe both a supervisor and several observers for a dynamic environment in which the task of piercing is to be carried out.

The role of a supervisor is to control the behavior of a system. In order to do so, the supervisor must have a complete view of the task and in particular the ability of determining, based on the previous events, the state of the system. We will give examples when discussing the mapping of the task to the force sensor.

Figure 11 shows a DEDS description identifying a piercing task as seen by the supervisor. We notice that some of the events, (Table 1), are clearly observable only by some sensor modalities. Event α_1, for instance, is observable only by vision. α_3 identifies the event of the object coming into contact with the surface. Furthermore, it is only upon contact. α_3 being asserted, that the state of the position sensor becomes defined.

Piercing Task Events Description

Controllable	*Uncontrollable*
α_1: when tool begins to move	α_2: when tool misses target object
α_3: contact is made with object	
α_5: failing to break the surface	
α_6: penetration actually taking place	$\alpha_4, \alpha_7, \alpha_{10}$: object or tool failure
α_8: failing to penetrate to desired depth	
α_9: goal accomplished	
α_{11}, α_{12}: logical transition to extraction state	
α_{13}: extraction failure	
α_{15}: extracted to contact level	$\alpha_{14}, \alpha_{17}, \alpha_{20}$: object or tool failure
α_{16}: logical transition to contact state	
α_{18}: moved from contact to free space	
α_{19}: logical transition to depart state	
α_{21}: moved to start state	

Table 1: Controllable and uncontrollable events in Figure 11

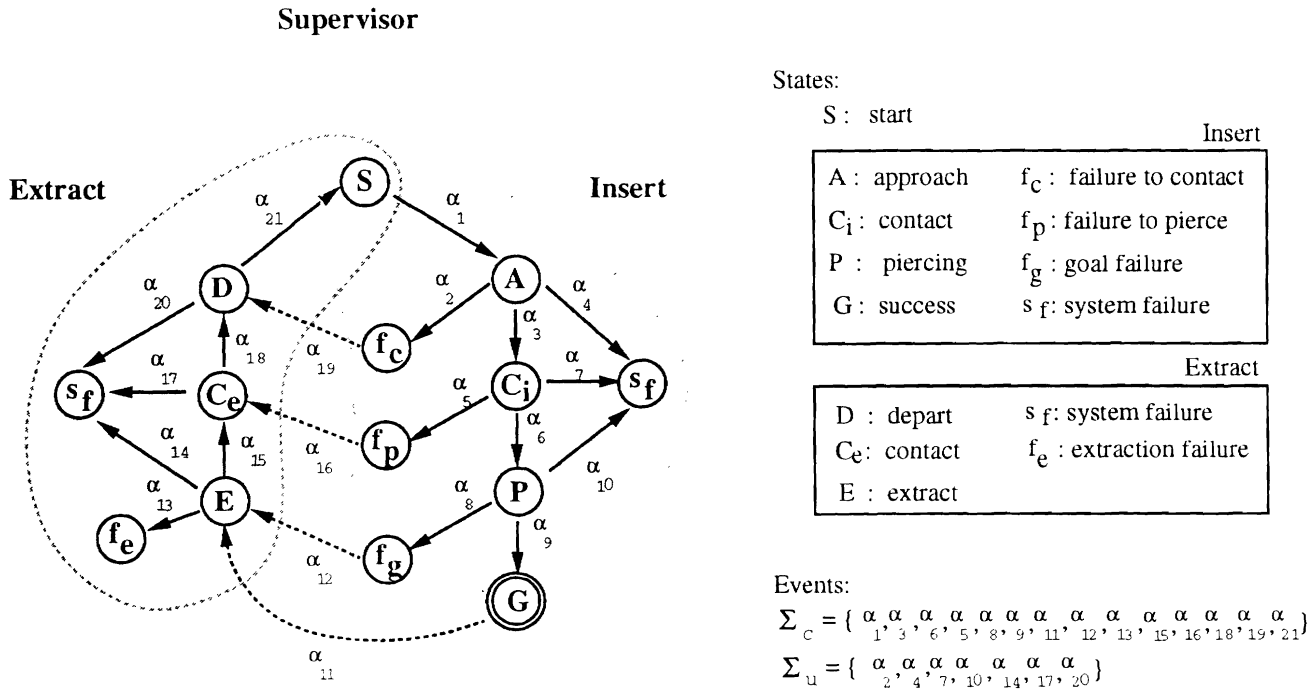


Figure 11: Definition of Piercing Task. Distinguished are the the two operations, insertion and extraction, which compose the piercing task. The dotted arches define controlled event which are asserted by the supervisor.

The definition of success in the context of piercing can be expressed as a function of position, force, or both. The behavior of the interaction will greatly vary depending on the type of material encountered.

In the case, for instance, that the object being pierced is thin and shows elastic properties or has a different lower internal hardness, success can be determined only of the basis of variation of force. In particular a raise and fall in the magnitude of the force will occur once the surface is pierced. [Stein and Paul, 1993] adopt this definition of successful piercing in their investigation of local control of simple behaviors for telerobotics. In particular they investigate the task of cutting the tape joining insulation panels on satellites.

If the target object were to be elastic, then the position of the end effector would have to be observed not only by a force sensor and an position sensor but also by an external vision sensor. It is only by obtaining observation through additional sensor modalities that we are able to identify the behavior of the material. In particular, if the tool were to be partially elastic, both position and force sensor may be asserting events indicating that the tool is penetrating the target object. Vision can at this point contradict such an assertion by revealing that a deformation of the tool is

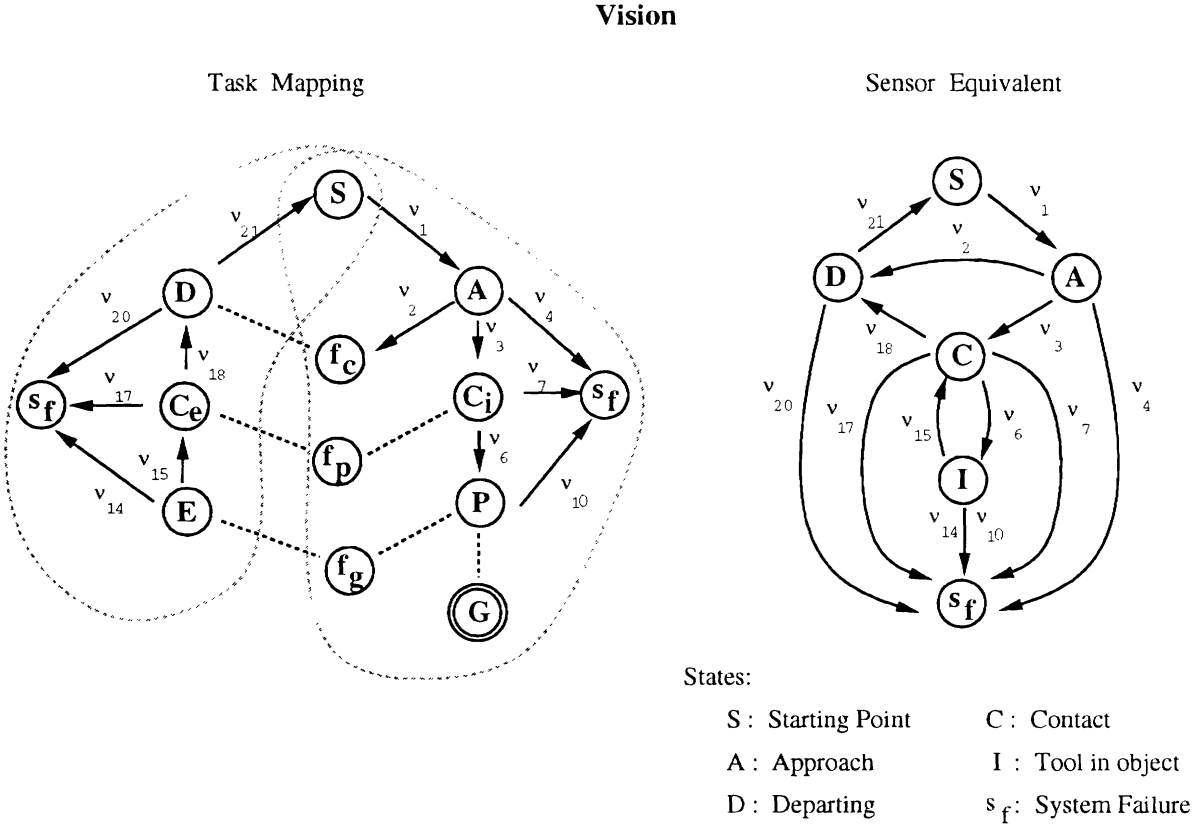


Figure 12: Mapping of piercing task to vision sensor(s).

taking place.

It is important to notice that even if the task was not accomplished, as the operation is carried out, the overall interaction was successful because information about the properties of the object have been gained. In particular the analysis of the cause of failure may lead to reconsider the interaction and possibly replan it. Unless it is prespecified, it is only by failing that certain properties about objects are discovered. Furthermore, by comparing success and failures we aim at acquiring knowledge about those physical properties of objects which render the interaction successful. Once these are discovered, they can be actively searched in a new object to test.

6.1.1 Mapping of the Task to Sensors

We now discuss the individual mappings to the different sensors.

As we can see in Figures 12–14, some of the nodes from the task description do not map

Vision Events Description

Controllable	*Uncontrollable*
ν_1: when tool begins to move	ν_2: when tool misses target object
ν_3: contact is made with object	ν_4, ν_7, ν_{10}: generic system failure
ν_6: tool is penetrating	
ν_{18}: loss of contact	ν_{17}, ν_{20}: system failure
ν_{21}: transition to start position (end of motion)	

Table 2: Controllable and Uncontrollable events in Figure 12 as mapped to the vision sensor.

to the sensors and others still are aliased since events which differentiate between the nodes are unobservable by that particular sensor. However, we notice that the overall task is piecewise observable. In the mapping we have preserved the numbering to exhibit the relationships between the events in the task and in the sensor. For each of the mappings presented we have provided a sensor equivalent mapping with state description.

Mapping to Vision Sensor

The mapping of the task to the vision sensor is given in Figure 12. Table 2 identifies the events. We have expressed the operations carried out by vision as a single sensor; nevertheless, the operation could be carried out with more that one vision modality. Tracking an object to contact and modeling the relationship between tool and target may involve more than one sensor and rather sophisticated algorithms.

Observing the mapping we notice that states $\{C_i,\ f_p,\ C_e\}$, $\{P,\ f_g,\ G,\ E\}$ and $\{D,\ f_c\}$ are indistinguishable by the vision sensor.

Mapping to Force Sensor

Table 3 outlines the events which are observable by the force sensor. We note that in the case depicted in Figure 13, once contact is defined there is no transition to the piercing state or to the goal state. This is because an increment in force does not necessarily identify a transition to a piercing state. The object could be too hard and hence unpierceable by the tool being investigated. The only transition which can be controlled is β_5 which identifies a failure due to reaching of a

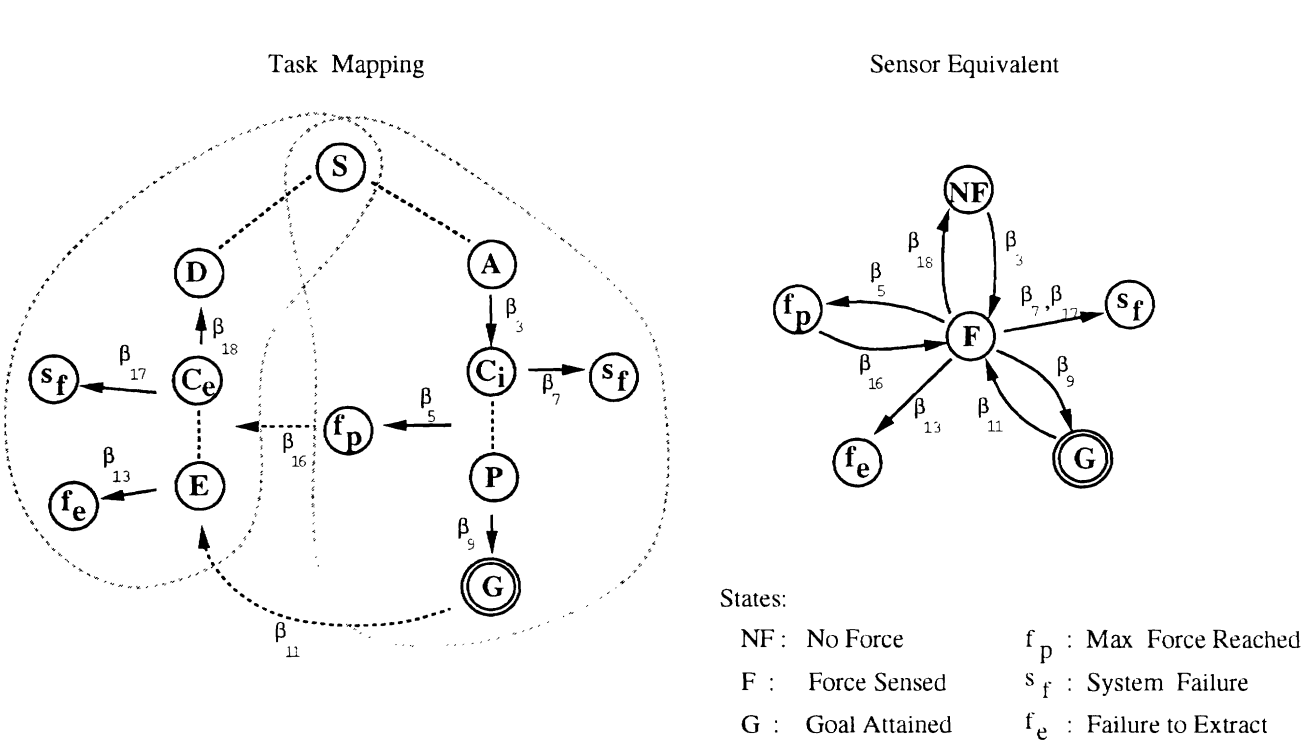


Figure 13: *Mapping of piercing task to Force sensor.*

maximum threshold for the force which the sensor can sustain. This type of event was identified in section 4 as a *guarded value* in $\mathcal{G}$.

On the other hand, as mentioned above, if the material were to be nonuniform in hardness then the force sensor would be able to assert that a transition to the goal state has occurred. Such an event β_9 would be triggered by a sudden drop in force. This type of transition in force characterizes also the behavior of a target object or tool shattering under pressure, denoted by event β_7. Hence we note the importance of an additional sensor for disambiguating the state after the event was asserted.

Examining the mapping in Figure 13, we notice that in the sensor equivalent description there are many events which originate from state F. While there are several which might occur simultaneously, there are some which will occur only when the tool is penetrating and others when the tool is being extracted. The function of enabling or disabling certain events will be carried out by the supervisor. There are six events which originate from state F. While some events, β_7 and β_{17}, are uncontrollable and hence always enabled, others are controlled by the supervisor. In particular,

Force Events Description

Controllable	*Uncontrollable*
β_3: contact is made with object	β_7: system failure: object shatters
β_5: maximum force reached, could be a failure	β_9: sudden drop in force. object pierced
β_{11}: transition to change force	
β_{13}: failure to extract	β_{17}: system failure
β_{18}: no force due to loss of contact	

Table 3: Controllable and uncontrollable events in Figure 13 as mapped to the force sensor.

when the system is in state F and piercing, events β_5 and β_9 are enabled and events β_{13} and β_{18} are disabled. On the other hand, when the tool is being extracted, β_{13} and β_{18} are enabled and β_5 and β_9 are disabled.

This analysis emphasizes both the role and the need for the supervisor.

Mapping to Position Sensor

Finally, table 4 lists the events which are observable by the position sensor. We notice that the failure to penetrate to a given depth can only be observed as a consequence of β_5 rather than directly from the position sensor. This is because the position sensor can not tell whether the operation is only temporarily stalled because we are increasing the force or because something else is happening. Hence α_8 is asserted only in the supervisor. This identifies one of the events, defined in section 4, which falls under the classification of logical assertions in set Λ.

We notice that no uncontrollable event is observable by the position sensor. Transition ρ_{11} is induced as a transition by the supervisor to initiate the removal of the tool from the target object. It identifies a transition to previous positions in the object.

6.2 Covering of the Events

Up to this point we have discussed the importance of mapping the events of the supervisor to the different sensors, yet we have not verified that the events, as mapped, cover the supervisor event space.

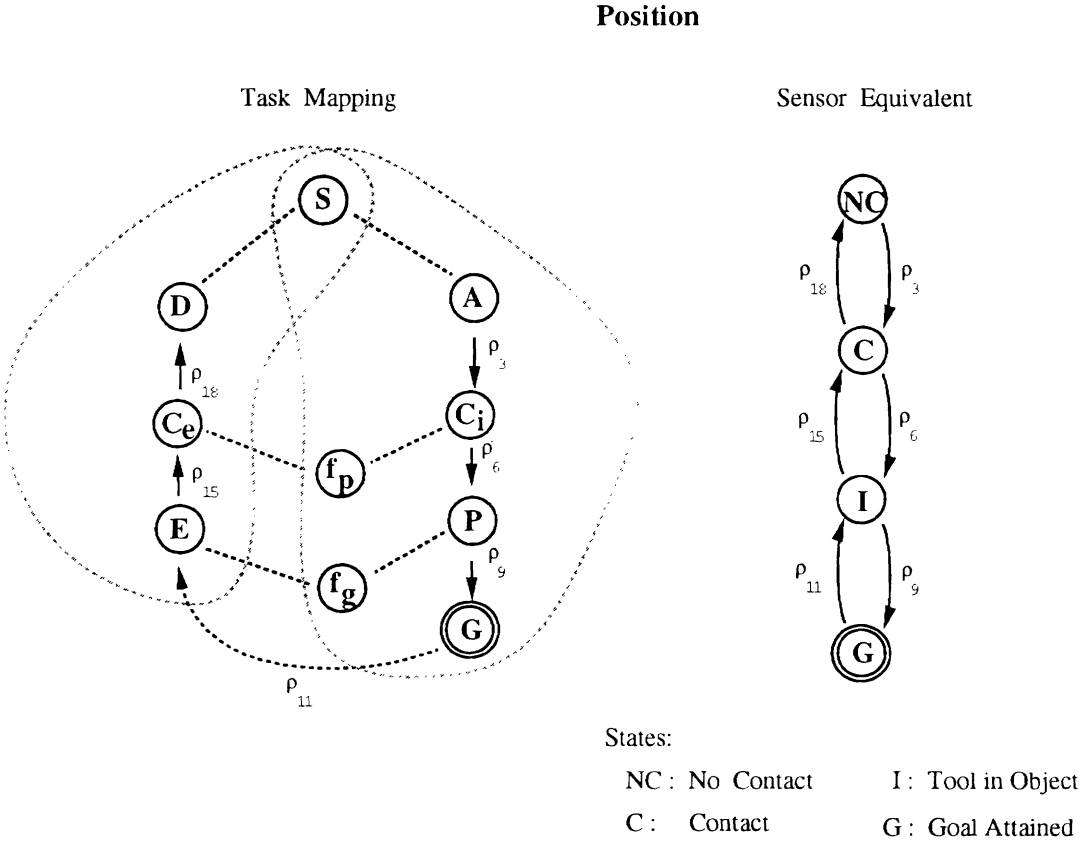


Figure 14: Mapping of piercing task to position sensor.

In table 5 we list the events in the supervisor and the corresponding sensor-mapped events. In the last column we mark the covering. We notice that all events, except for α_{12} and α_{19}, are mapped and hence observable by some sensors. The events which are not accounted for in the table represent *logical* events and are asserted by the supervisor. They provide, as mentioned in section 4, for transitions between components in the task and can easily be compared to what in automata theory is known as ϵ–transitions. In this case they identify the transition between insertion and extraction.

In table 6 we have outlined the state mapping. This table allows us to notice the amount of aliasing between the different states indicated by the shaded areas in Figures 12–14.

Position Events Description

Controllable	*Uncontrollable*
ρ_2: contact is made: position is defined	ϵ: none noticeable by position sensor
ρ_6: penetration actually takes place	
ρ_9: desired position reached	
ρ_{11}: logical transition to extract object	
ρ_{15}: transition to contact position	
ρ_{18}: tool no longer in contact (position undefined)	

Table 4: Controllable and uncontrollable events in Figure 14 as mapped to the position sensor.

6.3 Introducing Confidence Measures in the Event Set

Up to now we have addressed events as if their occurrence were easily detectable and thus allowing a sensor to assert them. Yet, while it is possible in some cases, for the majority of events it is not. Furthermore, more than one sensor could report that some event has occurred within a period of time. In that case it is important to establish some mechanism to decide the state the system is in based on the events occurred.

A vision sensor can not quite detect contact at the moment it occurs from a fixed view point unless some particular conditions are met. When the sensitivity of the force sensor determines that contact has occurred it is often the case that the two surfaces have already been in physical contact or that the contact is only partial. Furthermore, if the target object, for instance, is rather soft, then some degree of compression must take place in the target object before the force sensor registers contact.

While the above instances may be better handled if there is some a priori information about the objects interacting, the problem does not vanish; on the contrary, it may be only that the granularity of the problem is reduced. Often that is sufficient to allow the assertion to take place with high confidence.

It is, therefore, necessary to define a confidence measure over the set of events and determine a way to combine or resolve the information that is provided by the different sensors. Hence we distinguish

- the *confidence measure of an event* as a measure of the observability of the transition by a

Event Mapping Table

SUPERVISOR	FORCE	POSITION	VISION	COVERED
α_1	—	—	ν_1	$\surd$
α_3	—	—	ν_2	$\surd$
α_3	β_3	ρ_3	—	$\surd$
α_4	—	—	ν_4	$\surd$
α_5	β_5	—	—	$\surd$
α_6	—	ρ_6	ν_6	$\surd$
α_7	β_7	—	ν_7	$\surd$
α_8	β_5	—	—	$\surd$
α_9	β_9	ρ_9	—	$\surd$
α_{10}	—	—	ν_{10}	$\surd$
α_{11}	β_{11}	ρ_{11}	—	$\surd$
α_{12}	—	—	—	
α_{13}	β_{13}	—	—	$\surd$
α_{14}	—	—	ν_{14}	$\surd$
α_{15}	—	ρ_{15}	ν_{15}	$\surd$
α_{16}	β_{16}	—	—	$\surd$
α_{17}	β_{17}	—	ν_{17}	$\surd$
α_{18}	β_{18}	ρ_{18}	ν_{18}	$\surd$
α_{19}	—	—	—	
α_{20}	—	—	ν_{20}	$\surd$
α_{21}	—	—	ν_{21}	$\surd$

Table 5: The table shows the events mapping from the supervisor to the sensors. We notice that all events are covered except for events: α_8, α_{12}, and α_{19}. These represent logical transitions which only take place in the supervisor.

State Mapping Table

SUPERVISOR	FORCE	POSITION	VISION
S	NF	NC	N
A	NF	NC	A
Ci	F	C	C
sf	sf	—	sf
P	F	I	I
G	G	G	I
fc	—	—	D
fp	fp	C	C
fg	—	I	I
E	F	I	I
fe	fe	—	—
Ce	F	C	C
sf	sf	—	sf
D	NF	NC	D

Table 6: Aliasing in the states after the mapping.

sensor.

- the *likelihood of an event* as the probability that a given transition between nodes occurs based on the combination of evidence from different sensors.

The first measure describes a transition which takes place in the observer while the second one identifies a transition taking place in the supervisor.

6.4 Verification of the Task

The functional behavior defined so far has not addressed the verification of the success of the operation. This is because we have assumed that the events provided between states are observable and that they provide means of disambiguating clearly whether or not the interaction was successful.

Assume, for instance, that the vision sensor did not provide enough resolution to observe penetration of the tool in the target; and that the conclusion deduced from the force sensor and the position sensor was that the object was pierced. It is still possible that either the target object or the tool were (partially) elastic.

It is clear that in this instance a verification mechanism is needed. A very simple one, provided that the operations are repeatable, would be to repeat the functional test. If indeed the object has been pierced, the tool will encounter a different type of resistance from the surface. If the functional test for verification produces forces in the same range as in the original test, then one may conclude that the initial operation was unsuccessful. Further examinations and reasoning should reveal the reason.

We can conclude that verification is required if the data provided by the different sensors lead to contradicting or differing conclusions. In order to make our investigation feasible and address the verifiability of a task, we assume that if the interaction requires verification the verification can be carried out. Hence verification can be understood as another task meant to resolve the ambiguity left from the previous functional task.

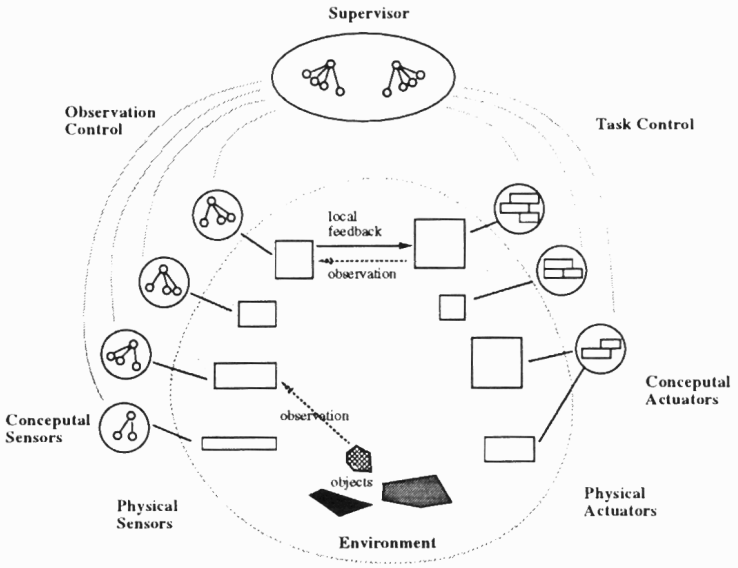


Figure 15: Conceptual description of a task and its mapping into an environment.

7 System Description

In the first portion of this section we present the task mapping from abstract task description to the instantiated task. Furthermore, we highlight the role of the supervisor. In the second part we give the actual system description employed in the experiments described in section 8.

7.1 From Abstract to Instantiated Task

Thus far we have seen a formalism for describing a manipulatory task (section 4). We have presented an abstract description of the task of piercing, addressed its composition, and reasoned about its piecewise observability (section 6.1). In this section we will discuss further the role of the supervisor, and we will investigate how the abstract description is to be instantiated into a specific context.

A supervisor, in its function of controlling a plant, has the dual role of *observation* and *task* control. In Figure 15 we have shown a task mapped into the supervisor on the observation side of the control and its mirror image on the task side of the control. The monitoring of a behavior is reflected in the constant interweaving of observation, from the sensors, and commands, sent to the actuators.

An abstract task description in the supervisor is mapped to the conceptual sensor description and to the actuators. These mappings allow to address the observability of the events. The

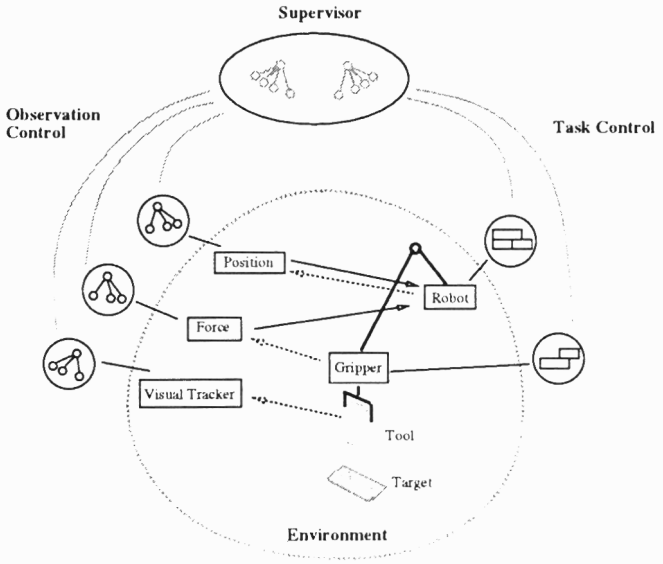


Figure 16: Conceptual description of a piercing task and its mapping into an environment.

individual subtasks are mapped to the physical sensors and actuator in the environment (Figure 16). Since we are dealing with a system needing real-time interaction and control, in some cases a feedback control loop needs to bypass the communication with the supervisor and to be handled locally. This would be the case when a guard event is observed and acted upon.

The mapping from the abstract task description at a sensor level and the physical level constitutes a wide area of research. Ideally one would like to be able to define a translator–like interface able to take into consideration the task constraints and goals as well as the physical constraints of the context in which the task is mapped to. Efforts addressing this issues are rather limited due to the complexity of the problems and the wide variety of both sensors and algorithms to obtain observations and interact with the environment.

7.2 System Implementation

We are currently developing a system for testing manipulatory functionalities which can be described and observed in terms of visual tracking and contact forces. The system set up presented here does not include the tracking portion. The current focus is on the force and position sensors; the vision sensor will be added subsequently.

The contact sensor, a compliant wrist [Xu, 1989; Lindsay, 1992] with 6 degrees-of-freedom, is mounted on the end-effector of a Puma 560 arm and holds the tool (Figure 17). The diagram of

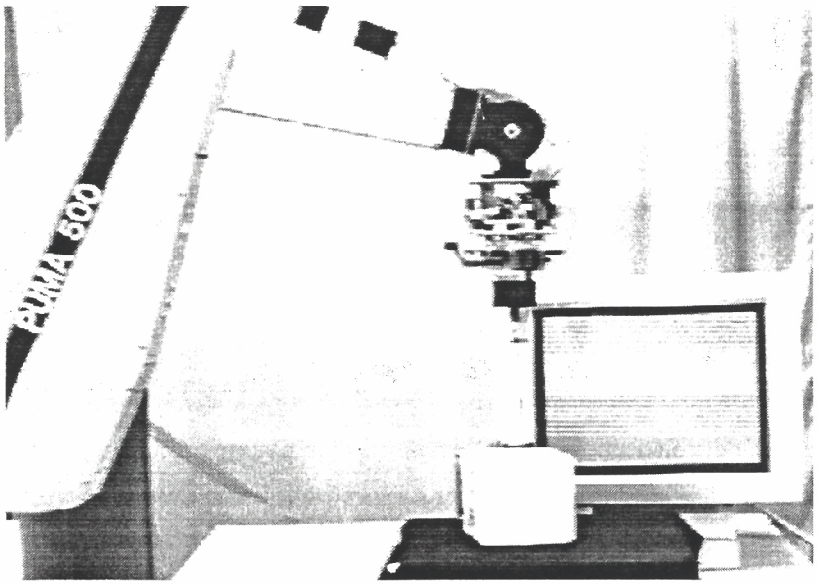


Figure 17: Puma robot 560 with compliant wrist holding a screwdriver and below, on the black background, a target object to operate on.

the system is schematically described in Figure 16.

The basic classification of force applied to the target object by the tool is currently described as belonging to three different classes: *No Contact*, *Contact*, *Too Large*. This class distinction on based on the work by [Stein and Paul, 1993]. This approach allows the classes to be defined dynamically based on the value obtained at contact time and on the noise in the sensor.

8 Experiments

In this section we describe four experiments we carried out: the first three are unsuccessful while the last one is successful. The purpose of these experiments is to illustrate the control behavior accomplished for the task of piercing.

The experiments performed are as follow:

1. Contact with a hard surface (wood) using a screwdriver, yielding a failed piercing. Maximum guard force threshold is quickly achieved without any piercing.

2. Contact with a compliant surface (pressed foam) using a mallet, yielding a failed attempt. In this case, while the surface is elastic, the tool fails to pierce and eventually reaches the guarded value for force threshold.

3. Contact with a compliant surface (pressed styrofoam) using a screwdriver. In this case, too, the operation is unsuccessful. This experiment is shown in Figure 23.

4. Contact with a compliant surface (insulation styrofoam for constructions) with a screwdriver, results in a successful interaction. This operation is presented in detail below and shown in Figures 19, 20, and 21.

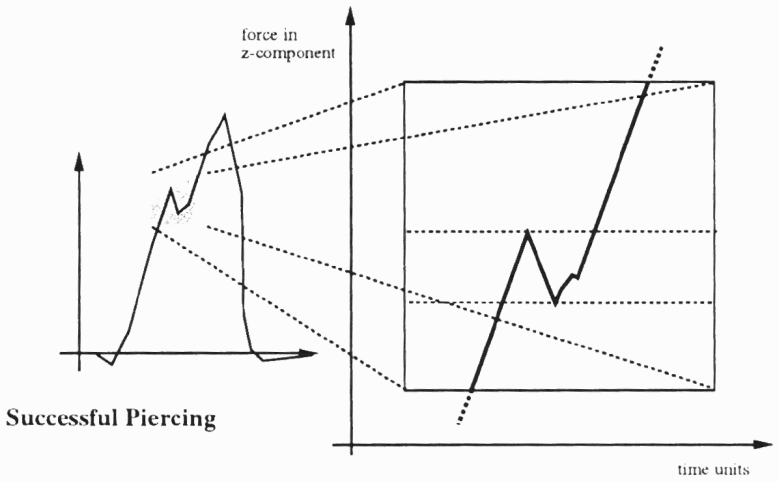


Figure 18: 1 Force transition upon breaking surface.

In the experiments carried out, we have identified a successful transition to piercing as the occurrence of two events measured by different sensors:

- the force profile in the z-component indicating a breaking of the surface (shown in Figure 18),
- a penetration of a certain depth after breaking the surface.

The reason for requiring both force and displacement was that we wanted to differentiate piercing from denting and that we wanted to investigate the interaction of both sensors.

In the experiments outlined here we show a list (Figures 19, 20, and 21) the successful sequencing through the states presented in the task description of section 6.1. These are followed by two force profiles of interactions highlighting the transition of force occurring when the surface is penetrated,(Figure 22). This experiment is followed by a sequence illustrating an interaction leading to a failure to pierce (Figure 23). Only the insertion phase of the interaction is shown. We then follow with two force profiles of interactions, (Figure 24), indicating that there is no transition identifying the piercing event.

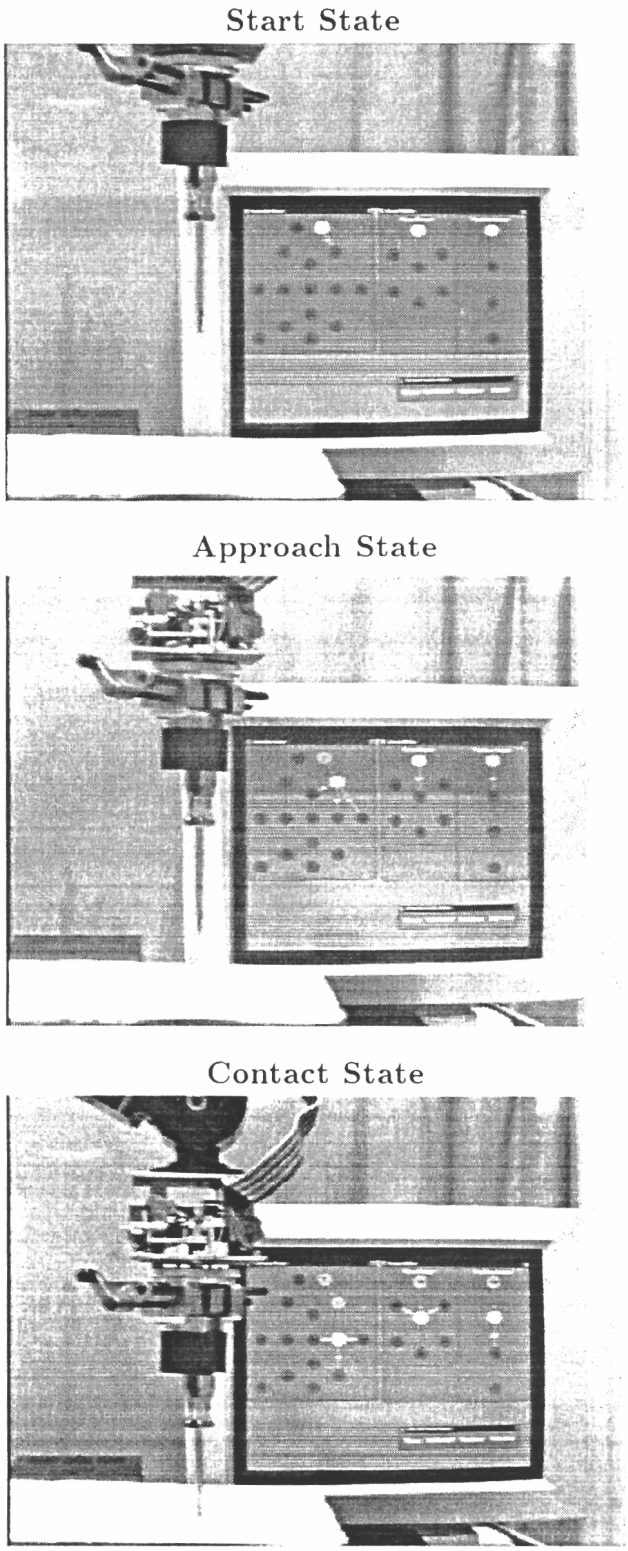
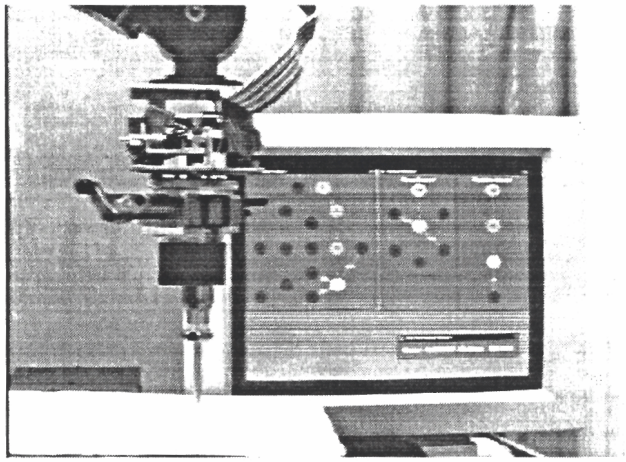
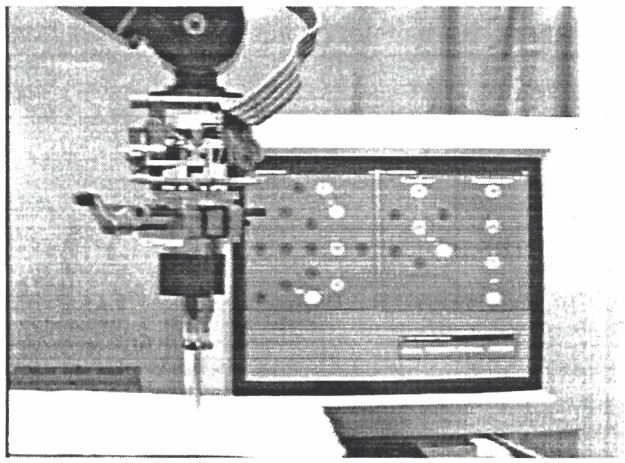


Figure 19: Approach phase in the piercing task.

Piercing State



Goal State



Extraction State

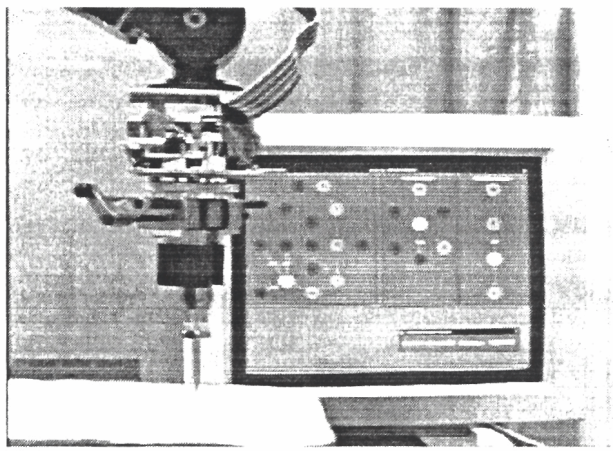


Figure 20: Second Piercing, Achieving of the Goal, and Extraction.

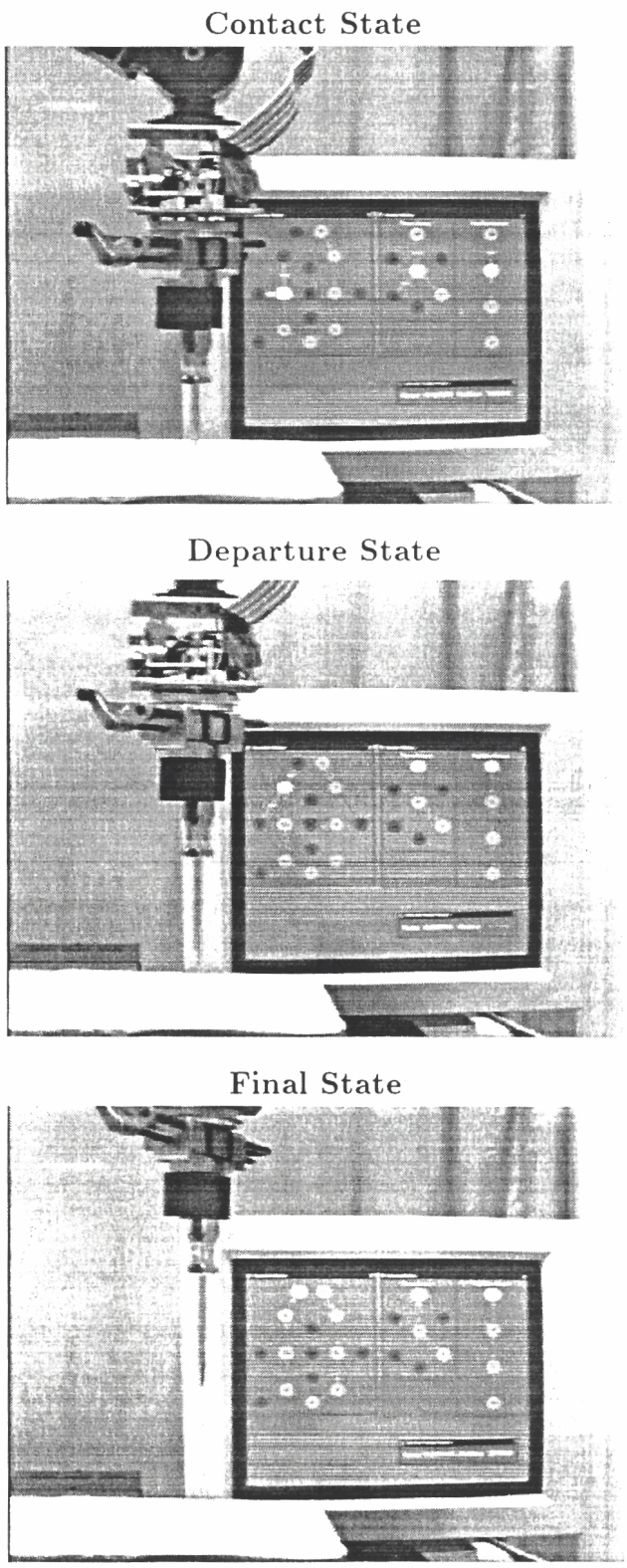


Figure 21: Departure phase in the piercing task.

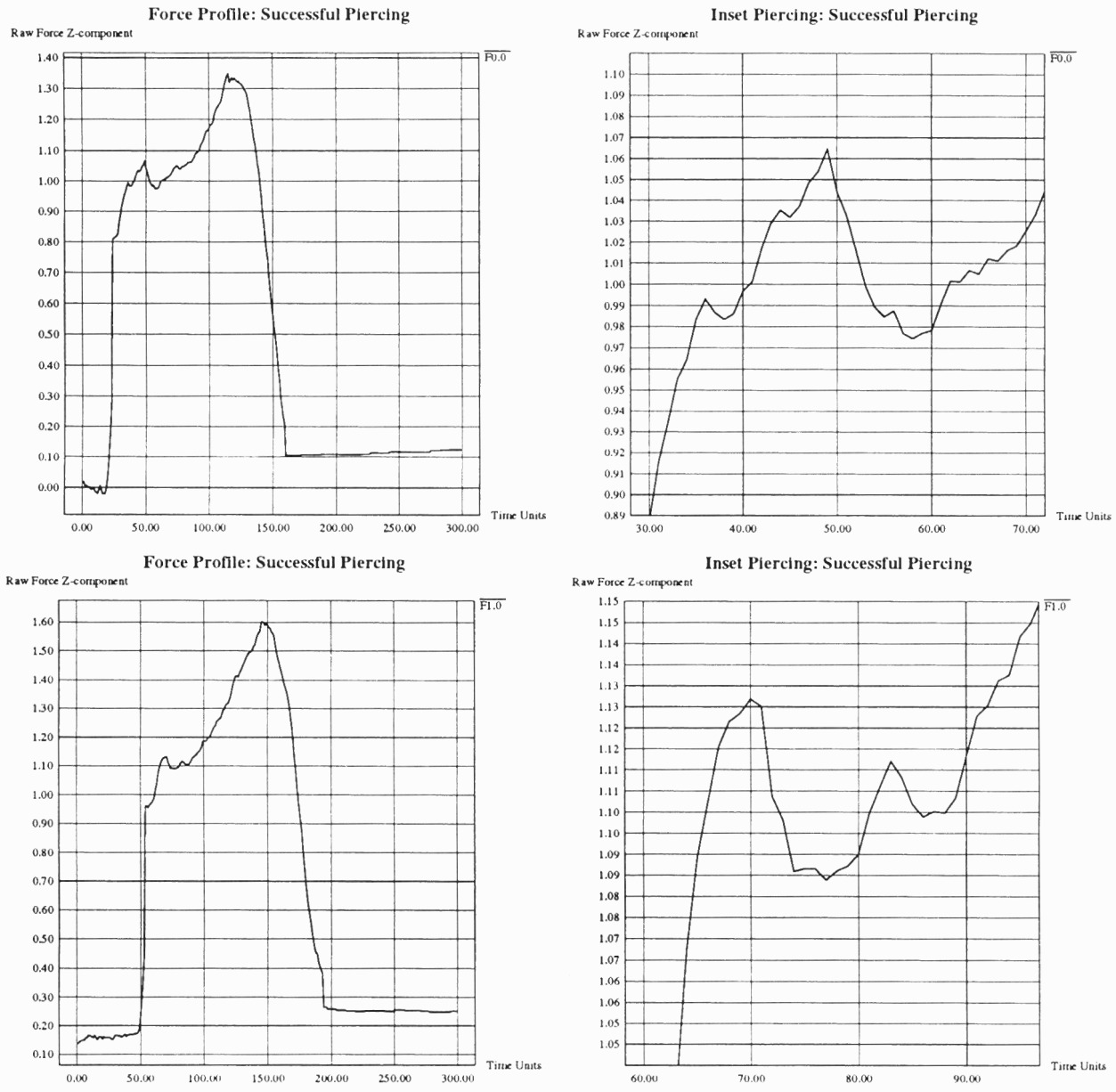


Figure 22: On the Left: Force profiles of successful piercing. On the right: Graphs of the insets of the force profile highlighting the transition in force observed by the sensor upon breaking the surface.

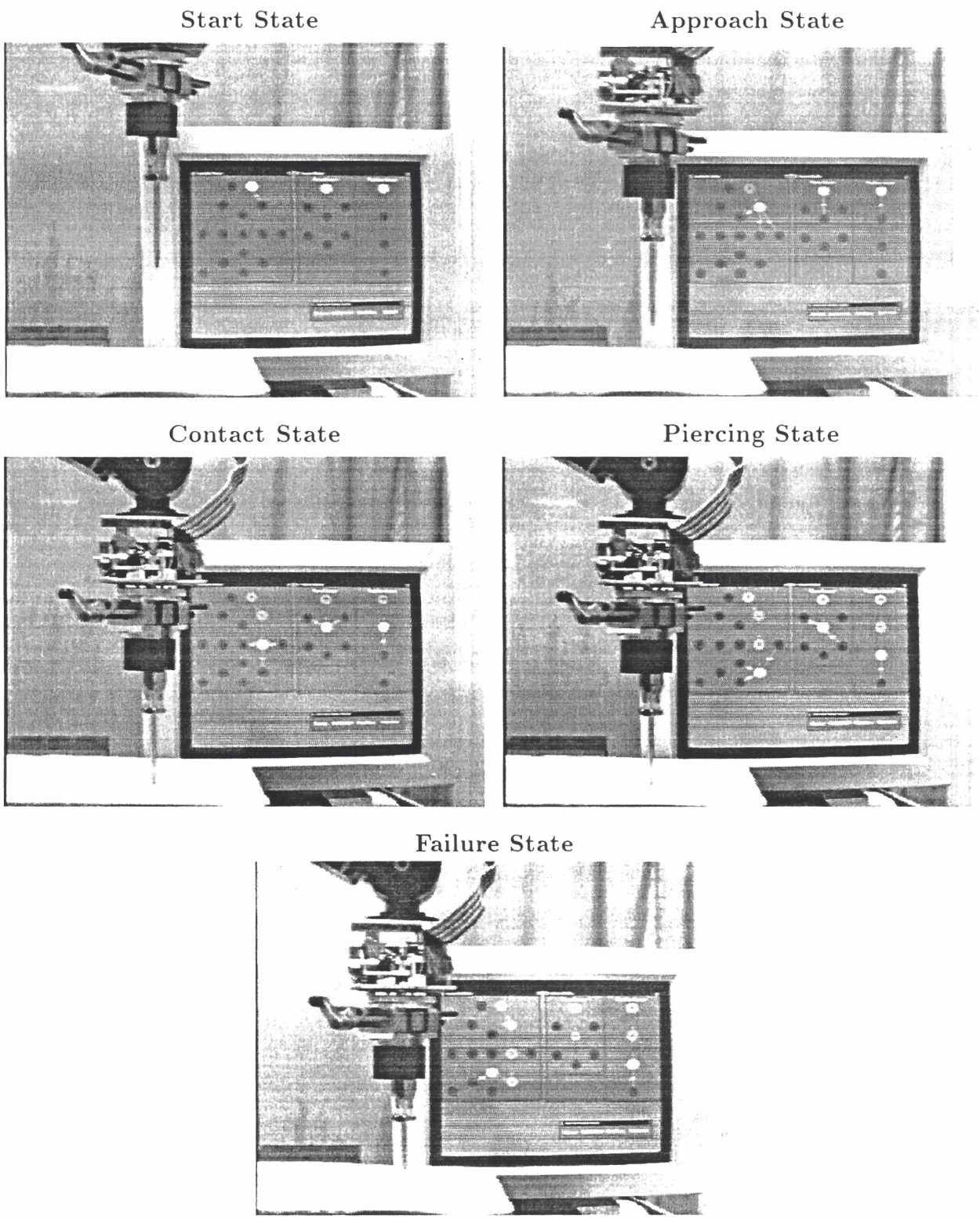


Figure 23: Failed piercing task. The last image shows a different path taken due to encountering too much resistance and having reached the force guard value.

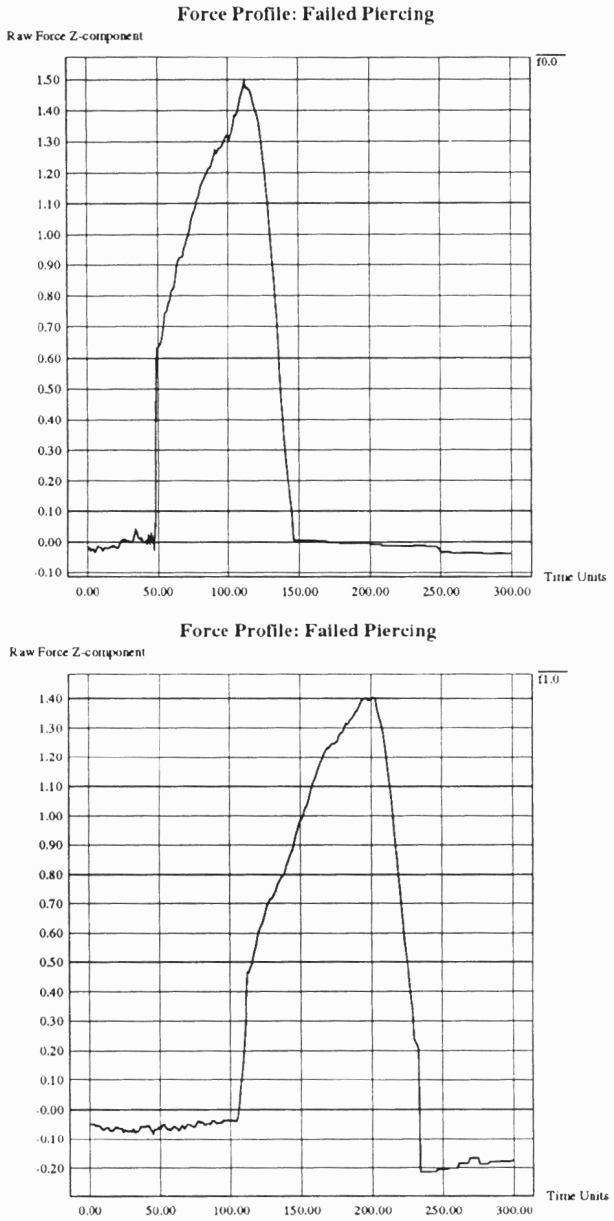


Figure 24: Sensor profile of unsuccessful piercing. We notice that, unlike in the graphs in Figure 22, there is no change in force that would indicate a piercing event.

9 Conclusions

In this paper we have

- given a definition for functionality and presented the importance of addressing functionality recovery or verification in the context of a dynamic environment,
- formalized a representation for a functional behavior and shown how such behavior can be expressed using the formalism of Discrete Event Dynamic Systems,
- focused on one such behavior expressing the functionality of piercing,
- integrated different sensor modalities to allow the observation of a task,
- showed the importance of piecewise observability by different sensors.
- gone from an abstract description of functionality to an instantiated one in an attempt to bridge the gap between the description and the actual execution of a task for a functionality,
- shown three experiments illustrating the interaction between tool and target object controlled by a supervisor module,
- developed a system for conducting experiments on functionality using several sensory modalities and an actuator system.

Future experiments will extend our work to:

- incorporate a visual observer for tracking the tool and monitor the interaction,
- investigate properties of both tools and target object by varying
  - the material of the target object, thus allowing to investigate the range of feasibility of the tool,
  - the material and properties of the tool, both physical and dynamic, to determine, based on previous experiments on ranges of materials, the best tool for a particular task in a given context,
- investigate functional *feasibility* and *performance* of a tool,
- apply learning strategies to the approach so that once a set of experiments has been carried out, the acquired knowledge may be used to guide interactions.

References

[Ambros-Ingerson and Steel, 1990] J.A. Ambros-Ingerson and A. Steel. *Readings in Planning*, chapter Integrating Planning, Execution and Monitoring, pages 735–740. Morgan Kaufmann, 1990.

[Asfahl, 1992] C.R. Asfahl. *Robots and Manufacturing Automation*, chapter Getting ready to Automate, pages 1–21. John Wiley and Sons, 1992.

[Bogoni and Bajcsy, 1993] L. Bogoni and R. Bajcsy. An active approach to characterization and recognition of functionality and functional properties. In *AAAI Workshop: Reasoning about Function*, pages 9–16, 1993.

[Brady *et al.*, 1985] M. Brady, P.E. Agre, D.J. Braunegg, and J.H. Connell. The mechanic's mate. In *Advances in Artificial Intelligence: European Conference of Artificial Intellingece*, pages 79–94, 1985.

[Brand, 1993] M. Brand. Vision system that see in terms of function. In *AAAI Workshop on Reasoning about Function*, pages 17–22, 1993.

[Brooks, 1980] R.A. Brooks. Symbolic reasoning among 3-d models and 2-d images. *Artificial Intelligence*, 17:285–348, 1980.

[Campos, 1992] M. Campos. *Robotic Exploration of Material and Kinematic Properties of Objects.* PhD thesis, University of Pennsylvania, 1992.

[Chapman and Agre, 1987] D. Chapman and P. Agre. *Reasoning about Actions and Plans*, chapter Abstract Reasoning as Emergent from Concrete Activity. Morgan Kaufmann, 1987.

[Connell and Brady, 1987] J.H. Connell and M. Brady. Generating and generalizing model of visual objects. *Artificial Intelligence*, 31:159–183, 1987.

[Cutkosky, 1989] M. R. Cutkosky. On grasp choice, grasp models, and the design of hands for manifacturing tasks. *IEEE Transactions on Robotics and Automation*, 5(3):269–279, 1989.

[Fikes and Nilsson, 1971] R. E. Fikes and N. J. Nilsson. Strips: a new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189 – 208, 1971. An early planning system used as the basis for many subsequent simulation and planning systems.

[Freeman and Newell, 1971] P. Freeman and A. Newell. A model for functional reasoning in design. In *Proceedings of IJCAI-71*, pages 621–632, 1971.

[Goodall, 1986] J. Goodall. *The Chimpanzees of Gombe, Patterns of Behaviour*, chapter Object Manipulation, pages 535–564. Belknap Press of Harvard University Press, 1986.

[Grace, 1989] R. Grace. *Interpreting the Function of Stone Tools*, pages 9–16, 106–115. B.A.R. International Series, Oxford, England, 1989.

[Hammond, 1986] K. Hammond. Chef: a model of case-based planning. In *Proceedings of AAAI*, 1986.

[Iberall *et al.*, 1988] T. Iberall, J. Jackson, L. Labbe, and R. Zampano. Knowledge-based prehension: capturing human dexterity. In *International Conference on Robotic Research*, pages 82–87, 1988.

[Ikeuchi and Hebert, 1990] K. Ikeuchi and M. Hebert. Task oriented vision. In *Image Understanding Workshop*, pages 497–507, 1990.

[Jones and Lozano-Perez, 1990] J.L. Jones and Tomas Lozano-Perez. Planning two-fingered grasps for pick-and-place operation on polyhedra. In *International Conference on Robotic Research*, pages 683–688, 1990.

[Jordan, 1991] D.S. Jordan. *The Role of Physical Properties in Understanding the Functionality of Objects.* PhD thesis, Stanford University, 1991.

[Kaebling, 1990] L. Kaebling. *Readings in Planning*, chapter An Architecture for Intelligent Reactive Systems, pages 713–728. Morgan Kaufmann, 1990.

[Kanazawa and Dean, 1989] K. Kanazawa and T. Dean. A model for projection and action. In *Proceedings of the International Joint Conference of Artificial Intelligence*, pages 985–999, 1989.

[Klatzky *et al.*, 1987] R. Klatzky, S. Lederman, and C. Reed. *There's more to touch than meets the eye: The salience of Object's Attributes for haptics with and without vision.* Technical Report, University of California, Cognitive Science, Santa Barbara, CA, 1987.

[Koivunen and Bajcsy, 1993] V. Koivunen and R. Bajcsy. Geometric methods for building CAD models from range data. In *Geometric Methods in Computer Vision II*, pages 205–216, 1993.

[Košecká and Bajcsy, 1993] J. Košecká and R. Bajcsy. Cooperation of visually guided behaviors. In *Fourth International Conference on Computer Vision*, pages 502–506, 1993.

[Lederman and Klatzky, 1987] S. Lederman and R. Klatzky. Haptic classification of common objects: knowledge-driven exploration. *Cognitive Psycology*, 22():421–459, 1987.

[Lindsay, 1992] T.S. Lindsay. *Teleprogramming: Remote Site Robot Task Execution.* PhD thesis, University of Pennsylvania, 1992. Technical Report MS-CIS-92-64.

[Lowry, 1982] M.R. Lowry. Reasoning between structure and function. In *Proceedings fo the Image Understanding Workshop*, pages 260–264, Science Applications In., Mclean VA, 1982.

[Lozano-Perez *et al.*, 1987] T. Lozano-Perez, J.L. Jones, E. Mazer, P.A. O'Donnell, W.E. Grimson, P. Tournassound, and A. Lanusse. Handey: a robot system that recognizes, plans and manipulates. In *International Conference on Robotic Research*, pages 843–849, 1987.

[McAllewster and Rosenblitt, 1991] McAllewster and Rosenblitt. Systematic non linear planner. In *Proceedings of AAAI*, 1991.

[Pentland, 1986] A.P. Pentland. Perceputual organization an dthe representation of natural form *Artificial Intelligence*, 28:293–331, 1986.

[Ramadge and Wonham, 1989] P.J.G. Ramadge and W. M. Wonham. The control of discrete event systems. In *Proceedings of the IEEE*, pages 81–97, 1989.

[Rivlin *et al.*, 1993] E. Rivlin, A. Rosenfeld, and D. Perlis. Recognition of object functionality in goal-directed robotics. In *Working Notes of AAAI Workshop: Reasoning about Function*, pages 126–130, 1993.

[Rosch, 1973] E. H. Rosch. *Cognitive Development and the Acquisition of Language*, chapter On the Internal Structure of Perceputal and Semantic Categories, pages 111–144. Academic Press, NY, NY, 1973.

[Rosch, 1978] E. H. Rosch. *Cognition and Categorization*, chapter 2 Principles of Categorization, pages 27–48. Erlbaum, Hillsdale NJ, 1978.

[Rosenschein and Kaebling, 1988] S.J. Rosenschein and L.P. Kaebling. The synthesis of digital machines with provable epistemic properties. In *Proceedings of the Conference on Theoretical Aspects of Reasoning about Knowledge*, pages 83–98, 1988.

[Sacerdoti, 1973] E.D. Sacerdoti. Planning in a hierachy of abstraction spaces. In *Proceedings of the International Joint Conference of Artificial Intelligence*, Palo Alto, CA., 1973.

[Sacerdoti, 1977] E.D. Sacerdoti. *A Structure of Plans and Behaviour.* Elsevier-North Holland, 1977.

[Sakaguchi and Nakano, 1992] Y. Sakaguchi and K. Nakano. Active perception with intentional observation. In *Proceedings of the Second International Symposium on Measurement and Control in Robotics*, pages 241–247, Tsukuba Science City, Japan, 1992.

[Salganicoff, 1992] M. Salganicoff. *A Robotic System for Learning Visually-Driven Grasp Planning.* Technical Report, University of Pennsylvania, 1992. Ph.D. Dissertation Proposal.

[Shirai, 1987] Y. Shirai. *Three Dimensional Computer Vision.* Springer-Verlag, 1987

[Sinha, 1992] P. Sinha. *Robotic Exploration Of Surfaces and Its Application to Legged Locomotion.* PhD thesis, University of Pennsylvania, 1992. Technical Report MS-CIS-92-12, GRASP Lab 301.

[Smith and Medin, 1981] E. E. Smith and D. L. Medin. *Categories and Concepts.* Harvard Unversity Press, Cambridge, MA, 1981.

[Sobh, 1991] T. Sobh. *Active Observer: A Discrete Event Dynamic System Model for Controlling an Observer under Uncertainty.* PhD thesis, University of Pennsylvania, 1991. Technical Report MS-CIS-91-99.

[Stark and Bowyer, 1991] L. Stark and K. Bowyer. Generic recognition through qualitative reasoning about 3-d shape and object function. In *Proceedings of CVPR*, pages 251–256, 1991.

[Stark and Bowyer, 1993] L. Stark and K. Bowyer. Function-based generic recognition for multiple object categories. In *Proceedings of CVGIP*, 1993. To Appear.

[Stein and Paul, 1993] M. Stein and R.P. Paul. *Behaviour Based Control in Time Delayed Teleoperation.* Technical Report, University of Pennsylvania, Dept. of Computer and Information Science, Philadelphia, PA., 1993.

[Tate, 1977] A. Tate. Generating project networks. In *Proceedings of the International Joint Conference of Artificial Intelligence*, Boston, MA, 1977.

[Tate *et al.*, 1990] A. Tate, J. Hendler, and M. Drummond. *Readings in Planning*, chapter A Review of AI Planning Techniques, pages 26–49. Morgan Kaufmann, 1990.

[Tsikos, 1987] C.I. Tsikos. *Segmentation of 3-D scenes using machine vision and programmable mechanical scene manipulation.* PhD thesis, University of Pennsylvania, 1987. Technical Report MS-CIS-87-103.

[Winston, 1980] P.H. Winston. Learning and reasoning by analogy. *Communication of ACM*, 23(12):689–703, 1980.

[Winston *et al.*, 1984] P.H. Winston, T.O Binford, B. Katz, and M. Lowry. *Learning Physical Descriptions from Functional Definitions, Examples, and Precedents*, chapter Connecting Perception to Action, pages 117–135. MIT Press, Cambridge, MA., 1984.

[Xu, 1989] Yangsheng Xu. *Compliant wrist design and hybrid position/force control of robot manipulators.* PhD thesis, University of Pennsylvania, 1989.