

Rule-based Contact Monitoring Using Examples Obtained by Task Demonstration

Pavan Sikka and Brennan J. McCarragher

Department of Engineering, F. E. I. T.

The Australian National University

Canberra, ACT 0200, Australia

Abstract

This paper presents a new rule-based hybrid-system approach to contact monitoring. The rules are formulated in terms of temporal sequences of the contact force and they recognize temporal patterns of force associated with contact states and transitions. The rule-base is built using inductive learning techniques on force data obtained by human demonstration. This approach is suitable for monitoring robotic as well as human tasks. An advantage of this approach is that it allows process monitors for different tasks to be built quickly and easily by learning new sets of rules from a demonstration of the tasks. Experimental results are presented to demonstrate the effectiveness of this approach.

1 Introduction

Robotic systems are prone to errors due to uncertainties in the model of the process and of the environment. This makes process monitoring an important part of a robotic system. Process monitoring is based on continuously observing and interpreting the data from sensors that report on the system state, and is used for error detection, recovery, control, adaptation and learning. The focus of this paper is the design of process monitors using inductive learning techniques within a hybrid dynamic systems framework.

Robotic tasks can often be modeled as hybrid dynamic systems [Stiver and Antsaklis, 1992]. A task level description is based on discrete event dynamic systems [Ramadge and Wonham, 1989], while the physical aspects of the system are modeled using continuous variable dynamic systems. In this hybrid dynamic systems framework, a process monitor is defined as a mapping that transforms the physical state of the continuous system into discrete states and events at the task level. This approach can be contrasted with statistical change de-

tection techniques for contact tasks. For example, Eberman and Salisbury [Eberman and Salisbury, 1994] consider a temporal sequence of the force signal to determine contact type; however, there is no consideration of a discrete-event task model and their approach is focused on the continuous variable representation. This paper presents a rule-based method for contact monitoring where temporal sequences of quantized continuous variables are matched with rules to determine the states and transitions of the discrete-event system.

Task demonstration has been used for task-level robot programming. Delson and West [Delson and West, 1996] considered the problem of robot programming from demonstrated motion during one task-level state and did not address the problem of contact monitoring. Wang *et al.* [Wang *et al.*, 1996] present a method to derive compliant motion programs from human demonstration. However, an explicit discrete-event model of the task is not considered, and hence the subtasks cannot always be identified with task-level states. In our approach, the rules are obtained from data generated by several demonstrations of the task being considered. The data obtained from demonstration is clustered into a small number of groups corresponding to quantized values of the continuous variables. Inductive learning techniques are then applied to the discretized data to obtain rules for monitoring and classifying the discrete states and events of the hybrid system. Since the rules pertain to both discrete states and events, this approach exploits the dynamics of the task, as opposed to previous work in assembly based on the quasi-static assumption which ignores dynamic effects (see, for example, [Whitney, 1982; Bruyninckx *et al.*, 1995]).

Our approach to obtaining the rules from examples can be contrasted with the model-based approach in [McCarragher and Asada, 1993] where qualitative reasoning techniques are used to obtain simple templates for recognizing contact state transitions. There is only one template for each contact state transition. The templates are based on the current values of quantized variables and

do not consider temporal sequences. Furthermore, qualitative reasoning techniques are difficult to apply consistently to continuous dynamic models since it is difficult to obtain consistent qualitative algebras [Williams and de Kleer, 1991].

There are several advantages of this approach. Since the analysis is based on real sensory data, the approach has the advantage of taking into account various task parameters such as friction, which are difficult to model analytically. Most important, the approach can be adapted easily and quickly to different tasks by simply learning new sets of rules from the data obtained by task demonstrations. Since the learning phase is based on a demonstration of the task, this approach is applicable both to robotic process monitoring and to the study of human demonstration of tasks. Finally, our approach is general in that it can be applied to one or more sensory modalities taken together, for example position and force.

2 Contact Monitoring using a Hybrid System Model

We consider contact tasks, such as assembly, to illustrate our approach to process monitoring. An example of modeling an assembly task as a hybrid dynamic system is presented in [McCarragher and Asada, September 1995]. These tasks can be characterized by the different configurations of contact between the parts being assembled. The configurations are described geometrically in terms of contact between the edges and surfaces of the objects involved and are called *contact states*. The contact states are organized into a graphical structure $G = (\Gamma, E)$, called the *contact state network*, where Γ is the set of nodes representing the contact states γ and E is the set of arcs connecting contact states and representing the allowed changes in contact states [Asada and Hirai, 1989]. This is illustrated in Figure 1, which shows a part of the contact state network corresponding to the planar peg-in-hole task. At the plant level, the contact states determine the geometrical constraints which, together with the equations of motion, determine the plant behavior in terms of state variables such as position, velocity, acceleration, and force.

The contact monitor tracks the state of the continuous plant by monitoring variables of interest such as the position of the robot and the force acting on the robot. The organization of the contact monitor within the hybrid-dynamic framework is shown in Figure 2. The system is modeled as a combination of a continuous-domain system interacting with a discrete-event system through an interface. The interface acts as a mapping between the two sub-systems, transforming and inter-connecting the discrete and continuous inputs and outputs between the two sub-systems. Within this framework, the transformation between the discrete event system and the con-

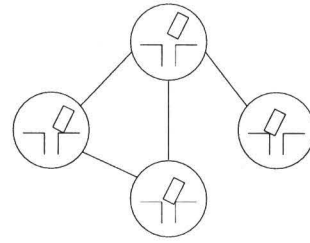


Figure 1: A part of the contact-state network for the peg-in-hole task. Each circle denotes a contact-state, with the figure inside the circle illustrating the corresponding peg-in-hole configuration. Each arc represents an allowed change in contact-state.

tinuous system becomes a key issue, and the process monitor is represented as a part of the interface. The vector $\mathbf{z}(t)$ denotes the state of the plant as seen by the process monitor, and consists of variables that are of interest to the process monitor. For example, in the planar case with force and changes in force being the variables of interest, $\mathbf{z}(t) = [f_x(t), f_y(t), \dot{f}_x(t), \dot{f}_y(t)]^t$, where f_x and f_y denote the planar components of force.

The contact monitor works in two steps. In the first step, the process monitor samples the variable $\mathbf{z}$ and then *quantizes* or *discretizes* it into the variable $\mathbf{z}_q$. For example, in an assembly task, $\mathbf{z}_q = [f_{xq}, f_{yq}, \dot{f}_{xq}, \dot{f}_{yq}]^t$. The discretization is performed using discriminant functions g_{γ_c} based on the current contact state γ_c . The discriminant functions are also obtained from the data generated by human demonstration. The details of this step can be found in [Sikka and McCarragher, 1996].

The second step of the process monitor consists of using a rule-based system [Barr and Feigenbaum, 1981] to identify the process state γ based on a sequence of discretized variables. This process is described in detail in the next section.

3 Rule-based Contact Monitoring

Each contact state transition can be characterized by temporal patterns of force. These patterns can easily be encoded in the form of rules for identifying the corresponding contact state transitions. This observation is the basis for our approach to rule-based contact state and transition identification.

A rule is defined as a *condition-action* pair where the condition part specifies the conditions under which the rule can be applied and the action part specifies the actions that are taken when the rule is applied. For example, in the context of contact monitoring, the following general rule can be used for recognizing a contact state transition:

IF
state = γ_1 AND

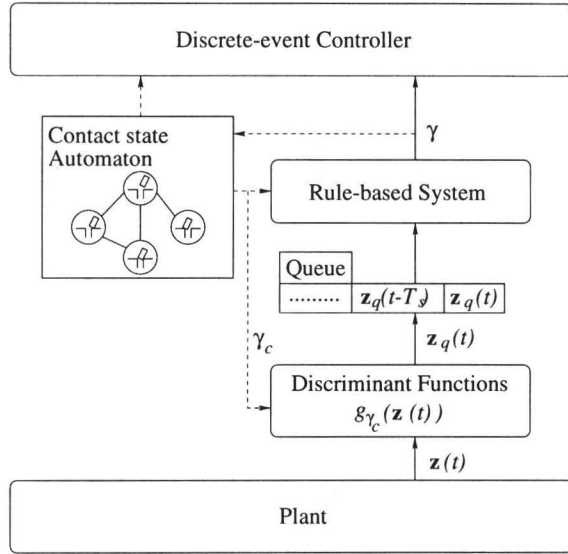


Figure 2: The process monitor within a hybrid system framework. The dotted line encloses the main functional blocks of the process monitor.

$$\begin{aligned}
 \dot{f}_{xq}(k) &= d_1 & \text{AND} \\
 \dot{f}_{xq}(k-1) &= d_1 & \text{AND} \\
 \dot{f}_{xq}(k-2) &= d_2 & \text{AND} \\
 \dot{f}_{xq}(k-3) &= d_3 & \text{AND} \\
 \dot{f}_{xq}(k-4) &= d_1 \\
 \text{THEN} \\
 \text{state} &= \gamma_2
 \end{aligned}$$

Here, k refers to the current sampling period, $d_1 \dots d_3$ refer to the discrete values of the quantized variables and $\gamma_1, \gamma_2 \in \Gamma$ denote contact states. To simplify notation, the remainder of this paper uses the corresponding integers i to refer to the discrete values d_i .

The rule shown above checks for a sequence of discretized values of the rate of change of force to recognize a transition from contact state γ_1 to γ_2 . In general, several variables are monitored and the rule-based process monitor consists of sets of rules for each variable and for each contact transition. This leads to a very general system for monitoring contact.

There are several important issues for rule-based systems. The matching of rules to data is computationally expensive [Barr and Feigenbaum, 1981]. A large rule-base coupled with a large database can make the system too inefficient to be of use. Another important issue for rule-based systems is the integrity of the rule-base. A rule-base with conflicting rules can lead to ambiguous results, and hence the rule-base must be checked for conflicts. These problems are addressed in the paper by the choice of finite-state machines as the representation for rules and by the organization of the rule-base using the contact state network.

3.1 Finite-state machines: a representation for rules

A finite-state machine (FSM) is defined as the structure $M = \{S, \Sigma, f, s_0, F\}$, where S is the set of states, Σ is the set of input symbols, $f : S \times \Sigma \rightarrow S$ is the state transition function, $s_0 \in S$ is the initial state, and $F \subseteq S$ is the set of final, accepting states. A FSM operates as follows: The state of the machine is initialized to s_0 . The input to the machine consists of a sequence σ of symbols taken from the input alphabet Σ of the FSM. The input string is processed sequentially. Thus, if σ_i is the next unprocessed symbol in the input, and s is the current state of the FSM, then the FSM changes its state to $f(s, \sigma_i)$ and σ_{i+1} becomes the next symbol to be processed. The FSM stops after processing the last symbol in the input string. Let s_t be the state of the FSM when it stops. If $s_t \in F$, then the FSM is said to *accept* the string σ .

A set of rules can be represented by a FSM under certain conditions. In particular, pattern classification rules can be easily implemented as a set of FSMs with one FSM for each set of patterns representing a class. The input alphabet of the FSM consists of the symbols used to express the patterns. A path from the initial state to a final state of the FSM corresponds to a pattern belonging to the corresponding class, and hence to a rule for that class.

The representation of a set of rules by a FSM significantly reduces the computational effort required to determine the rules that apply to the current information in the database. For a given pattern in the database, the task of finding the matching rule is reduced to determining if the FSM accepts the sequence representing the pattern.

3.2 The contact monitor

Let $\{z_q(k)\}$ denote a temporal sequence of the discretized plant state vector. When this sequence corresponds to a contact state transition, it can be used to characterize the particular transition. In general, a contact state transition can be characterized by a set of temporal sequences or patterns of the discretized plant state vector z_q . Thus, each component of z_q contributes an associated set of patterns for a given contact state transition. Each pattern becomes a rule for recognizing the corresponding contact state transition and these rules together are encoded as a FSM for recognizing the given transition.

The contact monitoring system is built around the contact state network G . Let $e_1, \dots, e_p$ represent the transitions allowed from a contact state γ . As before, let the discretized plant state vector be defined by: $z_q(k) = \mathbf{f}_q(k) = [f_{xq}(k), f_{yq}(k), \dot{f}_{xq}(k), \dot{f}_{yq}(k)]^t$. Then, for a transition e from γ , a set of 4 FSMs are defined,

one for each discretized variable: $M_{f_{xq}}^c, M_{f_{yq}}^c, M_{f_{xq}}^c$ and $M_{f_{yq}}^c$.

The rule-based contact monitor works as follows: Let $\{\dots, \mathbf{f}_q(j), \dots, \mathbf{f}_q(k)\}$ denote the sequence of values of the discretized plant state vector. Then, the input strings for the FSMs consist of the sequence $\{\mathbf{f}_q(k - m\Delta), \dots, \mathbf{f}_q(k - \Delta), \mathbf{f}_q(k)\}$, where Δ is an integer representing a system-dependent delay and m is an integer representing the system-dependent sequence length. The sequence of each discretized variable is used as the input for the corresponding FSM. The transition for which all the FSMs accept the input sequence is taken as the transition occurring at instant k . The parameters m and Δ in the input sequence are chosen based on the considerations such as the computing power available and the sampling rate. For example, in the implementation described later in the paper, the sampling rate is 120 Hz and typically about 60 samples make up a transition. The computing resources limit m to 5. Hence, to represent the transitions, Δ is chosen to be 12.

Organizing the FSMs according to the contact state network has the advantage of simplifying the problem of determining the consistency of rules. The rules corresponding to transitions from a contact state can be examined independently of transitions from other contact states. Furthermore, for a given state, conflicts may exist among rules for a particular variable as long as there are no conflicts when the rules for all the variables are considered together.

4 Contact monitoring and human demonstration

This section addresses the problem of constructing a contact monitor for a given task. This involves several steps. First, the contact state network must be constructed and the variables making up the plant state vector must be specified. Then, for each state, the discriminant functions for quantizing the plant state variables need to be specified. Finally, the FSMs corresponding to the rules associated with each contact state transition must be constructed. In this section, we describe the use of task demonstration to construct the discriminant functions and the FSMs for the given task. The problem of constructing the contact state network for a given task is not addressed in this paper.

A person demonstrates the task several times. Variables constituting $\mathbf{z}(t)$, like position and force, are recorded during each demonstration of the task. Thus, human demonstration of the task yields a set of traces of the plant state $\mathbf{z}$, denoted by $\{y_1, \dots, y_p\}$, where $y_i = \{\mathbf{z}(k)\}$. Each trace y_i is used to generate sub-traces y_{ij} corresponding to the individual contact state transitions present in the trace. This must be done manually,

although automatic change detection algorithms, for example [Eberman and Salisbury, 1994], can be used to aid in this process. These sub-traces are then organized according to contact states. Thus, the set X_{γ_1} contains all the subtraces corresponding to contact state transitions from the contact state γ_1 .

The next step in setting up the contact monitor is to discretize the traces obtained from human demonstration. The discretized sequences are then used to obtain the discriminant functions as well as the rules for the contact monitor. For each state γ_i , the sub-traces in X_{γ_i} are combined into one large set of data points X . Using clustering techniques, the points in X are then grouped into a small number of classes for each component of the plant state vector $\mathbf{z}$. These classes correspond to the discretized values for the variable. In this manner, the clusters are then used to learn the discriminant functions for each of the variables and for each contact state. This process is described in detail in [Sikka and McCarragher, 1996].

Finally, the rules for contact state and transition recognition must be obtained from the clustered data. First, for each state γ_i , the clustered data is regrouped into sub-traces corresponding to the sub-traces in the set X_{γ_i} to obtain the set of sub-traces $X_{q\gamma_i}$. Then, for each variable in $\mathbf{z}$, the sub-traces in $X_{q\gamma_i}$ are used to obtain examples of the corresponding contact state transition. This step is due to computational considerations and is required to reduce the length of long temporal sequences generated as a result of high sampling rate. Thus, we finally obtain the sets $X'_{q\gamma_i}$ corresponding to each contact state γ_i and containing examples of contact state transitions.

In the last step for setting up the contact monitor, the sets $X'_{q\gamma_i}$ are used to learn the FSMs for each variable in $\mathbf{z}$. The FSM generated for each variable recognizes exactly the set of examples used. The FSM is constructed using the algorithm described in [Biermann and Feldman, 1972]. This algorithm constructs an FSM given a set of examples in an iterative manner, with each constructed FSM closer to the given set. The algorithm is based on progressively identifying longer and longer prefixes that are present in the given set of examples.

The approach is illustrated in Figures 3 and 4. Figure 3(a) shows the raw signal corresponding to a gain of contact. The first-difference for this signal, corresponding to the time-derivative, is shown in Figure 3(b). This signal is then clustered into 4 discrete values to obtain the discretized first-difference signal shown in Figure 3(c). Finally, Figure 3(d) shows the input string of length 5 obtained by further sampling the discretized signal, 13111. Such input strings are obtained for each demonstration of the particular contact state transition to generate a set of input strings, for example, {13111,

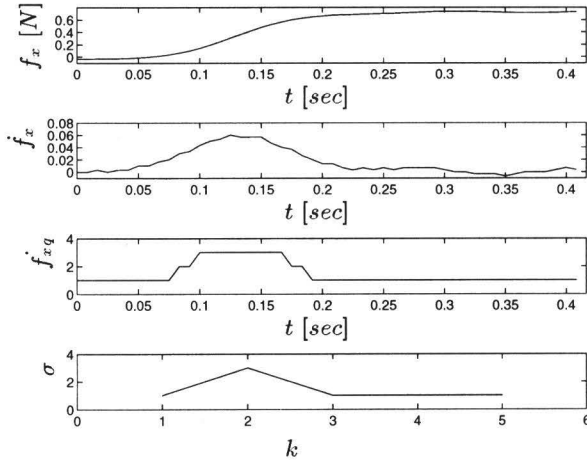


Figure 3: The transformation of a continuous signal f_x corresponding to a transition into a discrete sequence σ of five symbols, 13111.

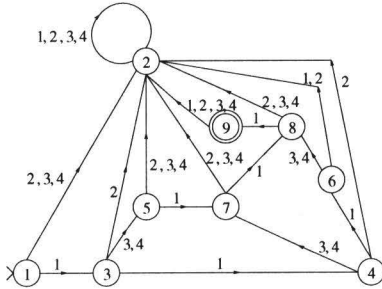


Figure 4: The automaton obtained from the training set {13111, 14111, 11311, 11411, 11131, 11141}.

14111, 11311, 11411, 11131, 11141}. Figure 4 shows the FSM obtained for this set using the algorithm described in [Biermann and Feldman, 1972]. The FSM consists of 9 states, with the state labeled 1 being the initial state s_0 , and the state labeled 9 being the only accepting state, i.e., $F = \{s_9\}$.

5 Experimental Results

The rule-based process monitor was tested on a task consisting of several contact states resulting from contact between a rectangular peg and a solid block. These contact states are illustrated in Figure 5 and are denoted by γ_1 , γ_2 and γ_3 . Figure 5 also shows the contact state network for the experimental setup, and the allowed transitions $\tau_1 \dots \tau_6$ between the contact states. The task used to test the process monitor consists of the sequence of contact states $\gamma_1 \rightarrow \gamma_2 \rightarrow \gamma_3 \rightarrow \gamma_1 \rightarrow \gamma_3 \rightarrow \gamma_2 \rightarrow \gamma_1$.

The task was demonstrated several times by a person and the data from three demonstrations was used to set up the rule-base of the process monitor as described in Section 4. Figure 6 shows the output of the two stages of the process monitor used during a task demonstration.

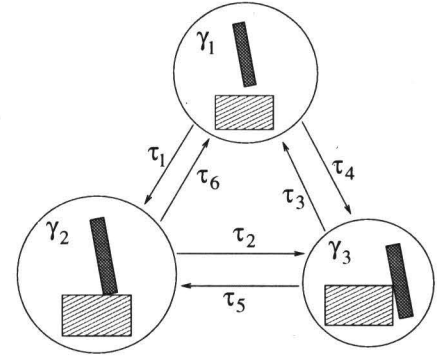


Figure 5: The setup used to demonstrate the process monitoring techniques.

The topmost graph in Figure 6 shows $\dot{f}_x$, the derivative of the force acting tangential to the top surface of the solid block. The second graph shows the discretized variable $\dot{f}_{xq}$ produced by the process monitor. Similarly, the third and fourth graphs relate to the force in the normal direction. The graph at the bottom of Figure 6 shows the discrete events recognized by the rule-based process monitor.

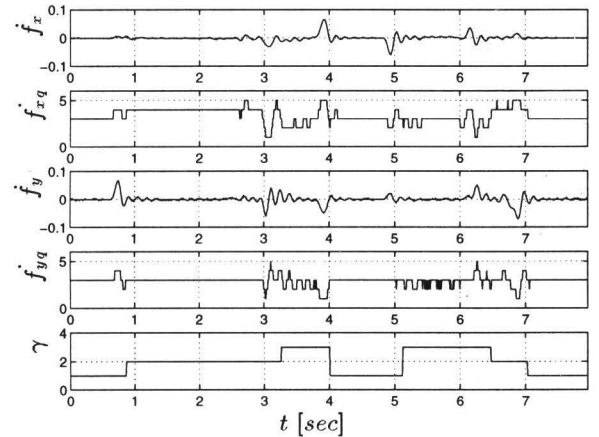


Figure 6: The results obtained from the process monitor for a demonstration of the task.

The performance of the contact monitor was measured by using it to monitor 25 demonstrations of the task described above. The percentage of transitions recognized correctly by the rule-based process monitor is summarized in Table 1 which shows the overall recognition accuracy for the task and the recognition accuracy for each individual transition. The first set of numbers in Table 1 show the performance after initially training the rule-base using 3 demonstrations. However, note that the recognition accuracy of the process monitor for transition τ_2 from contact state γ_2 to γ_3 is very poor at 48 %.

Closer examination of the data revealed qualitative

Performance (%)

	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6	All
After initial training							
	100	48	75	68	93	100	79
After retraining							
	100	72	61	83	93	95	84.5

Table 1: The performance of the process monitor.

differences corresponding to this transition. The contact monitor was therefore changed by replacing one of the demonstrations in the training set and then obtaining a new set of rules for the rule-base. The second set of numbers in Table 1 show the recognition accuracy of the process monitor based on the new rule-base. The Table shows that the recognition accuracy of the process monitor for transition τ_2 improved to 72 % and the overall recognition accuracy also improved to 84.5 %. This demonstrates that the process monitoring system is very flexible and changes can easily be made to improve performance.

6 Conclusion

We have presented a new rule-based method for process monitoring within the framework of hybrid dynamic systems, based on human demonstration. The experimental results indicate that this method is effective and captures the dynamic nature of the sensory data corresponding to gain and loss of contact. Thus, rule-based transition detection, together with pattern recognition based techniques of discriminant functions and clustering methods, provide a flexible and robust method for robotic process monitoring.

References

- [Asada and Hirai, 1989] H. Asada and S. Hirai. Towards a symbolic-level feedback: Recognition of assembly process states. In *5-th International Symposium on Robotics Research*, Tokyo, 1989.
- [Barr and Feigenbaum, 1981] Avron Barr and Edward A. Feigenbaum, editors. *The Handbook of Artificial Intelligence: Vol I*. William Kaufman, Inc., 1981.
- [Biermann and Feldman, 1972] A. W. Biermann and J. A. Feldman. On the synthesis of finite-state machines from samples of their behavior. *IEEE Transactions on Computers*, C-21:592–597, June 1972.
- [Bruyninckx et al., 1995] H. Bruyninckx, S. Dutré, and J. De Schutter. Peg-on-hole: A model based solution to peg and hole alignment. In *IEEE International Conference on Robotics and Automation*, pages 1919–1924, 1995.
- [Delson and West, 1996] Nathan Delson and Harry West. Robot programming by human demonstration: Adaptation and inconsistency in constrained motion. In *IEEE International Conference on Robotics and Automation*, 1996.
- [Eberman and Salisbury, 1994] Brian Eberman and J. Kenneth Salisbury, Jr. Application of change detection to dynamic contact sensing. *The International Journal of Robotics Research*, 13(5):369–394, October 1994.
- [McCarragher and Asada, 1993] Brennan J. McCarragher and Haruhiko Asada. Qualitative template matching using dynamic process models for state transition recognition of robotic assembly. *The ASME Journal of Dynamic systems, Measurement and Control*, 115(2A):261–275, June 1993.
- [McCarragher and Asada, September 1995] Brennan J. McCarragher and Haruhiko Asada. The discrete event modeling and planning of robotic assembly tasks. *The ASME Journal of Dynamic systems, Measurement and Control*, 117(3):394–400, September September 1995.
- [Ramadge and Wonham, 1989] Peter J. G. Ramadge and W. Murray Wonham. The control of discrete event systems. *Proceedings of the IEEE*, 77:81–98, 1989.
- [Sikka and McCarragher, 1996] Pavan Sikka and Brennan J. McCarragher. Monitoring contact using clustering and discriminant functions. In *IEEE International Conference on Robotics and Automation*, 1996.
- [Stiver and Antsaklis, 1992] James A. Stiver and Panos J. Antsaklis. Modeling and analysis of hybrid control systems. In *Proceedings of the 31st Conference on Decision and Control*, pages 3748–3751, 1992.
- [Wang et al., 1996] Qi Wang, Joris De Schutter, Wim Witvrouw, and Sean Graves. Derivation of compliant motion programs based on human demonstration. In *IEEE International Conference on Robotics and Automation*, pages 2616–2621, April 1996.
- [Whitney, 1982] Daniel E. Whitney. Quasi-static assembly of compliantly supported rigid parts. *The ASME Journal of Dynamic systems, Measurement and Control*, 1982.
- [Williams and de Kleer, 1991] B C Williams and J de Kleer. A theory of interactions: Unifying qualitative and quantitative algebraic reasoning. *Artificial Intelligence*, 51:1–79, 1991.