Programming by Demonstration using Hybrid Dynamic System Modelling

Jason Chen & Brenan McCarragher
Department of Engineering, FEIT
Australian National University, Canberra, 0200
chen@faceng.anu.edu.au & brenan@faceng.anu.edu.au

Abstract

This paper presents robot programming by demonstration using hybrid dynamic system modelling. A hybrid dynamic system has been identified as a good way to model tasks involving contacts and constrained motion. We propose a framework where mappings required within the hybrid dynamic system model are extracted from a demonstration. In this paper we concentrate on extracting a mapping for the Event Path Planner, which determines a good path plan for the task. The approach was implemented on an interesting assembly task involving insertion of a compressible spindle between two supports..

1 Introduction

One of the main problems to be solved if robots are to leave their current industrial domain is that of end user programming. An end user with little or no technical expertise needs a method of communicating to a robot what he wants done. A promising solution is Programming by Demonstration (PbD). Here the end user provides a demonstration of the task, with a PbD interface interpreting what is to be done, and communicating it to the robot for execution. This is an easy and natural method for humans to program a robot.

Also to be solved if robots are to be removed from their factory environments is the problem of spatial uncertainty. This is the problem of the robot not knowing the exact position of its end effector relative to other objects in the workspace. For many factory tasks such as welding or spray painting the task tolerance is sufficiently large to not make the spatial uncertainty a problem. However, for tasks involving contacts and constrained motion, (such as in assembly) heavy rigid robot arms and parts located by expensive jigs and fixtures are required to reduce spatial uncertainty sufficiently for the task to be completed. Clearly these methods cannot be transfered outside the factory environment, for example to the home.

A body of research exists which attempts to address the problem of spatial uncertainty. McCarragher and Asada [7] propose a hybrid dynamic system which can be naturally applied to model tasks involving contacts such as assembly. A hybrid dynamic model captures both the continuous and discrete nature of such tasks. The underlying dynamics are continuous, however the series of contact formations made between task objects during execution can be viewed as discrete. This method greatly reduces spatial uncertainty because confirmation of the relative position between the end effector and task objects is repeatedly provided by sensors as contact configurations change. Spatial uncertainty built up while in one contact state can be removed during the transition to the next.

Our research then aims to combine the PbD and Hybrid Modelling paradimes to provide a system that is capable of completing tasks involving workpiece contact, while also allowing new tasks to be programmed easily and naturally by humans. This paper presents one of the first steps in providing such a system, that of constructing a path plan from demonstration for a simple assembly task. The hybrid dynamic system model provides an ideal environment for describing such a path plan. At its discrete level, a path is nicely described as the series of contact formations which were traversed between the start and goal states.

Robot programming by demonstration is an active research area. Atkeson and Schaal [1], and Asada and Lui [5] concentrate on the PbD of continuous type tasks. Atkeson and Schaal teach a robot to invert and balance a pendulum using optimal control and vision sensing. Asada and Lui program deburring tasks by demonstration, using a neural network to map sensed process parameters to an appropriate feedrate and normal stiffness of the grinding tool. Work by these authors [1][5] concentrates on tasks suited to continuous modelling. A task level model is not used.

By contrast, Ikeuchi et al [4], Delson and West [2] use task level discrete state models. Ikeuchi et al [4] use a vision system to identify a change in discrete state during a demonstration. At execution time, a generic database provides the actions required to achieve the observed change in state. No attempt is made to build a path plan from the demonstration. Delson and West [2] model an assembly task as a series of discrete contact formations. However, they concentrate on the continuous motion within each contact state, identifying that variations in demonstrated trajectory result from both adaptation to part misalignment and human inconsistency. They do not address what discrete state path should be used.

Like the authors in this paper, Skubic and Voltz [8] propose a hybrid dynamic model for PbD. A simple assembly task is modelled as an automaton of discrete contact formations. The system is trained to recognise discrete states from force sensor signals using fuzzy logic. Actions taking assembly from one state to another are taken from the demonstration as the smoothed trajectories of the human. The state path through the automaton used by the robot is the one demonstrated. Information from the model is not used to improve the path. What should happen if more than one path is demonstrated is not addressed.

In previous work Chen and McCarragher [3] have partially addressed the issue of selecting a path plan. Suitability of the paths demonstrated were looked at on a complete path basis. This meant the path proposed by the planner needed to be one that was demonstrated. In this paper we break demonstrated paths down into segments called transitions, and select a path based on the suitability of the transitions. Fewer demonstrations are needed because information in each path is used more efficiently, ie. common elements of different paths are analysed as the same entity, not as different entities as occurs when analysing on a complete path basis. The method also allows paths that are not demonstrated to be selected. Good components of different paths can be combined to form a path better than any of those demonstrated.

2 Programming by Demonstration

2.1 Hybrid Dynamic Framework

The task we use for PbD is shown in Figure 1. It involves inserting an axially compressible spindle between two supports, where insertion cannot take place unless the spindle is at least partially compressed. The task is modelled on the household chore of changing a roll on a toilet roll holder, although the roll is left out here without fundamentally changing the task. The task is also simplified to 2D, motion is only considered in the horizontal plane.

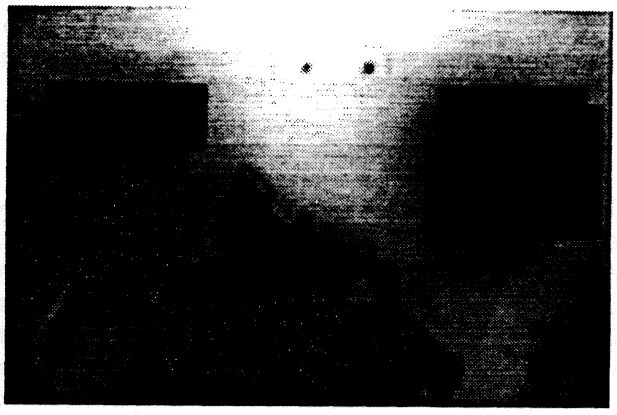


Figure 1: The spindle insertion task chosen for PbD

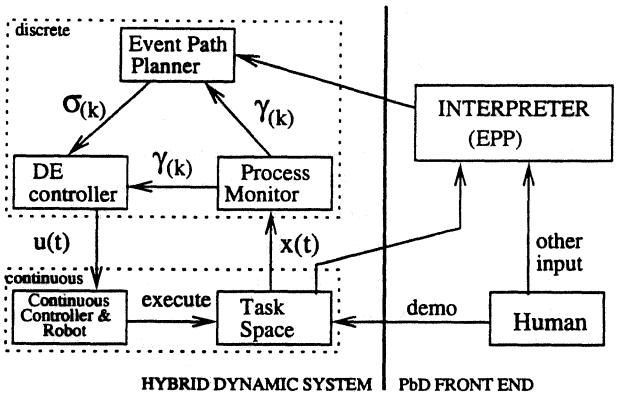


Figure 2: The Programming by Demonstration System

We model the spindle insertion task as a hybrid dynamic system. Figure 2 shows the components of such a system. It consists of the robot and continuous controller, the task space (spindle and supports), the Process Monitor (PM), the Event Path Planner (EPP), and the Discrete Event Controller (DEC). The Process Monitor provides an input into the discrete part of the system from the underlying continuous system, being characterised by the equations:

$$\tau(k) = \psi(\mathbf{x}(t))$$

$$\gamma(k) = \alpha(\gamma(k-1), \tau(k))$$

The kth discrete event $\tau(k)$ is identified by the mapping ψ from $\mathbf{x}(t)$, where $\mathbf{x}(t)$ is the position and orientation state vector of the spindle in the underlying continuous system. The kth discrete state $\gamma(k)$ is then determined by the mapping α from the previous discrete state $\gamma(k-1)$ and the event $\tau(k)$ that occured. For the task presented here, the discrete state $\gamma(k)$ corresponds to one of the spindle-support contact configurations (eg. no contact, or left end of spindle in contact with front

edge of left support, etc). The event $\tau(k)$ corresponds to a change from one contact configuration to another.

The Event Path Planner is characterised by:

$$\sigma(k) = (\tau_{k,d}, \tau_{d,d+1}, ..., \tau_{g-1,g}) = \xi(\gamma(k))$$

The function ξ maps the current state $\gamma(k)$ to discrete event path $\sigma(k)$, where $\sigma(k)$ consists of a sequence of events taking the assembly process from the current state $\gamma(k)$ to the goal state γ_g. The components of the path include events $\tau_{k,d}$(event from $\gamma(k)$ to the next desired state γ_d), $\tau_{d,d+1}$ (event from γ_d to subsequent desired state γ_{d+1}), etc. The final event in the path takes the task from the state γ_{g-1} to the goal state γ_g.

The Discrete Event Controller provides an output from the discrete system back into the underlying continuous system. It is governed by the equation:

$$\mathbf{u}(t) = \phi(\gamma(k), \sigma(k))$$

The function ϕ maps the current state $\gamma(k)$ and an event path $\sigma(k)$ to the continuous command $\mathbf{u}(t)$ sent to the robot continuous controller. $\mathbf{u}(t)$ specifies a trajectory which takes the task from the current state $\gamma(k)$ to the next desired state γ_d in $\sigma(k)$. The feedback loop is completed with the robot executing in the task space, modifying the state vector $\mathbf{x}(t)$.

Each of the mappings ϕ, ξ, ψ and α in the hybrid dynamic system need to be implemented before execution by the robot is possible. These mappings need to come from the demonstration. Here we concentrate on extracting the EPP mapping ξ from the demonstration. The right hand part of Figure 2 represents the PbD front end to the hybrid dynamic system. Human demonstration in the task space is monitored by an interpreter which extracts the relevant information and constructs the mapping ξ in the EPP. Figure 2 shows another input into the interpreter other than the demonstration. There are many other possible forms of human input into the system. In this paper we propose another human input which allows the disposition of the robot to be specified when it selects an event path.

2.2 Discrete System Task Model

The hybrid dynamic framework results in both a continuous and discrete level model of the task. Because here we concentrate on the EPP, for which both its input and output are discrete entities, we focus on the discrete level. Here the task can be modelled as a finite state automaton. Each node of the automaton represents a task state, ie. a distinct contact formation, and each arc represents a transition from one contact formation to another. Figure 3 presents part of the discrete state model of the spindle insertion task, and shows how these states correspond to actual spindle-support contact

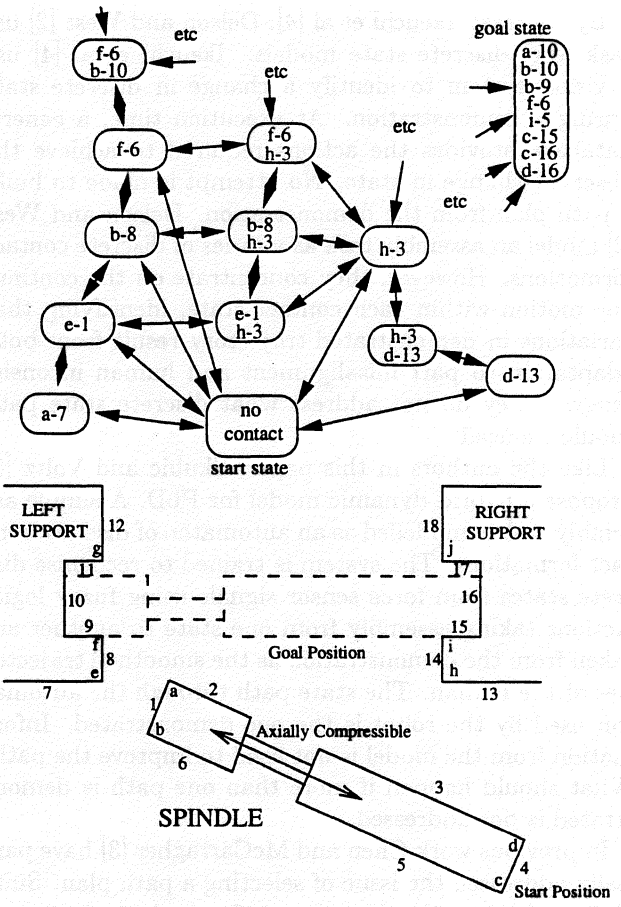


Figure 3: The discrete state representation of the spindle insertion task

formations. Spindle and support vertices were labelled with lower case letters, while edges were labelled with numbers. Contacts can then be described by vertex-edge pairs (eg, a-7 corresponding to the front left corner of spindle in contact with front edge of left support). A state was then defined as corresponding to a unique vertex-edge pair or group of vertex-edge pairs. In total the discrete model had 74 states.

3 The Event Path Planner

The EPP mapping ξ needs to be constructed from information provided by the demonstration. In Figure 4 we present a framework to do just this. A demonstration is first provided by the human. This produces force and position data via a position sensor mounted on the spindle and two force sensors, one beneath each support. The interpreter reads in the force and position data, and as its first operation converts it into the state path to which it corresponds. It does this using the positional information to narrow the choice of states down to a few contiguous states. As in [6], the force data can then be used to identify the appropriate state since

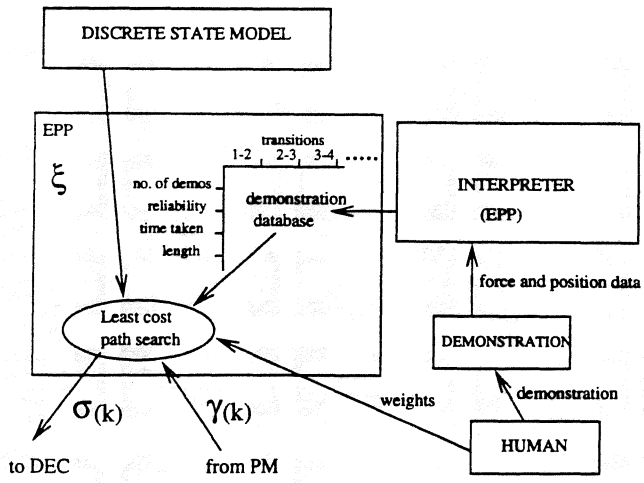


Figure 4: The Event Path Planning Framework

the force spikes produced by each will generally differ in their direction.

The second stage of the Interpreter's operation is to use the state paths generated in stage one to construct the demonstration database. The demonstration database is a two dimensional array with one axis containing the transitions that were demonstrated and the other containing four key performance measures. The database describes the attributes of every demonstrated transition in each of the four performance areas. The four measures we have chosen are:

(N) The number of times the transition was demonstrated.

(L) The length of the transition.

(T) The time taken for the transition to be completed during the demonstration. If the transition was demonstrated more than once, the average time of the group was taken.

(R) The reliability of the transition.

The performance information in the database is used to assign a cost to the transitions in the discrete task model automaton. If a transition is generally performed well, a low cost can be assigned to it in the automaton. Conversely a badly performed transition will be assigned a high cost. The mapping ξ can then be achieved by performing a search for the least cost path $\sigma(k)$ through the automaton between the current state $\gamma(k)$ and goal state.

Because a transition may be good in one performance area and bad in another, a method is required to combine individual performance areas into an overall performance, and then assign this as a cost in the task model automaton. We achieve this by assigning the overall cost of a transition to be:

$$C_{i,j} = W_N Cn_{i,j} + W_T Ct_{i,j} + W_R Cr_{i,j} + W_L Cl_{i,j} \quad (1)$$

where $C_{i,j}$ is the overall cost of transition $\tau_{i,j}$ between states i and j. The costs $Cn_{i,j}, Ct_{i,j}, Cr_{i,j}, Cl_{i,j}$ are derived from how transition $\tau_{i,j}$ was demonstrated in the N, T, R and L performance areas respectively. The cost $Cn_{i,j}$ was defined as:

$$Cn_{i,j} = N_{max} - N_{i,j} + 1$$

where N_{max} was the number of times the most demonstrated transition was demonstrated, and $N_{i,j}$ was the number of times $\tau_{i,j}$ was demonstrated. This converts the number of demonstrations into a cost where the most demonstrated transition has a cost of one and other transitions a cost greater than one.

The cost $Cr_{i,j}$ was defined as:

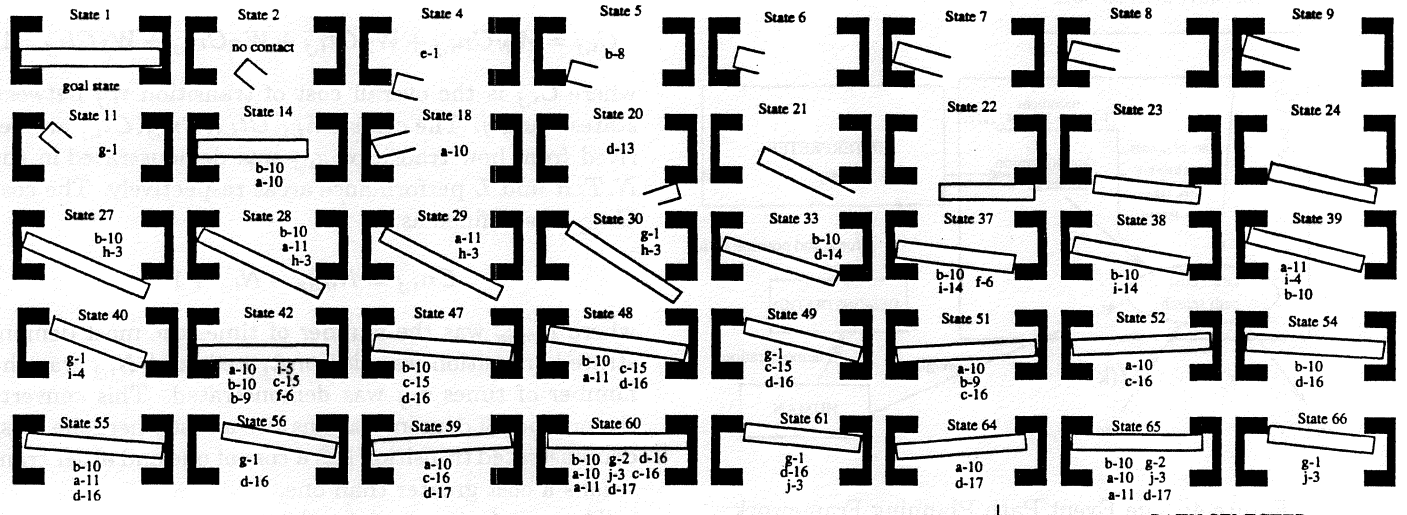
$$Cr_{i,j} = A_i S_i + 1$$

where A_i is the number of degrees of freedom in state i and S_i is the number of transitions to other states from state i. Reliability implies successful execution of an attempted transition. Unsuccessful execution mainly results from an error in correctly assessing spindle location and orientation in the state prior to the transition. On this basis, the degrees of freedom of the prior state are used as the basis of reliability of the transition. The greater the number of degrees of freedom, the less reliable the transition. The degrees of freedom of the prior state are taken as the number of degrees of freedom in the task minus the number of constraints in that state. This is multiplied by the number of possible next states because disorientation in a state could cause any of the transitions out of that state to occur. The addition of one is included in this term so that states with zero degrees of freedom do not have a cost of zero, but rather a cost of one.

The time taken for a transition and the length of a transition convert naturally to costs, since the larger the time or length of a transition the larger should be the cost assigned to it. Hence, costs $Ct_{i,j}$ and $Cl_{i,j}$ were defined simply as:

$$Ct_{i,j} = Tav_{i,j} \quad and \quad Cl_{i,j} = L_{i,j}$$

where Tav is the average time taken for transition $\tau_{i,j}$ in the demonstration, and $L_{i,j}$ is the length of the transition, currently assigned a value of one. The length parameter has the effect of promoting an event path with few states. This will be the case since any prospective path will have a component of its cost equal to the number of states in the path. The greater the number of states in the path, the greater the likely cost of the path.



DEMONSTRATED PATHS

Path 1: 2-4-5-6-7-8-27-33-8-38-14-18--64-59-52-51-1

Path 2: 2-20-22-21-23-24-21-27-8-54--47-1

Path 3: 2-5-6-7-8-27-8-38-8-9-65-60--55-54-47-1

Path 4: 2-21-24-21-30-11-40-11-66-61--56-49-48-47-1

Path 5: 2-21-29-28-9-39-9-65-60-55-54-47-1

Weight Stressed	PATH SELECTED	
	Wu Large (no exploration)	Wu Small (exploration)
all equal	2-5-6-7-8-54-47-1	2-8-54-47-1
Wn stressed	2-21-27-8-54-47-1	n/a
Wl stressed	2-21-27-8-54-47-1	2-8-54-47-1
Wr stressed	2-21-27-8-54-47-1	2-6-7-37-42-1
Wt stressed	2-5-6-7-8-54-47-1	2-8-54-47-1

Figure 5: States in the task model, demonstrated paths, and executed paths as weights are varied

The weights W_N, W_T, W_R, W_L in (1) are used to allow the human to specify what aspect of performance is important. That is, the user gives the robot a disposition. For example, if a short execution time is important, W_T can be increased relative to the other weights and a path with a short execution time will be selected. The robot can also be made to focus on reliability, on a path with few transitions, or on what the demonstrator did most often.

In general the task model automaton will also contain transitions that were not demonstrated. The cost $Cundem_{i,j}$ associated with an undemonstrated transition in the automaton was calculated as:

$$Cundem_{i,j} = W_U * [W_N Cn_{i,j} + W_T Ctdemav_{i,j} + W_R Cr_{i,j} + W_L Cl_{i,j}]$$

where $Cn_{i,j}, Cr_{i,j}, Cl_{i,j}, W_N, W_T, W_R$, and W_L are calculated in the same way as for demonstrated transitions, and $Ctdemav_{i,j}$ is the average time taken for all the demonstrated transitions. W_U is a weight applied to undemonstrated transitions. Executing an undemonstrated transition essentially amounts to exploration. A human who wants the robot to perform only demonstrated transitions can set W_U to be large. If exploration for possible better paths than those demonstrated is desired W_U can be set to a low value.

4 Results

A total of five demonstrations were provided by the human. The state paths taken during each are shown in Figure 5. The demonstrated paths were then fed into the EPP which produced different paths depended on what disposition was selected (ie. which weight was stressed). Two batches of tests were done, one where exploration was promoted (Wu set to be small) and one where exploration was discouraged (Wu set to be large). In each batch the weights Wl, Wt and Wr were individually stressed to see what paths were produced. Only in the batch where exploration was discouraged was Wn also individually stressed, since stressing the use of demonstrated transitions when exploration is being promoted didn't make sense. The paths selected by the EPP as the different weights were stressed are shown in figure 5.

When exploration is discouraged and all weights are made the same the EPP selected path 2-5-6-7-8-54-47-1

, which as you would expect is a middle of the road path. This path wasn't demonstrated but was constructed by the EPP from good components of other paths. Its attractiveness lies mainly in its short length and low time for execution, although many of the path's components were also frequently demonstrated. For example the sequence 54-47-1 was a favoured final approach to the goal state.

When Wr, Wl or Wn is stressed the EPP selects path 2-21-27-8-54-47-1. This is a good selection with regard to path length because it is shorter than 2-5-6-7-8-54-47-1. Also, with regard to reliability it contains more states where the spindle is in contact with both supports, resulting in less degrees of freedom than with the single sided contact path segment of 2-5-6-7-8 in 2-5-6-7-8-54-47-1. It is also a good selection from a number of demonstrations point of view. In addition to the well used 54-47-1 path segment, it contains the transition 2-21 which was also a favourite of the demonstrator when making initial contact between the spindle and supports. This was because there is a large range of valid movement of the spindle over which this transition can still be successfully made, resulting in it being a safe transition for leaving the no contact state. When Wt is stressed, the EPP returns to selecting 2-5-6-7-8-54-47-1. This is due to the speed with which the human was able to demonstrate the 5-6-7-8 and 54-47-1 state sequences.

When exploration is promoted by making Wu small (with all other weights equal to promote an even disposition) the EPP selects path 2-8-54-47-1. The 2-8 transition of this path was not demonstrated. It was selected by the EPP as a good transition to explore in order to shorten the 'even disposition' path 2-5-6-7-8-54-47-1 selected when exploration was discouraged above. This path was selected mainly on its short length and quick execution time attributes. This is shown by the fact that when Wn or Wt are stressed this path is also selected. When Wr is stressed the path selected is 2-6-7-37-42-1. Much of this path (2-7, 7-37, 37-42, 42-1) was undemonstrated, however it provides a very sure path to goal due to the very constrained spindle motion which results from it, especially within the sequence 7-37-41-1.

The process of building the EPP mapping ξ was successful. The EPP selected sensible paths according to what characteristic was stressed. It was also able to suggest undemonstrated transitions that could be explored which would improve the final event path. The approach is flexible because it allows the disposition of the robot to be varied according to what is required by the task. If a human demonstrator is very experienced at a task, then N can be stressed to allow the robot to choose his path. If the task to be learned has large penalties associated with failure during execution, then the robot can be given a risk adverse disposition. Conversely if task execution failure leads only to minor penalties, then the robot can be made to favour a fast execution. If the path should only contain demonstrated transitions then Wu can be set high. If some exploration can be tolerated then Wu can be set low.

5 Summary

We have proposed a method for the PbD of assembly tasks, where the task is modelled as a hybrid dynamic system. Using such a model allows the dual nature of assembly tasks to be captured. The dual nature includes the underlying continuous dynamics and the higher level discrete contact formation representation. We have shown the discrete level representation provides a good environment for describing a path plan, and have concentrated on constructing an event path for robot execution from the demonstration. The method selected good path solutions, and was able to change its selection according to what characteristics were desired in the selected path.

References

[1] Christopher G. Atkeson and Stefan Schaal. Learning tasks from a single demonstration. In *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*, pages 1706–1712, April 1997.

[2] Nathan Delson and Harry West. Robot programming by human demonstration: Adaptation and inconsistency in constrained motion. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, 1996.

[3] J.Chen and B.J.McCarragher. Robot programming by demonstration, selecting optimal event paths. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, pages 0–0, April 1998.

[4] K.Ikeuchi, M.Kawade, and T.Suehiro. Toward assembly plan from observation, task recognition with planar, curved and mechanical contacts. In *Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2294–2301, 1993.

[5] Sheng Liu and Haruhiko Asada. Transfer of human skills to robots: Learning from human demonstrations for building an adaptive control system. In *Proceedings of the 1992 American Control Conference*, pages 2607–2612, 1992.

[6] Brenan J. McCarragher and Haruhiko Asada. Process monitoring for robotic assembly using dynamic process models. *Journal of Dynamic Systems, Measurements and Control*, 115(2):261–275, June 1993.

[7] Brenan J. McCarragher and Haruhiko Asada. The discrete event modelling and trajectory planning of robotic assembly tasks. *Journal of Dynamic Systems, Measurements and Control*, 117(3):394–400, October 1995.

[8] Marjorie Skubic and Richard A. Volt. Learning force based assembly skills from human demonstration for execution in unstructured enviroments. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, pages 1281–1288, May 1998.