

## ROBUST DISCRETE EVENT CONTROLLER SYNTHESIS FOR CONSTRAINED MOTION SYSTEMS

David J. Austin, Brenan J. McCarragher

*Department of Engineering  
Faculty of Engineering and Information Technology  
The Australian National University  
Canberra, Australia  
email: {david,brenan}@faceng.anu.edu.au  
Fax: int + 61 6 249 0506*

**Abstract:** A new, robust discrete event controller synthesis methodology for the successful convergence of assembly tasks is presented. The modelling of an assembly process as a hybrid dynamic system has been shown to be a very effective strategy to incorporate both the continuous and discrete natures of the interaction between the workpiece and its environment. Prior works have presented controller synthesis methodologies for velocity- and force-controlled systems and here we present synthesis techniques to develop controllers which are robust to modelling and measurement errors. A number of examples are given which demonstrate the effectiveness of this method.

**Keywords:** Robust control, discrete event control, constrained motion, assembly.

### 1. INTRODUCTION

This paper will consider the control of constrained motion systems. Examples of such systems are numerous, particularly in the field of robotics. For example, walking machines, robot manipulators performing assembly or grinding tasks, and robotic hands grasping objects are all constrained motion systems. A great deal of research has been conducted in the field of constrained motion systems. Efforts have been focussed on control systems for the control of contact forces and transition behaviour. In particular, Hogan ? proposed an impedance control scheme and many alternate hybrid position/force control schemes have since been developed. Despite these efforts, the application of constrained motion systems has not had the burgeoning success of comparable technologies. This lack of success is due to a continuing focus on the low-level details. We would

argue that work for constrained motion systems should proceed at a more abstract level. Hybrid dynamic systems provide a good framework for modelling and analysis of abstract concepts linked to continuous systems.

A hybrid dynamic system ? consists of a discrete event system interacting with a continuous time system. Usually, the discrete event system is a decision-maker or controller and operates at an abstract level. A simple example is a furnace system in a typical home. The thermostat is an abstract, discrete event system with two states "too cold" and "warm enough", whereas the house and furnace are continuous time systems. The control algorithm is to turn the furnace on when in the "too cold" state and turn the furnace off otherwise. Even this simple example illustrates the important feature of hybrid dynamic models, which is to combine abstract concepts with continuous systems.

There have been many wide-ranging applications of hybrid dynamic systems, most notably in manufacturing systems and network protocols. To date, they have been used in detailed assembly tasks ?, for stable contact transitions ?, and telephone network protocols ?. Most of the research work in the area of hybrid systems is concerned with developing and proving control-theoretic ideas for specific classes of systems. Ostroff and Wonham ? provide a powerful and general framework (TTM/RTTL) for modelling and analysing real-time discrete event systems. Brandin and Wonham have developed supervisory controllers for timed discrete event systems ?. These papers have tried to capture a wide range of system attributes in simple models to allow for tractable analysis and control optimisation. McCarragher ? proposed a discrete event controller synthesis methodology for velocity-controlled systems. Since then, the authors have developed a controller synthesis methodology for force-controlled systems ?. This paper proposes a robust discrete event controller synthesis methodology which results in improved performance in the presence of measurement and modelling errors.

For many types of constrained motion tasks, a major cause of failure is errors in control commands resulting from measurement and modelling uncertainty. Many control schemes do not even address the issue of measurement uncertainty. However, ? consider the problem of uncertain constraints but not within a discrete event framework. This paper extends prior works in the area of discrete event controller synthesis by developing synthesis methodologies which make allowances for measurement and modelling uncertainties. The synthesised controllers will combine the excellent error recovery characteristics of discrete event control with the ability of robust command generation to avoid most errors.

## 2. HYBRID DYNAMIC MODELLING OF CONSTRAINED MOTION SYSTEMS

We will consider a constrained motion system which involves the motion of a rigid, polyhedral *workpiece* with possible constraints introduced by contact between the workpiece and a fixed, rigid and polyhedral *environment*. Systems of this type are typical of assembly processes. In a previous work, McCarragher ? demonstrates that all possible states of contact between two polyhedral parts in Cartesian space can be described as combinations of edge-edge and surface-vertex contacts. Hybrid dynamic modelling is particularly appropriate for assembly processes as there are a small number of possible contact configurations and, hence, we can dramatically reduce the complexity of the continuous time process by abstracting to a higher level.

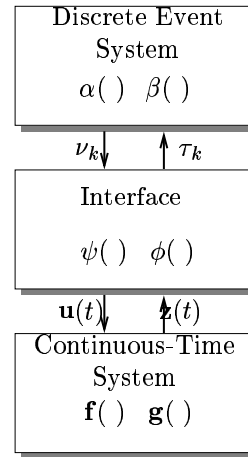


Fig. 1. *Block Diagram Representation of Hybrid Dynamic System*

We will consider a specific type of hybrid dynamic system, consisting of a discrete event system (in this case a discrete event controller) interfaced to a constrained motion system involving two polyhedral parts (as discussed above). The structure of the adopted hybrid dynamic model is as shown in Fig. 1. The system consists of three parts, which are the continuous time plant, the discrete event controller, and the interface. Each component is detailed below, with the description tailored to the restricted constrained motion systems that we consider. Mathematically, the continuous plant will be defined by a set of differential equations describing the motion of the workpiece. The discrete event controller is modelled as an automaton describing task-level decision making, whereas the interface serves as the communication between the continuous manipulation process and the decision maker.

### 2.1 The Continuous-Time System

Consider the general motion control of an object or workpiece. The equation of motion of the workpiece in free-space is given generally as

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)) \quad (1)$$

where  $\mathbf{x}(t)$  is the continuous time state vector, and  $\mathbf{u}(t)$  is the input vector. We will consider systems where  $\mathbf{u}(t)$  is a velocity command which is executed by a robot manipulator.

As the workpiece interacts with the inertially fixed environment, the free-space dynamics of (1) become constrained. For each edge-edge or surface-vertex contact, this constraint may be written as

$$g_j(\mathbf{x}(t)) = 0 \quad (2)$$

where  $g_j$  is the constraint equation for the  $j^{th}$  edge-edge or surface-vertex contact. Distance

functions are ideal candidates for  $g_j$  (e.g. the shortest distance between a surface and vertex). Note, equation (2) is only valid when the  $j^{\text{th}}$  edge-edge or surface-vertex pair is actually in contact.

## 2.2 The Discrete Event System

The continuous time plant is controlled by a task-level, discrete event controller modelled as an automaton. The automaton is a quintuple,  $(S, E, C, \alpha, \beta)$ , where  $S$  is the finite set of states,  $E$  is the set of plant events,  $C$  is the set of controller events,  $\alpha : S \times E \rightarrow S$  is the state transition function, and  $\beta : S \rightarrow C$  is the output function. Each *state* in  $S$ , denoted  $\gamma_i$ , identifies a distinct state of contact between the workpiece and the environment. For this paper, the states will be taken as the possible combinations of edge-edge and surface-vertex contact as discussed above. Each *controller event* in  $C$ , denoted  $v_k$ , is generated by the discrete event controller, whereas each *plant event* in  $E$ , denoted  $\tau_k$ , is generated by the conditions in the continuous plant. Here, plant events are generated when the continuous-time system changes contact state.

The dynamics of the discrete event controller are given generally by

$$\gamma_{k+1} = \alpha(\gamma_k, \tau_k) \quad (3)$$

$$v_k = \beta(\gamma_k) \quad (4)$$

where  $\gamma_k \in S, \tau_k \in E$  and  $v_k \in C$ . The input and output signals of the discrete event controller are asynchronous sequences of events, rather than continuous time signals. The function  $\alpha$  is functionally dependent on  $f$  and  $g$  as defined by the continuous-time system. Note that the state of the continuous-time system is  $\mathbf{x}$ , whereas  $\gamma$  is the state variable of the discrete event system and is dependent on  $\mathbf{x}$ .

## 2.3 The Interface

The interface is responsible for the communication between the plant and the controller, since these components cannot communicate directly due to the different types of signals being used. The interface consists of two maps  $\phi$  and  $\psi$ . The first map  $\phi$  converts each controller event into a plant input as follows

$$\mathbf{u}(t) = \phi(v_k) \quad t_k \leq t < t_{k+1} \quad (5)$$

where  $v_k$  is the most recent controller event before time  $t$  and  $t_k$  is the time of the  $k^{\text{th}}$  discrete event. For synthesis purposes it is often easier to combine the control equations (4) and (5) such that the

control input is directly a function of the discrete state

$$\mathbf{u}(t) = \phi(\beta(\gamma_k)) \quad t_k \leq t < t_{k+1} \quad (6)$$

The second map  $\psi$  converts the state space of the plant into the set of plant events.

$$\tau_k = \psi(\mathbf{x}(t)) \quad (7)$$

Note that equation (7) does not imply that  $\tau_k$  changes continuously as  $\mathbf{x}(t)$  changes. The state space of the plant is partitioned into contiguous regions. The function  $\psi$  generates a new plant event only when the state first enters one of these regions. The map  $\psi$  is called a *process monitor*.

## 3. CONTROL COMMAND SYNTHESIS

The *control commands* are the input to the continuous-time plant from the discrete event controller. For our purposes, the control commands depend solely upon the state of the discrete event controller. We will impose the minor restriction that commands can cause only one contact state change at a time (i.e. only one contact can be gained or lost per command). This restriction simplifies the following analysis and results in a more robust controller. The control commands are determined by first establishing a *desired event* for each state. The desired event is chosen such that the system moves towards the target state. The desired events may be determined manually or automatically, depending upon the application. For any given state, we use the desired event and geometric considerations of the workpiece and environment to establish conditions on the command to be executed.

There are three conditions upon which the control law (6) for motion control is selected. First, the *maintaining condition* ensures that the currently active constraints (2) remain satisfied, if desired. Second, the *enabling condition* is a necessary condition that ensures that the next desired discrete event  $\tau_{k+1}$  is allowed to occur. Third, the *disabling condition* is a sufficient condition that ensures an undesired discrete event is not allowed to occur.

### 3.1 Maintaining Condition

The motion of the system described by (1) is constrained by (2). The first possible task of the controller is to ensure that the control commands satisfy this geometric constraint. To derive admissible velocities that satisfy the geometric constraint, we can differentiate (2) to give

$$\frac{\partial}{\partial \mathbf{x}} [g_j(\mathbf{x})] \frac{d}{dt} \mathbf{x}(t) = \mathbf{0} \quad t_k \leq t < t_{k+1} \quad (8)$$

where  $g_j$  is the constraint function for this contact. Noting that  $\frac{d}{dt}\mathbf{x}(t) = \mathbf{u}(t)$ , this can be rewritten as

$$\mathbf{a}_j^T \mathbf{u}(t) = 0 \quad t_k \leq t < t_{k+1} \quad (9)$$

where  $\mathbf{a}_j = \frac{\partial}{\partial \mathbf{x}} g_j(\mathbf{x})$  is a column vector with length equal to the number of degrees of freedom. Equation (9) is our maintaining condition in that it must be satisfied to maintain the contact or geometric constraint. When  $g_j$  is a distance measure, equation (9) becomes a requirement that the distance between the points of contact remains zero (i.e. the points remain in contact).

### 3.2 Enabling Condition

In addition to determining motion that maintains a constraint, it is desired to determine the motion such that the workpiece encounters the next discrete state  $\gamma_{k+1}$ . Since the system is not in  $\gamma_{k+1}$ , the following must be true

$$g_j(\mathbf{x}(t)) = K \quad t_k \leq t < t_{k+1} \quad (10)$$

where, without loss of generality,  $K$  is a positive constant. In order to direct the system such that  $K \rightarrow 0$ , we require the time derivative to be negative.

$$\mathbf{a}_j^T \mathbf{u}(t) < 0 \quad t_k \leq t < t_{k+1} \quad (11)$$

Equation (11) is our enabling condition. It is a necessary condition for discrete event  $\tau_{k+1}$  to occur. When  $g_j$  is a distance measure, equation (11) becomes a requirement that the distance decreases or that the points of contact move closer together.

### 3.3 Disabling Condition

The third condition, the disabling condition, is derived directly from the enabling condition. Since (11) is a necessary condition for a discrete event to occur, a sufficient condition for a discrete event not to occur is obtained by changing the direction of the inequality.

$$\mathbf{a}_j^T \mathbf{u}(t) \geq 0 \quad t_k \leq t < t_{k+1} \quad (12)$$

where  $j$  indicates the discrete states (constraint equations) that are not desired to occur. Essentially, this disabling condition prevents  $K$  from decreasing in magnitude. When  $g_j$  is a distance measure, equation (12) becomes a requirement that the distance between the points of contact does not decrease (i.e. the points stay apart).

### 3.4 Solving for the Control Command

The desired event determines which of the above conditions should be applied for each possible edge-edge or surface-vertex contact. The maintaining condition is used when it is desired to maintain a contact. Note, when it is desired to immediately violate the current constraint by breaking the contact, the maintaining condition is not used. The enabling condition is used to enable the loss or gain of a contact. The disabling condition is used to prevent unwanted gains of contact. From the desired event, we now find a set of conditions on the control command, one for each possible edge-edge or surface-vertex contact. The control command is determined by satisfying this set of conditions. Any method for satisfying the set of constraints will yield an acceptable input command. It is suggested to use a search technique to maximise the minimum distance to each constraint.

## 4. ROBUST CONTROLLER SYNTHESIS

In Section 3 we have developed constraints upon the input command. These may be represented as

$$\Phi \cdot \mathbf{u} \begin{cases} < \\ = \\ \geq \end{cases} 0 \quad (13)$$

where  $\Phi$  is a  $1 \times n$  row vector which is a function of the geometry of the system and  $n$  is the dimension of the system. The relational operator depends upon the condition, as discussed above.

Due to measurement or modelling uncertainty, the exact value of  $\Phi$  is usually unknown. Instead we assume that upper and lower bounds have been established on each of the components of  $\Phi$  so that

$$\Phi_{\mathbf{L}}[i] < \Phi[i] < \Phi_{\mathbf{U}}[i] \quad 1 \leq i \leq n \quad (14)$$

where  $\Phi_{\mathbf{L}}$  is the vector of lower bounds and  $\Phi_{\mathbf{U}}$  is the vector of upper bounds. These bounds may be established by finding bounds on each of the uncertain parameters to  $\Phi$ . If  $\Phi$  is complicated then numerical techniques must be used to establish the bounds. For simple functions, such as sinusoids, the bounds can be established analytically.

Clearly we cannot obtain a solution to the equality case of (13) when  $\Phi$  is uncertain. Instead we will consider the remaining cases

$$\Phi \cdot \mathbf{u} \begin{cases} < \\ \geq \end{cases} 0 \quad (15)$$

### 4.1 Solving for the Control Command

As in Section 3.4, we wish to solve the set of constraints for a suitable control command. In

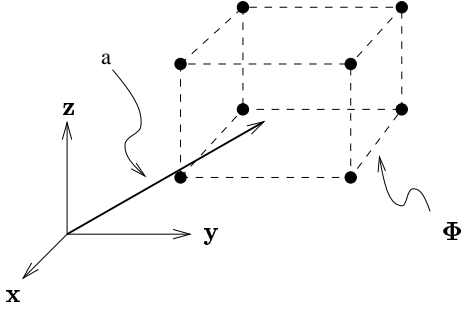


Fig. 2.  $\Phi$  and Bounding Box: As the constraints are linear we only need to consider the extremal points (solid dots) of the bounding box

this case, however, the constraints are uncertain. Hence, we wish to obtain a solution which is valid for all possible constraints within the specified ranges. This solution is a *robust control command*. A “brute-force” solution technique is presented in the next section, followed by a faster solution technique and a discussion of robust solution existence.

**4.1.1. Brute-Force Solution** As equation (15) is linear, only the extremal points of the box bounding  $\Phi$  need be considered, as shown in Fig. 2. Formally, equation (15) is satisfied if

$$\Phi_X \cdot \mathbf{u} \begin{cases} < \\ > \\ \geq \end{cases} 0 \quad \forall \Phi_X \quad (16)$$

where  $\Phi_X$  is an extremal point of the bounding box:

$$\Phi_X[i] \in \{\Phi_L[i], \Phi_U[i]\} \quad 1 \leq i \leq 6 \quad (17)$$

To determine a robust solution we solve the set of constraints with each uncertain constraint replaced by a number of constraints evaluated at the extremal points of the bounding box. Unfortunately, this technique leads to a proliferation of constraints with each uncertain constraint contributing many new constraints. For example, if a constraint is uncertain in each of its six components, then there are  $2^6 = 64$  extremal points on the bounding box and so 64 new constraints must be considered.

Despite the additional constraints it is still computationally feasible to solve the complete set. However, this is an inelegant or “brute force” solution. The next section illustrates methods for dramatically pruning the set of constraints that need to be considered.

**4.1.2. Fast Solution Technique** First we define  $\Phi_M$  as the mid-point of the bounding box:

$$\Phi_M[i] = \frac{\Phi_L[i] + \Phi_U[i]}{2} \quad 1 \leq i \leq n \quad (18)$$

and we determine a trial solution  $\mathbf{u}_T$  using  $\Phi_M$  for the uncertain constraint.

Under the assumptions that:

- a robust solution exists, and
- the trial solution is close to the final solution ( $\mathbf{u}_T \sim \mathbf{u}$ )

we can determine which of the extremal points is the most “critical” and, using only this point, we can solve for the input vector,  $\mathbf{u}$ . The point which is most critical depends upon the type of constraint being solved. If the constraint is a greater-than constraint then the critical point is the extremal point which minimises the quantity  $\Phi_X \cdot \mathbf{u}_T$ . If the constraint is a less-than constraint then the critical point is the extremal point which maximises the quantity  $\Phi_X \cdot \mathbf{u}_T$ . Intuitively, the critical point is the constraint value which the trial solution is closest to violating.

We require that the trial solution is close to the final solution ( $\mathbf{u}_T \sim \mathbf{u}$ ) so that the selected critical point for the trial solution is the critical point for the final solution.

Thus, we may obtain a fast solution by the following steps:

- Find the midpoint of the bounding box,  $\Phi_M$ ,
- Determine the trial solution,  $\mathbf{u}_T$  using  $\Phi_M$ ,
- Choose the extremal point of the box which is closest to violating the constraint equation (15) for the trial solution, and
- solve for  $\mathbf{u}$  using the chosen extremal point(s).

## 4.2 Solution Examples

In this section, we present two examples illustrating the application of the fast and brute-force solution techniques.

**4.2.1. Fast Solution Example** Consider a simple two-dimensional example which has the single constraint equation

$$\mathbf{A} \cdot \mathbf{u} > 0 \quad (19)$$

where  $\mathbf{A}$  is uncertain and is bounded as

$$\mathbf{A} = \begin{bmatrix} 0.3 \pm 0.12 \\ 0.3 \pm 0.1 \end{bmatrix} \quad (20)$$

This example is illustrated in Fig. 3.

We now solve the uncertain constraint using the fast solution technique. First, we find  $\mathbf{A}_M = [0.3 \ 0.3]^T$ . Hence,

$$\mathbf{u}_T = \begin{bmatrix} 0.707 \\ 0.707 \end{bmatrix} \quad (21)$$

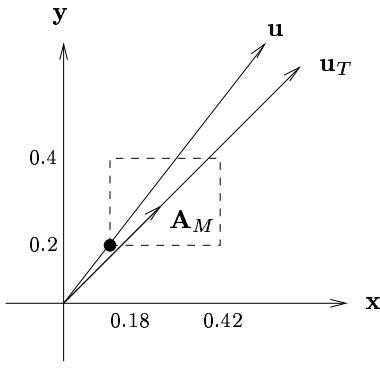


Fig. 3. *Example of Fast Solution: Solution process for  $\mathbf{A} \cdot \mathbf{u} > 0$  with  $\mathbf{A}$  uncertain and bounded by the dotted rectangle.  $\mathbf{A}_M$  is the mid-point of the bounding box,  $\mathbf{u}_T$  is the trial solution and  $\mathbf{u}$  is the final solution*

Thus, the extremal point of interest is  $\mathbf{A}_X = [0.18 \ 0.2]^T$  (as shown in Fig. 3) and the solution is determined as

$$\mathbf{u} = \begin{bmatrix} 0.669 \\ 0.743 \end{bmatrix} \quad (22)$$

Clearly, this is a robust solution to the constraint equation (19). Note that (21) is also a robust solution but that (22) is a better solution because it maximises the quantity  $\mathbf{A}_X \cdot \mathbf{u}$  for the critical point.

**4.2.2. Brute-Force Solution Example** The existence of a robust solution to the set of constraints is not guaranteed and Fig. 4 demonstrates an example of this. Here we have two constraint equations

$$\mathbf{B} \cdot \mathbf{u} > 0 \quad (23)$$

$$\mathbf{C} \cdot \mathbf{u} > 0 \quad (24)$$

where  $\mathbf{B}$  and  $\mathbf{C}$  are uncertain and are bounded as

$$\mathbf{B} = \begin{bmatrix} 0.0 \pm 0.8 \\ 0.7 \pm 0.1 \end{bmatrix} \quad (25)$$

$$\mathbf{C} = \begin{bmatrix} 0.7 \pm 0.1 \\ 0.0 \pm 0.8 \end{bmatrix} \quad (26)$$

As there is no robust solution, we cannot apply the fast solution technique and instead we use the brute-force solution method. Using a gradient search technique to solve the set of 8 constraints results in

$$\mathbf{u} = \begin{bmatrix} 0.707 \\ 0.707 \end{bmatrix} \quad (27)$$

This solution technique minimises the areas (shown shaded in Fig. 4) of each bounding box for which the solution is invalid.

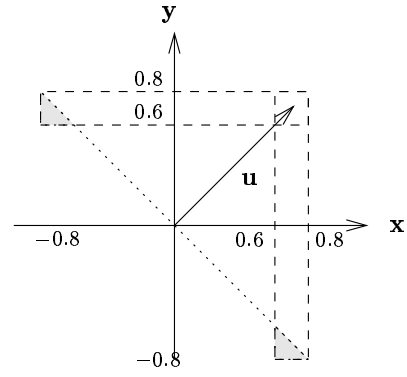


Fig. 4. *Example of Brute-Force Solution: No robust solution exists,  $\mathbf{u}$  is the brute-force solution and minimises the areas (shaded) of each bounding box for which the solution is invalid.*

### 4.3 Summary

In this section we have presented a method for determining robust control commands. Unfortunately this method increases the complexity of the solution by a factor of  $2^n$  where  $n$  is the number of uncertain variables. Two solution techniques are presented: one which solves all of the  $2^n$  new constraints and a faster method which requires much less computation. The fast solution technique requires that a robust solution exist. If this cannot be guaranteed then the brute force technique must be used. The examples of each solution technique demonstrated the application of these methods.

## 5. EXAMPLE OF ROBUST COMMAND SYNTHESIS

We now give a complete example where the uncertainties of the constraint vectors are derived from the original measurement uncertainties and the expressions for the constraint vectors. Consider the two-dimensional assembly task shown in Fig. 5 with parameters as presented in Table 1. Here we have a peg being inserted into a hole by a robot. It is desired to prevent gain of contact at 1 and to gain contact at point 2.

We can determine the constraint function for contact 1 as follows:

$$\mathbf{a}_1 = \frac{\partial}{\partial \mathbf{x}} g_i(\mathbf{x}) \quad (28)$$

using the shortest distance between the corner and the surface as the distance function  $g_i$  gives

$$\mathbf{a}_1 = [1 \ 0 \ -r_1 \sin\alpha + r_2 \cos\alpha]^T \quad (29)$$

To prevent contact at 1 we require

$$\mathbf{a}_1 \mathbf{u} \geq 0 \quad (30)$$

and with the parameters from Table 1 we have

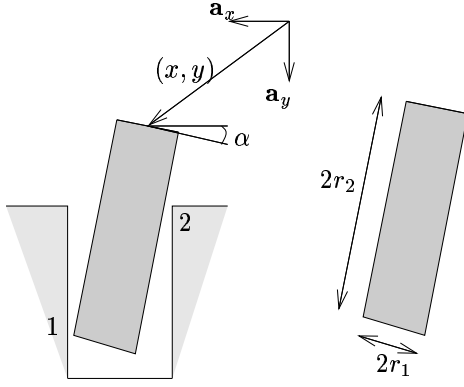


Fig. 5. Example: Peg-in-Hole Assembly Task

Parameter	Value
$x$	$2.0 \pm 0.5\text{cm}$
$y$	$2.0 \pm 0.5\text{cm}$
$\alpha$	$10 \pm 5^\circ$
$r_1$	$1.0\text{cm}$
$r_2$	$3.0\text{cm}$

Table 1. System Parameters for Example Calculations

$$\mathbf{a}_1 = \begin{bmatrix} 1.0 \pm 0.0 \\ 0.0 \pm 0.0 \\ 0.028 \pm 0.001 \end{bmatrix} \quad (31)$$

We can determine the constraint function for contact 2 as:

$$\mathbf{a}_2 = [-\cos\alpha \ \sin\alpha \ x\sin\alpha + y\cos\alpha]^T \quad (32)$$

and with the parameters from Table 1 we have

$$\mathbf{a}_2 = \begin{bmatrix} 0.981 \pm 0.015 \\ 0.173 \pm 0.086 \\ 0.024 \pm 0.008 \end{bmatrix} \quad (33)$$

To gain contact at 2 we require

$$\mathbf{a}_2 \mathbf{u} < 0 \quad (34)$$

Using the fast solution technique, we determine the trial solution as  $\mathbf{u}_T = [-0.01 \ -0.92 \ 0.38]^T$ . Hence, the critical extremal point for  $\mathbf{a}_1$  is the point which minimises the quantity  $\mathbf{a}_1 \mathbf{u}_T$  which is

$$\mathbf{a}_{1X} = [1.0 \ 0.0 \ 0.027]^T \quad (35)$$

Similarly, the critical extremal point for  $\mathbf{a}_2$  is the point which maximises the quantity  $\mathbf{a}_2 \mathbf{u}_T$  which is

$$\mathbf{a}_{2X} = [0.966 \ 0.087 \ 0.032]^T \quad (36)$$

and solving using the critical extremal points gives

$$\mathbf{u} = [-0.01 \ -0.97 \ 0.26]^T \quad (37)$$

Clearly, this command satisfies (30) and (34) and hence will achieve the desired gain of contact at 2 whilst preventing gain of contact at 1.

## 6. CONCLUSIONS

It is highly desirable to develop control systems which are robust to measurement and modelling uncertainties. In this paper, we presented a framework for robust discrete event controller synthesis. Both a brute-force and a fast solution technique were developed for control command synthesis. Two illustrative examples were presented which demonstrate the features of the two solution techniques. A complete example of a simple assembly task was also given. This example demonstrated the generation of the bounding boxes for the constraint vector and the successful solution of the uncertain constraint equations. These examples have demonstrated a highly successful robust discrete event controller synthesis technique.

## REFERENCES

- D. J. Austin and B. J. McCarragher. Force control command synthesis for assembly using a discrete event framework. In *IEEE Intl. Conf. on Robotics and Automation*, April 1997.
- B. A. Brandin and W. M. Wonham. Supervisory control of timed discrete event systems. In *IEEE Transactions on Automatic Control*, volume 139, February 1994.
- R. Brockett. Hybrid models for motion control systems. In *Essays on Control: Perspectives in the Theory and its Applications*, pages 29–53. Birkhauser Boston, 1993.
- Y. K. Ho, D. Wang, and Y. C. Soh. Robust control of a manipulator performing constrained motion with uncertainties in the constraint functions. *Journal of Robotic Systems*, 12, November 1995.
- N. Hogan. Impedance control: An approach to manipulation, parts i, ii and iii. *Journal of Dynamic Systems, Measurement and Control*, 107:1–24, March 1985.
- B. McCarragher. Task primitives for the discrete event modeling and control of 6-dof assembly tasks. *IEEE Transactions on Robotics and Automation*, 12(2):280–289, April 1996.
- B. J. McCarragher and H. Asada. The discrete event modelling and trajectory planning of robotic assembly tasks. *ASME Journal of Dynamic Systems, Measurement, and Control*, 117(3):394–400, September 1995.
- J.S. Ostroff and W.M. Wonham. A framework for real-time discrete event control. In *IEEE Transactions of Automatic Control*, volume 35, April 1990.
- A. Overkamp. Design and verification of a communication protocol for a telephone network. In *Belgian-French-Netherlands Summer School on Discrete Event Systems*, June 1993.
- T.-J. Tarn, Y. Wu, N. Xi, and A. Isidori. Force regulation and contact transition control. *IEEE Control Systems*, pages 32–39, February 1996.