Generating a Configuration Space Representation for Assembly Tasks from Demonstration

J. R. Chen
Department of Engineering, FEIT
The Australian National University
Canberra, ACT, 0200, Australia
E-mail: chen@faceng.anu.edu.au

A.Zelinsky
Department of Systems Engineering, RSISE
The Australian National University
Canberra, ACT, 0200, Australia
E-mail: Alex.Zelinsky@anu.edu.au

Abstract

Removing suboptimal actions that can exist in a demonstration is s key problem to be solved in Robot Programming by Demonstration. In this paper we present the first step of an approach for solving this problem. We present how the Configuration Space (C-space) of a task can be derived from demonstration. A demonstration traces out paths on a number of C-surfaces in C-space. The idea is to use statistical regression analysis on data from these paths to determine the unknown equation parameters of a C-surface. Experimental results show the validity of the approach. Accurate parameter estimates were obtained so long as a sufficiently rich set of demonstrated paths existed on the C-surface. The approach has the advantage that it tends to provide accurate parameter estimates for C-surfaces where they were most needed. That is, for C-surfaces (i) critical to task completion, and (ii) whose paths contained suboptimal actions.

1 Introduction

Recently there has been growing interest in the field of service robotics, where robots are utilized for tasks in a domestic environment. A major obstacle to be overcome before robots can enter such environments is of end user programming. A typical householder does not have the expertise to program a robot in the usual way, ie. by writing computer code. A new programming method is required that allows non-technical users to program robots. A promising solution is Programming by Demonstration (PbD). Here, the end user provides a demonstration of the task to be programmed. A PbD interface then interprets the demonstration and determines the low level control details required by the robot to achieve the task. This provides an easy and natural method for end users to program a robot.

Programming by demonstration is an active research area and many approaches have been presented [1, 3, 5, 10, 11, 13, 16]. A key result in the area is that it is generally suboptimal to have the robot directly copy the demonstration [3, 10, 13, 17]. For example, Delson and West [3] identify that, in a pick and place task through a field of obstacles, a human will naturally introduce noise into the demonstration by using different paths to traverse regions were the gap between obstacles is large. De Schutter et al [13] found that a demonstration of a peg-in-hole task could contain actions by the demonstrator that were suboptimal, erroneous, or even unintended. Similar findings were made by Kaiser and Dillman in [10]. Clearly, noise in the demonstration should be identified and removed by the PbD interface before demonstrated paths are programmed into the robot.

In this paper we present the first step of a new approach to removing noise from a demonstration. We show how a representation of Configuration Space (C-space [18]) can be constructed from the demonstration using standard statistical regression techniques. Such a first step is attractive since, once derived, C-space information can be used as the basis for noise removal. For example, we present in [9] a method for noise removal that uses the C-space representation derived in this paper.

Many approaches to constructing the C-space of a task exist [2, 4, 14, 18, 19]. Most are presented as a part of path planning approaches that use C-space eg. [4, 14, 19]. However, all the work we have seen in this area uses some sort of geometric description of the task, eg. a CAD model. Our approach to constructing C-space has the advantage over this work that a geometric model of the task is not required. This is important given our presentation is of PbD in a domestic environment, where such a geometric model is unlikely to be available. A limitation of our approach compared to [2, 4, 14, 18, 19] is that it only constructs regions of C-space that were visited in the demonstration. That is, it only provides a partial knowledge of C-space.

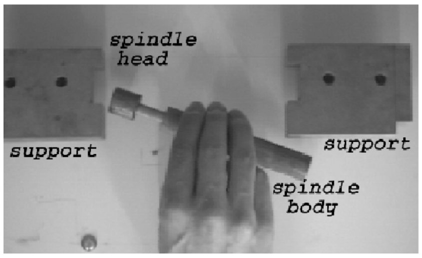


Figure 1: The spindle insertion task chosen for PbD

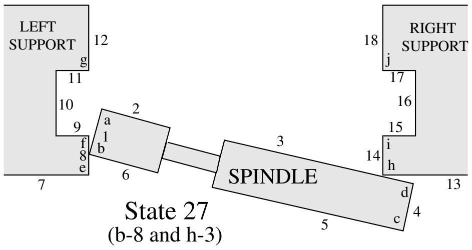


Figure 2: States in the HDS are defined on the basis of motion constraints on the spindle

2 Problem Formulation

Our aim is to present how C-space can be derived for a typical household task. The task chosen is shown in Figure 1. It is based on the domestic chore of changing rolls on a paper roll holder, and involves inserting an axially compressible spindle between two supports. The task involves four degrees of freedom, three to describe the position/orientation of the spindle body relative to the supports (y,z, and θ), and one to describe the compression of the spindle head relative to the spindle body (δ).

We model our task as a Hybrid Dynamic System (HDS). The spindle insertion task is in essence an assembly task, involving contact and constrained motion. Hybrid Dynamic Systems have been presented as a good way to model assembly tasks [12]. In its most general form, a HDS involves a continuous-time system interacting with a discrete-event system [15]. For assembly, the continuous-time system represents the continuous-time dynamics of the spindle. That is, as (i) a differential equation describing the free-space motion of the spindle relative to the supports, and (ii) a set of constraint equations describing the constraint on spindle motion when the spindle is in contact with the supports. In contrast, the discrete-event system captures the *discrete* nature of the assembly dynamics. It describes the assembly as a sequence of asynchronous *discrete events* occurring through time. A discrete event is defined to occur when the set of constraints on the spindle motion changes. Each distinct constraint set possible in the task is defined as a *discrete state*. Generally a discrete state will correspond to a unique contact formation between the spindle and supports. For example, we show in Figure 2 a spindle-support contact formation that defines one of the states in our task. The constraints existing in this state are caused by contacts h-3 and b-8 (note how a single point contact is coded as a letter-number pair the figure). In general, each single point contact defines one constraint in the state. As such, we reference a constraint by using the same letter-number pair of the contact that causes it, eg. h-3 and b-8 are the constraints that define the state in Figure 2. To make state referencing easier we give each state in the HDS a number. The state shown in Figure 2 has been labeled as state number 27 in the task. We show in Figure 3 six assembly sequences that were demonstrated for the task. Notice then how HDS modeling allows an assembly sequence to be nicely described as a sequence of discrete states. The state sequences start in the no-contact state (state 2), pass though a set of intermediate states, and end in the final fully assembled state (state 1). The six paths shown in the figure form the demonstration set we use to derive C-space in this paper. Note that the spindle position, orientation, and compression (ie, y,z,θ,δ) were recorded for these paths in the demonstration using two Polhemus sensors [7], one attached to the spindle head, and the other attached to the spindle body.

The use of HDS modeling is advantageous given our desire to construct C-space, because it provides a rigorous and well structured description of the topology of C-space. C-space consists of an obstacle free region (C_{free}), an obstacle defining region, and a region defining the boundary between the two ($C_{contact}$). Then the no-contact state in the HDS (state 2) corresponds to C_{free}. That is, any spindle configuration in state 2 will correspond to a point in C_{free}. All other states in the HDS involve contact between the spindle and supports, and so together correspond to $C_{contact}$. Individually, they each define a *C-surface*, a patch of curved surface (or hyper-surface) that defines part of $C_{contact}$. Our interest here is in deriving a representation of C-space. Let γ_i be an arbitrary state in the HDS that (a) defines contact between the spindle and supports (ie. we exclude state 2), and (b) was visited in the demonstration. Let c_i be the C-surface of γ_i. Then our aim is to derive a representation of C-space by determining an equation for each c_i in C-space.

Let C-space have dimension n. Then the problem of deriving an equation for c_i is a complex one, because c_i can be a surface ranging in dimension from zero to $n-1$. However, the problem is simplified if we note that any c_i can be specified as the intersection of C-

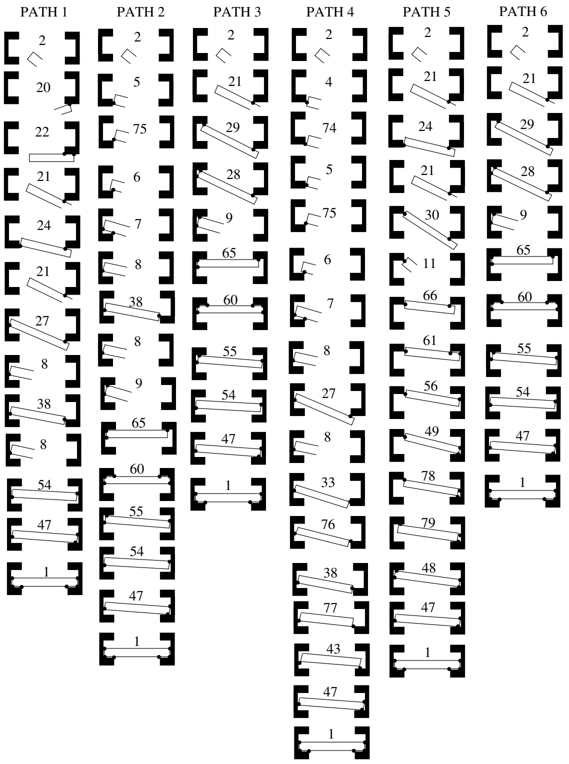


Figure 3: The set of demonstrations use to construct C-space

surfaces of dimension $n-1$. We denote such C-surfaces as *n-1 dimensional C-surfaces*. Recall that a state in the HDS is defined by a unique set of constraints, eg. the constraint set (b-8,h-3) defined state 27 in Figure 2. Let Ω_i be the constraint set defining γ_i. That is, $\Omega_i = (\rho_{i1}, \ldots, \rho_{ij}, \ldots, \rho_{n_j})$, where ρ_{ij} is the j^{th} constraint in state γ_i, and n_j is the number of constraints in γ_i. Each constraint ρ_{ij} results in the loss of one spindle dof, and hence determines an $n-1$ dimensional C-surface in C-space. Let c_{ij} be the $n-1$ dimensional C-surface determined by ρ_{ij}. Then c_i can be specified as:

$$c_i = \bigcap_{j=1}^{n_j} c_{ij}$$

That is, to determine c_i we need only know (a) the ρ_{ij} that exist in Ω_i, and (b) the equation of each c_{ij}. However, we note that many constraints are common to a number of states in the HDS. We denote as Ω^* the unique set of constraints existing in the demonstration set. Ω^* is calculated as:

$$\Omega^* = \bigcup_{i=1}^{n_k} \Omega_i = (\rho_1^*, \ldots, \rho_m^*, \ldots, \rho_{n_m}^*)$$

where n_k is the number of distinct states that were demonstrated, ρ_m^* is the m^{th} distinct constraint in the demonstration set, and n_m is the number of distinct constraints that exist in the demonstration set. Denote as c_m^* the n-1 dimensional C-surface corresponding to ρ_m^*. Then our problem of deriving an equation for any c_i can be recast as two sub-problems:

(a) determine the constraint set Ω_i that defines γ_i.

(b) determine the equation of every c_m^*.

We have a solution to both problems (a) and (b). However due to limited space we concentrate in this paper on presenting the solution for problem (b). We stated in the introduction that our method for constructing C-space does not require a geometric description of the task. One may think that determining the constraints existing in a state (ie. problem (a)) would require such a model. We note that our solution for (a) is based on work in [16], where a system is trained by demonstration to recognize the contact formation made between task objects. We use the contact formation information determined by this work to decide what constraints are present in a state, without the need for a geometric model of the task. For the remainder of this paper, we assume problem (a) has been solved. That is, we take it that all constraints ρ_{ij} that exist in a state γ_i are known.

3 Deriving the equation of a c_m^*

We base our approach for deriving the equation of each c_m^* on statistical regression analysis. Each demonstration in Figure 3 traced out a path though C-space. Some segment of one or more of those paths will exist on c_m^*. Our idea is to use regression analysis on the data from these segments to determine an equation describing c_m^*. For example, we show in Figure 4 a simple 3-D C-space defined by axes x, y, and z. Three simple, planar c_m^* are shown (labelled c_1^*, c_2^*, and c_3^*). The figure shows how a demonstration traces out paths on a number of c_1^*, c_2^*, and c_3^*. We have labelled the demonstrated segments existing on c_1^*, c_2^*, and c_3^* as 1, 2, and 3 respectively. Then our idea is to use data points in segment 1 to determine the equation of c_1^*, data points in segment 2 to determine the equation of c_2^*, etc. Two things are required for each c_m^* before the regression analysis can take place. They are:

- A regression model. The regression model is a generic equation for c_m^*. By *generic* we mean the form of the equation is known, but that its parameters are not.

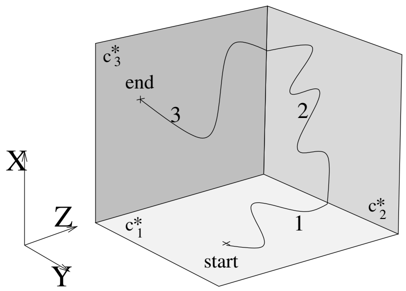


Figure 4: Simple example of how we use demonstrated paths to derive the equation of a c_m^*

- A data set. That is, the set of points recorded from the demonstration where the human was traversing on c_m^*.

3.1 The Regression Model and Data Set

Determining the data set for c_m^* is straightforward. It is formed by data recorded from any state in the demonstration where constraint ρ_m^* was present. Determining a regression model for c_m^* is more complex. We now present the details of how a regression model for c_m^* is determined.

For a planar assembly task involving two polyhedral objects, two possible constraint types ρ_m^* can exist in Ω^* [18]. If one object represents a workpiece (eg. spindle body) and the other the environment (supports), then the first constraint type is caused by a *vertex* of the manipulated workpiece in contact with an *edge* of the environment. We show in Figure 5(a) how the c_m^* of this type of ρ_m^* is given by the vector equation:

$$({}_\mathbf{a}\mathbf{A} + {}_\mathbf{b}\mathbf{C} - {}_\mathbf{a}\mathbf{B}) \cdot {}_\mathbf{a}\mathbf{n} = \mathbf{0} \tag{1}$$

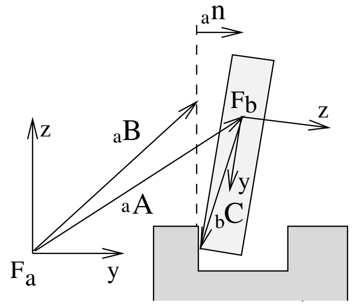
The second constraint type is formed by an *edge* of the workpiece in contact with a *vertex* of the environment. We show in Figure 5(b) how the c_m^* of this type of ρ_m^* is given by the vector equation:

$$({}_\mathbf{a}\mathbf{A} + {}_\mathbf{b}\mathbf{B} - {}_\mathbf{a}\mathbf{C}) \cdot {}_\mathbf{b}\mathbf{n} = \mathbf{0} \tag{2}$$

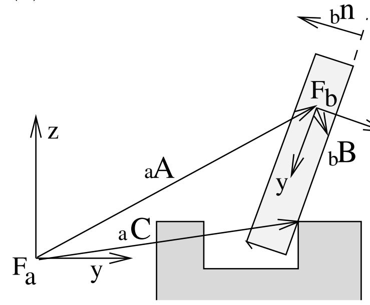
Equations (1) and (2) form the set of possible regression models for the c_m^* that exist in a planar task with a single manipulated object. They can both be expanded to give scalar equations of the form:

$$\phi_1(y, z, \theta; b, c, d, e, f) = 0 \tag{3}$$

where y,z, and θ are the position and orientation of the manipulated body, and b,c,d,e and f are the regression model's unknown parameters. Note that the parameters have physical meaning. For example, pair (c, d) gives the position of the vertex in the contact, relative to frame F_b in (1), and relative to frame F_a (2). This fact allows us to obtain the actual value of



(a)



(b)

Figure 5: Two of four possible constraint types for the spindle insertion task

parameters by measurement, something we use later to verify the accuracy of parameter values obtained from the regression analysis.

Our spindle insertion task is a planar task consisting of two manipulated objects (the spindle head and spindle body) with a single degree of freedom between them. Then a set of four regression models exists for the spindle insertion task. The first two correspond to edge-vertex and vertex-edge contacts between the spindle body and supports, and are given by equation (3). That is, we use equation (3) as the regression model for the c_m^* of ρ_m^* caused by spindle body edge-vertex and vertex-edge contacts with the supports. The second two regression models for the spindle insertion task correspond to edge-vertex and vertex-edge contacts between the spindle head and supports. They can be derived in a similar way to (3), however we do not present the details due to limited space. We only note that these models can be written as scalar equations of the form:

$$\phi_2(y, z, \theta, \delta; b, c, d, e, f) = 0 \tag{4}$$

where the additional variable δ in (4) compared to (3) describes the position of the spindle head relative to the spindle body. We use equation (4) as the regression model for the c_m^* of ρ_m^* caused by spindle head edge-vertex and vertex-edge contacts with the supports.

We note that our regression models (3) and (4) are non-linear in parameters b, c, d, e, f. However each model can be made linear in the set of transformed parameters $B_2 \ldots B_n$ by appropriate rearrangement of variables y,z,θ, and δ into a set of transformed variables $X_1 \ldots X_n$ [8] (recall that n is the dimension of C-space). That is, each regression model can be written as a linear model of general form:

$$[1, B_2, \ldots, B_n][X_1, \ldots, X_n]^T = 0 \quad (5)$$

We linearize each model in this way to simplify the regression problem to be solved in the next section.

3.2 Regression Analysis

With the model and data set determined for a c_m^*, the regression analysis can proceed. We first form a system of linear equations out of the model and data set, of the form:

$$\mathbf{X}[1, B_2, \ldots, B_n]^T = 0 \quad (6)$$

where $\mathbf{X}$ is the data set, whose i^{th} row we denote axss $X_{1_i}, \ldots, X_{n_i}$. Values in the data set $\mathbf{X}$ will contain some level of error since they are determined from the measured values y,z,θ, and δ returned by the Polhemus sensor [1]. The idea in regression is to form a system of equations (6) that is over-constrained. The over-constraint is then used to minimize the effect of error in the data set $\mathbf{X}$ by finding parameters $B_2, \ldots, B_n$ that see the model best fit the points in the data set. There are a number of ways that *best fit* can be defined . We choose the total least squares fitting method [6] (also known as linear orthogonal regression). Here the best fit is defined to occur when the Sum Square Error (SSE) is minimized, where SSE is given by:

$$SSE = \sum_{i=1}^{q} \frac{(X_{i_1} + B_2 X_{i_2} + \ldots + B_n X_{i_n})^2}{(1 + B_2^2 + \ldots + B_n^2)} \quad (7)$$

where q is the number of rows in the data set. Geometrically the approach can be interpreted as fitting to the data set, a hyper-plane which minimizes the sum of the squared Euclidean distances between each point in the data set and the hyper-plane. To solve equation (7) for the unknown parameters B_2 to B_n, we use the method based on Singular Value Decomposition outlined in [6]. We have chosen to use the total least squares fitting approach because it is the most suitable method for our situation for the following reasons. First, all variables $X_1 \ldots X_n$ in the model contain error. This is in contrast to the more widely adopted fitting approach of ordinary least squares where only one variable in the model is assumed to contain error. Second, it is reasonable to assume that the error in each variable is independent of the error in other variables. Third, it is reasonable to assume that the error in all variables are normally distributed with zero mean and equal variance.

[1]measurement made by any sensor will contain some level of error. For example, measurements made by the Polhemus sensor are transmitted as an electromagnetic signal, and so can have errors introduced by metallic objects or stray magnetic fields existing in the vicinity of the sensor

4 Results

Table 1 presents the results of our C-space construction method for the selected set of c_m^* shown in column 1. Columns 2 and 3 of the table show the contact formation and constraint corresponding to each c_m^* in column 1. Columns 3 to 7 show two rows of parameter values for each c_m^*, an upper row showing the parameter estimates obtained by our method, and a lower row showing a set of parameter values obtained by measurement (ie. the true parameter values). Finally, column 8 lists the states that contributed to the data set of each c_m^*.

On the whole, parameter estimates determined by our method were accurate. In some cases the estimates were excellent, eg. c_3^*, c_1^* while in others they were less accurate, eg. c_{12}^*, c_{13}^*, c_9^*. Two requirements for accurate parameter estimates were identified. First, a sufficient amount of data, ie. that the system of equations formed by the data set in the regression analysis was sufficiently over-constrained. The position sensor was capable of data output at a rate of 120 Hz, so sufficient data was generally available for all c_m^*. The second requirement for accurate parameter estimation was a good range of data. ie. that the demonstrator traced out paths over a wide range on the C-surface. This was the reason for less accurate estimates in our case. For c_m^* with a number of paths over distinct areas of the C-surface, eg. c_3^*, c_1^*, the parameter estimates were excellent. However, for cases where the range of data was more limited, eg. c_{12}^*, c_{13}^*, c_9^* less accurate parameter estimates were obtained.

There were two reasons why a limited range of paths were traced out on a c_m^* in the demonstration. The first was because the c_m^* was only briefly visited. For example, constraint j-3 was made in passing and existed only in two states in Demonstrated Path no.5. This resulted in less accurate estimates for c_9^*. What is required in these cases is a larger demonstration set so that more paths on distinct parts of the C-surface become available. The second reason for limited path range on a c_m^* was because the geometry of the task limited the range of motion that could be demonstrated, ie. that the c_m^* only exists over a small region

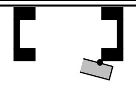
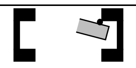
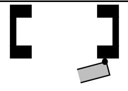
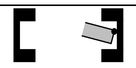
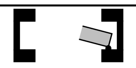
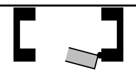
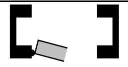
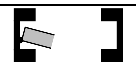
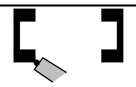
c_m^*	Contact Formation	ρ_m^*	$\hat{b}$ / b	$\hat{c}$ / c	$\hat{d}$ / d	$\hat{e}$ / e	$\hat{f}$ / f	States in Data Set
c_1^*		h-3	0.008	0.524	0.227	0.019	-0.998	(D_1) 22,21,24,21,27 (D_2) 27 (D3) 21,29,28
			0.011	0.525	0.223	0	1	(D_4) 27 (D_5) 21,24,21,30 (D_6) 21,29,28
c_9^*		j-3	-0.023	0.523	0.310	0.730	0.634	(D_5) 66,61
			0.011	0.570	0.223	0	1	
c_4^*		d-13	0.466	0.086	0.016	-0.972	0.234	(D_1) 20,22
			0.525	0.086	0.011	-1	0	
c_{11}^*		d-16	0.183	0.096	0.002	0.033	0.999	(D_1) 54,47,1 (D_2) 55,54,47,1 (D_3) 55,54,47,1
			0.213	0.086	0.011	0	1	(D_4) 47,1 (D_5) 79,48,47,1 (D_6) 55,54,47,1
c_{12}^*		c-15	0.581	0.078	-0.004	0.996	0.088	(D_1) 47,1 (D_2) 47,1 (D_3) 47,1 (D_4) 77,43,47,1
			0.540	0.086	0.011	1	0	(D_5) 49,78,79,48,47,1 (D_6) 47,1
c_6^*		d-14	0.191	0.118	0.043	0.001	0.996	(D_4) 33,76
			0.223	0.086	0.011	0	1	
c_{13}^*		b-8	0.470	-0.097	0.157	0.157	-0.988	(D_1) 24 (D_5) 24
			0.373	0	-0.011	0	-1	
c_3^*		b-10	0.377	0.005	-0.012	0.009	-0.997	(D_1) 27,8,38,8,54,47,1 (D_3/D_6) 28,9,65, 60,55,54,47,1 (D_2) 7,8,27,8,38,8,9,65,
			0.383	0	-0.011	0	-1	60,55,54,47,1 (D_4) 7,8,27,8,33,76,38,77,43,47,1
c_8^*		e-1	0.024	0.512	0.353	-0.999	-0.105	(D_4) 4,74
			0	0.525	0.373	-1	0	

Table 1: Regression analysis results for selected c_m^*

of C-space. For example, many paths contained constraint c-15, so one would expect precise parameter estimates for c_{12}^*. However for c-15, motion is naturally constrained by the geometry of the task, ie. the spindle cannot move very far from a vertical orientation. Although parameter estimates in these cases are not overly accurate, they still do in fact provide an accurate description of the c_m^* over the limited range of motion allowed by the task. The process of noise removal means deriving noise-free paths that lie on c_m^* [9]. Then parameter estimates that describe c_m^* well over the limited range allowed by the task will be useful for the noise removal process. That is, our derived path will move onto a new C-surface (ie. we will move into a new state) before reaching regions on c_m^* described badly by the parameter estimates. Many of the c_m^* with less accurate parameter estimates in Table 1 do so for this reason, eg. c_9^*, c_{11}^*, c_{12}^*.

We have seen that in many cases the approach results in accurate parameter estimates. Where limited demonstration data is available, estimates are less accurate. A feature of the approach is that it has the natural tendency to generate accurate estimates for c_m^* where noise removal is most in need. We most desire noise removal for c_m^* in two categories. First, c_m^* of constraints pivotal to completing the task. In our spindle insertion task, b-10 is a constraint of this type. It occurs at some point in every demonstration, and is critical for the completion of the task. For such constraints we desire accurate c_m^* estimates so that paths derived for the robot from these estimates exactly reflect the topology of C-space. The derived paths must be of high quality because they will be used often and are critical to the completion of the task. Our approach tends to provide accurate parameter estimates for the c_m^* of pivotal constraints because such constraints are generally demonstrated often, leading to a large data set with good range. Besides b-10, constraints h-3, c-15, and d-16 are also pivotal constraints in the task.

The second category of c_m^* where we desire noise removal are those on which particularly noisy paths were demonstrated. Our approach will tend to produce accurate parameter estimates for these c_m^* because the noisy paths will by definition visit diverse parts of the C-surface. For example, we highlight the parameter estimates obtained for c_8^* shown in Table 1. These estimates are quite accurate given that only one path from the demonstration was available for the regression analysis. The reason was because a relatively noisy path was demonstrated. The human produced a path that saw the spindle orientation move from between 6.5 and 39 degrees to the vertical, and 0 mm to 11.2 millimeters of spindle compression. In comparison we see the parameter estimates for c_4^* are less accurate. Although roughly the same amount of data was available in this case, the motion demonstrated was relatively noise free with the spindle moving in close to a direct line through state 20 to state 22.

5 Conclusion

We have presented an approach for constructing the C-space for a task from demonstration. It was presented as an alternative to well known methods for constructing C-space that require a geometric model of the task. Our motivation here was to derive C-space for the purpose of noise removal in PbD, however the approach is general and could be used in other applications where a geometric task model is not available. Experiments showed the validity of the approach. They showed that the approach could derive a representation of C-space in regions that were visited in the demonstration. The representation was generally accurate, although it could be less accurate in regions that were visited only briefly in the demonstration. Experiments also showed the suitability of the method for the purposes of noise removal in PbD. The method tended to provide an accurate description of C-space in those regions where noise removal was most required. That is, for regions (i) that must be visited to complete the task, and (ii) that contained noisy demonstrated paths. On these basis, we found the approach to be a valid way to produce a a C-space representation for a task. In particular, it was found to be particularly suitable for deriving C-space for the purpose of noise removal in the context of PbD.

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, May 1997.

[2] C.Bajaj and M.S.Kim. Generation of configuration space obstacles: The case of a moving sphere. *IEEE Journal of Robotics and Automation*, 4(1), 1988.

[3] 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.

[4] B.Faverjon F.Avnaim, J.D.Boissonnat. A practical exact motion planning algorithm for polygonal objects amidst polygonal obstacles. In *Proceedings. IEEE International Conference on Robotics*, pages 1656–1661, April 1988.

[5] H.Asada and H.Izumi. Automatic program generation from teaching data for the hybrid control of robots. In *IEEE Transactions on Robotics and Automation*, pages 163–173, 1989.

[6] S.Van Huffel and J.Vandewalle. *The Total Least Squares Problem: Computational Aspects and Analysis.* Frontiers in Applied Mathematics. Society for Industrial and Applied Mathematics, Philadelphia, 1991.

[7] Polhemus Inc. *3SPACE FASTRAK Users Manual Revision F.* 1993.

[8] C.J.Nachtsheim J.Neter, M.H.Kutner and W.Wasserman. *Applied Linear Regression Models.* IRWIN, 3rd edition, 1996.

[9] J.R.Chen and A.Zelinsky. Robot programming by demonstration: Removing suboptimal actions in a partially known configuration space. In *Submitted to the 2001 IEEE International Conference on Robotics and Automation.*

[10] M. Kaiser and R. Dillman. Building elementary skills from human demonstration. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, pages 2700–2705, April 1996.

[11] 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.

[12] 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.

[13] Wim Witvrouw Qi Wang, Joris De Schutter and Sean Graves. Derivation of compliant motion programs based on human demonstration. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2616–2621, April 1996.

[14] R.A.Brooks and T.Lozano-Perez. A subdivision algorithm in configuration space for findpath with rotation. *IEEE Transactions on Systems, Man, and Cybernetics*, 15(2):224–233, March/April 1985.

[15] R.W.Brockett. Hybrid models for motion control systems. In H.L.Trentelman and J.C.Willems, editors, *Essays on Control: Perspectives in the Theory and Its Applications*, chapter 2, pages 29–5. Birkhauser, Boston, MA, 1993.

[16] Marjorie Skubic and Richard A. Volt. Identifying contact formations from sensory patterns and its applicability to robot programming by demonstration. In *Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 458–464, 1996.

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

[18] T.Lozano-Perez. Spatial planning: A configuration space approach. *IEEE Transactions on Computing*, C32:108–120, February 1983.

[19] T.Lozano-Perez. A simple motion-planning algorithm for general robot manipulators. *IEEE Journal of Robotics and Automation*, 3(3):224–238, June 1987.