Modeling Planar Assembly Paths from Observation

George V Paul and Katsushi Ikeuchi

21 March 1996
CMU-RI-TR-96-13

Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213-3890

This work was sponsored in part by National Science Foundation under IRI-9224521 and in part by the Avionics Laboratory, Wright Research and Development Center, Aeronautical Systems Division (AFSC), U.S. Air Force, Wright-Patterson, AFB, OH 45433-6543 under F33615-90-C-1465 and F33615-93-1-1282.

Keywords: assembly plan, robot programming, task level programming, task model, c-surface, configuration space, contact states.

Abstract

This report describes a system for obtaining the compliant motion plan for a planar assembly task, given a sequence of observations of a human performing it. The compliant motion plan in configuration space is a series of connected path segments lying on the configuration space obstacle. We use the observed configurations of the assembled objects to selectively compute the features of the c-space obstacle on which the path lies. We project the observed configurations onto these features and reconstruct the path segments. The connected path segments form the model of the observed task and can be used to program a robot to repeat the task. We demonstrate the system using the planar peg in hole task.

Table of Contents

1 Introduction

Automatic planning of robots for assembly tasks has been attempted by a number of researchers[9][10]. This approach is computationally expensive. Another approach is to program robots for assembly tasks by demonstrating the task [4][12][14]. Our assembly plan from observation (APO) system is an example of the latter approach. Automatic planning typically involves the combinatorics of computing all possible contacts to build the configuration space (c-space) obstacle, and then searching for a feasible path in c-space. Unlike that approach, our system computes only the features of the c-space obstacle which are relevant to the observed assembly task. In addition, our system reconstructs a feasible path, instead of searching for one.

We use a real time vision system to record the assembly task. We then recognize and track the parts in each observed scene. The position and geometry of the objects are used to find the contacts between the objects in each observed scene. Using this, we identify the distinct contact configurations involved in the assembly task. Next, we analytically compute the feature on the c-space obstacle corresponding to each observed configuration. We then use the observed configurations to reconstruct the path segments lying on each computed c-space obstacle feature. These path segments constitute the model of the assembly task. Finally, we use the modeled path to program a robot to repeat the assembly task.

Figure 1 illustrates the theory behind this work. Consider a series of observed 2D scenes of the dark rectangle coming into contact with a large rectangle as shown in Figure 1(a). If we consider only 2D translation, the c-space obstacle will be the shaded rectangle as shown in Figure 1(b). Instead of computing this complete c-space obstacle, we find the contacts made in each observed scene and compute only the necessary features of the c-space obstacle, C_1 and C_2, which correspond to the contacts. The observations can then be corrected by projecting onto their corresponding c-obstacle feature. We can also find the intersections between adjacent path segments explicitly. The compliant motion path is then a series of path segments s_0, s_1, and s_2 as shown in Figure 1(c).

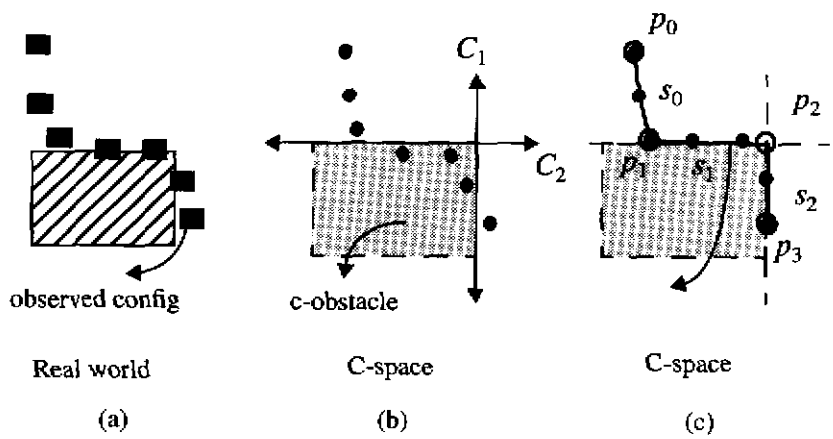

Figure 1 Path in c-space for 2D (translation only).

This work demonstrates a working system which can reconstruct the compliant motion plan of a planar assembly task using both rotation and translation motion. We also propose a numerical technique to find the local intersection of path segments lying on adjacent c-space obstacle features.

2 Related Work

The analysis of human assembly actions covers three broad areas. The finger-finger relations of the hand for understanding the grasp, the finger-object relations for segmenting the task and understanding the grasp, and finally the object-object relations for obtaining the compliant motion plan.

Work such as by Kang[6], Tung[14] and others use the dataglove as the primary sensor to analyze finger-finger and finger-object relations while performing assembly tasks. The dataglove is an invasive sensor and has limited accuracy. This has limited such works to mostly pick and place tasks. Inaba[5], Kuniyoshi[8] and others have also used stereo vision systems to observe the task. But these systems are also limited to simple pick and place tasks. This work uses a more precise range data using a stereo system and a robust localization system, which allows us to interpret actions such as compliant motion.

This work primarily deals with extracting the compliant motion path from object-object relations. Previous work on the APO system[4][12] used sparse observations of the assembly task. The system then computed the contact state of the object at the beginning and end of the task. The motion plan was then obtained by searching a transition graph of contact states. This approach is sufficient for most simple assembly tasks, but it cannot recover the compliant motion in the case of assembly tasks such as peg in hole insertion. Our new approach tries to extract the compliant motion of assembly tasks by observing the task continuously. We accomplish this by adapting some of the results of the configuration space approach to automatic path planning.

The configuration space approach for automatically planning the fine motion paths for assembly tasks was proposed by Lozano-Perez[9][10], Mason[11] and others. The idea is to compute the c-space obstacle and plan the path in c-space. Computing the c-space obstacle is crucial for assembly tasks where there is contact between the assembled object and the environment. Work such as by Bajaj et al [2] deal with algebraically computing the c-space obstacle, but is limited to translational motion only. Other work such as by Avnaim et al [1], and Brost [3] deal with computing the c-space obstacles in the planar case with both rotation and translation. We use Brost's results for our work to algebraically compute the features of the c-space obstacle.

3 Observation of the Assembly Task

The analysis of human assembly actions is based on the relations between the hand and the assembled parts at every instant of the assembly. Parts of the assembly task, such as during the compliant motion, consists of relatively small but significant motion. We capture this motion by recording the human assembly using a real-time stereo system [6]. The stereo images give us a sequence of dense and accurate 3D images of the scene. The assembled parts are then identified in the initial scene and tracked continuously through the whole sequence using a robust localization algorithm.

3.1 Localization and Tracking

We use a geometrical modeler called Vantage to build the models of the assembled parts. These models are then used to localize the parts and fingers in the scene using a 3D template matching algorithm called (3DTM) [15]. The 3DTM has features which make it well suited for localization in assembly scenes where occlusion and noise are inevitable.

The 3DTM can localize an object in a 3D scene, given a rough estimate of the location of the object in the scene. The algorithm uses sensor modeling, probabilistic hypothesis generation and robust localization techniques to make localization fast and accurate. We extend the localization algorithm from localizing one object in one image to tracking multiple objects in a series of images. We achieve tracking by using the previous locations of each object to compute the starting location for the next localization.

The pose of each object in a scene is given by (x,y,z,θ,ϕ,ψ) in a global reference frame as shown in Figure 2. The output of the localization is a list of poses of the objects in the scene. The output of the tracking system is a list of list of poses of the objects being tracked.

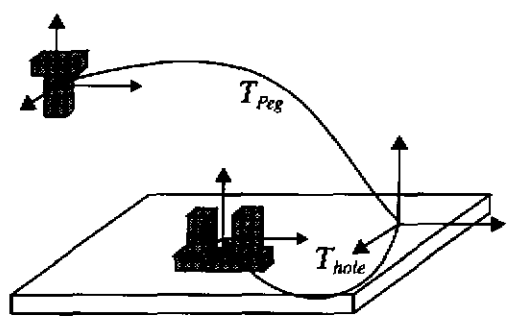

Figure 2 Assembly scene coordinates

The APO system uses the geometrical modeler called Vantage to create scenes of the task by instantiating models of the assembly objects at their observed poses. The sequence of observed poses of the peg during the peg in hole task is shown in Figure 3. The localized models of the peg and hole are superimposed as dark triangulated meshes on each of the

observed intensity images.

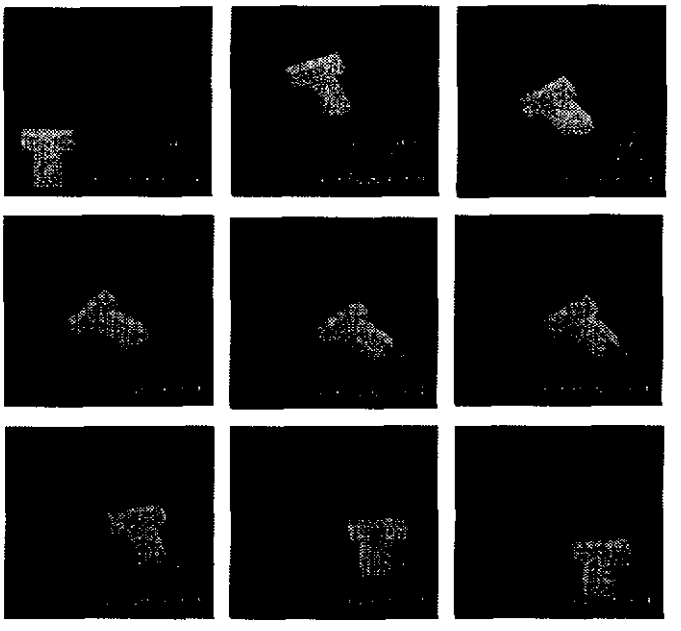

Figure 3 Object tracking in the observed scenes (only 9of the 64 scenes are shown here)

It should be noted that the raw pose obtained from the observation system is error prone as shown in Figure 4. Despite this, we can extract the essential contact information in each observation, and use it to reconstruct the path.

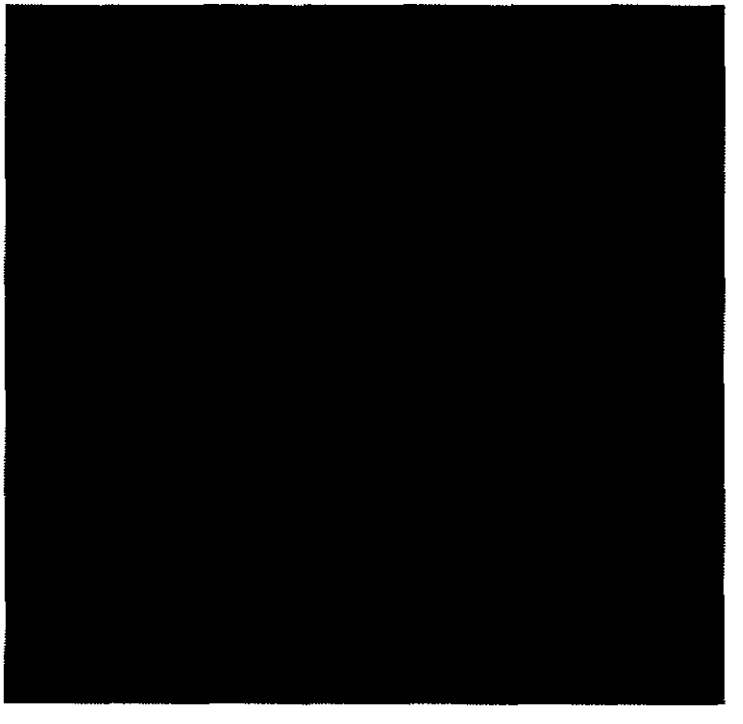

Figure 4 Observed configurations of the peg in hole task

4 Assembly Contacts and C-Surfaces

The key information in an observed scene of an assembly task is the contacts that are made between the assembled object and the environment. This involves identifying the feature pairs which are in contact.

4.1 Assembly Contacts

Each feature pair in contact involves a feature of the assembled object (vertex and edge for polygonal objects) and a feature of the objects in the world as shown in Figure 5.

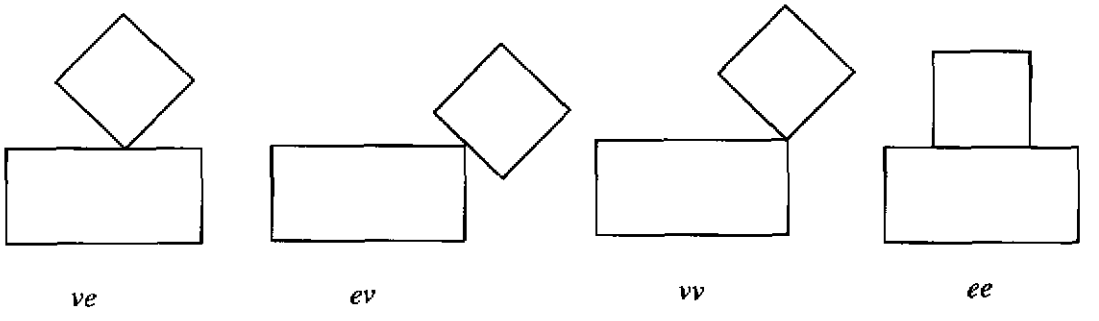

Figure 5 Types of contact pairs

Given the pose of the assembled object and the objects in the world, we can obtain the geometry of each feature of the assembled object and that of the other objects in the environment from the geometrical modeler, Vantage. We then check each feature pair for contact using one of the following contact conditions.

For vertex-on-edge (*ve*) and edge-on-vertex (*ev*) contacts, if the vertex coordinates are (v_x, v_y) and the edge equation[1] is $e_a x + e_b y + e_c = 0$, then the primary condition for the pair to be in contact is given by (1).

$$e_a v_x + e_b v_y + e_c < \delta_{ve} \tag{1}$$

In addition, we check if the projection of the vertex lies within the end points of the edge.

For vertex-on-vertex (*vv*) contacts if the two vertex coordinates are (v_x^1, v_y^1) and (v_x^2, v_y^2), then the condition for the pair to be in contact will be given by (2).

$$\sqrt{(v_x^1 - v_x^1) + (v_y^1 - v_y^1)} < \delta_{vv} \tag{2}$$

For edge-on-edge (*ee*) contacts if the edge equations are $e_a^1 x + e_b^1 y + e_c^1 = 0$ and $e_a^2 x + e_b^2 y + e_c^2 = 0$, then the primary condition for the pair to be in contact will be given by (3).

1. (e_a, e_b) is a 2d unit vector.

$$1-(e_a^1 e_a^2 + e_b^1 e_y^2) < \delta_{ee1} \wedge \left| e_c^1 - e_c^2 \right| < \delta_{ee2} \quad (3)$$

In addition, for *ee* contacts we check if the edge segments overlap.

For every observed configuration o_i, we end up with a set of feature pair sets in contact $\{c_{i1}, c_{i2}, \ldots\}$.

4.2 Configuration Space Obstacle Features

By definition, the set of configurations of the assembled object which keeps a set of its features in contact with a set of features of another object will be a feature of the c-space obstacle in configuration space. The order of this feature in c-space depends on the degree of contact. For example, a single *ve* contact results in a two dimensional c-surface in (x,y,θ) space, whereas two simultaneous *ve* contacts correspond to a one dimensional edge in c-space. Given the set of feature pairs in contact, we can algebraically compute the infinite surface on which the feature of the c-space obstacle will lie.

For polygonal objects in a plane, the features of the c-space obstacles can be two dimensional facets, one dimensional edges and zero dimensional vertices. There are two types of facets of configuration obstacles corresponding to single *ve* and single *ev* contacts. There are four classes of edges which correspond to contact configurations with more than one *ve* or *ev* contacts or when there are *ee* or *vv* contacts. The vertices of the c-space obstacle are defined by their coordinates.

We briefly introduce the representation of a two dimensional surface in c-space corresponding to a single vertex-on-edge (*ve*) contact here. The representations of the all other possible contact configurations can be found in Brost's work [3].

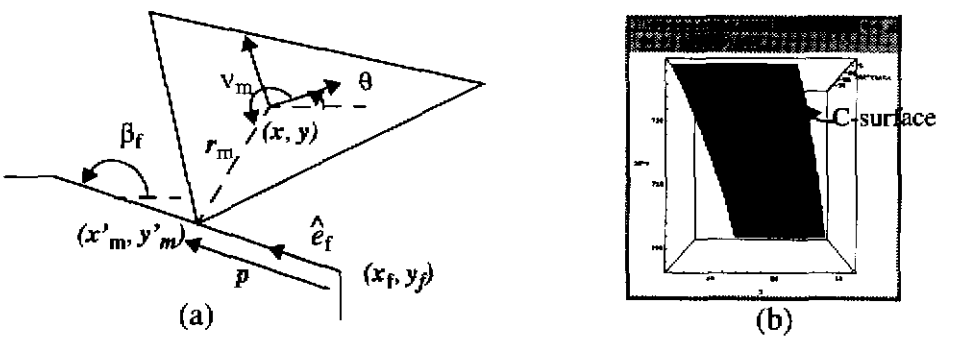

Figure 6 A single *ve* contact

Consider two polygons in contact in a plane as shown in the figure. The vertex of the moving object is in contact with the edge of the fixed object. The configuration of the moving triangle is (x,y,θ). The configuration obstacle feature corresponding to this contact will be a subset of an infinite two dimensional surface embedded in (x,y,θ) space. This surface (c-surface) can be parameterized by two variables (p, θ). Brost defines p as a perimeter variable which is the distance of the contact vertex from the clockwise vertex of the contact edge as

shown in Figure 7. The parameter θ is identical to the angle of the moving object.

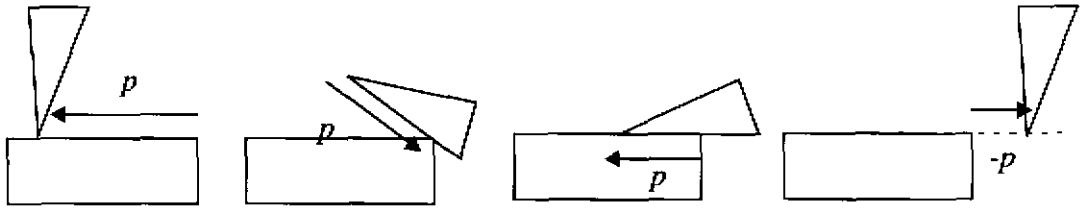

Figure 7 The parameter p

The infinite c-surface can then be described in parametric form by (4), (5) and (6).

$$x = f_x(p, \theta) \tag{4}$$

$$y = f_y(p, \theta) \tag{5}$$

$$\theta = f_\theta(p, \theta) = \theta \tag{6}$$

The x and y functions for a single *ve* contact are given by the equations (7) and (8). A example of the 2D c-surface corresponding to a *ve* contact is shown in Figure 10.

$$\begin{aligned} f_{x_{ve}}(p, \theta) &= k_{1_x} + k_{2_x} p + k_{3_x} \cos(\theta + \kappa_{3_x}) \\ k_{1_x} &= x_f \quad k_{2_x} = \cos(\beta_f) \quad k_{3_x} = -r_m \quad \kappa_{3_x} = \nu_m \end{aligned} \tag{7}$$

$$\begin{aligned} f_{y_{ve}}(p, \theta) &= k_{1_y} + k_{2_y} p + k_{3_y} \sin(\theta + \kappa_{3_y}) \\ k_{1_y} &= y_f \quad k_{2_y} = \sin(\beta_f) \quad k_{3_y} = -r_m \quad \kappa_{3_y} = \nu_m \end{aligned} \tag{8}$$

There exists an inverse mapping of a configuration space point in (x,y,θ) space to a point on the c-surface defined by the (p, θ) variables. The form of these equations are given by (9) and (10).

$$p = f_p(x, y, \theta) \tag{9}$$

$$\theta = f_\theta(x, y, \theta) = \theta \tag{10}$$

The equation (9) for the *ve* contact is given by (11).

$$\overrightarrow{e_{cw}v} = \begin{bmatrix} (x + r_m \cos(\theta + \nu_m) - x_f) \\ (y + r_m \sin(\theta + \nu_m) - y_f) \end{bmatrix} \tag{11}$$

$$p = f_p(x, y, \theta) = \overrightarrow{e_{cw}v} \cdot \hat{e}_f$$

$$\theta = f_\theta(x, y, \theta) = \theta \tag{12}$$

Given any configuration space point (x,y,θ) the equations (9) and (10) will find the (p, θ) of the closest point which lies on the c-surface. We use this property to correct the raw config-

urations obtained from the observation system.

If the i-Th. observed configuration is o_i, (x_i,y_i,θ_i) and the c-surface C_j is parameterized by (p,θ_i), then the projected configuration r_i (p_i,θ_i) can be obtained from (9)-(10). The coordinates (p_i,θ_i) can be converted to the r_i (x_i,y_i,θ_i) using (4)-(6). The error in the observation corresponding to the c-surface C_j will be given by (13).

$$Error(o_i, C_j) = \|r_i - o_i\| \tag{13}$$

The counterparts of the equations (4)-(6) and (9)-(10) exist for the *ev* facets and the four classes of edges. These are derived in [3]. The form of these equations are listed in the Appendix for reference.

4.3 C-Surface Selection

The true configuration of the object r_i corresponding to any observed configuration o_i depends on the c-surface the o_i corresponds to. This c-surface can be unambiguously obtained when the contact pairs are well defined as shown in Figure 8(a). Here, there are two simultaneous contacts $\{ev_1, ev_2\}$, satisfying the contact conditions (1)-(3). The c-surface defined by this contact pair set, $\{ev_1, ev_2\}$ is a class-2 edge. The coefficients defining the c-surface can be obtained from the geometry of the contacts using the equations (4)-(6).

In certain configurations, such as the one as shown in Figure 8(b), not all the three contact pairs obtained using the conditions (1)-(3), $\{ev_1, ev_2, ve_1\}$ will correspond to real contacts. The c-surface in this case can correspond to only one of the two simultaneous contacts $\{ev_1, ev_2\}$ or $\{ev_1, ve_1\}$. We obtain the correct contact pair set as follows.

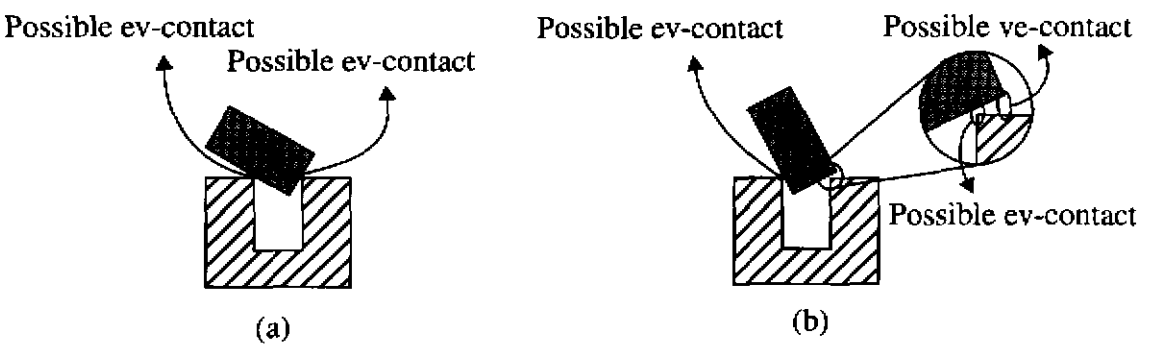

Figure 8 Ambiguous contacts

We generate all feasible contact pair combinations from the contact pair set. Each contact pair combination corresponds to a c-surface. Next we project the observed configuration o_i onto each of these c-surfaces as explained in the previous section. The c-surfaces are mathematical constructs, hence there will always be a projected configuration, but these projected configurations may not be legal as it might cause collision of other features of the object with the environment as shown in Figure 9. Hence we check for collision of objects in the projected configuration. All contact pair combinations whose projection onto c-surfaces

cause collisions (Figure 9(a)) or are infeasible (Figure 9(b)) are discarded.

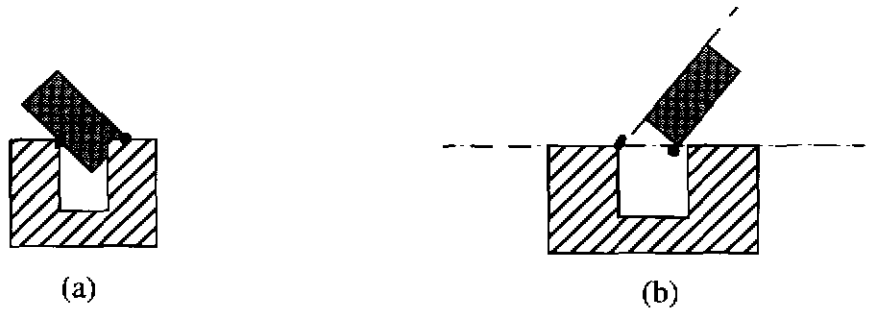

Figure 9 Illegal but mathematically feasible ev-ve contact configurations

Among those that remain, we find the c-surface that has the minimum projection distance using equation (13). This is the c-surface which has the most likelihood to correspond to the observed configuration.

4.4 Path Segments

The compliant motion of an assembly task in c-space is composed of path segments lying on the c-space obstacle. All configurations in a path segment will maintain the same contacts and correspond to the same c-surface. The motion in this path segment will be a smooth curve lying on this c-surface.

Suppose all consecutive observations $\{o_{j_1}, o_{j_2}, ..., o_{j_k}\}$ correspond to the same c-surface C_j, the path segment is a curve which is fit to the projections of $\{o_{j_1}, o_{j_2}, ..., o_{j_k}\}$. Any point on this curve will maintain the set of contacts $\{c_j\}$. This will be the path segment PS_j. The continuous path segment is obtained by interpolating in the (p,θ) space as shown in Fig-

ure 10.

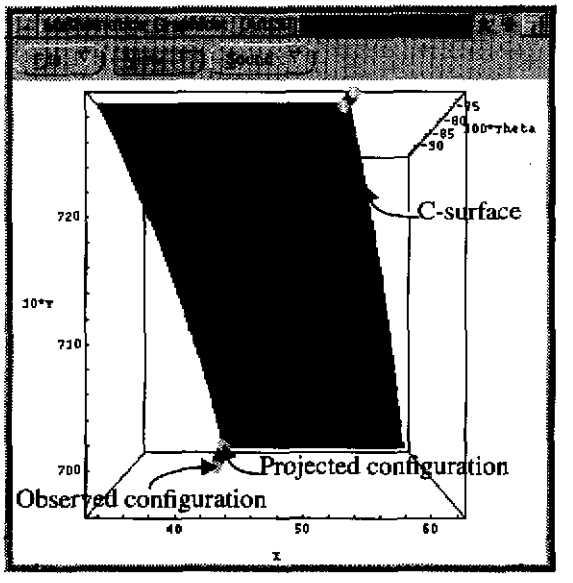

Figure 10 C-surface corresponding to a *ve* contact

There is an important aspect of compliant motion paths lying on the c-space obstacle, where C_{i-1}, C_i and C_{i+1} are consecutive c-space obstacle features with finite path lengths. The start point of a path segment on C_i lies on the intersection of C_i and C_{i-1}. The end point of the same path segment will lie on the intersection of C_i and C_{i+1}. These points are critical points in the path. We compute these start and end points of each path segment explicitly using a numerical technique.

The technique can be illustrated by considering a simple 2D case as shown in Figure 11. Let the last observation corresponding to the c-surface C_i be o_i, and let C_{i+1} be the c-surface corresponding to the next contact configuration. We want to compute the intersection q of the two c-surfaces C_i and C_{i+1} closest to o_i.

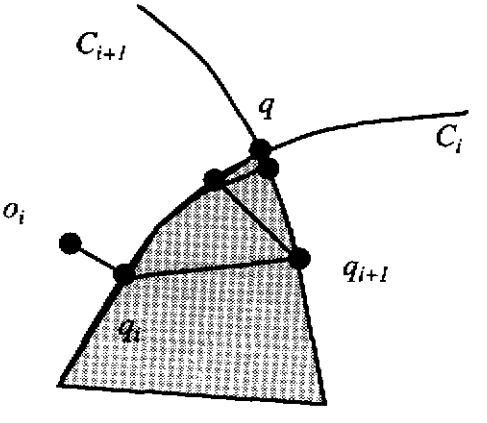

Figure 11 Intersection of two path segments

We find the intersection by successively projecting the observed point onto C_i and the C_{i+1}

until the point converges. The procedure is as follows.

1. Project o_i onto C_i using equations (9)-(10) and obtain (p_i, θ_i). Convert (p_i, θ_i) back to (x_i, y_i, θ_i) using equations (4)-(6). This is the projected point q_i.

2. Project q_i onto C_{i+1} using equations (9)-(10) and obtain (p_{i+1}, θ_{i+1}). Convert (p_{i+1}, θ_{i+1}) back to $(x_{i+1}, y_{i+1}, \theta_{i+1})$ using equations (4)-(6). This is the projected point q_{i+1}.

3. If $|q_i - q_{i+1}| < \delta$ (threshold), return q_{i+1}; else set $o_i = q_{i+1}$ and go to step 1.

The procedure converges because the initial configuration o_i is close to the intersection point of two c-surfaces whose intersection exists. An advantage of this technique is that it can find intersections of path segments lying on different classes of c-surfaces.

We compute the end points of all the path segments using this procedure. The start points of a path segment coincides with the end point of the previous path segment. The path segment is then obtained by interpolating the (p, θ) parameters from the start point (p_s, θ_s) to the end point (p_e, θ_e).

$$PS_i \rightarrow \left(f_{x_{C_i}}(p, \theta), f_{y_{C_i}}(p, \theta), \theta \right) \Big|_{(p_s, \theta_s)}^{(p_e, \theta_e)} \tag{14}$$

5 Implementation

The three main steps of the algorithm which converts the observed poses of the assembled object to the compliant motion path are shown in Figure 12.

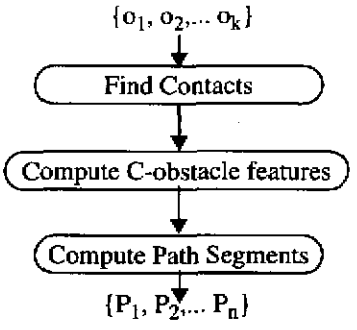

Figure 12 Main steps of the Algorithm

The output of the observation system is list of observed configurations of the assembled object, $\{o_1, o_2, \ldots o_k\}$. The algorithm instantiates the model of the assembled object at each observed configuration, o_i.

The next step is to find the feature pairs which are in contact for each o_i of the assembled object. This is done by comparing all possible combinations of features of the assembled object with the features of the other objects in the environment using the conditions (1), (2) and (3). Let the contact feature pair set for the observed configuration o_i *be* $FP_i = \{fp_1, fp_2, \ldots, fp_{k_i}\}$.

The next step is to find the most likely c-surface corresponding to the observation o_i as explained in the previous section. We then segment the observed configurations $\{o_1, o_2, \ldots o_k\}$ into contiguous segments $\{S_1, S_2, \ldots S_n\}$ such that the feature pair sets for all observations in a segment S_j maintain the same contact pair set c_j.

$$\begin{aligned} S_j &= \{o_{j_1}, o_{j_2}, \ldots, o_{j_k}\} \\ FP_{j_1} \cap FP_{j_2} \cap \ldots \cap FP_{j_k} &= c_j \end{aligned} \tag{15}$$

The next step is to use the geometry of the contact feature pair set c_j to identify and compute the infinite c-space obstacle feature C_j using equations (4), (5) and (6). The infinite c-surface corresponding to the single *ve* contact is shown in Figure 10.

The last step of the algorithm is to fit a path segment PS_j lying on the c-space obstacle feature C_j through the observed configurations in $S_j = \{o_{j_1}, o_{j_2}, \ldots, o_{j_k}\}$.

The complete compliant motion path is then the concatenation of the path segments PS_i. The reconstructed path in c-space of the peg in the task observed in Figure 3 is shown in Figure

13.

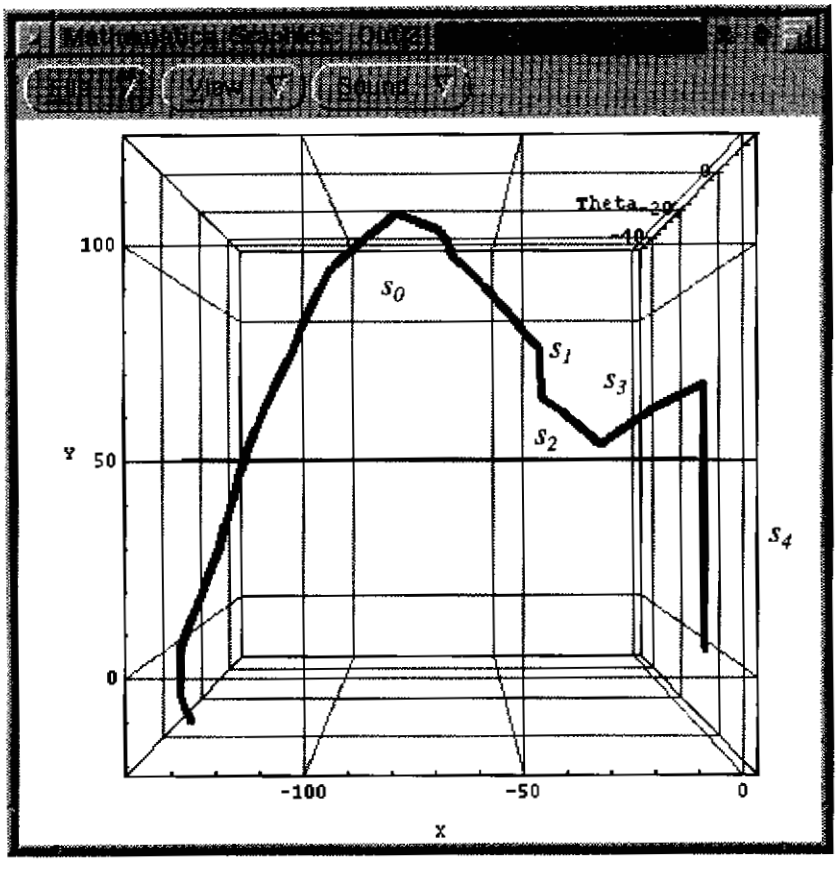

Figure 13 Path of the peg in configuration-space

The path segment s_0 shows the path the peg in free space. This is considered special because no contacts exist and the feature corresponding to this state is the whole 3D space (x,y,θ). This path usually avoids obstacles. Hence unlike other path segments, this path is interpolated between all observed configuration points. The path segment s_1 corresponds to the single *ve* contact that is made initially. The segment s_2 corresponds to the single *ev* contact following s_1. s_3 corresponds to the *ev-ev* contact. Finally, s_4 corresponds to the multiple *ee* contacts when the peg is in the hole. The modeled path in c-space will correspond to the

motion of the assembled peg as shown in Figure 14.

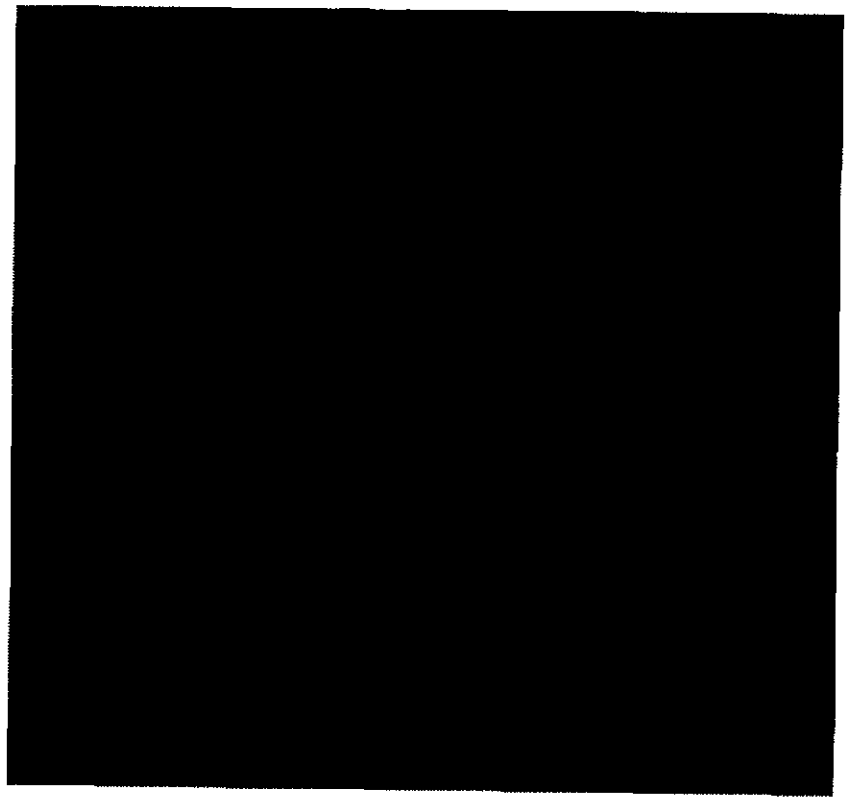

Figure 14 Path of the peg in Vantage

6 Assembly Task Execution

Once we have modeled the exact assembly path used in the task, we can program a robot to execute the trajectory. Since we have all the contact information at every instant of the task, we can compute both the positional and force references for the assembly path. For our implementation we used only the positional information and a relatively accurate robot (RobotWorld) to execute the modeled assembly path. The snapshots of the execution are shown in Figure 15.

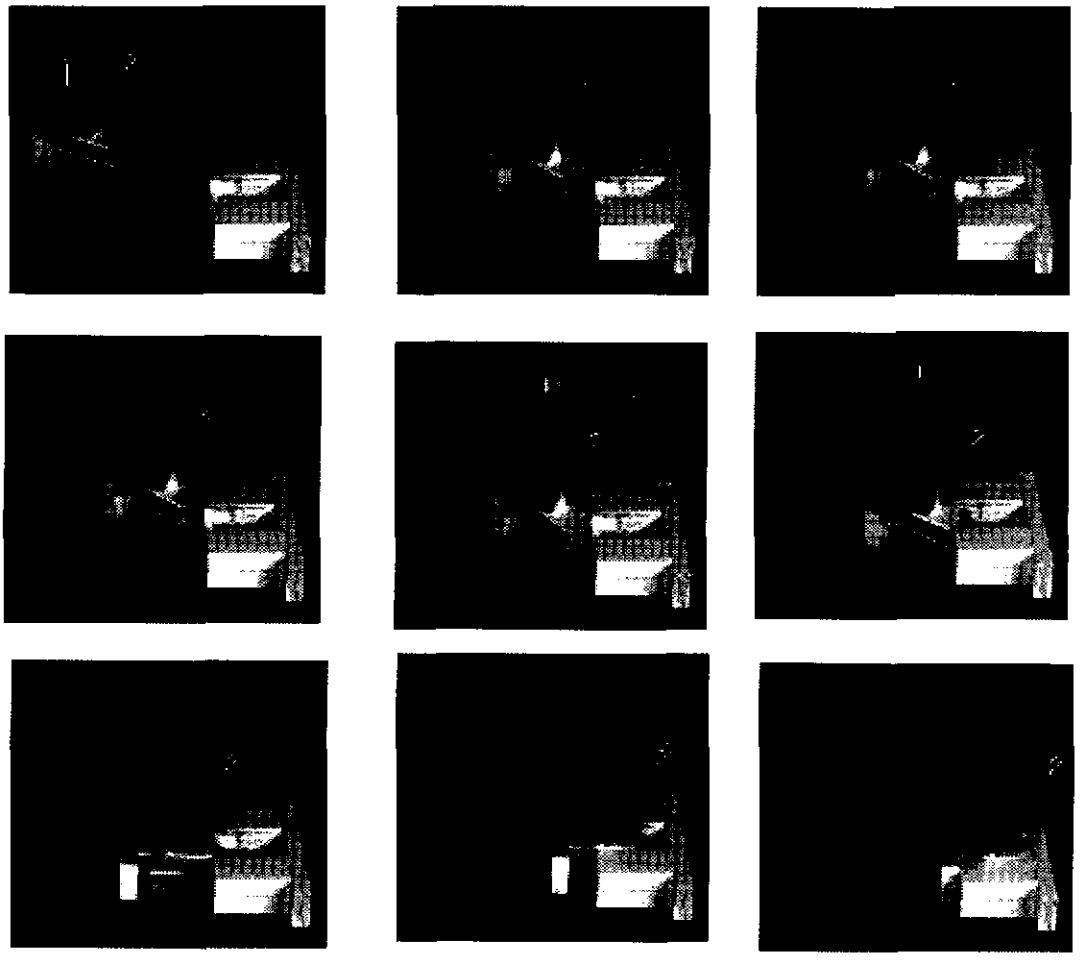

Figure 15 Assembly Task Execution on RobotWorld

7 Conclusions

We have successfully modeled the compliant motion path of planar assembly tasks from observation involving polygonal objects. We reconstructed the path by first computing the relevant features of the c-space obstacle. Then we computed the path segments lying on these features. The connected path segments constitute the model of the observed assembly task.

We plan to extend the system to model assembly paths of 3D polyhedral objects in the 3D space.

Acknowledgments

I wish to thank Santiago E Conant for his help in converting VANTAGE to C++, Mark. D Wheeler for his help with 3DTM, and Yunde Jiar for his help with the stereo system.

Appendix

The features of the configuration space obstacle involving polygonal objects in a plane will be facets, edges and vertices in the 3D space (x,y,θ). The analytical equations describing these features are parametrized by p or θ, or both, or neither. (Only the x component of the features are given here for brevity). The complete derivation and details are in [3].

There are two types of facets, one due to a single vertex on edge (*ve*) contact, and the other due to a single edge on vertex (*ev*) contact. The analytical functions describing these facets are given by equations (16) and (17).

$$f_{x_{ve}}(p,\theta) = k_{1_x} + k_{2_x} p + k_{3_x}\cos(\theta + \kappa_{3_x}) \tag{16}$$

$$f_{x_{ev}}(p,\theta) = k_{1_x} + k_{2_x} p\cos(\theta + k_{2_x}) + k_{3_x}\cos(\theta + \kappa_{3_x}) \tag{17}$$

There are four classes of edges of the c-obstacle which result from multiple *ve* or *ev* contacts or *ee* contacts or *vv* contacts. The x coordinate of a *class-0* edge is parametrized by p and is given by (18).

$$f_{x_{class-0}}(p) = k_{0_x} + k_{1_x} p \tag{18}$$

The x coordinate of *class-1*, *class-2* and *class-3* edges are parameterized by θ as given by (19), (20) and (21).

$$f_{x_{class-1}}(\theta) = k_{0_x} + k_{1_x}\sin(\theta) + k_{2_x}\cos(\theta) \tag{19}$$

$$f_{x_{class-2}}(\theta) = k_{0_x} + k_{1_x}\sin(\theta) + k_{2_x}\cos(\theta) + k_{3_x}\cos(\theta)\sin(\theta) + k_{4_x}\sin(\theta)^2 \tag{20}$$

$$f_{x_{class-3}}(\theta) = k_{0_x} + k_{1_x}\sin(\theta) + k_{2_x}\cos(\theta) + \frac{k_{3_x} + k_{4_x}\sin(\theta) + k_{5_x}\cos(\theta)}{k_{6_x}\sin(\theta) + k_{7_x}\cos(\theta)} \tag{21}$$

The vertex on a c-space obstacle can be defined by its coordinates.

The inverse functions which map from the (x,y,θ) space to the parameter space (p,θ) exists for each class of feature given above. θ is identical in both parameter spaces. The $f_p(x,y,\theta)$ function is defined only for the single *ve*, *ev* contacts and *class-0* edges. The $f_p(x,y,\theta)$ function can be computed using the dot product given by (22), where $\overrightarrow{e_{cw}v}$ and $\hat{e}$ can be computed from the geometry of the contacts.

$$p = f_p(x,y,\theta) = \overrightarrow{e_{cw}v} \cdot \hat{e} \tag{22}$$

References

[1] Avnaim, F. Boissonnat, J.D, and Faverjon, B. A practical exact motion planning algorithm for polygonal objects amidst polygonal obstacles. *IEEE Int. Conf. on R & A* 1988, pages 1656-1661.

[2] Bajaj, C. and Kim, M. Generation of Configuration Space Obstacles: Moving Algebraic Surfaces. *Int. Journal of Robotics Research*, 1990, vol.9, No 1, pages 92-112.

[3] Brost, R C. Analysis and Planning of Planar Manipulation Tasks. *Ph.D. Thesis*, CMU-CS-91-149, 1991, pages 96-103, 264-268.

[4] Ikeuchi, K. and Suehiro, T. Towards an Assembly Plan from Observation, part I: Assembly Task Recognition with Polyhedral Objects. *IEEE Trans on R&A*, vol 10, no 3, 1994, pages 368-385.

[5] Inaba, M. and Inoue, H. Visual-based Robot Programming. *Int. Sym on Robotics Research*, 1989, pages 129-136.

[6] Kang, S.B, Webb, J.A, Zitnick, C.L and Kanade, T, An active multibaseline stereo system with real-time image acquisition, Tech. Rep. CMU-CS-94-167, 1994.

[7] Kang, S.B, Robot Instruction by Human Demonstration. Ph.D. Thesis, Tech. Rep. CMU-RI-94-167, 1994.

[8] Kuniyoshi, Y, Inaba, M, and Inoue, H, Learning by Watching: Extracting Reusable Task Knowledge from Visual Observation of Human Performance, *IEEE Trans on R&A*, Vol 10, No 6, 1994, pages 799-822.

[9] Lozano-Perez, T. Automatic Planning of Manipulator Transfer Movements. *IEEE Trans. System Man and Cybernetics,* 1981, SMC-11(10): pages 681-689.

[10] Lozano-Perez, T. Mason, M. T. and Taylor, R. H. Automatic Synthesis of Fine Motion Strategies for Robots, *Int. Journal of Robotics Research*, 1984, vol.3, No 1, pages 3-24.

[11] Mason, M.T. "Compliance and force control for computer controlled manipulators" *IEEE Trans. Systems, Man and Cybernetics,* vol. 11, no. 6, 1981, pages 418-432.

[12] Paul, G.V. and Ikeuchi, K. Modeling Planar Assembly Tasks: Representation and Recognition. *IEEE Int Conf on Intelligent Robots & Systems,* Aug 1995, Vol 1, pages 17-22.

[13] Suehiro, T. and Ikeuchi, K. Towards an Assembly Plan from Observation, part II. Correction of Motion Parameters Based on Face Contact Constraints. *IEEE Int Conf on Intelligent Robots & Systems,* Jul 1992.

[14] Tung, C.P. and Kak, A. Automatic Learning of Assembly Tasks Using a DataGlove System. *IEEE Int Conf on Intelligent Robots & Systems,* Aug 1995, Vol 1, pages 1-8.

[15] Wheeler, M.D and Ikeuchi, K. Sensor Modeling, Probabilistic Hypothesis Generation, and Robust Localization for Object Recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence.* Vol 17, no 3, pages 252-265, Mar, 1995.