



UNIVERSITAT POLITÈCNICA
DE CATALUNYA

Haptic Rendering of Compliant Motions using Contact Tracking in C -space

Jan Rossell and Israel Vázquez

IOC-DT-P-2004-17
Octubre 2004

Haptic Rendering of Compliant Motions using Contact Tracking in \mathcal{C} -space

Jan Rosell and Israel Vázquez

Institut d'Organització i Control de Sistemes Industrials (IOC-UPC)
Barcelona, Spain. Email: jan.rosell@upc.es

Abstract

This paper presents a new approach for the haptic rendering of compliant motions during the execution of virtual assembly tasks between simple polyhedral objects. The method, based on the task configuration space, analyzes the type of contacts that take place between objects, since this knowledge allows a better haptic rendering when face-face or edge-face contacts occur. Making use of spacial and temporal coherence, the paper presents an efficient procedure to keep track of the current contacts. This allows to comply to the hard temporal constraints of the haptic servo loop. The presented procedures are focused on the haptic rendering during the interaction between convex polyhedra.

I. INTRODUCTION

Haptic devices are used to interact with virtual worlds by allowing to feel the reaction forces and torques that arise when the object attached to the user-manipulated probe touches the other objects in the virtual environment. Starting from a configuration of the manipulated object where there is no interference with the obstacles in the environment, the haptic rendering loop consists of the following steps. First the user moves the manipulated object and the interference with the obstacles is checked. Then, in case of interference, the penetration distance is computed and used to estimate the new (contact) position of the manipulation object and the reaction force and torque. These steps must be carried out in less than one millisecond in order to obtain a smooth and stable haptic rendering [12].

Collision detection procedures developed for computer graphics are usually used [3], [6], [9], since they allow to efficiently detect interference between models composed of thousands of triangles and give information about the collision points and the penetration distances. When several contacts take place simultaneously, the reaction force and torque is computed in an approximate way by the interpolation or the sum of the forces computed at each contact point [4], [12]. Alternatively, other approaches use an approximate discretized representation of the manipulated object (e.g. as a set of points each one with an associated force vector dependant on its location with respect to the obstacle [2]), and/or an approximate discretized representation of the obstacles (e.g. as a set of regular small volumes called voxels [8]).

All these procedures are useful when the virtual world is complex (i.e. constituted by curved objects modelled by huge triangular meshes), since the kind of contacts that usually take place when the user moves the manipulated object are point contacts (i.e. vertex-face contacts). Nevertheless, when virtual assembly tasks between simple polyhedral objects are considered, face-face contacts and edge-face contacts are common and, moreover, compliant motions are usually performed maintaining these types of contacts. In these situations, the previous approaches do not provide a good enough haptic rendering.

To cope with this problem, the kind of contacts that take place between the manipulated object and the obstacles in the environment must be considered. This idea is first introduced by You and Xiao [14]. These authors compute the reaction forces and torques once the contact situation has been identified as a collection of principal contacts¹. Contact identification is done by using classical collision detection algorithms, reasoning procedures and a precomputed graph of possible contact situations.

Following this idea of determining the reaction force and torque from the knowledge of the current type of contact taking place, we propose a framework based on the task configuration space (\mathcal{C} -space). Figure 1 shows the proposed framework, that was first explored in a preliminary version [13], providing promising results. The use of \mathcal{C} -space has the following advantages:

- The \mathcal{C} -space captures the contact constraints and eases the contact identification since the manipulated object collapses to a single point [7].
- The \mathcal{C} -space can be used to compute, using gross-motion planning techniques, a force field to guide the user in performing the virtual assembly task (e.g. [11]).
- The visualization of \mathcal{C} -space, together with that of physical space, aids the user in performing the constrained motions involved in low-clearance assembly tasks (e.g. [5] [10]).

0

This work was partially supported by the CICYT projects DPI2004-03104 and DPI2002-03540. Israel Vázquez has a scholarship from COFAA-IPN and COTEPABE-SUPERA

¹A principal contact is a contact between a pair of topological elements (vertices, edges or faces) that are not the boundary of other topological elements in contact.

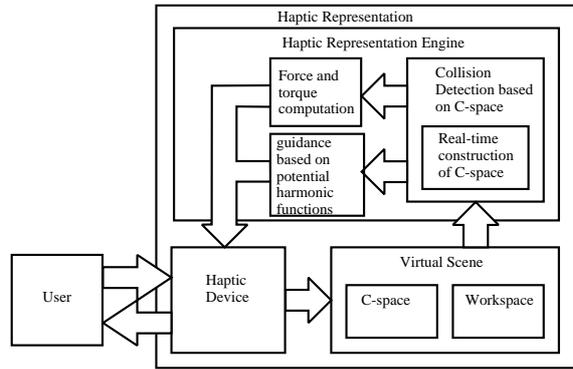


Fig. 1. Haptic rendering of an assembly task using C -space.

The obvious disadvantage of using the C -space is the complexity of its computation. Nevertheless, for haptic rendering purposes and taking into account the spacial and temporal coherence, the C -space only needs to be computed locally. This can be done by keeping track of the current contacts taking place or of the nearest potential contacts (if no contact is taking place). This paper presents an efficient procedure with that purpose, that allows to comply to the hard temporal constraints of the haptic servo loop, and is focused on the haptic rendering during the interaction between convex polyhedra.

The paper is structured as follows. Section II introduces the C -space and its modelling, which is the base of the contact tracking procedure presented in Section III. Then, Section IV introduces the haptic rendering. Finally, Section V discusses the contribution of the proposed approach.

II. CONFIGURATION SPACE

Let \mathcal{A} and \mathcal{B} be two polyhedra describing the manipulated object and a static object, respectively. Let \mathcal{F}_A be reference frame attached to \mathcal{A} and \mathcal{F}_W be the fixed reference frame of the workspace. Let $q^A = (x^A, \Theta^A)$ be a configuration of \mathcal{A} , where x^A and Θ^A describe, respectively, the position and the orientation of \mathcal{F}_A with respect to \mathcal{F}_W . Let also $q_0^A = (x_0^A, \Theta_0^A)$ be the current configuration of \mathcal{A} .

The Configuration Space (C -space) of \mathcal{A} is the space defined by all of its configurations. The subset of configurations where there is interference between \mathcal{A} and \mathcal{B} is called the C -obstacle \mathcal{CO}_B . The border of \mathcal{CO}_B is composed of C -faces, each C -face being the subset of configurations of a five-dimensional hyper-surface $f(q^A) = 0$ where a given basic contact takes place. There are three types of basic contacts between \mathcal{A} and \mathcal{B} :

- Type-A: a face of \mathcal{A} against a vertex of \mathcal{B} .
- Type-B: a vertex of \mathcal{A} against a face of \mathcal{B} .
- Type-C: an edge of \mathcal{A} against an edge of \mathcal{B} .

For each type of basic contact, an applicability condition can be defined to determine if for a given orientation of \mathcal{A} the contact can take place [1].

For orientation Θ_0^A , the subset of the C -faces corresponding to the basic contacts that satisfy their applicability condition are planar polygons over the corresponding planes $f(x^A, \Theta_0^A) = 0$. These are the faces of the 3D polyhedron that represent the C -obstacle for that orientation Θ_0^A .

A. Modelling of C -obstacles

From now on let consider all polyhedra convex (non-convex polyhedra are previously decomposed into convex ones). Then, the modelling of the (convex) C -obstacles is done as follows.

Each C -obstacle \mathcal{CO}_B will be represented as a graph, \mathcal{G}^{AB} , that captures its topology. The nodes of \mathcal{G}^{AB} are all the possible basic contacts between the topological elements of \mathcal{A} and those of \mathcal{B} , and its arcs show the neighboring relationship, as computed in the Appendix. The neighbor nodes of a node of \mathcal{G}^{AB} are called \mathcal{G} -neighbors.

For a given orientation of the manipulated object \mathcal{A} , only a subset of the nodes of \mathcal{G}^{AB} can take place, i.e. those satisfying the applicability condition. Those nodes form a subgraph, called *applicability subgraph*, $\mathcal{G}_{\Theta_0^A}^{AB}$.

Due to the spatial and temporal coherence, i.e. due to the fact that the C -space changes very slightly around q_0^A for a motion of \mathcal{A} between two consecutive instants of time, only a subset of nodes of $\mathcal{G}_{\Theta_0^A}^{AB}$ near q_0^A is to be considered. This subset is called *local subgraph*, \mathcal{G}_L^{AB} . This local subgraph is computed from a seed node set, called *Near*, that is the set of C -faces that are nearest to x_0^A (i.e. those such that the distance between x_0^A and the plane $f(x^A, \Theta_0^A) = 0$ is minimum²).

²Note that, since \mathcal{A} and \mathcal{B} are convex, *Near* will contain only one element except for the orientations where face-face contacts or edge-face contacts are possible (and in those cases the C -faces of *Near* are coplanar).

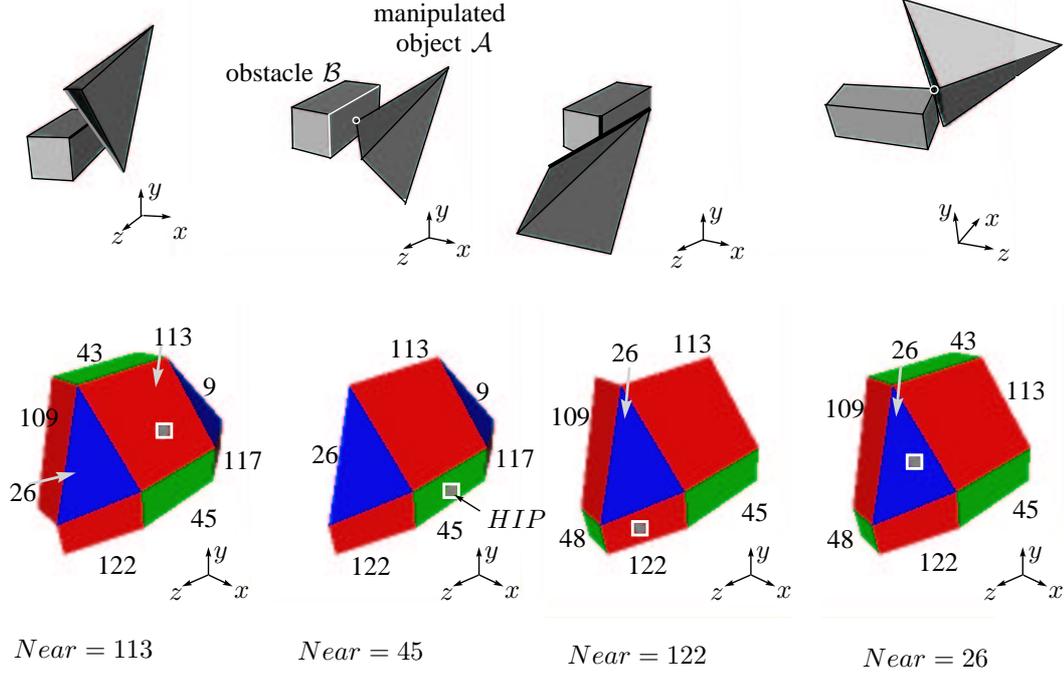


Fig. 2. Snapshots of a translation motion of the manipulated object both in physical space (top) and C -space (bottom). Only the C -faces of the nodes of each local subgraph \mathcal{G}_L^{AB} are shown.

When the orientation of \mathcal{A} changes, $\mathcal{G}_{\Theta_0^A}^{AB}$ may change (and thus \mathcal{G}_L^{AB}) since some basic contacts may no longer be possible (i.e. their applicability condition may no longer hold) and others may become possible. Moreover, if the position of \mathcal{A} changes, \mathcal{G}_L^{AB} may change since the nearest basic contact(s) may change. The update of \mathcal{G}_L^{AB} is tackled in Section III, as well as the collision detection test, which must be applied to the basic contacts of \mathcal{G}_L^{AB} .

III. CONTACT TRACKING / COLLISION DETECTION

Each time the position and/or orientation of \mathcal{A} changes the following steps are executed for each \mathcal{CO}_B in order to recompute the corresponding subgraphs \mathcal{G}_L^{AB} , and perform the collision detection test:

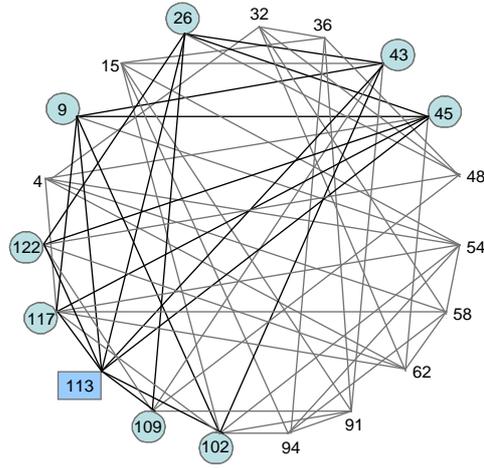
- 1) Verify the applicability condition at the \mathcal{G} -neighbors of the basic contact(s) of $Near$. Group those satisfying it in a set called $N1$.
- 2) Verify the applicability condition at the basic contact(s) of $Near$.
 - a) If it is satisfied set \mathcal{G}_L^{AB} with the the basic contact(s) of $Near$ and those of $N1$.
 - b) If it is not satisfied verify the applicability condition at the \mathcal{G} -neighbors of the basic contact(s) of $N1$. Group those satisfying it in a set called $N2$. Set \mathcal{G}_L^{AB} with the the basic contacts of $N1$ and those of $N2$.
- 3) Compute the distance from the current position x_0^A to the supporting planes $f(x^A, \Theta_0^A) = 0$ of the C -faces represented by the nodes of \mathcal{G}_L^{AB} . If all distances are negative then x_0^A is inside the C -obstacle and a collision is taking place.
- 4) Set $Near$ with the nearest node(s) found in step (3).

As an example Figure 2 shows four snapshots of a translation motion of the manipulated object both in physical space and C -space. It can be seen how the nearest C -face and the local graph \mathcal{G}_L^{AB} change. The graph \mathcal{G}^{AB} of this simple assembly task is composed of 128 nodes (\mathcal{A} is composed of 4 faces, 4 vertices and 6 edges and \mathcal{B} is composed of 6 faces, 8 vertices and 12 edges). The applicability subgraph $\mathcal{G}_{\Theta_0^A}^{AB}$ for the orientation used in Figure 2 has 19 nodes. It is shown in Figure 3, together with the local subgraph \mathcal{G}_L^{AB} for the position of the first snapshot of Figure 2, that is composed of 9 nodes.

IV. HAPTIC RENDERING

A. Single basic contacts

The situation where the manipulated object \mathcal{A} is interfering with an obstacle \mathcal{B} is represented in C -space by q_0^A being inside \mathcal{CO}_B . This situation, as commented in the previous section, is detected by verifying that all the distances from x_0^A to



$$Near = 113$$

$$\mathcal{G}_L^{AB} = \{9, 26, 43, 45, 102, 109, 113, 117, 122\}$$

Fig. 3. Applicability subgraph $\mathcal{G}_{\Theta_0}^{AB}$ and local subgraph \mathcal{G}_L^{AB} for the first snapshot of Figure 2.

the planes of the \mathcal{C} -faces of the nodes of \mathcal{G}_L^{AB} are negative. If $Near$ contains only one \mathcal{C} -face, a single basic contact is taking place. The reaction force and torque is then computed as follows. Let:

- The *HIP*, or *haptic interface point*, be x_0^A .
- Π_{Near} be the plane in \mathcal{C} -space that contains the \mathcal{C} -face of $Near$.
- The *SCP*, or *surface contact point*, be the point over Π_{Near} which is nearest to the *HIP* when the *HIP* is inside \mathcal{CO}_B .
- \vec{n}_{Near} be the outward normal to Π_{Near} .
- *depth* be the (unsigned) distance from the *HIP* to Π_{Near} .

The *HIP* is changed by the user as he moves the probe attached to the manipulated object. When there is no contact, the *SCP* is set coincident to the *HIP*, otherwise when the *HIP* is inside a \mathcal{C} -obstacle, the *SCP* is computed as:

$$SCP = HIP + \vec{n}_{Near} \cdot depth \quad (1)$$

Then, the reaction force is computed proportional to the distance between the *HIP* and the *SCP*:

$$\vec{F} = (SCP - HIP)k_F \quad (2)$$

where k_F represents the elasticity of the objects involved in the contact.

For the computation of the reaction torque, the contact point, App , where the force is applied must be computed. For a single basic contact, it coincides with the contact vertex (for type-A or type-B basic contacts) or with the point where edges cross (for type-C basic contacts). Then, the reaction torque is computed as:

$$\vec{\tau} = \vec{F} \times (x_0^A - App) \quad (3)$$

Note that, since \vec{F} and $\vec{\tau}$ are the reaction force and torque to be fed back to the user, they are computed with respect to the orientation of the fixed reference frame $\mathcal{F}_{\mathcal{W}}$.

B. Multiple contacts between two convex polyhedra

Let consider now the face-face and edge-face contact situations³, where a set of K basic contacts between two convex polyhedra take place simultaneously. The (coplanar) \mathcal{C} -faces of these K contacts are the subset of \mathcal{C} -faces of $Near$ that contain the *SCP*. Let:

- The *contact plane* $\Pi_{contact}$ be the plane in physical space that contains the contact face(s).
- The application point App_k be the contact point corresponding to basic contact $k \in 1 \dots K$.
- The *contact region* \mathcal{H} be the convex hull of the application points. It is a segment for edge-face contacts and a convex polygon with K vertices for face-face contacts.
- The *contact reference point* $\pi(x_0^A)$ be the orthogonal projection of x_0^A onto $\Pi_{contact}$.

³note that each of these contact situations can only take place at a unique orientation.

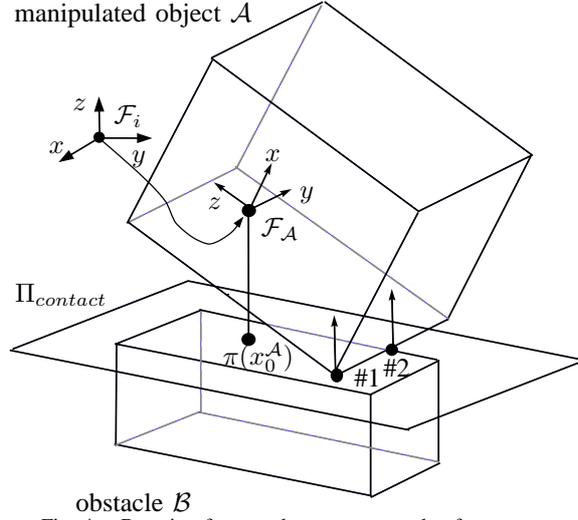


Fig. 4. Reaction force and torque at an edge-face contact.

- The *contact reference frame* \mathcal{F}_i be the orthogonal reference frame associated to a given multi-contact situation i . The origin of \mathcal{F}_i coincides with x_0^A and its orientation is such that the z -axis is normal to $\Pi_{contact}$, the x -axis is parallel to the contact edge (for edge-face contacts) or parallel to the nearest edge of \mathcal{H} (for face-face contacts), and the y -axis is orthogonal to the xz -plane and its sense is such that it makes \mathcal{F}_i right-handed.
- The rotation matrix \mathbf{R}_i be the matrix that relates the orientation of \mathcal{F}_i with that of \mathcal{F}_W .

The reaction force \vec{F} is computed as in the single contact case, since all the \mathcal{C} -faces involved in the multi-contact situation are coplanar. The reaction torque $\vec{\tau}$ is computed as:

$$\vec{\tau} = \mathbf{R}_i \vec{\tau}_i \quad (4)$$

where $\vec{\tau}_i$ is the torque at \mathcal{F}_i computed as follows. Let the x, y and z components of $\vec{\tau}_i$ be, respectively, τ_i^x , τ_i^y and τ_i^z .

Then, for edge-face contacts:

- $\tau_i^z = 0$ since the reaction force is (by construction of \mathcal{F}_i) in the direction of the z -axis.
- τ_i^x and τ_i^y are computed as follows. Let $\tau_{i,k}^x$ and $\tau_{i,k}^y$ be, respectively, the x and y components of the individual torques produced by the applied force acting at point App_k $k \in 1 \dots K$, and computed as in the single contact case using equation (3). Then:

$$\tau_i^y = \begin{cases} 0 & \text{if } \text{sign}(\tau_{i,k}^y) \neq \text{sign}(\tau_{i,j}^y) \quad k, j \in 1 \dots K \\ \frac{1}{K} \sum_{k=1}^K \tau_{i,k}^y & \text{otherwise} \end{cases} \quad (5)$$

$$\tau_i^x = \frac{1}{K} \sum_{k=1}^K \tau_{i,k}^x \quad (6)$$

And for face-face contacts:

- $\tau_i^z = 0$ since the reaction force is also, by construction, in the direction of the z -axis.
- $\tau_i^x = 0$ and $\tau_i^y = 0$ if $\pi(x_0^A) \in \mathcal{H}$; otherwise, they are computed as in the edge-face case using equations (5) and (6).

Figure 4 shows an example where the multi-contact situation i is an edge-face contact between the manipulated object \mathcal{A} and an obstacle \mathcal{B} . This is a two-contact situation composed of a type-B basic contact (point #1) and a type-C basic contact (point #2). The x -axis of \mathcal{F}_i is parallel to the contact edge and the z -axis is normal to the contact face. For the configuration shown in the figure, there is a positive torque around the x -axis of \mathcal{F}_i and a null torque around the y -axis, since $\tau_{i,1}^y < 0$ and $\tau_{i,2}^y > 0$.

Figure 5 (left) shows an example where the multi-contact situation i is a face-face contact between the manipulated object \mathcal{A} and an obstacle \mathcal{B} . This is a multi-contact situation composed of two type-C basic contacts (points #2 and #4) a type-A basic contact (point #1) and a type-B basic contact (point #3). Since $\pi(x_0^A)$ is next to the edge of \mathcal{H} between points #1 and #2, this edge determines the direction of the x -axis of \mathcal{F}_i . In this example $\pi(x_0^A) \notin \mathcal{H}$ and therefore the torque is non-null in the x -direction (and is null in the y -direction since $\tau_{i,2}^y > 0$ and $\tau_{i,4}^y < 0$). In Figure 5 (right), on the contrary, $\pi(x_0^A)$ lies inside \mathcal{H} and the torque is zero.

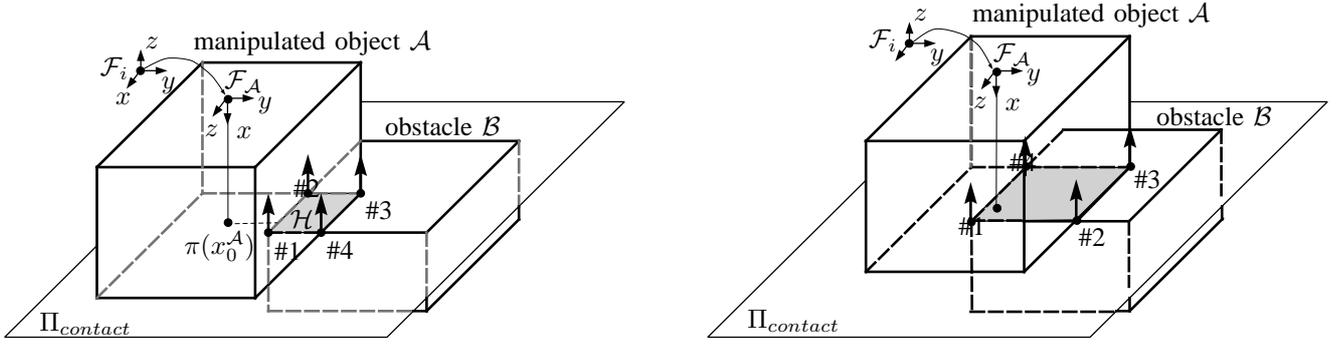


Fig. 5. Reaction force and torque at a face-face contact. Left: non-zero torque example ($\tau_i^x \neq 0$); Right: zero torque example.

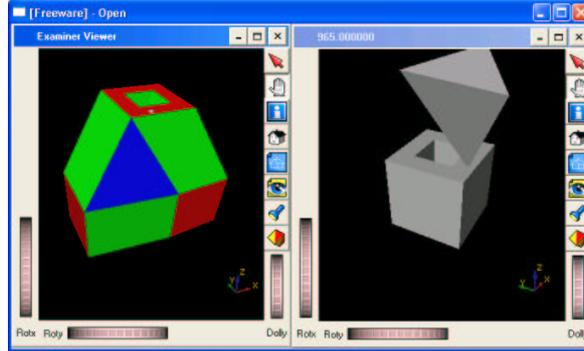


Fig. 6. Interaction between a convex object and a concave one. Left: \mathcal{C} -space, Right: physical workspace.

V. DISCUSSION

This paper copes with the problem of the haptic rendering of compliant motions performed during the execution of virtual assembly tasks. These tasks are normally performed between simple polyhedral objects, and compliant motions maintaining face-face contacts and face-edge contacts are usual. In order to obtain a smooth haptic rendering in these situations, the information of the current type of contact taking place is necessary. Taking into account this need, this paper introduces an approach based on the \mathcal{C} -space. Using spatial temporal coherence, a procedure to keep track of the (possible) contacts is presented that allows to consider the \mathcal{C} -space locally.

The developed procedures have been implemented in C++ on a PC computer. The interface has been developed using Qt and OpenInventor. Several simple assembly tasks have been considered. Figure 6 shows a peg-into-hole assembly task (the hole has been previously decomposed into five convex polyhedra). A 6 d.o.f. Phantom haptic device has been used for the experiments (Figure 7). First results show good performance in comparison to standard haptic rendering methods. Efforts are now directed towards optimizing the code and reducing the execution time in order to cope with more complex objects.

APPENDIX

This appendix explains how the neighboring relationship of the nodes of the graph \mathcal{G}^{AB} that captures the topology of a \mathcal{C} -obstacle is obtained. Let F , E and V represent, respectively, a face, an edge and a vertex of a polyhedron, and let the following neighborhood operators be defined as:

- $N_v(F)$: gives the vertices of face F .
- $N_e(F)$: gives the edges of face F .
- $N_f(F)$: gives the faces that contain an edge of $N_e(F)$, excluding F .
- $N_f(V)$: gives the faces that contain vertex V .
- $N_e(V)$: gives the edges that contain vertex V .
- $N_v(V)$: gives the vertices of the edges of $N_e(V)$, excluding V .
- $N_e(E)$: gives the edges that share a vertex with edge E .

Then, the procedure to compute the arcs of \mathcal{G}^{AB} has the following steps:

- 1) Connect each type-A basic contact (F_A, V_B) with type-A basic contacts $(N_f(F_A), V_B)$ and $(F_A, N_v(V_B))$, with type-B basic contacts $(N_v(F_A), N_f(V_B))$, and with type-C basic contacts $(N_e(F_A), N_e(V_B))$.

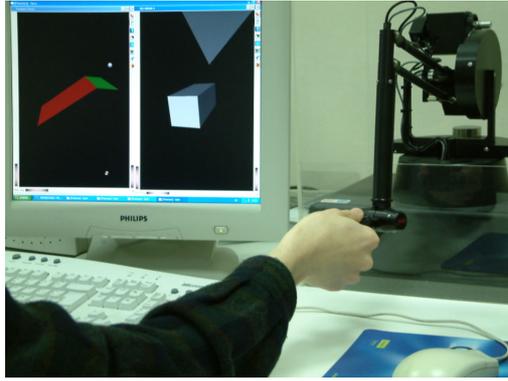


Fig. 7. Haptic interaction with the virtual assembly task using the 6 d.o.f. Phantom haptic device.

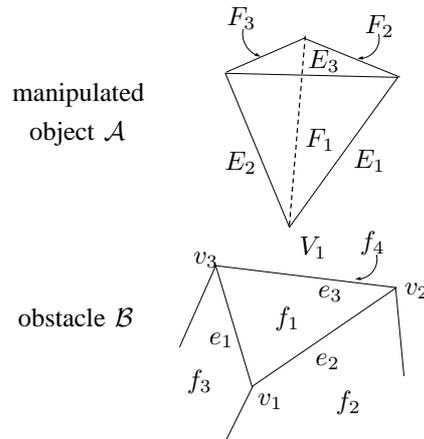


Fig. 8. Type-B basic contact (V_1, f_1) . Some of the basic contacts that neighbor on (V_1, f_1) are (V_1, f_2) , (E_1, e_1) and (F_1, v_3) .

- 2) Connect each type-B basic contact (V_A, F_B) with type-B basic contacts $(N_v(V_A), F_B)$ and $(V_A, N_f(F_B))$, and with type-C basic contacts $(N_e(V_A), N_e(F_B))$.
 - 3) Connect each type-C basic contact (E_A, E_B) with type-C basic contacts $(E_A, N_e(E_B))$ and $(N_e(E_A), E_B)$.
- As an example Figure 9 shows some of the \mathcal{G} -neighbors of a type-B basic contact.

REFERENCES

- [1] B.R. Donald. On motion planning with six degrees of freedom: Solving the intersection problems in configuration space. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 536–541, 1985.
- [2] D. Garroway. 6dof haptic rendering using geometric algebra. In *Proc. of the Int Workshop on Haptic Virtual Environments and Their Applications*, pages 103–107, 2002.
- [3] E.G. Gilbert, D.W. Johnson, and S.S. Keerth. A fast procedure for computing the distance between complex objects in three-dimensional space. *IEEE Tran. on Robotics and Automation*, 4(2):193–203, 1998.
- [4] A. Gregory, A. Mascarenhas, S. Ehmann, M. Lin, and D. Manocha. Six degree-of-freedom haptic display of polygonal models. In *Proc. on Visualization*, pages 139–146, 2000.
- [5] I. Ivanisevic and V. Lumelsky. Augmenting human performance in motion planning tasks - the configuration space approach. In *Proc. of the IEEE Int. Conf. on Robotics and Autom.*, pages 2649–2654, 2001.
- [6] M.C. Lin and J.F. Canny. A fast algorithm for incremental distance calculation. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, pages 1008–1014, 1991.
- [7] T. Lozano-Pérez. Spatial planning: A configuration space approach. *IEEE Trans. Comput.*, 32(2):108–120, 1983.
- [8] W. McNeely, K. Puterbaugh, and J. Troy. Six degree-of-freedom haptic rendering using voxel sampling. In *Proc. of ACM SIGGRAPH*, pages 401–408, 1999.
- [9] B. Mirtich. V-clip: fast and robust polyhedral collision detection. *ACM Transaction on Graphics*, 17(3):177–208, July 1998.
- [10] K. Morishige and H. Noborio. A tele-operation support system with the help of dual views of cartesian and configuration space. In *Proc. of the IEEE Int. Workshop on Robot and Human Interactive Communication*, pages 382–387, 2000.
- [11] J. Rosell and P. Iñiguez. A hierarchical and dynamic method to compute harmonic functions for constrained motion planning. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 2334–2340, 2002.
- [12] D. C. Ruspini, K. Kolarov, and O. Khatib. The haptic display of complex graphical environments. In *Computer Graphics Proceeding, Annual Conference Series*, pages 345–352, 1997.
- [13] I. Vazquez and J. Rosell. Using configuration space for the haptic rendering of assembly tasks. In *Proc. of the World Automation Congress*, 2004.
- [14] S. You and J. Xiao. Haptic display of contact states. In *Proc. of the 4th. World Congress on Intelligent Control and Automation*, pages 2854–2859, 2002.