Proceedings of the 2005 IEEE
International Conference on Robotics and Automation
Barcelona, Spain, April 2005

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

Jan Rosell and Israel Vázquez
Institute of Industrial and Control Engineering (IOC-UPC)
Barcelona, Spain. Email: jan.rosell@upc.edu

***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.**

***Index Terms*— Virtual Assembly, Haptic Rendering, Compliant Motions.**

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 [1].

Collision detection procedures developed for computer graphics are usually used [2]–[5], 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 [6], [7]. 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 [8]), and/or an approximate discretized representation of the obstacles (e.g. as a set of regular small volumes called voxels [9]).

Alternatively, in order to avoid interpenetration and costly interference detection algorithms, other works propose constrained and impulse based methods that, based on separation distances and body velocity/accelelration, compute the forces of eventual contact space constraints [10].

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 [11]. These authors compute the reaction forces and torques once the contact situation has been identified as a collection of principal contacts[1]. 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). Fig. 1 shows the proposed framework, that was first explored in a preliminary version [12], 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 [13].
- 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. [14]).
- 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. [15] [16]).

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

[1] A 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.

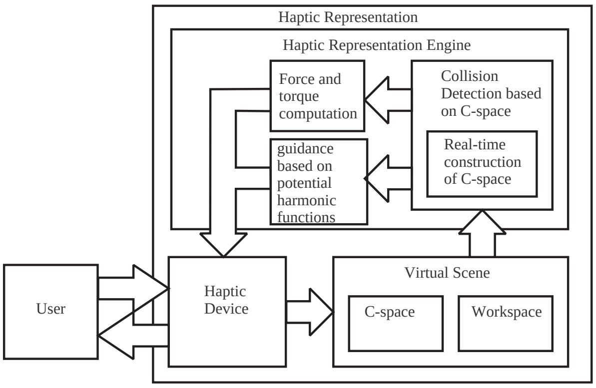

Fig. 1. Haptic rendering of an assembly task using $\mathcal{C}$-space.

The obvious disadvantage of using the $\mathcal{C}$-space is the complexity of its computation. Nevertheless, for haptic rendering purposes and taking into account the spacial and temporal coherence, the $\mathcal{C}$-space only needs to be computed locally (a similar idea in the field of robotic teleoperation is proposed by Cho et al. [17] for planar tasks). 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 $\mathcal{C}$-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 summarizes 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 $\mathcal{F}_\mathcal{A}$ be reference frame attached to $\mathcal{A}$ and $\mathcal{F}_\mathcal{W}$ be the fixed reference frame of the workspace. Let $q^\mathcal{A} = (x^\mathcal{A}, \Theta^\mathcal{A})$ be a configuration of $\mathcal{A}$, where $x^\mathcal{A}$ and $\Theta^\mathcal{A}$ describe, respectively, the position and the orientation of $\mathcal{F}_\mathcal{A}$ with respect to $\mathcal{F}_\mathcal{W}$. Let also $q_0^\mathcal{A} = (x_0^\mathcal{A}, \Theta_0^\mathcal{A})$ be the current configuration of $\mathcal{A}$.

The Configuration Space ($\mathcal{C}$-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 $\mathcal{C}$-obstacle $\mathcal{CO}_\mathcal{B}$. The border of $\mathcal{CO}_\mathcal{B}$ is composed of $\mathcal{C}$-faces, each $\mathcal{C}$-face being the subset of configurations of a five-dimensional hyper-surface $f(q^\mathcal{A}) = 0$ 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 [18].

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

A. Modelling of $\mathcal{C}$-obstacles

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

Each $\mathcal{C}$-obstacle $\mathcal{CO}_\mathcal{B}$ will be represented as a graph, $\mathcal{G}^{\mathcal{A}B}$, that captures its topology. The nodes of $\mathcal{G}^{\mathcal{A}B}$ 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 $\mathcal{G}^{\mathcal{A}B}$ are called $\mathcal{G}$-neighbors.

Let the distance from $x_0^\mathcal{A}$ to a $\mathcal{C}$-face i be the distance between $x_0^\mathcal{A}$ and the plane that contains the $\mathcal{C}$-face for the orientation $\Theta_0^\mathcal{A}$, i.e. plane $f_i(x^\mathcal{A}, \Theta_0^\mathcal{A}) = 0$. Then, the subset of nodes of $\mathcal{G}^{\mathcal{A}B}$ corresponding to the set of $\mathcal{C}$-faces that are nearest to $x_0^\mathcal{A}$ form a subgraph called *near subgraph*, $\mathcal{G}^{\mathcal{A}B}_{Near}$. Note that, since $\mathcal{A}$ and $\mathcal{B}$ are convex, $\mathcal{G}^{\mathcal{A}B}_{Near}$ will contain only one node except for the orientations where face-face contacts or edge-face contacts are possible (and in those cases the $\mathcal{C}$-faces of the nodes of $\mathcal{G}^{\mathcal{A}B}_{Near}$ are coplanar). The distance from $x_0^\mathcal{A}$ to the node(s) of $\mathcal{G}^{\mathcal{A}B}_{Near}$ is called D_{Near}. The nodes of $\mathcal{G}^{\mathcal{A}B}_{Near}$ and their $\mathcal{G}$-neighbors form a subgraph called *neighbor subgraph*, $\mathcal{G}^{\mathcal{A}B}_{Neigh}$.

Due to the spatial and temporal coherence, i.e. due to the fact that the $\mathcal{C}$-space changes very slightly around $q_0^\mathcal{A}$ for a motion of $\mathcal{A}$ between two consecutive instants of time, only those nodes of $\mathcal{G}^{\mathcal{A}B}_{Neigh}$ that are nearest to $q_0^\mathcal{A}$ and that satisfy the applicability condition are to be considered. This subset is called *local applicability subgraph*, $\mathcal{G}^{\mathcal{A}B}_{L,\Theta_0}$.

When the orientation of $\mathcal{A}$ changes, $\mathcal{G}^{\mathcal{A}B}_{L,\Theta_0}$ may change 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, $\mathcal{G}^{\mathcal{A}B}_{L,\Theta_0}$ may change since the nearest basic contact(s) may change. The update of $\mathcal{G}^{\mathcal{A}B}_{L,\Theta_0}$ is tackled in Section III, as well as the collision detection test, which is applied to the nodes of $\mathcal{G}^{\mathcal{A}B}_{L,\Theta_0}$.

III. Contact Tracking / Collision Detection

Each time the configuration of $\mathcal{A}$ changes from $q_0^\mathcal{A}$ to a new configuration $q^\mathcal{A}_{new} = (x^\mathcal{A}_{new}, \Theta^\mathcal{A}_{new})$, the following steps are executed for each $\mathcal{CO}_\mathcal{B}$ in order to recompute the corresponding subgraph $\mathcal{G}^{\mathcal{A}B}_{L,\Theta_0}$, and perform the collision detection test:

1) Compute the distances d_i (for orientation $\Theta^\mathcal{A}_{new}$) from $x^\mathcal{A}_{new}$ to the supporting planes of the nodes of $\mathcal{G}^{\mathcal{A}B}_{Neigh}$,

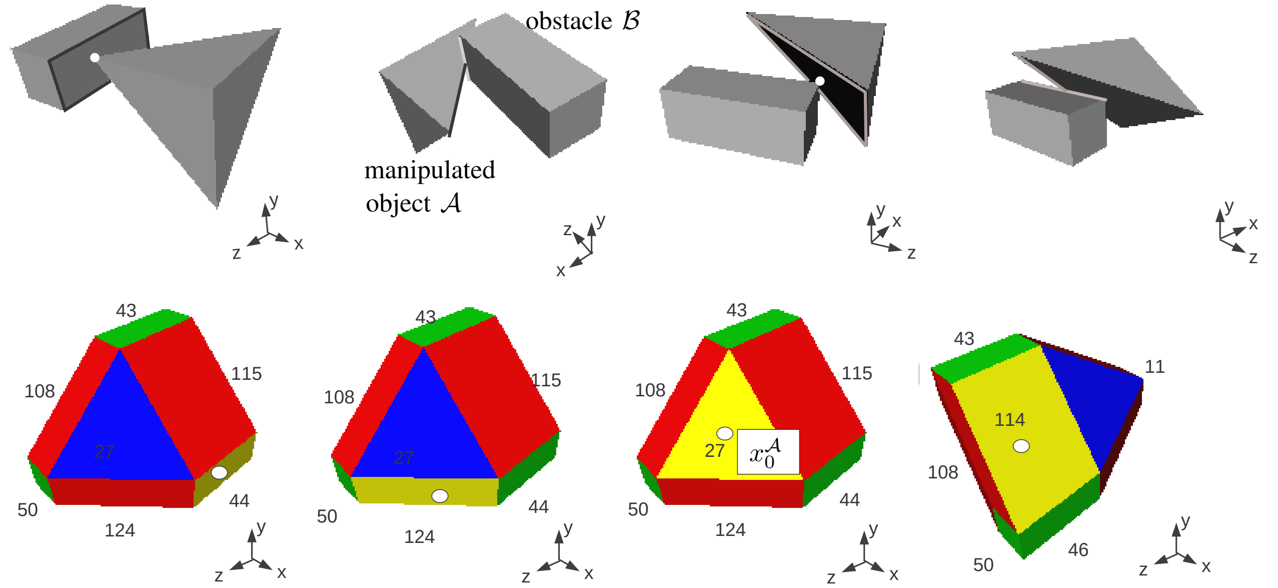

Fig. 2. Snapshots of a motion of the manipulated object both in physical space (top) and $\mathcal{C}$-space (bottom). The $\mathcal{C}$-faces of $\mathcal{G}_{Near}^{AB}$ are highlighted.

i.e. $d_i = f_i(x_{new}^{\mathcal{A}}, \Theta_{new}^{\mathcal{A}})$. Since the $\mathcal{C}$-obstacles are convex, the nearest $\mathcal{C}$-face is the one that has the greater value of d_i [19].

2) Set $\mathcal{G}_{L,\Theta_0}^{AB}$ with all the nodes of $\mathcal{G}_{Neigh}^{AB}$ whose distance d_i satisfies $|D_{Near} - d_i| < \delta$ and that satisfy the applicability condition (within some tolerance values). This step is performed to prune all the nodes of $\mathcal{G}_{Neigh}^{AB}$ whose $\mathcal{C}$-faces are, due to the spatial and temporal coherence assumed, too far from $x_0^{\mathcal{A}}$ to become the new nearest $\mathcal{C}$-face. The value of δ is small and related to $|x_{new}^{\mathcal{A}} - x_0^{\mathcal{A}}|$.
3) Order the nodes of $\mathcal{G}_{L,\Theta_0}^{AB}$ in decreasing order of d_i. Set $\mathcal{G}_{Near}^{AB}$ with the first node and update D_{Near} with its corresponding distance value. Add also to $\mathcal{G}_{Near}^{AB}$ any node whose distance d_i satisfies $|D_{Near} - d_i| < \epsilon$ with ϵ fixed to allow some tolerance in the handling of face-face and face-edge contact situations.
4) A collision is taking place if $D_{near} \leq 0$, i.e. when all the distances are negative.
5) Update $\mathcal{G}_{Neigh}^{AB}$ with the $\mathcal{G}$-neighbors of $\mathcal{G}_{Near}^{AB}$.
6) Update $q_0^{\mathcal{A}}$ with the value of $q_{new}^{\mathcal{A}}$.

As an example, Fig. 2 shows four snapshots of a motion of the manipulated object both in physical space and $\mathcal{C}$-space. The program highlights the $\mathcal{C}$-face(s) of $\mathcal{G}_{Near}^{AB}$ as the user manipulates the object. It can be seen how it changes following the sequence $44 \rightarrow 124 \rightarrow 27 \rightarrow 114$. 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 graph $\mathcal{G}_{Neigh}^{AB}$ ranges from 17 to 72 nodes depending on the number and type of contacts. The graph $\mathcal{G}_{L,\Theta_0}^{AB}$ is composed of less than 5 nodes when $\mathcal{G}_{Near}^{AB}$ has a single node (corresponding to a face-vertex or an edge-edge contact situation); and is composed of less than 10 nodes when $\mathcal{G}_{Near}^{AB}$ has up to four nodes (corresponding to a face-face contact situation).

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 $\mathcal{C}$-space by $q_0^{\mathcal{A}}$ being inside $\mathcal{CO}_{\mathcal{B}}$. This situation, as commented in the previous section, is detected by verifying that all the distances from $x_0^{\mathcal{A}}$ to the planes of the $\mathcal{C}$-faces of the nodes of $\mathcal{G}_{L,\Theta_0}^{AB}$ are negative. If $\mathcal{G}_{Near}^{AB}$ 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 $x_0^{\mathcal{A}}$.
- Π_{Near} be the the plane in $\mathcal{C}$-space that contains the $\mathcal{C}$-face of $\mathcal{G}_{Near}^{AB}$.
- The SCP, or *surface contact point*, be the point over Π_{Near} which is nearest to the HIP when the HIP is inside $\mathcal{CO}_{\mathcal{B}}$.
- $\vec{n}_{Near}$ 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:

$$SCP = HIP + \vec{n}_{Near} \cdot depth \tag{1}$$

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

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

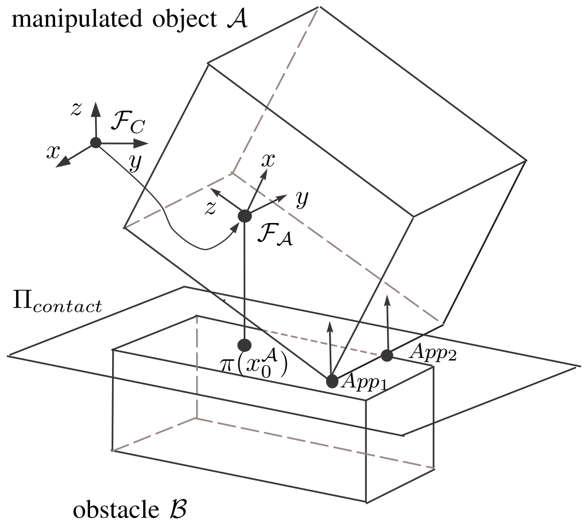

Fig. 3. Reaction force and torque at an edge-face contact.

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:

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

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 situations where a set of K basic contacts between two convex polyhedra take place simultaneously. For any orientation where these contacts situations occur, the $\mathcal{C}$-faces of the involved K basic contacts are coplanar. Let:

- The *contact plane* $\Pi_{contact}$ be the plane in physical space that contains the contact faces.
- 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* $\pi(x_0^\mathcal{A})$ be the orthogonal projection of $x_0^\mathcal{A}$ onto $\Pi_{contact}$.
- The *contact reference frame* $\mathcal{F}_C$ be the orthogonal reference frame associated to a given multi-contact situation C. The origin of $\mathcal{F}_C$ coincides with $x_0^\mathcal{A}$ 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 $\mathcal{F}_C$ right-handed.
- The rotation matrix $\mathbf{R}$ be the matrix that relates the orientation of $\mathcal{F}_C$ with that of $\mathcal{F}_\mathcal{W}$.

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:

$$\vec{\tau} = \mathbf{R}\vec{\tau}_C \tag{4}$$

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

Then, for edge-face contacts:

- $\tau_C^z = 0$ since the reaction force is (by construction of $\mathcal{F}_C$) in the direction of the z-axis.
- τ_C^x and τ_C^y are computed as follows. Let $\tau_{C,k}^x$ and $\tau_{C,k}^y$ 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 (3). Then:

$$\tau_C^y = \begin{cases} 0 \text{ if } \operatorname{sign}(\tau_{C,k}^y) \neq \operatorname{sign}(\tau_{C,j}^y) \;\; k,j \in 1 \dots K \\ \frac{1}{K}\sum_{k=1}^{K} \tau_{C,k}^y \text{ otherwise} \end{cases} \tag{5}$$

$$\tau_C^x = \frac{1}{K}\sum_{k=1}^{K} \tau_{C,k}^x \tag{6}$$

And for face-face contacts:

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

Fig. 3 shows an example where the multi-contact situation C 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 App_1) and a type-C basic contact (point App_2). The x-axis of $\mathcal{F}_C$ 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 $\mathcal{F}_C$ and a null torque around the y-axis, since $\tau_{C,1}^y < 0$ and $\tau_{C,2}^y > 0$.

Fig. 4 (left) shows an example where the multi-contact situation C 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 App_2 and App_4) a type-A basic contact (point App_1) and a type-B basic contact (point App_3). Since $\pi(x_0^\mathcal{A})$ is next to the edge of $\mathcal{H}$ between points App_1 and App_2, this edge determines the direction of the x-axis of $\mathcal{F}_C$. In this example $\pi(x_0^\mathcal{A}) \notin \mathcal{H}$ and therefore the torque is non-null in the x-direction (and is null in the y-direction since $\tau_{C,2}^y > 0$ and $\tau_{C,4}^y < 0$). In Fig. 4 (right), on the contrary, $\pi(x_0^\mathcal{A})$ lies inside $\mathcal{H}$ and the torque is zero.

Fig. 5 show the reaction forces and torques obtained from the contacts of Fig. 4 (left) when they are consecutively attained starting from a no-contact situation. The computation times (including the contact tracking) always ranged from 0.1 to 0.6 ms.

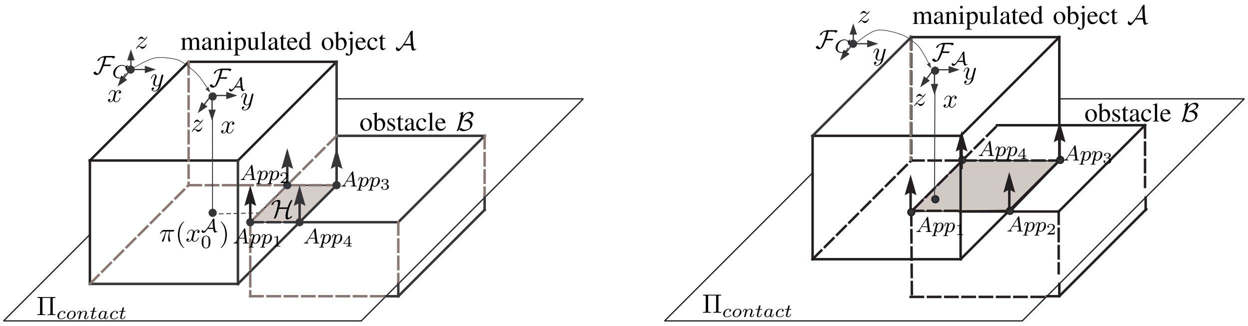

Fig. 4. Reaction force and torque at a face-face contact. Left: non-zero torque example ($\tau_i^x \neq 0$); Right: zero torque example.

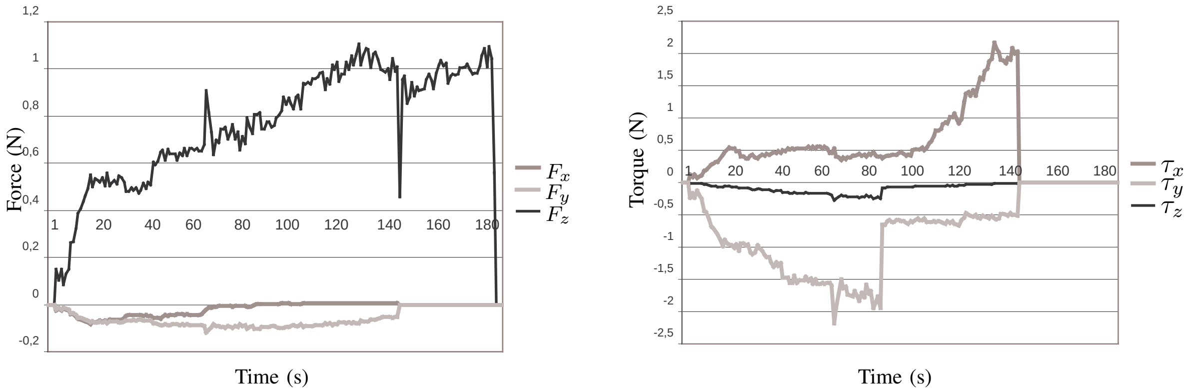

Fig. 5. Reaction forces (left) and torques (right) during the motion from a no-contact situation to attain the face-face contact situation of Fig. 4 (left), following the path $App_1 \rightarrow (App_1, App_2) \rightarrow (App_1, App_2, App_3, App_4)$.

V. Summary

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 an 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. Fig. 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 (Fig. 7). First results show good performance in comparison to standard haptic rendering methods. Efforts are now directed towards improving the control algorithm by considering a control law based on virtual passive coupling [20], and towards considering the contacts between several objects.

Appendix

This appendix explains how the neighboring relationship of the nodes of the graph $\mathcal{G}^{\mathcal{A}B}$ 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 $\mathcal{G}^{\mathcal{A}B}$ has the following steps:

1) Connect each type-A basic contact $(F_\mathcal{A}, V_\mathcal{B})$ with type-A basic contacts $(N_f(F_\mathcal{A}), V_\mathcal{B})$ and $(F_\mathcal{A}, N_v(V_\mathcal{B}))$, with type-B basic contacts $(N_v(F_\mathcal{A}), N_f(V_\mathcal{B}))$, and with type-C basic contacts $(N_e(F_\mathcal{A}), N_e(V_\mathcal{B}))$.
2) Connect each type-B basic contact $(V_\mathcal{A}, F_\mathcal{B})$ with type-B basic contacts $(N_v(V_\mathcal{A}), F_\mathcal{B})$ and

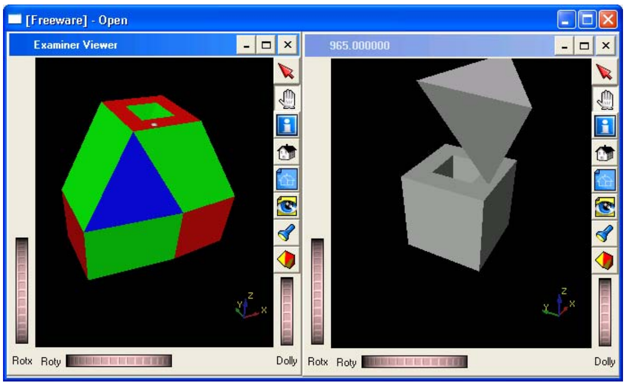

Fig. 6. Interaction between objects during a virtual assembly task, represented both in $\mathcal{C}$-space and in physical space.

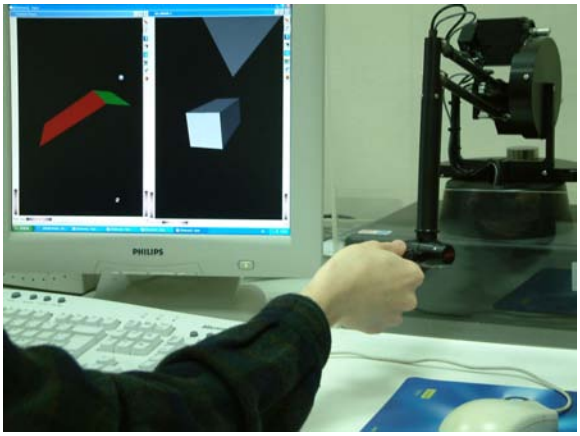

Fig. 7. Haptic interaction using the 6 d.o.f. Phantom haptic device.

$(V_{\mathcal{A}}, N_f(F_{\mathcal{B}}))$, and with type-C basic contacts $(N_e(V_{\mathcal{A}}), N_e(F_{\mathcal{B}}))$.

3) Connect each type-C basic contact $(E_{\mathcal{A}}, E_{\mathcal{B}})$ with type-C basic contacts $(E_{\mathcal{A}}, N_e(E_{\mathcal{B}}))$ and $(N_e(E_{\mathcal{A}}), E_{\mathcal{B}})$.

As an example Fig. 8 shows some of the $\mathcal{G}$-neighbors of a type-B basic contact.

REFERENCES

[1] M. A. Srinivasan and C. Basdogan, "Haptics in virtual environments: Taxonomy, research status, and challenges," *Computer and Graphics*, vol. 21, no. 4, pp. 393–404, 1997.

[2] B. Mirtich, "V-clip: fast and robust polyhedral collision detection," *ACM Transaction on Graphics*, vol. 17, no. 3, pp. 177–208, 1998.

[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 Trans. on Robotics and Automation*, vol. 4, no. 2, pp. 193–203, 1998.

[4] M. C. Lin and J. F. Canny, "A fast algorithm for incremental distance calculation," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pp. 1008–1014, 1991.

[5] M. C. Lin, D. Manocha, and M. K. Ponamgi, "Fast algorithms for penetration and contact determination between non-convex polyhedral models," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, vol. 3, pp. 2707–2712, 1995.

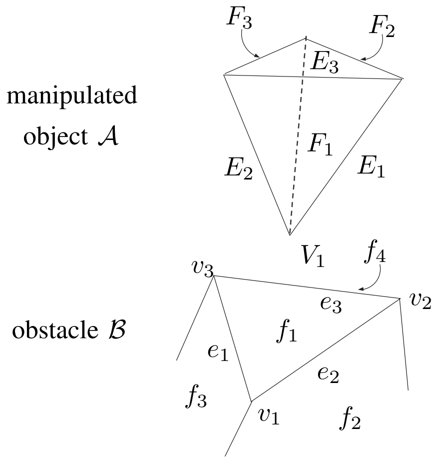

Fig. 8. Type-B basic contact (V_1, f_1). Some of the basic contacts that neighbor on (V_1, f_1) are (V_1, f_2), (E_1, e_1) and (F_1, v_3).

[6] D. C. Ruspini, K. Kolarov, and O. Khatib, "The haptic display of complex graphical environments," in *Proc. of the 24th annual Conf. on Computer graphics and Interactive Techniques*, pp. 345–352, 1997.

[7] A. Gregory, A. Mascarenhas, S. Ehmann, M. Lin, and D. Manocha, "Six degree-of-freedom haptic display of polygonal models," in *Proc. on Visualization*, pp. 139–146, 2000.

[8] D. Garroway, "6dof haptic rendering using geometric algebra," in *Proc. of the Int Workshop on Haptic Virtual Environments and Their Applications*, pp. 103–107, 2002.

[9] W. McNeely, K. Puterbaugh, and J. Troy, "Six degree-of-freedom haptic rendering using voxel sampling," in *Proc. of ACM SIGGRAPH*, pp. 401–408, 1999.

[10] B. Chang and J. E. Colgate, "Real-time impulse-based simulation of rigid body systems for haptic display," in *Proc. of the ASME Int. Mechanical Engineeering Congress and Exhibition*, 1997.

[11] S. You and J. Xiao, "Haptic display of contact states," in *Proc. of the 4th. World Congress on Intelligent Control and Automation*, pp. 2854–2859, 2002.

[12] I. Vázquez and J. Rosell, "Using configuration space for the haptic rendering of assembly tasks," in *Proc. of the World Automation Congress*, 2004.

[13] T. Lozano-Pérez, "Spatial planning: A configuration space approach," *IEEE Trans. Comput.*, vol. 32, no. 2, pp. 108–120, 1983.

[14] 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*, pp. 2334–2340, 2002.

[15] 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.*, pp. 2649–2654, 2001.

[16] K. Morishige and H. Nobioro, "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*, pp. 382–387, 2000.

[17] Y.-J. Cho, T. Kotoku, K. Komoriya, and K. Tanie, "A discrete event approach to the automatic command generation in a teleprogramming systems," *5th IEEE Int. Workshop on Robot and Human Communication*, pp. 238–243, Nov 1996.

[18] 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*, pp. 536–541, 1985.

[19] J. Rosell and I. Vázquez, "Contact tracking in configuration space for haptic rendering purposes." unpublished. Submitted to IEEE Int. Symp. on Assembly and Task Planning - July 2005.

[20] R. J. Adams and B. Hannaford, "Control law design for haptic interfaces to virtual reality," *IEEE Trans. on Control Systems Technology*, vol. 10, no. 1, pp. 3–13, 2002.