The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

Haptic Primitives Guidance Based on the Kautham Path Planner

Carlos Vázquez, Jan Rosell, Luciano Chirinos and Omar A. Domínguez

Abstract— **Path planning methods are well suited to automatically perform robotic tasks; haptic guidance is a powerful tool for disabled people rehabilitation, sports training, handcraft skills acquiring and such kind of enactive tasks. This paper proposes a novel an efficient method to accomplish guided movements by efficiently combing path planning methods and haptic guidance.**

The main contribution of this paper is the development of a reliable method to haptically guide the user within a virtual robotic task, by means of: on-line, collision free, path planning generated trajectories. To accomplish haptic guidance three stages must be correctly done: the first one where the user selects the desired objects and obstacles within a virtual environment to set up the task, the second one, where a path planner looks for the task trajectory and, the third one, where the user is constrained to on-line generated local channels and solution paths that offer a reliable path guidance system to achieve the task movements.

I. Introduction

Path planning methods are well suited to automatically perform robotic tasks [1]. Haptic Guidance is commonly used for training and rehabilitation, mostly within virtual environments [2]. From the combination of path planning and haptic devices two approaches can be distinguished: the first one, where the user aids the path planner to get a better path (aided path planning), for example, by improving a Road Map by adding free samples in difficult $\mathcal{C}$-space zones [3]. The second one is about the use of a path planner to find a trajectory that is subsequently used to guide the user by means of a haptic device. Path planning haptic guidance applications can be found, for example, in CAD and assembly tasks guidance [4], [5], [6] or to help users to guide virtual vehicles [7]. In the field of nanotechnology, potential field path planners have been used to position virtual reality nanoparticles using a haptic device [8], [9].

The main contribution of this work is the development of a reliable method based on haptic primitives to feedback the user into the correct movement to achieve a selected task without collisions. Our proposed methodology is able to construct the haptic guidance trajectories on the fly avoiding the need to have pre-calculated trajectories as some other haptic guidance methods do [10].

C. Vázquez and L. Chirinos are with the Mechatronics Automotive Research Center (CIMA), ITESM Toluca Campus, México. carlos.vazquez.hurtado@itesm.mx luciano.chirinos@itesm.mx

J. Rosell is with the Institute of Industrial and Control Engineering - Technical University of Catalonia, Barcelona, Spain. jan.rosell@upc.edu

O. Domínguez is with the ITESM Hidalgo Campus, México. omar.dominguez@itesm.mx

The proposed methodology has three stages: 1) the user selects the desired free-flying robot and the obstacles to use them within a virtual environment to set up the task. A XML file is used to set up the robots/obstacles configuration and the GNU/GPL C++ libraries Qt4 and Coin3D are used to create a Computer Graphic Interface to load those 3D models. 2) Our hierarchical $\mathcal{C}$-space decomposition, deterministic sampling, harmonic-function-based path planner, named Kautham Planner, looks for a collision free channel path connecting the start and goal configuration to complete the task. 3) the user is constrained to some haptic primitives conforming a reliable path guidance system to achieve collision free task movements, this is done using the OpenHaptics library [11].

This paper is structured as follows: In section II a brief explanation of the Kautham path planner is presented. In sections III and IV there is a detailed description of the guiding components. In section V a detailed description of the novel discrete method to obtain the haptic guiding forces, based on haptic primitives is presented. In section VI some results and an application example is developed, finally in section VII, some conclusions are given.

II. The Kautham Path Planner

The Kautham Path Planner was initially a $\mathbb{R}^2$ path planner based on Harmonic Functions and Probabilistic Cell Decomposition whose output was a collision free channel of cells [12]. The planner evolved to support $\mathbb{R}^2$, $\mathbb{R}^3$, $SO(n)$ and $SE(3)$ topologies based on: 1) Deterministic sampling sequence 2) Hierarchical $\mathcal{C}$-space decomposition, and 3) Harmonic functions and the solution channel was improved by considering distances [13], [14]. The Kautham Planner was then used to haptically guide a user on the execution of teleoperated assembly tasks, using a simple force-generation pattern based on balls [15].

The Kautham path planner is based on a deterministic sampling sequence because it offers an uniform and incremental coverage of the $\mathcal{C}$-space, within a latice structure; then a hierarchical cell decomposition of the $\mathcal{C}$-space is iteratively performed using Harmonic functions to get a collision-free channel of cells connecting the cell containing the initial configuration sample and the cell containing the goal configuration sample.

That means, the $\mathcal{C}$-space is hierarchically decomposed into non-uniform cells to group samples, capturing the $\mathcal{C}$-space structure. Cells are not just classified as free or forbidden as in traditional methods, instead, our method describes the cells using a transparency parameter dependent on the number of free and forbidden samples in the cell. This transparency parameter is used to control the collision detection and the cell partition procedures.

Collision detection procedure is used in a lazy evaluation manner, so that only few samples are collision checked. A cell partition procedure is used to create a cell division only if the cell is not homogeneous enough. The harmonic function is computed using the successive over-relaxation method (S.O.R) to guide hierarchical cell decomposition procedure.

The solution channel is obtained by following the negated gradient of the harmonic function H_1. Starting at the initial cell, the next cell is iteratively chosen among the neighbors such that it has the lowest H_1 value, until the goal cell with H_1 value fixed at U_L is reached. The obtained channel is composed of cells with different transparency values.

III. Local channels

A local channel is a channel of cells that can be used to guide the user from his current position $(\mathbf{u}_q)$ to the target configuration. A definition of the basic motion planning problem can be stated as follows [16]:

Let the world $\mathcal{W}$ be $\mathcal{W} = \mathbb{R}^3$, such world contains an obstacle region $\mathcal{O} \subset \mathcal{W}$, and let $\mathcal{A}$ be a robot $\mathcal{A} \subset \mathcal{W}$. Let $q \in \mathcal{C}$-space be a free configuration from $\mathcal{A}$, such that $q = (x_t, y_t, z_t, h)$ (where h is an unitary quaternion); The forbidden region, $\mathcal{C}_{obs} \subset \mathcal{C}$-space, is defined as:

$$\mathcal{C}_{obs} = \{q \in \mathcal{C}\text{-space} \mid \mathcal{A}(q) \cap \mathcal{O} \neq \emptyset\} \tag{1}$$

$\mathcal{C}_{obs}$ is the set of all configurations q such that $\mathcal{A}(q)$, the transformed robot, intersects the forbidden region. The rest of the $\mathcal{C}$-space configurations are known as the free space configurations, which are defined as $\mathcal{C}_{free} = \mathcal{C}\text{-space} \setminus \mathcal{C}_{obs}$.

For the path planning problem, a free start configuration $q_{ini} \in \mathcal{C}_{free}$ and a free goal configuration $q_{end} \in \mathcal{C}_{free}$ must be defined. Then, the complete path planning algorithm gets a free path $\tau : [0,1] \rightarrow \mathcal{C}_{free}$, where $\tau(0) = q_{ini}$ and $\tau(1) = q_{end}$ or it reports the absence of the free path.

A. Local channel components

Since our approach uses a discrete representation of the $\mathcal{C}$-space based on cells, we can adapt the previous definitions by using sets of cells instead of regions. Our $\mathcal{C}$-space model can be expressed in terms of three sets of cells differentiated about the transparency thresholds $\Delta_{obstacle}$ and $\Delta_{acceptance}$ as shown:

$$\mathcal{C} = \mathcal{C}_{H1} \cup \mathcal{C}_F \cup \mathcal{C}_O \tag{2}$$

where:

- $\mathcal{C}$ is the set of all the cells composing the $\mathcal{C}$-space model.
- $\mathcal{C}_{H1}$ is the set of the cells of the channel found with the Kautham planner. The transparency of these cells is high and satisfies:

$$T_j \geq \Delta_{acceptance} \tag{3}$$

- $\mathcal{C}_O$ is the set of the cells with a transparency lower than $\Delta_{obstacle}$, those cells may contain an obstacle.

$$\mathcal{C}_O = \{c_p \mid T_j < \Delta_{obstacle}\} \tag{4}$$

- $\mathcal{C}_F$ is the set of the free cells with transparency higher than a $\Delta_{obstacle}$, that are not in the set $\mathcal{C}_{H1}$.

$$\mathcal{C}_F = \{c_f \mid c_f \in \mathcal{C} \setminus \mathcal{C}_{H1} \cup \mathcal{C}_O\} \tag{5}$$

Since our haptic guidance method builds the local channel on the fly, the user movement is not restricted to the Kautham planner solution channel $\mathcal{C}_{H1}$ but to any local channel connecting the cell where the user position lies and the target cell. Then, the initial cell $c_{ini} \in \mathcal{C}_{H1} \cup \mathcal{C}_F$ is the cell which contains the initial configuration from where the user starts.

A local solution channel S_G is any ordered set of cells that can be used to guide the user from any cell $c_{ini} \in \mathcal{C} \setminus \mathcal{C}_O$ inside the $\mathcal{C}$-space model to the goal cell c_{end}. A local solution channel S_G can be expressed as the set of adjacent cells a , b, whose potential values are U_a and U_b respectively, ordered as indicated in (6).

$$S_G = (X, \leq) \Rightarrow \forall a, b \in X : a \leq b \Leftrightarrow U_a > U_b \tag{6}$$

where c_{ini} is the lower bound of the ordered set X and c_{end} is the upper bound of the set X. So, the relation $c_{ini} \leq a$, $a \leq b$ and $b \leq c_{end}$ holds. Finally, we know a cell since it will always become our current cell, so, the final task is to find the next cell b.

Let $\mathbf{N}$ be the set of all the neighbor cells of a, and $\mathbf{U}$ the set of the potential values linked to every value of $\mathbf{N}$, then $U_b = \min\{\mathbf{U}\}$ and b is the next cell with potential value U_b. Obtaining the neighbor cell is a procedure that can be sped up if $a \in \mathcal{C}_{H1}$ since in this case only the neighbor cells pertaining to $\mathcal{C}_{H1}$ are considered.

B. Local channel transitions

In the following paragraphs, the way to transit from cell to cell throughout a local channel is presented. To understand the methodology a Petri Net based model is presented as depicted in Fig. 1. The aim of the model is to let the user go from cell to cell in a stable way, that is, avoiding strong magnitude force generation due to the change on the direction of the magnitude force vector.

IV. Local paths

A local path is a path composed by linear segments, connecting the current user configuration u_q to the goal configuration q_{end}, that is $\tau : [\mathbf{u}_q, q_{end}] \rightarrow \mathcal{C}_{H1} \cup \mathcal{C}_F$. This local path is used to keep the user inside the local channel. In the following paragraphs the procedure needed to construct and navigate thorough local paths is presented.

A. Local path segments

In the previpus section we have defined the set $\mathbf{N}$ to be the set of all the current cell c_c neighbor cells and the set $\mathbf{U}$ to be the set of potential values of the cells $c_j \in \mathbf{N}$. Then, the next cell c_n in the guidance sequence is the c_c neighbor cell with a lowest potential value: $c_n = c_j \in \mathbf{N} \mid \min\{\mathbf{U}\} = U_j$.

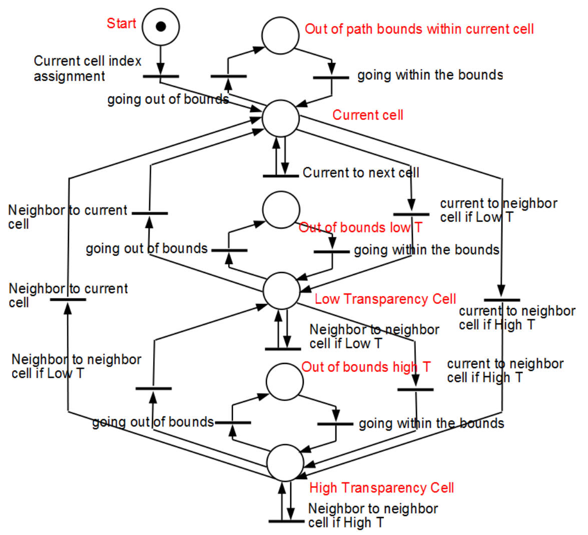


Fig. 1: Petri Net defining the channel navigation. Guiding forces will suggest the user's motions, i.e. they will try to control the firings of the transitions.

The position of each cell is given by its center, such as $\mathbf{p}_c$ is the position of the cell c_c and $\mathbf{p}_n$ is the position of the cell c_n. Fig. 2a shows a pair of neighbor cells where the elements c_c, c_n, $\mathbf{p}_c$ and $\mathbf{p}_n$ are depicted.

The smallest cell c_s between the pair c_c and c_n can be found by comparing the size w_c of c_c and the size w_n of c_n:

$$c_s = \begin{cases} c_n & \text{if } w_n < w_c \\ c_c & \text{if } w_n \geq w_c \end{cases} \tag{7}$$

Let the point $\mathbf{p}_s$ be the position of smallest cell c_s which size is w_s, and let the movement axis $\mathbf{v}_m$ be an unit vector in the $\hat{\mathbf{x}}$, $\hat{\mathbf{y}}$ or $\hat{\mathbf{z}}$ direction. The movement axis $\mathbf{v}_m$ is parallel to one of the coordinate axis and the point $\mathbf{p}_s$ lies on it. To get $\mathbf{v}_m$, the auxiliary vector $\mathbf{v}_d$ that goes from $\mathbf{p}_c$ to $\mathbf{p}_n$ should be obtained first.

$$\mathbf{v}_d = \mathbf{p}_n - \mathbf{p}_c \tag{8}$$

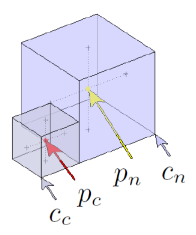
now, from $\mathbf{v}_d = \mathrm{v}_{dx}\hat{\mathbf{x}} + \mathrm{v}_{dy}\hat{\mathbf{y}} + \mathrm{v}_{dz}\hat{\mathbf{z}}$ we can get $\mathbf{v}_m$ by using (9).

$$\mathbf{v}_m = \begin{cases} \frac{\mathbf{v}_{d_x}}{|\mathbf{v}_{d_x}|}\hat{\mathbf{x}} & \text{if } max\left\{|\mathbf{v}_{d_x}|, |\mathbf{v}_{d_y}|, |\mathbf{v}_{d_z}|\right\} = |\mathbf{v}_{d_x}| \\ \frac{\mathbf{v}_{d_y}}{|\mathbf{v}_{d_y}|}\hat{\mathbf{y}} & \text{if } max\left\{|\mathbf{v}_{d_x}|, |\mathbf{v}_{d_y}|, |\mathbf{v}_{d_z}|\right\} = |\mathbf{v}_{d_y}| \\ \frac{\mathbf{v}_{d_z}}{|\mathbf{v}_{d_z}|}\hat{\mathbf{z}} & \text{if } max\left\{|\mathbf{v}_{d_x}|, |\mathbf{v}_{d_y}|, |\mathbf{v}_{d_z}|\right\} = |\mathbf{v}_{d_z}| \end{cases} \tag{9}$$

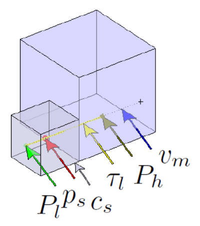
Finally, the local path segment τ_l is the line segment that intersects the point $\mathbf{p}_s$ in the direction of $\mathbf{v}_m$ and whose bounding points are given by the lower bounding point $\mathbf{P}_l$ and upper bounding point $\mathbf{P}_h$, defined below. Then, the local path is continuous in a closed interval $\tau_l : [\mathbf{P}_l, \mathbf{P}_h] \rightarrow \mathcal{C}_{H1} \cup \mathcal{C}_F$.

The lower bounding point can be obtained using (10).

$$\mathbf{P}_l = \begin{cases} \mathbf{p}_c - \left(\frac{w_c}{2}\right) \cdot \mathbf{v}_m & \text{if } \mathbf{p}_c = \mathbf{p}_s \\ \mathbf{p}_s - \left(w_c + \frac{w_s}{2}\right) \cdot \mathbf{v}_m & \text{otherwise} \end{cases} \tag{10}$$



(a) Axes and their boundaries.



(b) Axes and their boundaries.

Fig. 2: Local paths elements.

The upper bounding point can be obtained using (11).

$$\mathbf{P}_h = \begin{cases} \mathbf{p}_c + \left(\frac{w_c + w_n}{2}\right) \cdot \mathbf{v}_m & \text{if } \mathbf{p}_c = \mathbf{p}_s \\ \mathbf{p}_s & \text{otherwise} \end{cases} \tag{11}$$

Fig. 2b depicts the following local path components: c_s, $\mathbf{p}_s$, $\mathbf{v}_m$, τ_l, $\mathbf{P}_l$ and $\mathbf{P}_h$.

B. Local path transitions

In the previous paragraphs, a local path segment with fixed lower ($\mathbf{P}_l$) and upper ($\mathbf{P}_h$) bounds was generated. Nevertheless, our goal is to have a local path connecting the start and goal configurations. Such a local path can be obtained by joining the local path segments, which implies a flexible bound definition. A mechanism to do so consists on keeping the previous cell c_o local path segment until the user enters the current cell c_c, then an auxiliary local path segment may be constructed to connect the previous local path segment and the current one.

Then, to realize correct haptic guidance through local paths three cells are needed: the previous cell c_o, the current cell c_c an the next cell c_n. Furthermore, at least two local paths segments are needed, the previous one τ_{lo} and the current one τ_{lc}.

To guide the user from c_o to c_n, the previous local path τ_{lo} is kept from c_o by enlarging its upper bounding point $\mathbf{P}_{ho}$ to a new point fixed inside the current cell c_c. Due to the fact that cells c_o and c_c could have different size and, distinct configurations between the previous movement axis $\mathbf{v}_{mo}$ and the current movement axis $\mathbf{v}_{mc}$ exist, some auxiliary path segments connecting v_{mo} and v_{mc} may be needed. Finally τ_{lc} is used to guide the user from the current to next cell.

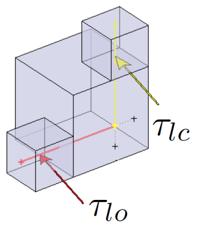
Two basic configurations between the previous movement axis $\mathbf{v}_{mo}$ and the current one $\mathbf{v}_{mc}$ are described bellow:

Perpendicular axes: The axis $\mathbf{v}_{mo}$ and $\mathbf{v}_{mc}$ are perpendicular if the dot product between $\mathbf{v}_{mo}$ and $\mathbf{v}_{mc}$ is zero:

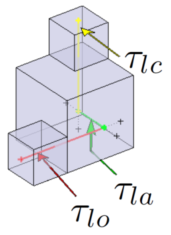
$$\mathbf{v}_{mo} \cdot \mathbf{v}_{mc} = 0 \tag{12}$$

Two subcases may occur:

1) Coplanar perpendicular axes: Fig. 3a shows a coplanar case.
2) Non-coplanar perpendicular axes: this case is depicted in Fig. 3b.



(a) Coplanar perpendicular.



(b) Not-coplanar perpendicular.

Fig. 3: Perpendicular axes cases.

Parallel axes: The axis $\mathbf{v}_{mo}$ and $\mathbf{v}_{mc}$ are parallel if the magnitude of the cross product between $\mathbf{v}_{mo}$ and $\mathbf{v}_{mc}$ is zero:

$$|\mathbf{v}_{mo} \times \mathbf{v}_{mc}| = 0 \tag{13}$$

Then, the first step is to identify the case in which the axes $\mathbf{v}_{mo}$ and $\mathbf{v}_{mc}$ are, to do so, two vectors will be defined.

- The vector $\mathbf{v}_{au}$ between the center of the previous smallest cell center $\mathbf{p}_{so}$ and the center of the current smallest cell $\mathbf{p}_{sc}$

$$\mathbf{v}_{au} = \mathbf{p}_{sc} - \mathbf{p}_{so} \tag{14}$$

- If $\mathbf{v}_{au}$ has zero magnitude, then, we can conclude the axes are collinear parallel. If the product magnitude is different from zero, then a new vector $\mathbf{v}_{um}$ is obtained from the cross product between $\mathbf{v}_{au}$ y $\mathbf{v}_{mc}$.

$$\mathbf{v}_{um} = \mathbf{v}_{au} \times \mathbf{v}_{mc} \tag{15}$$

Now, from those vectors, we can obtain the particular case in which the vectors are by using the two previous equations. The three different cases for parallel axes are:

1) Collinear parallel axes: the axes $\mathbf{v}_{au}$ and $\mathbf{v}_{mc}$ are collinear parallel if the following condition holds:

$$\hat{\mathbf{v}}_{au} - \mathbf{v}_{mc} = 0 \tag{16}$$

This case is depicted in Fig. 4a.
2) Coplanar parallel axes: the axes $\mathbf{v}_{au}$ y $\mathbf{v}_{mc}$ are coplanar parallel if the following condition holds:

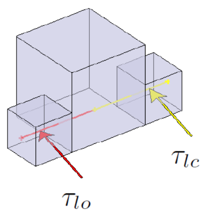
$$|\mathbf{v}_{um}| = 1 \tag{17}$$

This case is depicted in Fig. 4b.
3) Non-coincident parallel axes: The axes $\mathbf{v}_{au}$ and $\mathbf{v}_{mc}$ are non-coincident parallel if neither condition (16) nor condition (17) are satisfied.
Fig. 4c depicts a non-coincident parallel case.

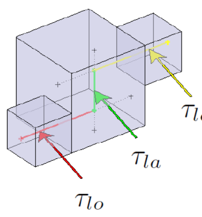
V. Guiding forces

Haptic guidance inside the local channels is done using a pair of neighbor cells c_c y c_n to get a force that results from the sum of three particular forces as shown in (18).

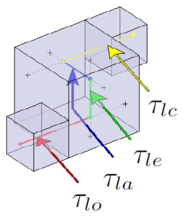
$$F = F_g + F_c + F_r + F_d \tag{18}$$



(a) Parallel collinear axes.



(b) Coplanar parallel axes.



(c) Non-coincident parallel axes.

Fig. 4: Parallel axes cases.

where:

- F is the local channel guiding force.
- F_g is the force that guides the user from the current cell c_c to the next cell c_n.
- F_c is the force that constrains the user to the local path τ_l.
- F_r is the force that repels the user from the neighbor cell c_j when the user tries to leave the current cell c_c and such a neighbor cell is different from the next cell (c_n).
- F_d is the damping force, it is a force proportional to user movements velocity that avoids oscillations troughout the guidance.

To guide the user, some haptic primitives are used, those primitives are: snap to linear local path (τ_l) and snap to a vertex (to achieve the final configuration). In the following paragraphs, the procedure to get the forces that constrain the user to the primitives is explained.

A. Current to Next cell guiding force F_g

F_g is a constant force, it has the same direction as $\mathbf{v}_m$ and points from cell c_c to cell c_n. This force can be got from the basis that where established in the previuos sections using (19).

$$F_g = k_g \mathbf{v}_m \tag{19}$$

where k_g is an empirically obtained constant.

B. Local path guidance force F_c

F_c is the force that constrains the user to the local path τ_l. The following lines explain how to get some useful vectors to obtain F_c using the current cell (c_c) and the next cell (c_n).

Let $\mathbf{v}_{qs}$ be the vector from the current user configuration $\mathbf{u}_q$ to the smallest cell center $\mathbf{p}_s$, then $\mathbf{v}_{qs}$ may be got using (20).

$$\mathbf{v}_{qs} = \mathbf{p}_s - \mathbf{u}_q \tag{20}$$

Let $\mathbf{v}_a$ be an auxiliary vector resulting form the cross product between $\mathbf{v}_{qs}$ and the vector defining the current movement axis direction $\mathbf{v}_m$, then $\mathbf{v}_a$ may be got using (21).

$$\mathbf{v}_a = \mathbf{v}_{qs} \times \mathbf{v}_m \tag{21}$$

Let $\mathbf{v}_p$ the projection of the vector $\mathbf{v}_{qs}$ onto the surface with normal $\mathbf{v}_m$ in which the point $\mathbf{u}_q$ lies, then $\mathbf{v}_p$ may be got using (22).

$$\mathbf{v}_p = \mathbf{v}_m \times \mathbf{v}_a \tag{22}$$

Finally $\mathbf{v}_p$ is the vector to be used to get the force that constrains the user to the local path using (23).

$$F_c = k_c \mathbf{v}_p \tag{23}$$

where k_c is an empirically obtained constant.

C. Repelling force

F_r is the force that is felt when the user enters a neighbor cell c_j different from the next cell c_n, then, the user is repelled from this c_j cell to let him back to the solution current channel. It should be noticed that when the user enters a c_j cell it does not turn on the current cell c_n as if he entered the next cell. The repelling force F_r is a strong force in which the influence of the current local path is augmented.

The repelling force is generated if: *1)* c_j is a free cell different from the next cell, in other words, if $c_j \in \mathbf{N} \neq c_n$ and if $(T_j \geq \Delta_{obstacle})$, *2)* when c_j belongs to the forbidden cells, in other words, if $c_j \in \mathbf{N} \neq c_n$ and if $(T_j < \Delta_{obstacle})$ and, 3) when the user is out of the current local path bounds. To get the repelling force, three kinds of penalty vectors have been proposed:

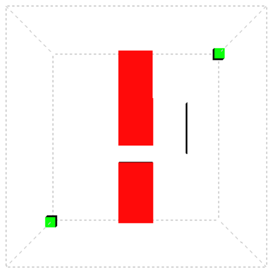
$$\mathbf{v}_{lp} = (|\mathbf{v}_p| - w_c)\,\hat{\mathbf{v}}_p \tag{24}$$

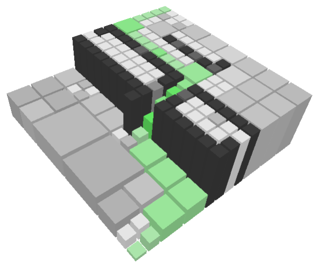
$$\mathbf{v}_{cp} = \left(|\mathbf{v}_p| + |\mathbf{v}_p|^3 - w_c\right)\hat{\mathbf{v}}_p \tag{25}$$

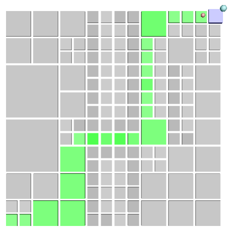
$$\mathbf{v}_{bp} = (|\mathbf{u}_q - \mathbf{p}_c| - w_c)\,\hat{\mathbf{v}}_m \tag{26}$$

The repelling force F_r is the product of a penalty constant times the selected penalty vector. If the neighbor cell c_j the user is trying to penetrate is a free cell then the linear penalty vector is chosen (24), is the neighbor cell is an obstacle cell then the cubic penalty vector is chosen (25), if the user tries to leave the current local path then the out of bounds penalty vector is chosen (26), finally if the neighbor cell is the next in the channel cell c_n then no repelling force is generated. The repelling force is then:

$$F_r = \begin{cases} 0 & \text{if } c_j = c_n \\ k_r \mathbf{v}_{bp} & \text{if out of path bounds} \\ k_r \mathbf{v}_{lp} & \text{if } c_j \in \mathcal{C} \backslash \mathcal{C}_{\mathcal{O}} \\ k_r \mathbf{v}_{cp} & \text{if } c_j \in \mathcal{C}_{\mathcal{O}} \end{cases} \tag{27}$$


(a) Virtual scene.


(b) Harmonic Function 1.


(c) Solution channel cells.

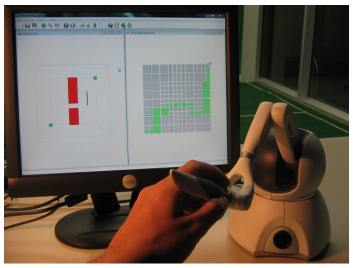

(d) Haptic Guidance Platform.

Fig. 5: Example on path planning based haptic guidance.

VI. Results

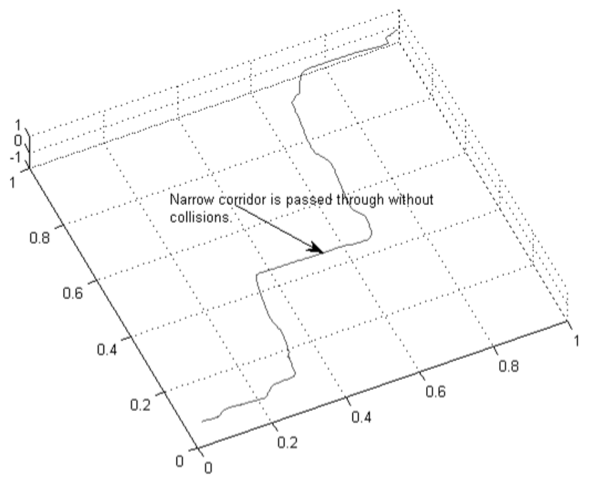
A $\mathbb{R}^2$ demonstration example is presented. Fig. 5a shows the virtual scene as the free flying robot and the obstacles have been loaded and graphically rendered. The scene consists of a cube robot that must pass through a narrow corridor. The initial position is at the top-right corner and the goal position at the bottom-left corner.

Fig. 5b shows the harmonic function values as they descend from the initial configuration cell to the goal cell. Since the harmonic function has been computed all over the $\mathcal{C}$-space model, the guiding local paths can be obtained on-line from any point within the workspace. That means that there are not pre-calculated trajectories but on the fly useful trajectories the user can have despite his position.

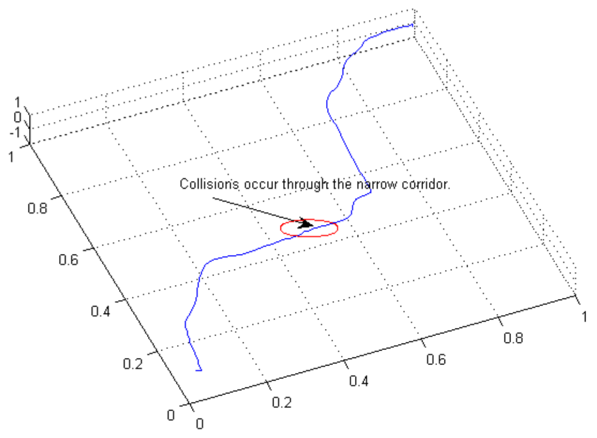
Fig. 5c shows the solution channel cells, i.e. those cells inside the local solution channel S_G whose potential values where found to be a solution to the harmonic function relaxation. User position is depicted using a big sphere, the current cell is highlighted and the next cell center is depicted using a small sphere. Notice that, despite the haptic guidance, this visual aid is presented so the user can have both visual and haptic feedback, making the system more reliable.

Fig. 5d shows the experimental platform, the user is guided using the Phantom Omni haptic device, the computer has an Intel Core Duo @ 2GHz CPU with 2GB RAM. The application can be compiled using Visual Studio 2008, furthermore Qt v4.6, Coin3D v3.0, Quarter v1.0 and OpenHaptics HD v3.0 C++ libraries must be installed, it works under Windows 7. Source code is GNU/GPL and it is available through subversion from [17].

Fig. 6a depicts the obtained trajectory from a user that has been haptically guided, notice that he passes through the narrow corridor without collisions. Fig. 6b shows the



(a) Haptically assisted trajectory.



(b) Not assisted trajectory.

Fig. 6: Haptically assisted vs. not assisted trajectories.

trajectory that results from a user moving the cube robot without any haptic aid, notice that the user collisions when entering the narrow corridor, considering the user is trying to apply the same velocity he used when he had been haptically guided.

VII. Conclusion

The development of a reliable method based on haptic primitives to feedback the user into the correct movement to achieve a selected task without collisions has been presented. Our system, based on the Kautham path planner is able to find a solution channel to complete a selected task within a virtual reality scenario. Then, stable haptic guidance is done within the local channel by means of local paths connecting the start and goal configurations.

The proposed methodology is able to construct the local paths on the fly, so there is no need to have pre-calculated trajectories to perform the haptic guidance. Furthermore, a visual aid is offered so the system feedback is complete and reliable.

As future work $\mathbb{R}^3$ and $SE(3)$ demonstrative examples are being prepared. An UDP client-server approach will be included so reaction forces running at another computer, from the virtual environment, could be added.

References

[1] S. R. Lindemann and S. M. LaValle., "Simple and efficient algorithms for computing smooth, collision-free feedback laws over given cell decompositions," *International Journal of Robotics Research*, vol. 28, no. 5, pp. 600–621, 2009.

[2] L. Lugo-Villeda, A. Frisoli, O. Sandoval-Gonzalez, M. Padilla, V. Parra-Vega, C. Avizzano, E. Ruffaldi, and M. Bergamasco, "Haptic guidance of light-exoskeleton for arm-rehabilitation tasks," in *Robot and Human Interactive Communication, 2009. RO-MAN 2009. The 18th IEEE International Symposium on*, pp. 903 – 908, 2009.

[3] O. Bayazit, G. Song, and N. Amato, "Enhancing randomized motion planners: exploring with haptic hints," in *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, vol. 1, pp. 529 – 536, 2000.

[4] D. Galeano and S. Payandeh, "Artificial and natural force constraints in haptic-aided path planning," in *Haptic Audio Visual Environments and their Applications, 2005. IEEE International Workshop on*, 2005.

[5] N. Ladeveze, J. Fourquet, B. Puel, and M. Taix, "Haptic assembly and disassembly task assistance using interactive path planning," in *Virtual Reality Conference, 2009. VR 2009. IEEE*, pp. 19 – 25, 2009.

[6] C. Christiand, J. Yoon, M. Auralius, and W. Yu, "An enhanced haptic assembly simulation system for the efficiency of assembly tasks," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pp. 1241 – 1246, 2009.

[7] R. Komerska and C. Ware, "Haptic task constraints for 3d interaction," in *Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2003. HAPTICS 2003. Proceedings. 11th Symposium on*, pp. 270 – 277, 2003.

[8] A. Varol, I. Gunev, and C. Basdogan, "A virtual reality toolkit for path planning and manipulation at nano-scale," in *Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2006 14th Symposium on*, pp. 485 – 489, 2006.

[9] Z. Gao and A. Lecuyer, "Path-planning and manipulation of nanotubes using visual and haptic guidance," in *Virtual Environments, Human-Computer Interfaces and Measurements Systems, 2009. VECIMS '09. IEEE International Conference on*, pp. 1 – 5, 2009.

[10] A. Jarillo-Silva, O. Dominguez-Ramirez, V. Parra-Vega, and J. Ordaz-Oliver, "Haptic guidance based on sub-optimal passivity control," in *Electronics, Robotics and Automotive Mechanics Conference, 2009. CERMA '09.*, pp. 175 – 180, 2009.

[11] B. Itkowitz, J. Handley, and W. Zhu, "The openhaptics toolkit: a library for adding 3d touch navigation and haptics to graphics applications," in *Haptic Interfaces for Virtual Environment and Teleoperator Systems, WHC 2005. First Joint Eurohaptics Conference and Symposium on*, pp. 590 – 591, 2005.

[12] J. Rosell and P. Iñiguez, "Path planning using harmonic functions and probabilistic cell decomposition," in *Proceedings of the IEEE Int. Conf. on Robotics and Automation - ICRA 2005*, pp. 1815–1820, 2005.

[13] J. Rosell, C. Vázquez, and A. Pérez, "Cspace decomposition using deterministic sampling and distances," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems* (IEEE, ed.), pp. 15–20, 2007.

[14] J. Rosell, M. Roa, A. Pérez, and F. García, "A general deterministic sequence for sampling d-dimensional configuration spaces," *J. Intell. Robot. Syst.*, vol. 40, no. 4, pp. 361–374, 2007.

[15] J. Rosell, C. Vázquez, A. Pérez, and P. Iñiguez, "Motion planning for haptic guidance," *Journal of Intelligent and Robotic Systems*, 2008.

[16] S. M. LaValle, *Planning Algorithms*, ch. Complexity of Motion Planning, pp. 247 – 296. http://planning.cs.uiuc.edu/, 2006.

[17] J. Rosell and C. Vázquez. `https://virtualtech.googlecode.com/svn/trunk/Kautham/KPlanner`.