A Framework for Robotized Teleoperated Tasks

Luis Basañez, Jan Rosell, Leopold Palomo, Emmanuel Nuño and Henry Portilla

***Abstract*— Teleoperation systems allow the extension of the human operator's sensing and manipulative capability into a remote environment to perform tasks at a distance, but the time-delays in the communications affect the stability and transparency of such systems. This work presents a teleoperation framework in which some novel tools, such as nonlinear controllers, relational positioning techniques, haptic guiding and augmented reality, are used to increase the sensation of immersion of the human operator in the remote site. Experimental evidence supports the advantages of the proposed framework.**

I. INTRODUCTION

Teleoperated robotic systems are characterized by a robot that executes the movements/actions commanded by an operator. The main objective of such systems is to reproduce—and if possible, enhance—in a remote environment, the sensing and actuation capabilities of an operator [1], [2], so that the mental and physical effort required to accomplish a given task be not negatively affected by its remote execution. However, until recently, performing complex tasks with classical teleoperated systems demanded very skillful operators. This work presents a teleoperation framework for robotized tasks in which, besides the variable time-delays control aspects, several advanced tools like relational positioning, virtual contacts, haptic guidance and augmented reality have been designed in order to aid the operator in performing the tasks. The contribution of this work is the framework as a whole, together with the aforementioned tools developed by the authors for turning bilateral robotic teleoperation secure and reliable.

In particular, the framework employs different control schemes for constant and variables time-delays, such as simple Proportional plus damping (P+d), scattering-based and adaptive schemes [3], [4], [5], [6]; it makes use of the PMF (Positioning Mobile with respect to Fixed) solver to perform tasks where an object has to be positioned with respect to its surroundings [7], [8]; it also uses a planning methodology for hapticaly guiding the human motions [9], [10]; and finally, the framework also offers an augmented reality module [11].

This work has been partially supported by the Spanish CICYT projects DPI2008-02448 and DPI2007-63665 and the Mexican projects CONACyT CB-129079 and SEP PROMEP-NPTC 103.5/10/4470.

All authors, except E. Nuño, are with the Institute of Industrial and Control Engineering (IOC) at the Technical University of Catalonia (UPC), Barcelona, Spain. Emails:{*luis.basanez, jan.rosell, leopold.palomo, henry.portilla*}*@upc.edu.*

E. Nuño is with the Department of Computer Science at the University of Guadalajara (UdG), Guadalajara, Mexico. Email: *emmanuel.nuno@cucei.udg.mx*.

A. The Proposed Teleoperation Framework

The logic structure of the proposed teleoperation framework is represented in the upper part of Fig. 1. The diagram contains two large blocks that correspond to the *local station*, where there are the human operator and the local robot (a haptic device) and to the *remote station*, that includes an industrial manipulator as remote robot. There is also an intermediate block representing the communication channel. The physical architecture of the whole teleoperation framework is shown in the lower part of Fig. 1.

In the local station the human operator physically interacts with a haptic device and with the teleoperation aids modules by means of a graphical user interface.

In the local station, the teleoperation control modules are the following:

- *Haptic Rendering module*. It calculates the force to be fed back to the operator as a combination of the following forces: a) *constraint force*, computed by the relational positioning module in order to restrict movements to a submanifold of free space; b) *virtual force*, computed by the virtual contacts module as a response to the detection of collision situations; c) *guiding force*, computed by the guiding module to swept the operator along a collision-free path towards the goal; and d) *control force*, produced by the controller as a result of tracking errors between the haptic and the robot manipulator.
- *Control Algorithms module*. It executes the motion and force control algorithms that allow position tracking while maintaining stability despite the communication delays between stations.
- *Geometric Conversion module*. It is in charge of the conversion between the coordinates of the haptic device and of the remote robot.
- *State Determination module*. It updates the system state variables that are used by the teleoperation control and the teleoperation aids sets.

The modules of the teleoperation aids are:

- *Relational Positioning module*. It is used by the operator to define geometric relationships between the part manipulated by the remote robot and the parts in the environment. The module finds the solution submanifold where the constraints imposed by the geometric relationships are satisfied, as well as the forces needed to be kept within this submanifold.
- *Virtual Contacts module*. It detects possible collisions of the manipulated part with the environment, and generates the corresponding repulsion forces for helping

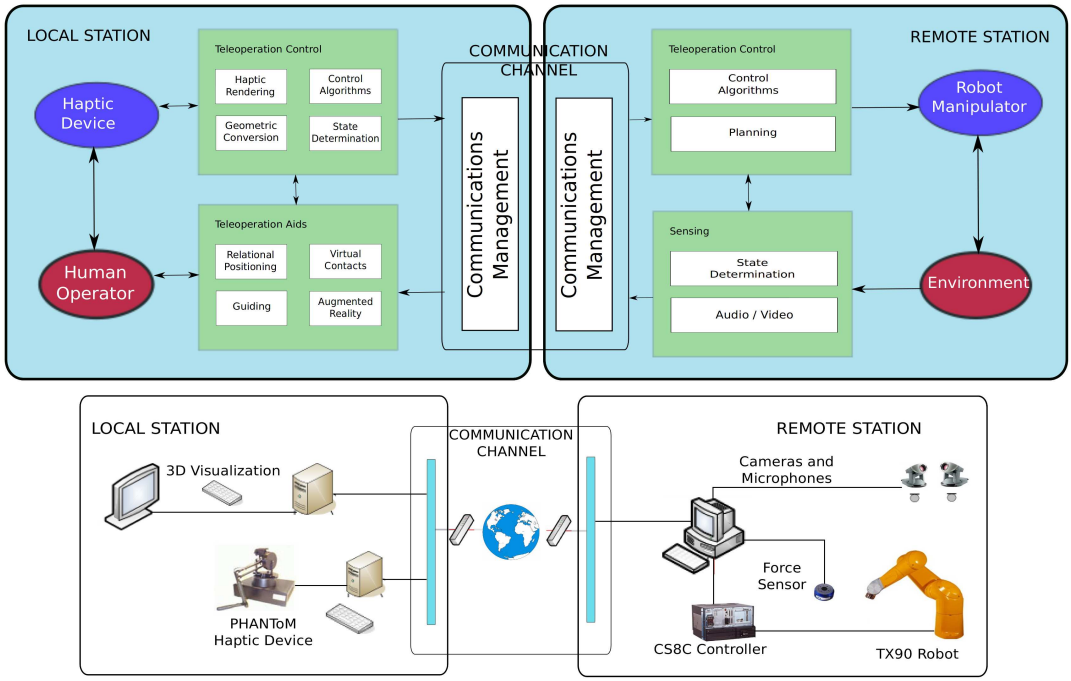

Fig. 1. Logic and physical diagrams of the teleoperation framework.

the operator to react as soon as possible when virtual contacts situations occur.

- *Guiding module*. It computes, by means of motion planning techniques, the forces that guide the operator during task execution.
- *Augmented Reality module*. It displays a stereoscopic view of the Remote Station enhanced with computer generated data such as co-located virtual objects.

In the remote station, a robot manipulator is controlled by means of the teleoperation control modules. It is also connected to the local station through the communication channel. The information generated at this station is captured and transmitted by the sensing modules. The teleoperation control consists of the following modules:

- *Control Algorithms module*. It receives motion commands from the local station and sends the corresponding torques to the joint actuators of the robot manipulator.
- *Planning module*. It computes how the part manipulated by the robot must follow the trajectory specified by the operator with the haptic device, avoiding, at the same time, collisions between the robot and the environment.

The rest of the available capabilities are gathered in the sensing modules:

- *State Determination module*. It gathers and processes the remote system state variables that are fed back to the local station.
- *Audio/Video module*. This module is the responsible of the capture and transmission of sounds and images of the remote environment.

Fig. 2 shows the flow of the main signals through the teleoperation framework: force, position/velocity, operator commands and audio/video signals. This figure points out the interaction paths between the human operator and the environment by means of the local haptic device and the remote robot manipulator and the teleoperation modules (control, aids and sensing).

II. Teleoperation Control

There are six different modules in charge of the teleoperation control task, four in the local station and two in the remote station (Fig. 1). This section discusses them in detail, except for the state determination module, which collects and handles all available data. However, before going through these modules, the dynamical model of the non-linear teleoperation framework will be introduced.

In the following, $\mathbb{R}$ stands for the real number set, $\mathbb{R}^+$ for the positive real number set and $\mathbb{R}_0^+$ for the set containing $\mathbb{R}^+$ and zero. The subscript i takes the values l and r for the local and remote robot manipulators, respectively.

The local and remote manipulators together with the human and environment interactions are modeled as a pair of *n*-Degree Of Freedom (DOF) serial links with revolute joints. Their corresponding non-linear dynamics are

$$\begin{aligned} \mathbf{M}_l(\mathbf{q}_l)\ddot{\mathbf{q}}_l + \mathbf{C}_l(\mathbf{q}_l,\dot{\mathbf{q}}_l)\dot{\mathbf{q}}_l + \mathbf{g}_l(\mathbf{q}_l) &= \boldsymbol{\tau}_l^* - \boldsymbol{\tau}_h \\ \mathbf{M}_r(\mathbf{q}_r)\ddot{\mathbf{q}}_r + \mathbf{C}_r(\mathbf{q}_r,\dot{\mathbf{q}}_r)\dot{\mathbf{q}}_r + \mathbf{g}_r(\mathbf{q}_r) &= \boldsymbol{\tau}_e - \boldsymbol{\tau}_r^*, \end{aligned} \tag{1}$$

where: $\mathbf{q}_i$, $\dot{\mathbf{q}}_i$, $\ddot{\mathbf{q}}_i \in \mathbb{R}^n$ are the joint position, velocity and acceleration; $\mathbf{M}_i(\mathbf{q}_i) \in \mathbb{R}^{n\times n}$ the inertia matrices; $\mathbf{C}_i(\mathbf{q}_i,\dot{\mathbf{q}}_i) \in \mathbb{R}^{n\times n}$ the Coriolis and centrifugal effects; $\mathbf{g}_i(\mathbf{q}_i) \in \mathbb{R}^n$ the gravitational forces; $\boldsymbol{\tau}_i^* \in \mathbb{R}^n$the controller forces; and $\boldsymbol{\tau}_h \in \mathbb{R}^n$, $\boldsymbol{\tau}_e \in \mathbb{R}^n$ the joint forces corresponding

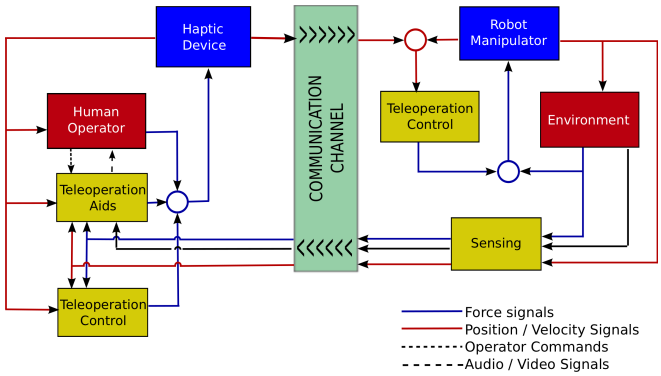

Fig. 2. Flow diagram of the main signals of the teleoperation framework.

to the ones exerted by the human and the environment, respectively.

A. Haptic Rendering

This module is intended to compose the total force $\boldsymbol{\tau}_l^*$ to be fed back to the operator through the haptic device. The total force is the resultant of the *constraint force* $\mathbf{f}_c$, the *guiding force* $\mathbf{f}_g$, the *virtual force* $\mathbf{f}_v$, all three initially expressed in the operational space but converted to the joint space, and the *control force* $\boldsymbol{\tau}_l$, provided by the Relational Positioning, Guiding, Virtual Contacts and Control Algorithms modules, respectively. The total force is given by $\boldsymbol{\tau}_l^* = \mathbf{J}^\top(\mathbf{q}_l)[\mathbf{f}_c+\mathbf{f}_g+\mathbf{f}_v]+\boldsymbol{\tau}_l$, where $\mathbf{J}^\top(\mathbf{q}_l)$ is the transpose Jacobian of the haptic device.

B. Geometric Conversion

The force capacity and the operational space of the local and remote manipulators differ from one another. In this framework the haptic device workspace and its force capacity are about ten times smaller than the corresponding ones on the remote robot manipulator.

The Geometric Conversion module is responsible for the position and force correspondence between the two robots. This module provides the framework with the capability to *mouse jump* the haptic device, allowing the operator to reposition the reference frame for the haptic device end-effector in order to enlarge the haptic workspace. When jumping, the module sets to zero all forces on the haptic device and stops motion on the robot manipulator during the process, and after jumping it resumes the teleoperation associating the last robot position to the new haptic position [10].

C. Control Algorithms

The haptic device and the remote manipulator are coupled to each other by means of a control algorithm that sends/receives information through the communication channel. Such channel imposes limited data transfer and, depending on its nature, time-delays that can be constant or variable. These delays affect the overall stability of the teleoperation system. Controlling these systems has become a highly active research field (for guides to teleoperators control, the reader may refer to [2], [6]).

TABLE I

CONTROL LAWS FOR THE LOCAL AND REMOTE ROBOT MANIPULATORS. WHERE $K_i, B_i, K_d, K_{di}, K \in \mathbb{R}^+$ ARE THE CONTROL GAINS AND THE SYMBOL $(\dot{\cdot})$ MEANS ESTIMATION.

Scheme	Control laws
P+d	$\boldsymbol{\tau}_l = K_l\mathbf{e}_l + B_l\dot{\mathbf{q}}_l - \mathbf{g}_l$ $\boldsymbol{\tau}_r = -K_r\mathbf{e}_r - B_r\dot{\mathbf{q}}_r + \mathbf{g}_r(\mathbf{q}_r)$
PD+d	$\boldsymbol{\tau}_l = K_d\dot{\mathbf{e}}_l + K_l\mathbf{e}_l + B_l\dot{\mathbf{q}}_l - \mathbf{g}_l$ $\boldsymbol{\tau}_r = -K_d\dot{\mathbf{e}}_r - K_r\mathbf{e}_r - B_r\dot{\mathbf{q}}_r + \mathbf{g}_r(\mathbf{q}_r)$
Scatt.	$\boldsymbol{\tau}_l = \boldsymbol{\tau}_{ld} + K\mathbf{e}_l + B_l\dot{\mathbf{q}}_l - \mathbf{g}_l$ $\boldsymbol{\tau}_{ld} = K_{dl}[\dot{\mathbf{q}}_l - \dot{\mathbf{q}}_{ld}]$ $\boldsymbol{\tau}_r = \boldsymbol{\tau}_{rd} - K\mathbf{e}_r - B_r\dot{\mathbf{q}}_r + \mathbf{g}_r$ $\boldsymbol{\tau}_{rd} = -K_{dr}[\dot{\mathbf{q}}_r - \dot{\mathbf{q}}_{rd}]$
Adaptive	$\boldsymbol{\tau}_l = \hat{\mathbf{M}}_l\boldsymbol{\lambda}\dot{\mathbf{e}}_l + \hat{\mathbf{C}}_l\boldsymbol{\lambda}\mathbf{e}_l - \hat{\mathbf{g}}_l + K_l(\dot{\mathbf{q}}_l + \boldsymbol{\lambda}\mathbf{e}_l) + B\dot{\mathbf{e}}_l$ $\boldsymbol{\tau}_r = \hat{\mathbf{g}}_r - \hat{\mathbf{M}}_r\boldsymbol{\lambda}\dot{\mathbf{e}}_r - \hat{\mathbf{C}}_r\boldsymbol{\lambda}\mathbf{e}_r - K_r(\dot{\mathbf{q}}_r + \boldsymbol{\lambda}\mathbf{e}_r) - B\dot{\mathbf{e}}_r$

The controllers that have been developed for the present framework are: a proportional plus damping (P+d), a proportional-derivative plus damping (PD+d), a scattering-based, and an adaptive controller. Table I depicts their respective mathematical expressions. Detailed descriptions of these controllers, along with proofs for the statements in this paper can be found in [3], [4], [5], [6]. Each controller exhibits different capabilities and, in general, all of them can handle time-delays and can provide asymptotic stability. The main differences between them are stated as follows. The P+d and PD+d can provide stiffer force reflections of the remote environment, however, an increase in time-delays represents an increase in damping, thus overdamped behavior can be obtained. For small time-delays the P+d provides better transparency than all the others. The scattering-based controller is more robust to changes in time-delays, but injects more damping than the P+d and the PD+d, and is potentially subject to wave reflections. These three controllers are able to handle variable time-delays, but on the downside, they need to compensate the gravity forces, requiring some previous knowledge of the teleoperators nonlinear model. Finally, the adaptive controller estimates the physical parameters of the manipulators and the rate convergence of the errors to zero is faster than with the other schemes, but it cannot handle variable time-delays. An statistical performance comparison, between these schemes, is currently underway. For another analysis of this kind see [12].

Let $T_i(t)$ represent the time-delays, and $\mathbf{e}_i \in \mathbb{R}^n$ the position errors, defined as

$$\mathbf{e}_l = \mathbf{q}_l - \mathbf{q}_r(t - T_r(t)); \quad \mathbf{e}_r = \mathbf{q}_r - \mathbf{q}_l(t - T_l(t)).$$

Using standard Lyapunov analysis together with Barbălat's Lemma (when the constraint, guiding and virtual forces are zero) it can be proved that using the P+d or the PD+d controllers in closed loop with the system, velocities and position error are bounded, provided that the control gains, $K_i, B_i > 0$, are set according to

$$4B_lB_r > ({}^*T_l + {}^*T_r)^2K_lK_r, \tag{2}$$

under the assumptions that the human operator and the environment are passive; that if time-delays are variable then they have known upper bounds ${}^*T_i$. *i.e.,* $T_i(t) \leq {}^*T_i < \infty$; and that their time derivatives do not grow or decrease faster than time itself, thus, $|\dot{T}_i(t)| < 1$.

Moreover, if the operator does not inject forces on the haptic device and the remote manipulator does not come in contact with the environment (*i.e.,* $\boldsymbol{\tau}_h = \boldsymbol{\tau}_e = \mathbf{0}$), then velocities and position errors asymptotically converge to zero, *i.e.,* $|\dot{\mathbf{q}}_i| \to 0$, $|\mathbf{e}_i| \to 0$ as $t \to \infty$. Note that the key feature for the stability of the teleoperator with these controllers, is condition (2), that clearly states that larger time-delays require injecting more damping in order to maintain stability and position tracking.

Using the scattering-based controller with gains satisfying (2), then, position tracking can be also established for variable time-delays. In this case, the desired velocities are encoded using the classic scattering transformation proposed in [13], [14]. For variable time-delays $T_i(t)$, the local and the remote manipulators are interconnected as $\mathbf{u}_r = \gamma_l \mathbf{u}_l(t - T_l(t))$ and $\mathbf{v}_l = \gamma_r \mathbf{v}_r(t - T_r(t))$ where $\gamma_i^2 \leq 1 - \dot{T}_i(t)$, $\mathbf{u}_l := \boldsymbol{\tau}_{ld} + \alpha\dot{\mathbf{q}}_l$, $\mathbf{v}_r := \boldsymbol{\tau}_{rd} - \alpha\dot{\mathbf{q}}_r$ and $\alpha \in \mathbb{R}^+$ is a control gain.

The adaptive controllers in Table I can be also written as

$$\begin{aligned} \boldsymbol{\tau}_l &= -\mathbf{Y}_l(\mathbf{q}_l, \dot{\mathbf{q}}_l, \mathbf{e}_l, \dot{\mathbf{e}}_l)\hat{\boldsymbol{\theta}}_l + K_l\boldsymbol{\epsilon}_l + B\dot{\mathbf{e}}_l \\ \boldsymbol{\tau}_r &= \mathbf{Y}_r(\mathbf{q}_r, \dot{\mathbf{q}}_r, \mathbf{e}_r, \dot{\mathbf{e}}_r)\hat{\boldsymbol{\theta}}_r - K_r\boldsymbol{\epsilon}_r - B\dot{\mathbf{e}}_r. \end{aligned} \quad (3)$$

where Y_i is a matrix that is function of the joint positions, velocities and position and velocity error, and θ_i is a vector of estimated parameters.

Defining a synchronizing signal $\boldsymbol{\epsilon}_i$, as

$$\boldsymbol{\epsilon}_i = \dot{\mathbf{q}}_i + \boldsymbol{\lambda}\mathbf{e}_i, \quad (4)$$

where $\boldsymbol{\lambda} > 0$ is diagonal, then with (3), (4) and $\boldsymbol{\tau}_h = \boldsymbol{\tau}_e = \mathbf{0}$, we can write (1) as

$$\mathbf{M}_i(\mathbf{q}_i)\dot{\boldsymbol{\epsilon}}_i + \mathbf{C}_i(\mathbf{q}_i, \dot{\mathbf{q}}_i)\boldsymbol{\epsilon}_i + \mathbf{K}_i\boldsymbol{\epsilon}_i + \mathbf{B}\dot{\mathbf{e}}_i = \mathbf{Y}_i\tilde{\boldsymbol{\theta}}_i$$

where $\tilde{\boldsymbol{\theta}}_i = \hat{\boldsymbol{\theta}}_i - \boldsymbol{\theta}_i$ is the error between the estimated and the real parameters. Now, using the estimation law $\dot{\hat{\boldsymbol{\theta}}}_i = -\boldsymbol{\Gamma}_i\mathbf{Y}_i^\top\boldsymbol{\epsilon}_i$, with $\boldsymbol{\Gamma}_i = \boldsymbol{\Gamma}_i^\top > 0$, yields $|\boldsymbol{\epsilon}_i| \to 0$ as $t \to \infty$ and $|\dot{\mathbf{q}}_i| \to |\mathbf{e}_i| \to 0$.

D. Planning

The role of the Remote Planning module is the avoidance of collisions between the robot manipulator and the environment. This is a necessary step since the teleoperation is done at task level. The operator commands the motions of the part being manipulated by the robot using both the Guiding module (to avoid collisions of the part with the environment) and the Relational Positioning module (to constrain some of its DOF). Nevertheless, the commanded motions are executed by the robot manipulator using all of its DOF. The remote Planning module provides a reactive behaviour that guarantees that their execution is collision-free.

III. Teleoperation Aids

A. Relational Positioning

Tasks where an object has to be positioned with respect to its surroundings are ubiquitous in robotics, and oftentimes can be decomposed into a series of constrained movements which do not require using the six DOF an object has in free space. For example, the spray-painting of a flat surface takes place in a two-DOF planar space, and the insertion of a prismatic peg in a hole also requires two DOFs, translation and rotation around the hole axis, provided that the axis of the two objects are aligned.

Although operator skills are needed for the successful execution of many teleoperated tasks, maintaining the tool or the manipulated object inside a specific region of space can be both challenging and tiring. Such region can be easily described in terms of geometric constraints that, when satisfied, define a submanifold of SE(3) of allowed movements. Haptic feedback can be used to assist the operator by restricting his/her movements to the submanifold of interest, lowering the mental burden needed to execute the task.

The Relational Positioning module explicitly addresses these issues. Its core consists of the PMF (Positioning Mobile with respect to Fixed), a geometric constraint solver that finds the map between constraint sets and parameterized solution submanifolds [7]. Constraints are defined between elements of the manipulated object and elements of its surroundings, which are considered fixed. PMF accepts as input constraints distance and angle relations between points, lines, and planes, and exploits the fact that in a set of geometric constraints, the rotational component can often be separated from the translational one and solved independently. By means of logic reasoning and constraint rewriting, the solver is able to map a broad family of input problems to a few rotational and translational scenarios with known closed-form solution. The solver can handle under-, well-, and over-constrained (redundant or incompatible) problems with multiple solutions, and is computationally very efficient, so it can be included in high-frequency loops that require response times within the millisecond order of magnitude (e.g., update solutions when the geometry of the problem changes, as in the case of moving obstacles). Fig. 3 shows the PMF user interface displaying a cone with its tip restricted to a spherical surface.

The present teleoperation framework makes use of an impedance-type haptic device (force/torque input, velocity output), so the generation of virtual constraint forces and torques requires translating the kinematic information provided by the geometric constraint solver into a dynamic model. The chosen scheme takes advantage of the high backdrivability and low inertia/friction of the haptic device: it leaves the dynamics of the unconstrained directions unchanged and generates forces in the constrained directions based on the difference e_l between the actual x_l and desired x_{ld} positions of the end-effector in operational space coordinates: $e_l = x_l - x_{ld}$, where x_{ld} represents the projection of x_l on the current solution submanifolds. e.g., the expression

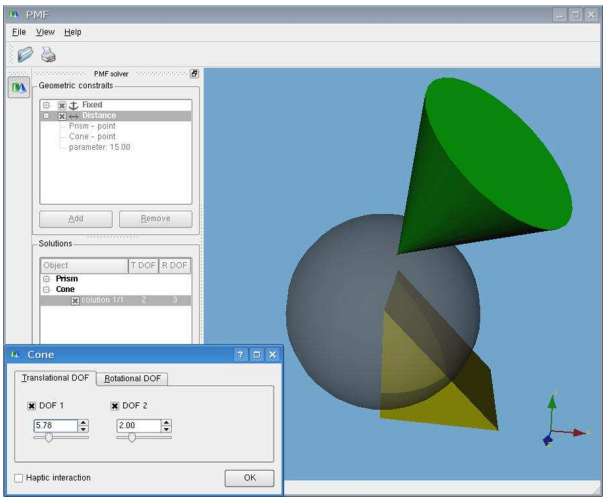

Fig. 3. PMF user interface displaying a cone with its tip restricted to a spherical surface.

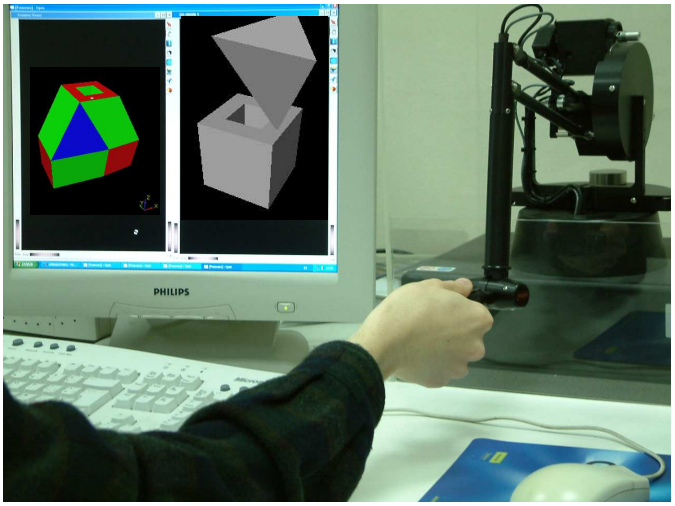

Fig. 4. Rendering of virtual contacts forces between 3D objects using the task configuration space.

TABLE II

TRANSLATIONAL AND ROTATIONAL SUBMANIFOLDS.

Translational submanifold	DOF	Rotational submanifold	DOF
$\mathbb{R}^3$	3	$SO(3)$	–
Plane	2	Vectors at an angle	2
Sphere	2	Parallel vectors	1
Cylinder	2	Fixed rotation	0
Line	1		
Ellipse	1		
Point	0		

of the force $f_c = K_P e_l + K_D \dot{e}_l$ (a PD-like controller) is analogous to attaching a virtual spring and damper between x_l and x_{ld}. A similar scheme is used by the Virtual Contacts and the Guiding modules. Table III-A shows the translational and rotational submanifolds to which the haptic device end-effector can be constrained. All combinations between rotational and translational submanifolds are possible.

B. Virtual Contacts

Haptic devices allow the operator to interact with a virtual world and to feel the reaction forces that arise when the manipulated virtual object collides with the objects in the virtual environment. The haptic rendering of virtual contacts can be a useful teleoperation aid because virtual contacts can make the user react on time since they may occur before the real ones (if bounding volumes are considered for the models and because no time delay exists).

Simple and efficient procedures have been developed for punctual haptic interaction. Nevertheless, the haptic rendering of virtual contacts forces between 3D objects requires the use of collision detection algorithms and the approximation of the reaction force and torque by the interpolation or the sum of the forces computed at each contact point. Considering the task composed of convex polyhedra representing the bounding volumes of the objects, face-face contacts or edge-face contacts are not uncommon (these type of contact may also occur with virtual fixtures defined by the operator). Approaches based on collision detection algorithms do not provide, in these situations, a good haptic rendering. This can be solved if the knowledge of the current type of contacts taking place is used. In this line, a method based on the task configuration space (C-space) is adopted in this project. Since in C-space the manipulated object is represented by a point, the method becomes similar to punctual haptic interaction methods. The procedure, illustrated in Fig. 4 is based on the following three steps [15].

1) *C-space modelling*: Assuming objects are modelled with convex polyhedra, their interference is represented by (convex) *C-obstacles*. Each *C-obstacle* can be modelled with a graph G whose nodes represent basic contacts. The node whose *C-face* is closest to the current position of the manipulated object, together with those neighbor nodes that satisfy the applicability condition for the current orientation, constitute a subgraph, $G_{near}(\theta)$, that represents the local *C-space*.
2) *C-space updating*: Each time the user changes the orientation θ of the manipulated object, $G_{near}(\theta)$ is updated (note that $G(\theta)$ remains unchanged whenever the user does a pure translation motion).
3) *Haptic rendering*: The collision detection between the manipulated object and the obstacles is done by evaluating whether the object position lies inside a *C-obstacle*. This is done only considering the nodes of $G_{near}(\theta)$ by verifying if the object position is below the plane that contains the *C-face* of the basic contact for the current orientation. The reaction force is computed proportional to the penetration depth. The reaction torque is computed as a function of the point where the reaction force is applied.

C. Guiding

Haptic devices can also provide guiding forces to assist the user to safely teleoperate a robot or to train him in the performance of a virtual task. Some simple guiding forces may constrain the user motions along a line or curve or over a given working plane or surface, e.g. for a peg-in-hole task

a line can be defined along the axis of the hole and the user may feel an increasing force as he moves the peg away from that line. Although these simple guides can already be a good help, some tasks may require more demanding guiding forces to aid the user all along the task execution. Motion planning strategies based on potential fields can cope with these guiding requirements, by defining trajectories that follow the gradient descent. The present framework follows this line by using harmonic functions that guarantee the existence of a single minimum at the goal configuration. The proposed approach, illustrated in Fig. 5 for the teleoperation of a bent-corridor planar task, relies on the following three points [9]:

1) *Configuration space modelling*: A 2^n-tree hierarchical cell decomposition of the configuration space is build based on an iterative procedure that samples configurations (using a deterministic sampling sequence), evaluates and classifies them, and updates the cell partition when necessary. A transparency parameter is associated to each cell as a function of the number of free and collision samples it contains.
2) *Harmonic function computation*: The computation of an harmonic function is interleaved with the configuration space modelling. The harmonic functions is not only computed over the free cells (fixing the obstacle cells at a high value), but over the whole set of cells (using the transparency as a weighting parameter). This is a new and efficient method to compute harmonic functions tailored to be used over partially-known configuration spaces.
3) *Generation of haptic guiding forces*: Guiding forces are generated from the harmonic function, using a simple point-attraction primitives [16]. From the current cell, the user is attracted to next one (following the negated gradient) by a force directed to its center. The force is felt until the user is located at a given distance threshold from the center. Then the current cell is updated and the procedure is repeated until the goal cell is reached.

D. Augmented Reality

Augmented reality refers to a real-world environment representation that has been enhanced with the addition of computer-generated data. Generally speaking, this augmentation can be multisensorial, but this discussion will focus on the visual augmentation of video streams.

Visual feedback of the teleoperated robot environment is of capital importance, because it provides the operator with information that will help him navigate through the remote workspace. This is especially true for task operations that take place in free space, where there is no force feedback originating from the contact with other objects. Depth perception also improves the operator's sense of immersion, but cannot be achieved with a monoscopic view of the remote station.

The current framework implements a stereoscopic visual feedback system that combines images gathered from two remotely actuated video cameras. The stereoscopic effect is

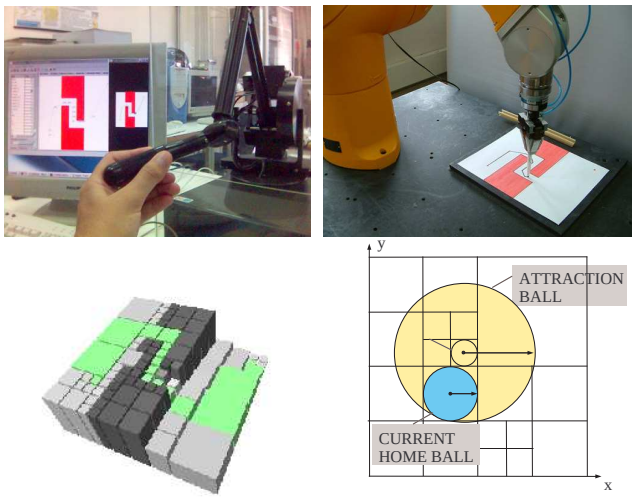

Fig. 5. Teleoperation of a bent-corridor task using guiding forces.

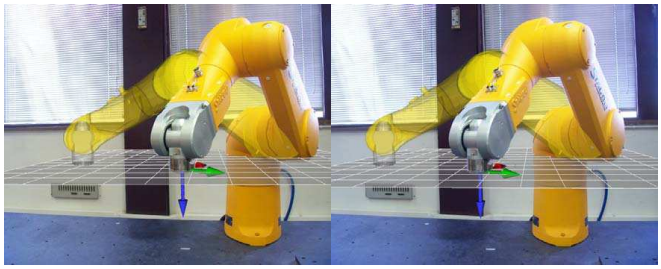

Fig. 6. Stereographic view with augmented reality aids.

achieved by alternatively displaying the image corresponding to the left and right eyes on a computer monitor or wall projector, switching between them at a frequency of 100-120Hz. Many people can visualize the 3D scene at the same time by wearing pairs of shutter glasses that are synchronized with the switching of the video display.

The operator experience can be enhanced at the local station by overlaying a co-located virtual scene on the video streams corresponding to each eye. This virtual scene contains visual cues and annotations that are not present in the real world, but can improve task performance. Examples of graphical entities that can be rendered are the geometric constraints an object is subject to, virtual objects an operator may be haptically interacting with, guidance paths, magnitude and direction of interaction forces, and boundaries of the robot workspace.

Limited-bandwidth communications may reduce the frame rate of the live video streams below acceptable levels. In such cases a virtual version of the remote robot can be displayed and refreshed at a higher frequency while using very little bandwidth, since it only requires updating the current joint positions.

The appearance of an augmented environment with both real and virtual objects must be visually compelling and, for this, must obey overlay and occlusion visibility rules. That is, parts of a virtual object that are in the foreground are

rendered and block the real objects that lie behind (if the virtual object is semitransparent, a blending effect occurs), and parts of a virtual object that are behind a real one are not rendered. Occlusions are achieved by rendering transparent models of the real objects in the virtual scene. Unmodeled real objects are unable to produce occlusion effects. Fig. 6 shows the left and right eye views of a scene augmented with three visible virtual entities: a planar surface that represents the geometric constraints acting on the robot end-effector, the end-effector coordinate frame, and a semitransparent rendering of the robot in a different configuration. Notice the overlay and occlusion effects between the real robot and the virtual entities.

IV. Sensing

A. State determination

The teleoperation aids and control modules explained in the previous sections require knowledge of the state (position and velocity) of the haptic device and of the robot manipulator in both joint and operational space coordinates, as well as a means for sending actuation commands to them.

Interaction with the haptic device is done using Sensable OpenHaptics library [16]. Queried parameters are positions and velocities in both joint and operational space coordinates, and actuation commands are given in the form of operational space forces and torques. On the other hand, interaction with the robot manipulator is done using Stäubli's Low Level Interface (LLI) library. The library is used for queries of joint space position, velocity and applied torque, and actuation commands are given as joint torques. The map to operational space coordinates is done externally using custom-made kinematic routines.

B. Video and audio capture and transmission

The remote station has two video cameras and microphones used for stereoscopic visual and audio feedback. The operator is able to control from the local station the pan, tilt, and zoom of either an individual camera or of the stereoscopic view (both cameras at the same time).

The two video streams are captured independently and are hardware-encoded to MPEG4 format. This type of encoding has high compression and produces high quality video. Video and audio data are transmitted separately from the data relevant to the teleoperation control loop using a client-server architecture and the Real Time Streaming Protocol (RTSP).

RTSP is based on the Transport Control Protocol (TCP) and manages the transmission session. During a transmission the Real-time Transport Protocol (RTP) is used for the transport of streaming data and the RTCP (where the 'C' stand for control) is used periodically to monitor the provided quality of service. Amongst the available control parameters figure lost packets, network jitter and round-trip delay.

C. Communication Channel

The communication channel used in this framework is a packet switched network like the Internet or the Internet2. It uses protocols that divide the messages into packets before sending them. Each packet is then transmitted individually and can follow different routes to their destination. Once all packets forming a message have arrived at destination, they are recompiled into the original message. The pros on using these means of communications are ubiquity and bandwidth. However, the cons are packet drops and variable time-delays. The teleoperation framework uses UDP as the transport protocol and can use either IPv4 or IPv6, versions 4 and 6 of the Internet Protocol, respectively.

V. Operation Principle

The operation principle for a teleoperated task using the teleoperation framework proposed is done as follows. The operator first determines the goal configurations of the task and sets the motion constraints that must be maintained during the task execution, by specifying the relative positions of the part or tool manipulated by the robot with respect to the environment. Then, in order to remotely control the robot performing the task, he moves the haptic device with the help of the following feedback forces:

1) Forces that restrain him within the task submanifold defined by the constraints set, letting him concentrate on the commanding of motions relevant to the task.
2) Forces resulting from collisions detected in the virtual environment, which allow him to react on time because they prevent him of imminent collisions in the remote station.
3) Guiding forces that, from any point within the task submanifold, swept him along a collision-free path towards the goal configuration, resulting in a faster task commanding.
4) Forces generated by the controller as a result of tracking errors that occur due to real interactions on the remote station.

Position and force correspondence between the haptic device and the robot is guaranteed, including the possibility to perform mouse jumps required when the size of the workspaces differs substantially. The performance of the task being teleoperated is continually monitored by the operator using the 3D image of the scene augmented with extra information relevant to the task.

VI. Experimental Results

In order to validate the proposed framework several experimental tests have been remotely performed using the P+d and PD+d controllers that provide position tracking. One of them consists on moving the remote robot manipulator end-effector along a rail with a line restriction.

The proposed test has the following characteristics:

- The motion of the remote robot manipulator end-effector is restricted to a line parallel to the x-axis and fixed orientation, as shown in Fig. 7.
- The P+d controller has been used in both the haptic device and the robot manipulator.
- The position commands correspond to the desired position of the haptic end-effector.

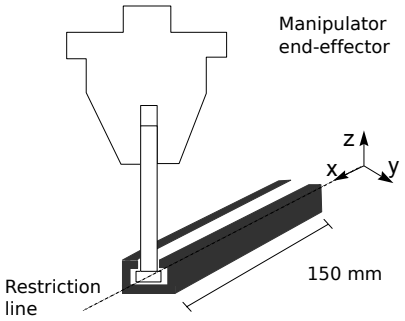

Fig. 7. Line restriction over a rail.

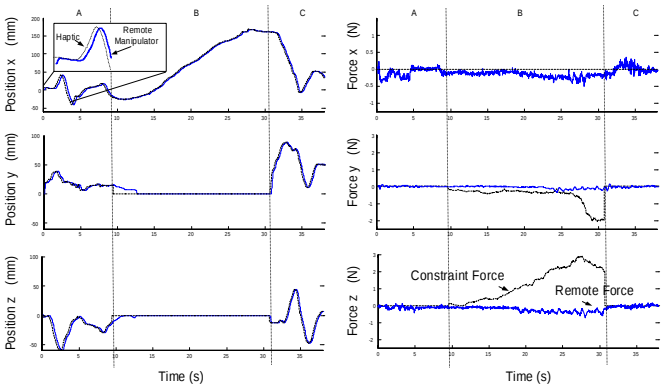

Fig. 8. Position and force plots of the experimental test.

- The communication channel is implemented using UDP/IPv6 sockets in a client-server application.
- Although the 6 DOF of the robot manipulator have been controlled, only data concerning the translational DOFs is reported.

Fig. 8 depicts the evolution of positions and forces along the x, y, z-axes. Position curves are shown in the left column, and feature three distinct zones separated by dashed vertical lines: Zones A and C correspond to unrestricted motion in free space, while Zone B corresponds to translations restricted to a line. It can be seen that position values in the y and z-axes drop to zero, and motion only takes place in the x-axis when the restriction is set (at around 10s). Note also that although the initial positions of the haptic device and the robot manipulator differ from one another, position error converges to zero and position tracking is achieved. Force curves are shown on the right column and plot the two components of the total force acting on the haptic device: the constraint force and the control force. This last force does not have a significant contribution since the time-delays are small, approximately 50 ms, so the constraint force dominates the force felt by the operator when he/she tries to violate the constraint, as it can be seen in the y and z axes in Zone B.

The reported framework has been tested under different scenarios. Some experiments have been performed from Barcelona, Spain, as the remote site, to Urbana-Champaign, Il., USA, as the local site. In this case, the P+d and PD+d controllers have been employed and it has been shown [4] that the closed-loop system is asymptotically stable. Also, there have been experiments from Barcelona, to Guadalajara, Mexico, in which the P+d controller was used. In all these experiments, Internet has been the adopted communication channel and the round trip time delay, for both cases, has been around 0.8 seconds. The User Datagram Protocol (UDP) has been the transport layer protocol used for the experiments.

VII. Conclusions

The teleoperation framework presented in this paper makes use of relational positioning, that allows the operator to easily introduce geometric relations; virtual contacts detection, that provide predictive forces arising from virtual collisions; augmented reality, that enhances the visual feedback; and control algorithms, that provide position tracking despite variable time-delays. These advanced tools increase the overall task performance, thus reducing operator fatigue and stress while remotely executing a task.

References

[1] L. Rosenberg, "Virtual fixtures: Perceptual tools for telerobotic manipulation," in *IEEE Annual Int. Symp. Virtual Reality*, 1993, pp. 76–82.
[2] P. Hokayem and M. Spong, "Bilateral teleoperation: An historical survey," *Automatica*, vol. 42, no. 12, pp. 2035–2057, 2006.
[3] E. Nuño, R. Ortega, N. Barabanov, and L. Basañez, "A globally stable PD controller for bilateral teleoperators," *IEEE Transactions on Robotics*, vol. 24, no. 3, pp. 753–758, June 2008.
[4] E. Nuño, L. Basañez, R. Ortega, and M. Spong, "Position tracking for nonlinear teleoperators with variable time-delay," *International Journal of Robotics Research*, vol. 28, no. 7, pp. 895–910, July 2009.
[5] E. Nuño, R. Ortega, and L. Basañez, "An adaptive controller for nonlinear teleoperators," *Automatica*, vol. 46, no. 1, pp. 155–159, January 2010.
[6] E. Nuño, L. Basañez, and R. Ortega, "Passivity-based control for bilateral teleoperation: A tutorial," *Automatica*, vol. 47, no. 3, pp. 485–495, March 2011.
[7] A. Rodríguez, L. Basañez, and E. Celaya, "A relational positioning methodology for robot task specification and execution," *IEEE Transactions on Robotics*, vol. 24, no. 3, pp. 600–611, 2008.
[8] A. Rodríguez, L. Basañez, J. E. Colgate, and E. L. Faulring, "A framework for the simulation and haptic display of dynamic systems subject to holonomic constraints," *Int. Jour. Robotics Research*, vol. 29, no. 4, pp. 336–352, 2010.
[9] J. Rosell, C. Vázquez, A. Pérez, and P. Iñiguez, "Motion planning for haptic guidance," *Journal of Intelligent and Robotic Systems*, vol. 53, no. 3, pp. 223–245, nov 2008.
[10] A. Pérez and J. Rosell, "An assisted re-synchronization method for robotic teleoperated tasks," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 2011, pp. 886–891.
[11] H. Portilla and L. Basañez, "Augmented reality tools for enhanced robotics teleoperation systems," in *3DTV Conference*, May 2007, pp. 1–4.
[12] E. Rodriguez-Seda, D. Lee, and M. Spong, "Experimental comparison study of control architectures for bilateral teleoperators," *IEEE Trans. Rob.*, vol. 25, no. 6, pp. 1304–1318, December 2009.
[13] R. Anderson and M. Spong, "Bilateral control of teleoperators with time delay," *IEEE Trans. Autom. Control*, vol. 34, no. 5, pp. 494–501, May 1989.
[14] G. Niemeyer and J. Slotine, "Stable adaptive teleoperation," *IEEE Jour. Oceanic Eng.*, vol. 16, no. 1, pp. 152–162, Jan. 1991.
[15] J. Rosell and I. Vázquez, "Haptic rendering of compliant motions using contact tracking in c-space," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 2005, pp. 4223–4228.
[16] B. Itkowitz, J. Handley, and W. Zhu, "The openhaptics toolkit: a library for adding 3d touch navigation and haptics to graphics applications," in *First Joint Eurohaptics Conf. and Symp. on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 2005, pp. 590–595.