Visual-Based Shared Control for Remote Telemanipulation with Integral Haptic Feedback

Nicolò Pedemonte, Firas Abi-Farraj and Paolo Robuffo Giordano

***Abstract*— Nowadays, one of the largest environmental challenges that European countries must face consists in dealing with the past half century of nuclear waste. In order to optimize maintenance costs, nuclear waste must be sorted, segregated and stored according to its radiation level. Towards this end, in [1] we have recently proposed a visual-based *shared control* architecture meant to facilitate a human operator in controlling two remote robotic arms (one equipped with a gripper and another with a camera) during remote manipulation tasks of nuclear waste via a master device. The operator could then receive force cues informative of the feasibility of her/his motion commands during the task execution. The strategy presented in [1], albeit effective, suffers however from a *locality* issue since the operator can only provide instantaneous velocity commands (in a suitable task space), and receive instantaneous force feedback cues. On the other hand, the ability to 'steer' a *whole* future trajectory in task space, and to receive a corresponding *integral* force feedback along the whole planned trajectory (because of any constraint of the considered system), could significantly enhance the operator's performance, especially when dealing with complex manipulation tasks. The aim of this work is to then extend [1] towards a *planning-based* shared control architecture able to take into account the mentioned requirements. A human/hardware-in-the-loop experiment with simulated slave robots and a real master device is reported for demonstrating the feasibility and effectiveness of the proposed approach.**

I. INTRODUCTION

When operating in hazardous environments such as nuclear sites [2], outer space [3] or underwater [4], robotic telemanipulation becomes essential to guarantee the operator's safety. The panoply of assisted teleoperation systems proposed in the literature includes classical techniques such as virtual fixture [5], [6] or shared control frameworks that merge robot autonomy and human supervisory capabilities [1], [7]–[10]. In a typical shared control architecture, the human operator is in charge of the high-level decision making and all low-level operations are autonomously controlled. Nevertheless, the task allocation between the human operator and the robotic system remains one of the main challenge of assisted teleoperation [11].

In [1], a shared control framework for teleoperating a system on a nuclear site was presented. The work was motivated by the European H2020 "Robotic Manipulation for Nuclear Sort and Segregation" (RoMaNS) project[1]. In the RoMaNS scenario, a human operator has access to a system consisting of two robotic arms, one equipped with a gripper and the other one with a camera, with the goal of approaching and grasping nuclear waste for sort and segregation purposes (see Fig. 1). The shared control architecture presented in [1] consisted in an instantaneous motion control of the gripper manipulator for approaching the target object, and of corresponding instantaneous force cues informing the operator about the feasibility of her/his (again instantaneous) motion commands. Due to the complexity of its motion, the camera was instead autonomously controlled. The main limitation of the architecture presented in [1] lies in its 'local' nature, which does not provide the operator with the possibility of modifying the future behavior of the robot motion, nor of receiving force cues informative about the future consequences of her/his actions. On the other hand, the possibility of 'affecting the future' (i.e., over some future time window) and to receive a corresponding sensible force feedback would be of great importance for facilitating the operator's task, especially in complex manipulation environments.

Fig. 1: Left figure: containers with unknown (or partially known) contents and mixed contamination levels that need to be examined, sorted and separated. Right figure: new special storage containers where contaminated waste must be placed.

In order to cope with this issue, in this paper we consider the same scenario of [1] and extend it by including the possibility of acting on a whole future trajectory for the gripper. In particular, rather than controlling some DOFs of the gripper manipulator arm and obtaining an instantaneous force feedback w.r.t. any pre-defined constraint of the system (in [1] joint limits were considered), the human operator is now given the possibility of modifying *online* the trajectory for approaching the object of interest with the aid of predictive force feedback cues informing about any constraint of interest. The force feedback is denoted as predictive since it informs the operator about the feasibility of her/his planned (future) trajectory against the system constraints, and thus is evaluated on a planned motion yet to be executed. The concept of predictive haptic feedback was introduced in [10], where it was applied to the problem of shaping the 2D trajectory of a mobile robot via an input device. In this work, the ideas of [10] are instead extended to the case of a human

The authors are with the CNRS at Irisa and Inria Rennes Bretagne Atlantique, Campus de Beaulieu, 35042 Rennes Cedex, France {nicolo.pedemonte,firas.abi-farraj,prg}@irisa.fr

[1] http://www.h2020romans.eu/

operator in partial control of the pose of a 6-dof manipulator with the goal of approaching an object of interest to be later manipulated.

The rest of the paper is organized as follows. In Sect. II the general problem is introduced, while the shared control architecture is described in detail in Sect. III. Section IV reports some experimental results and Section V concludes the paper and discusses some future directions.

II. PROBLEM SETTING

The scenario considered in this paper consists of two 6-dof serial manipulators, one equipped with a monocular (calibrated) camera and the other one with a gripper, aiming at grasping an object of interest (see Fig. 2). We consider three frames of reference: a frame $\mathcal{F}_O$: $\{\boldsymbol{O}_O; \boldsymbol{X}_O, \boldsymbol{Y}_O, \boldsymbol{Z}_O\}$ attached to the object to be grasped, a frame $\mathcal{F}_G : \{\boldsymbol{O}_G; \boldsymbol{X}_G, \boldsymbol{Y}_G, \boldsymbol{Z}_G\}$ attached to the gripper, and a frame $\mathcal{F}_C$: $\{\boldsymbol{O}_C; \boldsymbol{X}_C, \boldsymbol{Y}_C, \boldsymbol{Z}_C\}$ attached to the camera. We assume that $\boldsymbol{Z}_G$ is aligned with the gripper approaching direction, and that (as usual) $\boldsymbol{Z}_C$ is aligned with the camera optical axis.

We let $({}^C\boldsymbol{P}_G, {}^C\boldsymbol{R}_G) \in \mathbb{R}^3 \times SO(3)$ represent the 3D pose of $\mathcal{F}_G$ w.r.t. $\mathcal{F}_C$ expressed in $\mathcal{F}_C$ and, similarly, $({}^C\boldsymbol{P}_O, {}^C\boldsymbol{R}_O) \in \mathbb{R}^3 \times SO(3)$ represent the 3D pose of $\mathcal{F}_O$ w.r.t. $\mathcal{F}_C$ expressed in $\mathcal{F}_C$. In the context of the RoMaNS project, we can assume that an accurate enough 3D model of both the object to be grasped and of the gripper is available beforehand. This allows to leverage any model-based tracker, such as those present in the ViSP library [12], for retrieving *online* a reliable estimation of the camera/object and camera/gripper relative poses in the camera frame.

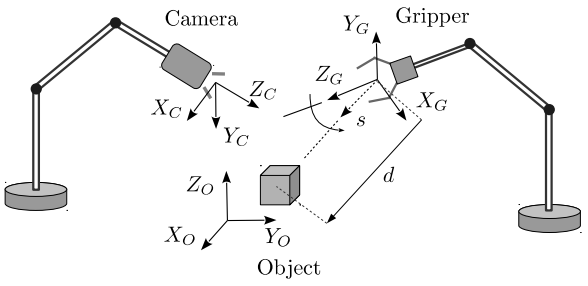

Fig. 2: An illustrative representation of the two 6-dof serial manipulator arms equipped with a camera and a gripper, respectively, together with other quantities of interest.

The goal of the proposed shared control architecture is to (i) let a human operator modify *online* the approaching trajectory towards the object to be grasped via a force-feedback device, (ii) let an autonomous algorithm verify that the commanded (desired) trajectory respects all the possible constraints of the robotic system and, in case it does not, modify it accordingly, (iii) provide the human operator with online force cues informing about any discrepancy between the commanded trajectory and the actual one (modified by the autonomous algorithm in order to cope with the robot constraints), thus informing about the future consequences of the operator's actions and, finally, (iv) let an autonomous algorithm control the camera motion so as to keep a suitable vantage point w.r.t. the observed scene (i.e., both the gripper and the object). We now proceed to detail the components of the shared control architecture.

III. SHARED CONTROL ARCHITECTURE

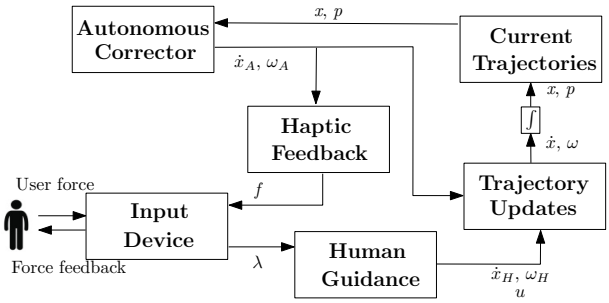

Fig. 3: An illustration of the proposed shared control framework.

Figure 3 illustrates the proposed framework. Given an initial trajectory for the gripper manipulator, the human operator is allowed to modify the trajectory via the input device while the autonomous corrector verifies that all operator's modifications respect the constraints of the system. The haptic feedback is designed in order to assist the user during the task execution, mantain the trajectory away from dangerous configurations and keep the user aware of any autonomously imposed adjustments of the trajectory. The trajectory planning problem for a 6-dof serial manipulator is split into the position and the orientation components. We will then consider two trajectories, $\boldsymbol{\gamma}(t)$ for the gripper position and $\boldsymbol{\eta}(t)$ for the gripper orientation. Both the human operator and the autonomous correcting algorithm will be allowed to modify $\boldsymbol{\gamma}$ and $\boldsymbol{\eta}$ at runtime.

Figure 3 is detailed in this section. In III-A the trajectory planning problem is introduced and the block "Trajectory updates" explained. In III-B, the "Human Guidance" component is presented. Then, section III-C focuses on the autonomous correcting algorithm while section III-D details the haptic feedback. Finally, in III-E, the control of the camera is reviewed.

A. Trajectory planning

The trajectory planning algorithm is performed on the gripper position and orientation $({}^C\boldsymbol{P}_G, {}^C\boldsymbol{R}_G)$ w.r.t. the target object $({}^C\boldsymbol{P}_O, {}^C\boldsymbol{R}_O)$. Both $({}^C\boldsymbol{P}_G, {}^C\boldsymbol{R}_G)$ and $({}^C\boldsymbol{P}_O, {}^C\boldsymbol{R}_O)$ are measured by the camera in order to provide the system with an accurate estimation of the gripper and object poses.

Consider the position problem. We choose to represent the trajectory $\boldsymbol{\gamma}$ as a cubic B-spline. Cubic B-splines are piecewise curves which guarantee the continuity of velocities and accelerations and are a typical choice for representing trajectories [10], [13]. A cubic B-spline can be completely

parametrized by a sequence of scalars U $(knot\ vector)$ and a vector of points $\boldsymbol{x} = [\boldsymbol{x}_0, ..., \boldsymbol{x}_n]$ $(control\ points)$ with $\boldsymbol{x}_i \in \mathbb{R}^3$ and $n \in \mathbb{N}$. The position trajectory will then be defined by:

$$\boldsymbol{\gamma}(\boldsymbol{x}, u) = \sum_{i=0}^{n} \boldsymbol{x}_i B_i(u), \tag{1}$$

where $u \in [0, 1]$ is the independent variable representing the position along the curve ($u = 0$ being the starting point and $u = 1$ the final point) and B_i are the basis functions [13], [14]. Note that, given the knot vector, a *finite* number $n + 1$ of control points is sufficient to define the whole trajectory. This property of B-splines is crucial for our application since it reduces an *infinite*-dimensional problem (optimization of a continuous trajectory) to a *finite*-dimensional one (optimization of the $n + 1$ control points).

For the orientation problem, we exploit [15] where an analytical expression for a cubic B-spline quaternion curve in $\mathbb{S}^3$ was presented. Consider the same knot vector U and a vector of control points (quaternions) $\boldsymbol{p} = [\boldsymbol{p}_0, ..., \boldsymbol{p}_n]$, with $\boldsymbol{p}_i \in \mathbb{S}^3$. The whole orientation trajectory is defined by:

$$\boldsymbol{\eta}(\boldsymbol{p}, u) = \boldsymbol{p}_0^{\widetilde{B_0}(u)} \prod_{i=1}^{n} exp(\boldsymbol{\omega}_i \widetilde{B_i}(u)), \tag{2}$$

where $\widetilde{B}$ are cumulative basis functions [15], $\boldsymbol{\omega}_i = log(\boldsymbol{p}_{i-1}^{-1}\boldsymbol{p}_i)$ and $log(\cdot)$ and $exp(\cdot)$ represent respectively the logarithmic map, $log : \mathbb{S}^3 \to \mathbb{R}^3$, and the exponential map, $exp : \mathbb{R}^3 \to \mathbb{S}^3$ (see Appendix A and [16]). The cubic B-spline quaternion curve is shown to respect some important B-spline properties such as $\boldsymbol{C}^1$-continuity and local controllability [15].

As shown in Fig. 3, the human operator and the autonomous corrector can modify the trajectories $\boldsymbol{\gamma}$ and $\boldsymbol{\eta}$ by commanding the linear and angular velocities of the control points of the two trajectories. Hence, the human operator will command $(\dot{\boldsymbol{x}}_H, \boldsymbol{\omega}_H)$ and the autonomous corrector will command $(\dot{\boldsymbol{x}}_A, \boldsymbol{\omega}_A)$. The human and the autonomous contributions are then merged as follows (corresponding to the block "Trajectory updates" in Fig. 3). Concerning the control points of the position trajectory, the following merging rule is implemented:

$$\boldsymbol{x}(t) = \boldsymbol{x}(t_0) + \int_{t_0}^{t} \dot{\boldsymbol{x}}_H(\tau)d\tau + \int_{t_0}^{t} \dot{\boldsymbol{x}}_A(\tau)d\tau, \tag{3}$$

where $\dot{\boldsymbol{x}}_H$ represents the velocity of the control points of $\boldsymbol{\gamma}$ commanded by the human operator, and $\dot{\boldsymbol{x}}_A$ corresponds to the velocity of the control points of $\boldsymbol{\gamma}$ commanded by the autonomous correction.

The equivalent of (3) for the control points of the orientation trajectory is:

$$\boldsymbol{p}(t) = \boldsymbol{p}(t_0) \cdot \boldsymbol{p}_H(t) \cdot \boldsymbol{p}_A(t) \tag{4}$$

where $\boldsymbol{p}_H$ corresponds to the rotation commanded by the human operator, $\boldsymbol{p}_A$ represents the orientation computed by the autonomous corrector and $\cdot$ is the usual quaternion multiplication rule. The quantities $\boldsymbol{p}_H(t)$ and $\boldsymbol{p}_A(t)$ can be obtained from the quaternion integration rule as follows:

$$\begin{aligned} \boldsymbol{p}_H(t) &= exp(\tfrac{1}{2}\textstyle\int_{t_0}^{t} \boldsymbol{\omega}_H(\tau)d\tau) \\ \boldsymbol{p}_A(t) &= exp(\tfrac{1}{2}\textstyle\int_{t_0}^{t} \boldsymbol{\omega}_A(\tau)d\tau) \end{aligned}, \tag{5}$$

where $\boldsymbol{\omega}_H$ represents the angular velocity applied by the operator and $\boldsymbol{\omega}_A$ corresponds to the angular velocity imposed by the corrector.

B. Human Input

In this section, the block "Human Guidance" of Fig. 3 will be detailed. As we stated above, the human operator is given the opportunity of commanding the quantities $\dot{\boldsymbol{x}}_H$ and $\boldsymbol{\omega}_H$ in order to modify the trajectories $\boldsymbol{\gamma}$ and $\boldsymbol{\eta}$ to his/her convenience. Moreover, we also provide the operator with he possibility of moving along the current trajectories backwards and forwards at will. The operator will then generate the commands $[\boldsymbol{\lambda}, \lambda_u]^T \in \mathbb{R}^4$ by controlling the position of the input device along four independent DOFs. The command λ_u permits to the operator to move the gripper manipulator along the planned trajectories when all the autonomous corrections are completed. On the other hand, the command vector $\boldsymbol{\lambda} \in \mathbb{R}^3$ is mapped onto the control point space $\mathbb{R}^7$ in order to modify both trajectories. We now proceed to detail this mapping.

We consider in this work the approaching (pre-grasping) phase. We assume that, during this specific phase, the operator may need to move the gripper around the target object in order to choose the best pose to grasp. Let the two initial trajectories $\boldsymbol{\gamma}_0(\boldsymbol{x}, u)$ and $\boldsymbol{\eta}_0(\boldsymbol{p}, u)$ be defined such that the gripper is, in its final pose, both aligned w.r.t. the target object and at a certain distance r from it, and let $\boldsymbol{Q} : \mathbb{R}^3 \to \mathbb{R}^{7\cdot(n+1)}$ be the nonlinear mapping matrix from the input device space to the control point space, where $\boldsymbol{Q}$ can be decomposed as $\boldsymbol{Q} = [\boldsymbol{Q}_1 \dots \boldsymbol{Q}_i \dots \boldsymbol{Q}_n]^T$. Then, the velocity of the control points $\boldsymbol{x}_i$ and $\boldsymbol{p}_i$ commanded by the human operator are defined as:

$$\dot{\boldsymbol{x}}_{H,i} = k_{H,i}\boldsymbol{Q}_i \cdot \begin{bmatrix} \boldsymbol{\lambda} \\ 0 \end{bmatrix}, \quad \boldsymbol{\omega}_{H,i} = k_{H,i}\boldsymbol{Q}_i \cdot \begin{bmatrix} 0 \\ \boldsymbol{\lambda} \end{bmatrix}, \tag{6}$$

where $k_{H,i} \in \boldsymbol{K}_H = [0, k_{H,1}, ..k_{H,n-1}, 1]$ is the vector of gains. Matrix $\boldsymbol{Q}_i \in \mathbb{R}^{7\times3}$ maps the input commands $\boldsymbol{\lambda}$ into the velocities of the i-th control points of $\boldsymbol{\gamma}(\boldsymbol{x}, u)$ and $\boldsymbol{\eta}(\boldsymbol{p}, u)$. In our framework, following [1], we define $\boldsymbol{Q}_i = [\boldsymbol{m}_{1,i},\ \boldsymbol{m}_{2,i},\ \boldsymbol{m}_{3,i}]$ where $\boldsymbol{m}_{1,i}$, $\boldsymbol{m}_{2,i}$ and $\boldsymbol{m}_{3,i}$ are the three motion directions defined as follows: let $\boldsymbol{s}_i$ be the 3D direction towards the object to be grasped for each control point $\boldsymbol{x}_i$ defined as $\boldsymbol{s}_i = \frac{\boldsymbol{P}_{s,i}}{\|\boldsymbol{P}_{s,i}\|}$, where $\boldsymbol{P}_{s,i} = \boldsymbol{O}_O - \boldsymbol{x}_i$ expressed in the object frame $\mathcal{F}_O$. We then take

$$\boldsymbol{m}_{1,i} = \begin{bmatrix} -[\boldsymbol{s}_i]_\times \boldsymbol{e}_y \\ -\dfrac{{}^G\boldsymbol{R}_{O,i}\boldsymbol{P}_{s,i}\boldsymbol{e}_y}{\| \boldsymbol{P}_{s,i} \|} \end{bmatrix}, \boldsymbol{m}_{2,i} = \begin{bmatrix} [\boldsymbol{s}_i]_\times \boldsymbol{e}_x \\ \dfrac{{}^G\boldsymbol{R}_{O,i}\boldsymbol{P}_{s,i}\boldsymbol{e}_x}{\| \boldsymbol{P}_{s,i} \|} \end{bmatrix},$$
$$\boldsymbol{m}_{3,i} = \begin{bmatrix} \boldsymbol{0} \\ {}^G\boldsymbol{R}_{O,i}\boldsymbol{s}_i \end{bmatrix}, \tag{7}$$

with $\boldsymbol{e}_x = [1\ 0\ 0]^T$, $\boldsymbol{e}_y = [0\ 1\ 0]^T$. As explained in [1], these three motion directions allow to control the gripper distance and alignment w.r.t. the target object with an intuitive physical meaning. In particular, $\boldsymbol{m}_{1,i}$ and $\boldsymbol{m}_{2,i}$ correspond to two coordinated motions (linear/angular velocity) that displace the gripper over a sphere centered at the target object and $\boldsymbol{m}_{3,i}$ realizes a rotation about $\boldsymbol{s}_i$, (see 4).

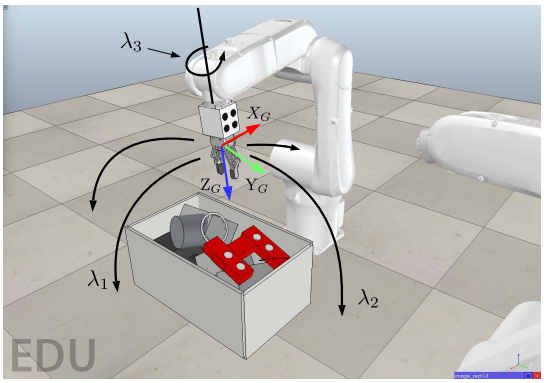

Fig. 4: Visualization of the 3 motion directions actuated by the human operator when commanding the input vector $\boldsymbol{\lambda}$.

The mapping (6) applies to all control points $\boldsymbol{x}_i$ and $\boldsymbol{p}_i$ such that $i \geq j_k$ with $j_k \in \mathbb{N} \leq n$. The index j_k can be readily found by implementing the classical algorithm for determining the knot span $[j_{k-1}, j_k)$ corresponding to the current value of the independent variable u [13], [14]. Given a certain value of u, for $i \geq j_k$, modifying the control points $\boldsymbol{x}_i$ and $\boldsymbol{p}_i$ will not change the current pose of the gripper manipulator. Hence, only the part of the two trajectories yet to be covered by the gripper/manipulator can be modified by the operator. Note that the control points $\boldsymbol{x}_0$ and $\boldsymbol{p}_0$ never move (being $k_{H,0}$ in (6) equal to 0).

C. Autonomous Correction

In this section, the block "Autonomous Corrector" of Fig. 3 will be detailed. In our framework, the goal of the autonomous algorithm is to ensure that the current trajectories respect any predefined constraint, e.g., joint limits, singularities, or collision avoidance. Let $H(\boldsymbol{q})$ be the cost function encoding such constraint(s) and $\boldsymbol{q} \in \mathbb{R}^6$ the joint vector of the manipulator carrying the gripper. The objective of the autonomous corrector is to modify the vectors of control points $\boldsymbol{x}$ and $\boldsymbol{p}$ in order to minimize the cost function $H(\boldsymbol{q})$ along the whole trajectories $\boldsymbol{\gamma}$ and $\boldsymbol{\eta}$. For a given u, the trajectories $\boldsymbol{\gamma}(\boldsymbol{x}, u)$ and $\boldsymbol{\eta}(\boldsymbol{p}, u)$ correspond to a specific robot configuration $\boldsymbol{q}(u)$ via the robot inverse kinematics. Letting $\boldsymbol{J}_P(\boldsymbol{q})$ and $\boldsymbol{J}_O(\boldsymbol{q})$ be the geometric Jacobian matrices related to the position and the orientation of the gripper, the following then holds

$$\begin{bmatrix} \dfrac{\partial \boldsymbol{\gamma}(\boldsymbol{x},u)}{\partial t} \\ \dfrac{\partial \boldsymbol{\eta}(\boldsymbol{p},u)}{\partial t} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial \boldsymbol{\gamma}(\boldsymbol{x},u)}{\partial \boldsymbol{q}(u)} \\ \dfrac{\partial \boldsymbol{\eta}(\boldsymbol{p},u)}{\partial \boldsymbol{q}(u)} \end{bmatrix} \frac{\partial \boldsymbol{q}(u)}{\partial t} = \\ = \begin{bmatrix} \boldsymbol{J}_P(\boldsymbol{q}(u)) \\ \boldsymbol{M}_{QP}(\boldsymbol{p}) \cdot \boldsymbol{J}_O(\boldsymbol{q}(u)) \end{bmatrix} \dot{\boldsymbol{q}}(u) \tag{8}$$

where $\boldsymbol{M}_{QP}(\boldsymbol{p})$ is obtained from the quaternion propagation rule (see Appendix B). For each position u along the trajectories $\boldsymbol{\gamma}(\boldsymbol{x}, u)$ and $\boldsymbol{\eta}(\boldsymbol{p}, u)$, one also has $H(\boldsymbol{q}) = H(\boldsymbol{q}(u)) = H(\boldsymbol{\gamma}(\boldsymbol{x}, u), \boldsymbol{\eta}(\boldsymbol{p}, u))$. In order to minimize the cost function $H(\boldsymbol{\gamma}(\boldsymbol{x}, u), \boldsymbol{\eta}(\boldsymbol{p}, u))$ along the whole trajectories, the autonomous correction $\dot{\boldsymbol{x}}_A$ and $\boldsymbol{\omega}_A$ are then defined as the negative partial derivative of the cost function w.r.t. $\boldsymbol{\gamma}(\boldsymbol{x}, u)$ and $\boldsymbol{\eta}(\boldsymbol{p}, u)$, respectively, see also [10].

Consider the position problem. The trajectory $\boldsymbol{\gamma}$ will be autonomously modified according to the following update rule

$$\frac{\partial \boldsymbol{\gamma}(\boldsymbol{x}(t), u)}{\partial t} = -\int_\gamma \left(\frac{\partial H(\boldsymbol{\gamma}(\boldsymbol{x}, u), \boldsymbol{\eta}(\boldsymbol{p}, u))}{\partial \boldsymbol{\gamma}(\boldsymbol{x}, u)} \right)^T du. \tag{9}$$

The velocity $\partial \boldsymbol{\gamma}(\boldsymbol{x}(t), u)/\partial t$ is mapped onto the $\mathbb{R}^{3(n+1)}$ space of control points by using the pseudo-inverse $(\partial \boldsymbol{\gamma}(\boldsymbol{x}, u)/\partial \boldsymbol{x})^\dagger$ in order to invert the relation

$$\frac{\partial \boldsymbol{\gamma}(\boldsymbol{x}, u)}{\partial t} = \frac{\partial \boldsymbol{\gamma}(\boldsymbol{x}, u)}{\partial \boldsymbol{x}} \dot{\boldsymbol{x}}.$$

Here $\dot{u} = 0$ because we assume that the operator does not move the robot along the trajectories while the autonomous correction is in progress. The autonomous correction in terms of control points can then be expressed by:

$$\dot{\boldsymbol{x}}_A = -\int_\gamma \left(\frac{\partial \boldsymbol{\gamma}(\boldsymbol{x}, u)}{\partial \boldsymbol{x}} \right)^\dagger \left(\frac{\partial H(\boldsymbol{q}(u))}{\partial \boldsymbol{q}(u)} \left(\frac{\partial \boldsymbol{\gamma}(\boldsymbol{x}, u)}{\partial \boldsymbol{q}(u)} \right)^\dagger \right)^T du, \tag{10}$$

where $\partial \boldsymbol{\gamma}(\boldsymbol{x}, u)/\partial \boldsymbol{q}(u) = \boldsymbol{J}_P(\boldsymbol{q}(u))$ from (8). The term $(\partial \boldsymbol{\gamma}(\boldsymbol{x}, u)/\partial \boldsymbol{x})^\dagger$ can be easily computed thanks to the basic B-spline properties (see Appendix B). The computation of $\partial H(\boldsymbol{q}(u))/\partial \boldsymbol{q}(u)$ is straightforward and depends on the particular choice of function $H(\boldsymbol{q})$.

In the same spirit, the trajectory $\boldsymbol{\eta}$ for the orientation problem are autonomously modified as follows:

$$\frac{\partial \boldsymbol{\eta}(\boldsymbol{x}(t), u)}{\partial t} = -\int_\eta \left(\frac{\partial H(\boldsymbol{\gamma}(\boldsymbol{x}, u), \boldsymbol{\eta}(\boldsymbol{p}, u))}{\partial \boldsymbol{\eta}(\boldsymbol{p}, u)} \right)^T du, \tag{11}$$

which leads to:

$$\dot{\boldsymbol{p}}_A = -\int_\eta \left(\frac{\partial \boldsymbol{\eta}(\boldsymbol{p}, u)}{\partial \boldsymbol{p}} \right)^\dagger \left(\frac{\partial H(\boldsymbol{q}(u))}{\partial \boldsymbol{q}(u)} \left(\frac{\partial \boldsymbol{\eta}(\boldsymbol{p}, u)}{\partial \boldsymbol{q}(u)} \right)^\dagger \right)^T du, \tag{12}$$

where $\partial \boldsymbol{\eta}(\boldsymbol{p}, u)/\partial \boldsymbol{q}(u) = \boldsymbol{M}_{QP}(\boldsymbol{p}) \cdot \boldsymbol{J}_O(\boldsymbol{q}(u))$ from (8). The term $\partial \boldsymbol{\eta}(\boldsymbol{p}, u)/\partial \boldsymbol{p}$ is described explicitely in Appendix B. Vector $\boldsymbol{\omega}_A$ can be easily computed from (12) as $\boldsymbol{\omega}_A = log(\dot{\boldsymbol{p}}_A)$. The autonomous correction $\dot{\boldsymbol{x}}_A$ and $\boldsymbol{\omega}_A$ are then merged with the human operator's modifications (6) as described in (3) and (4).

D. Haptic Feedback

The haptic feedback is designed in order to assist the operator during the task execution and inform him/her about how well the system is following the commanded motion $\boldsymbol{\lambda}$. Specifically, the operator must be aware if any modification is being applied to the trajectories.

Following the classical *bilateral force-feedback* framework [17], [18], we then assume the presence of a *master device* upon which the operator can act for sending the commands $\boldsymbol{\lambda}$ to the *slave side* (the gripper/manipulator arm) and receiving force feedback cues. The master device is modeled as a generic (gravity pre-compensated) mechanical system

$$\boldsymbol{M}(\boldsymbol{y}_M)\ddot{\boldsymbol{y}}_M + \boldsymbol{C}(\boldsymbol{y}_M, \dot{\boldsymbol{y}}_M)\dot{\boldsymbol{y}}_M = \boldsymbol{\tau} + \boldsymbol{\tau}_h \tag{13}$$

where $\boldsymbol{y}_M \in \mathbb{R}^m$ is the device configuration vector (with same dimension as the human commands $\boldsymbol{\lambda}$), $\boldsymbol{M}(\boldsymbol{y}_M) \in \mathbb{R}^{m\times m}$ is the positive-definite and symmetric inertia matrix, $\boldsymbol{C}(\boldsymbol{y}_M, \dot{\boldsymbol{y}}_M) \in \mathbb{R}^{m\times m}$ accounts for Coriolis/centrifugal terms, and $\boldsymbol{\tau}, \boldsymbol{\tau}_h \in \mathbb{R}^m$ are the control and human forces, respectively.

The human control actions are implemented by setting

$$\boldsymbol{\lambda} = \boldsymbol{K}_\lambda \boldsymbol{y}_M, \tag{14}$$

with $\boldsymbol{K}_\lambda \in \mathbb{R}^{m\times m}$ being a diagonal matrix of positive scaling factors. This coupling then allows the operator to directly affect the control points of the planned trajectories along the motion directions $\boldsymbol{m}_i$ by adjusting the position of the master device (Fig. 5). The force feedback is instead designed as

$$\boldsymbol{\tau} = -\boldsymbol{B}_M \dot{\boldsymbol{y}}_M - \boldsymbol{K}_M \boldsymbol{y}_M + \boldsymbol{f}. \tag{15}$$

Here, $\boldsymbol{B}_M \in \mathbb{R}^{m\times m}$ is a positive definite damping matrix

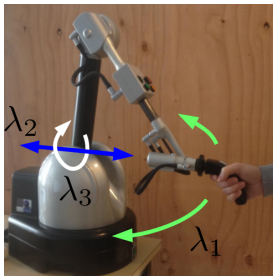

Fig. 5: Representation of $\boldsymbol{\lambda}$ on the haptic device.

for stabilizing the haptic device and $\boldsymbol{K}_M \in \mathbb{R}^{m\times m}$ is a positive definite diagonal matrix meant to implement a 'soft spring' centered at the device rest position[2]. Vector $\boldsymbol{f} = [\dots f_i \dots]^T \in \mathbb{R}^m$ represents instead the force cues provided to the human operator: as explained, the design of these cues is aimed at informing the operator about the feasibility of each trajectory update w.r.t. possible constraints/limitations of the gripper/arm system such as, for instance, proximity to joint limits, to singularities, self-collisions or to collisions with the surrounding environment. We now proceed to detail the cueing algorithm which will be then exploited in Sect. IV.

The haptic feedback must inform the operator about how well the actual trajectories $\boldsymbol{\gamma}(\boldsymbol{x}, u)$ and $\boldsymbol{\eta}(\boldsymbol{p}, u)$ are following his/her commands $\boldsymbol{\lambda}$ despite the possible modifications of the autonomous corrector component. In order to obtain this goal, we map the autonomous corrector's commanded velocities $\dot{\boldsymbol{x}}_A$ and $\boldsymbol{\omega}_A$ onto the human input command space using the pseudo-inverse of the mapping matrix $\boldsymbol{Q}$ introduced in III-B. Be $\boldsymbol{\Pi} = [\boldsymbol{\Pi}_1 \dots \boldsymbol{\Pi}_i \dots \boldsymbol{\Pi}_n]^T \in \mathbb{R}^{7(n+1)\times 1}$, with $\boldsymbol{\Pi}_i = [\dot{\boldsymbol{x}}_{A,i}, \ \boldsymbol{\omega}_{A,i}]^T \in \mathbb{R}^{7\times 1}$. The haptic cues $\boldsymbol{f}$ are then defined as:

$$\boldsymbol{f} = -\boldsymbol{K}_\tau \boldsymbol{Q}^\dagger \boldsymbol{\Pi}, \tag{16}$$

where $\boldsymbol{K}_\tau$ is a constant gain matrix. The forces $\boldsymbol{f}$ will push the operator away from any dangerous configuration with an intensity proportional to $H(\boldsymbol{q})$ along the directions of the trajectory modifications imposed by the autonomous corrector. In conclusion, the haptic cues will provide the operator with a high-level information about the feasibility of his/her commanded trajectories w.r.t. the system constraints along a *future time window* (the time extension of the trajectory itself), rather than by providing instantaneous cues as previously proposed in [1].

As a final step, we make use of the *passive set-position modulation* (PSPM) algorithm [19] for coping with the typical stability issues of any bilateral force feedback loop because of communication delays, packet losses, master/slave kinematic/dynamic dissimilarities, and other shortcomings. To this end, let $\boldsymbol{\psi}(t) = \boldsymbol{K}_M^{-1}\boldsymbol{f}(t)$ and rearrange (15) as

$$\boldsymbol{\tau} = -\boldsymbol{B}_M \dot{\boldsymbol{y}}_M - \boldsymbol{K}_M(\boldsymbol{y}_M - \boldsymbol{\psi}(t)). \tag{17}$$

The PSPM action modulates the (arbitrary) signal $\boldsymbol{\psi}(t)$ into a possibly attenuated version $\bar{\boldsymbol{\psi}}(t)$ which, when plugged into (17), ensures input/output stability (passivity) of the master device. This is then sufficient for guaranteeing stability (passivity) of the overall bilateral teleoperation, see [19] for more details and [1], [20] for some recent examples of the use of the PSPM algorithm.

E. Camera Control

The goal of the vision system goal is to retrieve the object and gripper poses required by the shared control framework presented above. The camera manipulator is then controlled using an IBVS approach so as to keep a suitable vantage point w.r.t. the observed scene, i.e., both the gripper and the object. All the details on this component of the architecture can be found in [1].

IV. EXPERIMENTAL RESULTS

In this section we report the results of a set of experiments designed to illustrate and validate the shared control architecture presented above and to prove the effectiveness of the force cues. The master side consists of the Haption VIRTUOSE 6D haptic device[3], a high performance force feedback device with three translational DOFs and three rotational DOFs. The maximum force/torque is about 30 [N]/3 [Nm], the workspace has a spherical-like shape with an approximated radius of 0.9 [m], and the device exchanges data over ethernet with a control PC at 1 kHz. Four DOFs of the Haption device were left unconstrained. Three of them are needed for actuating the $m = 3$ motion directions $\boldsymbol{m}_i$

[2]Therefore, by means of this spring the user will be provided with a perception of the distance from a zero-commanded velocity.

[3]`www.haption.com`.

detailed in (7). As explained in Sect. III-D, the position of the master along these three DOFs was coupled to the input commands $\boldsymbol{\lambda}$ via (14). The fourth DOF is used by the operator to move the gripper along the trajectory backwards and forwards. The remaining two DOFs were constrained via software to a constant value. The slave side consists of two 6-DOF Viper S850 robotic arms carrying the gripper and the camera, and simulated in the popular V-REP environment[4]. The poses of the gripper and of the target object in the camera frame were reconstructed by feeding the model-based ViSP tracker [12] with the segmented location of some fiducial markers acquired at 30 Hz, see Fig. 6.

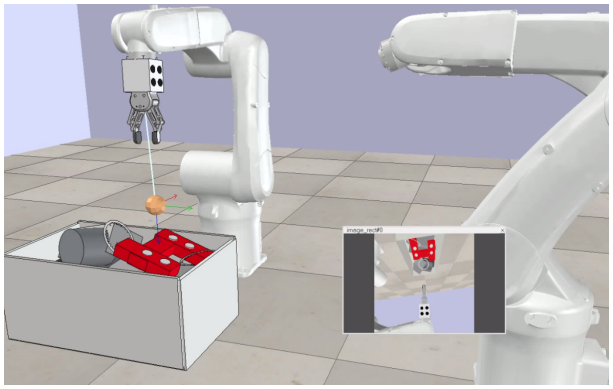

Fig. 6: The two simulated 6-DOF manipulators carrying the camera and the gripper and, in the small window on the bottom-right, the camera image. The yellow sphere represents the final pose of the gripper, whereas the cyan line represents the initial trajectory.

We considered in this work proximity to joints limits and to singularities as representative constraints of the gripper/manipulator arm system. The function $H(\boldsymbol{q})$ in (9) and (11) is then expressed as

$$H(\boldsymbol{q}) = H_{JL}(\boldsymbol{q}) + H_S(\boldsymbol{q}), \tag{18}$$

where $H_{JL}(\boldsymbol{q})$ represents the cost function for the joint limit constraint and $H_S(\boldsymbol{q})$ the cost function for the singularity constraint. The joint limit cost function is defined as a classical quadratic penalty cost

$$H_{JL}(\boldsymbol{q}) = k_{JL} \sum_{i=1}^{6} h_i(q),$$

with

$$h_i(q) = \begin{cases} (q_i - (q_{i,max} - q_{th}))^2, & \text{if } q_i \geq q_{i,max} - q_{th} \\ (q_{i,min} + q_{th} - q_i)^2, & \text{if } q_i \leq q_{i,min} + q_{th} \\ 0, & \text{otherwise} \end{cases},$$

where $(q_{i,max},\ q_{i,min})$ are the maximum/minimum range for the i-th joint and q_{th} is a user-defined threshold defining the activation region inside which the user will receive a force feedback. The cost function related to the singularity constraint is instead defined as the inverse of the determinant of the geometric Jacobian matrix $\boldsymbol{J}$:

$$H_S(\boldsymbol{q}) = \begin{cases} \frac{k_S}{\det(\boldsymbol{J})}, & \text{if } \det(\boldsymbol{J}) \leq \epsilon \\ 0, & \text{otherwise} \end{cases},$$

where ϵ is a user-defined constant parameter, and k_S and k_{JL} are scaling factors.

We now report two different experiments conducted for validating the described shared control architecture. The reader is also encouraged to watch the video attachment for a better understanding of the results of the experimental session.

A. First Experiment

The first experiment is designed in order to show the main features of our approach, i.e., (i) the possibility of modifying the current trajectories ($\boldsymbol{\gamma}$ and $\boldsymbol{\eta}$) by actuating the m motion directions, (ii) the possibility of moving along the current trajectories, and (iii) the assistance provided by the haptic feedback when the trajectories are in proximity of joint limits and singularities. The experiment can be split into three phases where, in each of them, a single motion command is activated. The trajectories are modified until they reach a joint limit/singular configuration. In the first phase (7 [s] $\leq t \leq 34$ [s]), the operator sends the motion command λ_2. In the second phase (51 [s] $\leq t \leq 57$ [s]), λ_1 is activated. Finally, in the third phase (68 [s] $\leq t \leq 88$ [s]), the operator commands λ_3. Between the first and the second phase (43 [s] $\leq t \leq 47$ [s]) and between the second and the third phase (64 [s] $\leq t \leq 69$ [s]) the operator commands the manipulator to move forward along the trajectory.

Figures 7(a–c) report the experimental results. In particular, the behavior of the haptic cues $\boldsymbol{f}$ (Fig. 7b), the value of the cost function $H(\boldsymbol{q})$ (Fig. 7c), and the user commands $\boldsymbol{\lambda}$ and λ_A (Fig. 7a) are shown over time. The three phases show the system reaction to the three possible motion commands. It is worth noting that in the first two phases the highest force cue is produced along the direction of the commanded motion. In the third phase, on the other hand, the user activates the input λ_3 but the haptic feedback is mainly projected along the motion direction $\boldsymbol{m}_1$ (see Fig. 7b). This behavior might not seem intuitive but it can be explained by the gradient-based algorithm: in this situation, the fastest way to reduce the cost function was to move along the direction $\boldsymbol{m}_1$ rather than pushing back the user along $\boldsymbol{m}_3$. Finally, it is interesting to observe how the haptic cues in Fig. 7b are activated when the cost function $\boldsymbol{H}(\boldsymbol{q}) \geq 0$ (Fig. 7c). The algorithm will autonomously correct the trajectories $\boldsymbol{\gamma}$ and $\boldsymbol{\eta}$ until $H(\boldsymbol{q})$ is minimized.

B. Second Experiment

The second experiment is meant to assess the effectiveness of the haptic feedback during a possible task execution. In this case, the human operator commands a random trajectory modification for purposely reaching joint limits and experiencing the corresponding force feedback. Figures 8(a–c) report the experimental results. As it can be observed in

[4] www.coppeliarobotics.com.

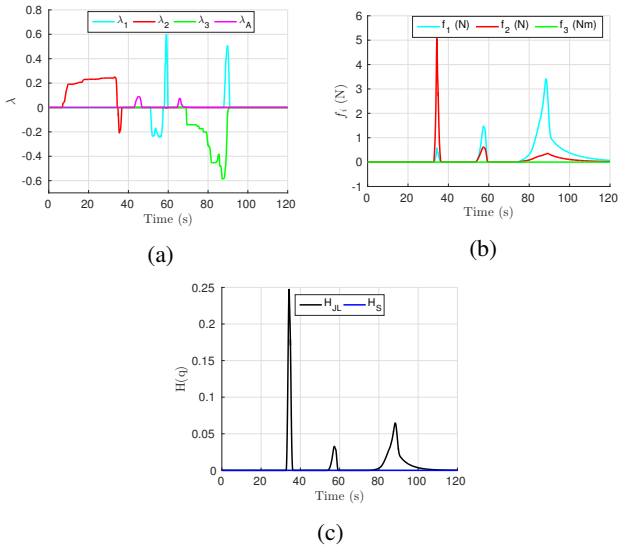

Fig. 7: Results of the first experiment.(a) Behavior of the operator's commands $\boldsymbol{\lambda}, \lambda_A$ for actuating the motion directions $\boldsymbol{m}_i$ and moving along the trajectories. (b) Behavior of the three force cues f_i displayed to the human operator during the robot motion; (c) behavior of the scalar cost functions $H(\boldsymbol{q})_{JL}$ for joint limits and $H(\boldsymbol{q})_S$ for singularities, quantifying the proximity to the relative constraint.

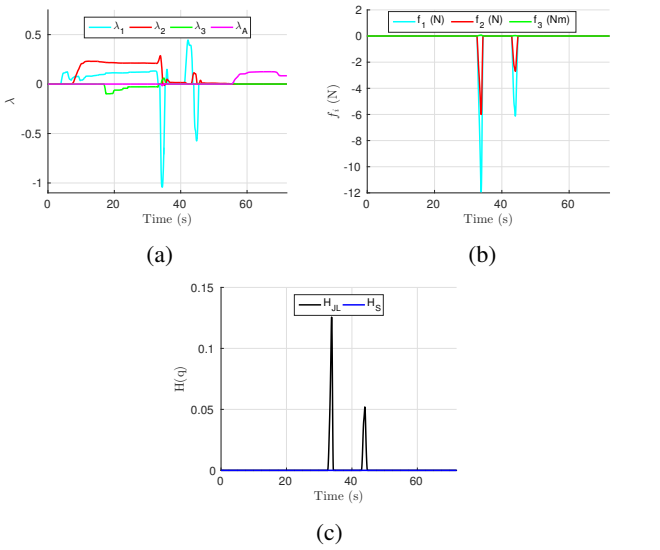

Fig. 8: Results of the second experiment. (a) Behavior of the operator's commands $\boldsymbol{\lambda}, \lambda_A$ for actuating the motion directions $\boldsymbol{m}_i$ and moving along the trajectories. (b) Behavior of the three force cues f_i displayed to the human operator during the robot motion; (c) behavior of the scalar cost functions $H(\boldsymbol{q})_{JL}$ for joint limits and $H(\boldsymbol{q})_S$ for singularities, quantifying the proximity to the relative constraint.

Fig. 8a, the operator initially sends the motion commands λ_1 (4 [s] $\leq t \leq 33$ [s]) and λ_2 (7 [s] $\leq t \leq 33$ [s]) and successively also λ_3 (17 [s] $\leq t \leq 33$ [s]). At $t \simeq 33$ [s], a joint limit is reached and $H(\boldsymbol{q})$ rapidly increases (see Fig. 8c). The haptic cues (Fig. 8b) are consequently activated and the operator is forced to move backwards along $\boldsymbol{\lambda}$. This backward motion is particularly evident along λ_1, where the haptic feedback is the highest. The operator tries one more time to move forward along λ_1 but she/he is repulsed again by the haptic feedback (at $t \simeq 43$ [s]). Eventually, the operator commands the robot to move along the trajectory and the gripper reaches the final pose (55 [s] $\leq t \leq 73$ [s]).

In conclusion, the designed force cues proved to be both informative and efficient while assisting the user in "steering" the trajectories away from undesired configurations by either moving back along the operator's commanded direction, or by manoeuvring over the other available motion directions as a function of the magnitude (and sign) of the received haptic information.

V. CONCLUSIONS

This paper extended the work presented in [1] to a planning-based shared control architecture. In the proposed framework, the human operator is given the possibility of modifying two initial trajectories ($\boldsymbol{\gamma}_0$ for the position of the gripper/manipulator and $\boldsymbol{\eta}_0$ for its orientation) via a haptic device. In parallel, an autonomous corrector verifies that all human modifications respect the system constraints, such as (in the reported cases) joint limits and singularities. When the human-modified trajectories are in proximity of any of these constraints, the autonomous system is able to steer the trajectories away from the constraints by minimizing the associated cost function. The autonomous intervention is then converted into a set of force cues fed to the human operator in order to keep him/her aware of the feasibility of his/her commanded motions, and informed about any change autonomously imposed to the current trajectories. The effectiveness of our proposed shared control architecture with integral haptic cues was then demonstrated in a set of experiments presented in the last section of the paper.

In the next future, two real 6-dof manipulators equipped with a gripper/camera will be employed to validate the planning-based shared control architecture. To this end, we will use vision-based reconstruction methods able to handle more complex scenes than the ones considered in this work. Furthermore, we are interested in evaluating other criteria for generating force cues other than the system intrinsic constraints, such as, for instance manipulability or collision avoidance measures. Finally, we also planning to run a set of user studies in order to assess the effectiveness of the described algorithm in a principled way.

APPENDIX A

$\triangleright$ Consider $\boldsymbol{p} = [w, \boldsymbol{v}]^T \in \mathbb{S}^3$. The logarithmic map $log : \mathbb{S}^3 \rightarrow \mathbb{R}^3$ is defined by $log(\boldsymbol{p}) = (\arccos w \cdot \dfrac{\boldsymbol{v}}{||\boldsymbol{v}||}) \in \mathbb{R}^3$ [16].

$\triangleright$ Consider $\boldsymbol{v} \in \mathbb{R}^3$. The exponential map $exp : \mathbb{R}^3 \rightarrow \mathbb{S}^3$ is defined by $exp(\boldsymbol{v}) = \left[\cos||\boldsymbol{v}||, \dfrac{sin||\boldsymbol{v}||}{||\boldsymbol{v}||} \cdot \boldsymbol{v}\right]^T$ if $\boldsymbol{v} \neq \boldsymbol{0}$, else $exp(\boldsymbol{v}) = [1, \boldsymbol{0}]^T$ [16].

APPENDIX B

▷ For the position trajectory, the term $\dfrac{\partial \boldsymbol{\gamma}(\boldsymbol{x},u)}{\partial \boldsymbol{x}} = \left[\dfrac{\partial \boldsymbol{\gamma}(\boldsymbol{x},u))}{\partial \boldsymbol{x}_1}, ... \dfrac{\partial \boldsymbol{\gamma}(\boldsymbol{x},u)}{\partial \boldsymbol{x}_i}, ... \dfrac{\partial \boldsymbol{\gamma}(\boldsymbol{x},u)}{\partial \boldsymbol{x}_n}\right]$ where $\dfrac{\partial \boldsymbol{\gamma}(\boldsymbol{x},u)}{\partial \boldsymbol{x}_i} = \begin{bmatrix} B_i & 0 & 0 \\ 0 & B_i & 0 \\ 0 & 0 & B_i \end{bmatrix}$, remembering that for a cubic B-spline curve, no more than 4 basis function B are not null for each knot span [13].

▷ For the orientation trajectory, the term $\dfrac{\partial \boldsymbol{\eta}(\boldsymbol{p},u)}{\partial \boldsymbol{p}} = \left[\dfrac{\partial \boldsymbol{\eta}(\boldsymbol{p},u)}{\partial \boldsymbol{p}_1}, ... \dfrac{\partial \boldsymbol{\eta}(\boldsymbol{p},u)}{\partial \boldsymbol{p}_i}, ... \dfrac{\partial \boldsymbol{\eta}(\boldsymbol{p},u)}{\partial \boldsymbol{p}_n}\right]$ where $\dfrac{\partial \boldsymbol{\eta}(\boldsymbol{p},u)}{\partial \boldsymbol{p}_i} = \boldsymbol{p}_0^{\widetilde{B_0}(u)} \prod_{j=1}^{i-1} exp(\boldsymbol{\omega}_j \widetilde{B_j}(u)) \cdot \boldsymbol{D}_1 \cdot \prod_{j=i+1}^{n} exp(\boldsymbol{\omega}_j \widetilde{B_j}(u)) + \boldsymbol{p}_0^{\widetilde{B_0}(u)} \prod_{j=1}^{i} exp(\boldsymbol{\omega}_j \widetilde{B_j}(u)) \cdot \boldsymbol{D}_2 \cdot \prod_{j=i+2}^{n} exp(\boldsymbol{\omega}_j \widetilde{B_j}(u))$, with $\boldsymbol{\omega}_j = log(\boldsymbol{p}_{j-1}^{-1}\boldsymbol{p}_j)$ and $\boldsymbol{D}_k = \dfrac{\partial exp(\cdot)}{\partial log(\cdot)} \dfrac{\partial log(\cdot)}{\partial \boldsymbol{\rho}} \mathbf{d}_k$, with $\boldsymbol{\rho} = \boldsymbol{p}_{j-1}^{-1}\boldsymbol{p}_j$.

Consider $\boldsymbol{v} = [v_1, v_2, v_3]^T \in \mathbb{R}^3$, from the definition of $exp(\boldsymbol{v})$, it follows that $\dfrac{\partial exp(\boldsymbol{v})}{\partial \boldsymbol{v}} = \begin{bmatrix} -\frac{s}{\|v\|}v_1 & -\frac{s}{\|v\|}v_2 & -\frac{s}{\|v\|}v_3 \\ \left(\frac{c}{\|v\|^2} - \frac{s}{\|v\|^3}\right)v_1^2 + \frac{s}{\|v\|} & \left(\frac{c}{\|v\|^2} - \frac{s}{\|v\|^3}\right)v_1 v_2 & \left(\frac{c}{\|v\|^2} - \frac{s}{\|v\|^3}\right)v_1 v_3 \\ \left(\frac{c}{\|v\|^2} - \frac{s}{\|v\|^3}\right)v_1 v_2 & \left(\frac{c}{\|v\|^2} - \frac{s}{\|v\|^3}\right)v_2^2 + \frac{s}{\|v\|} & \left(\frac{c}{\|v\|^2} - \frac{s}{\|v\|^3}\right)v_2 v_3 \\ \left(\frac{c}{\|v\|^2} - \frac{s}{\|v\|^3}\right)v_1 v_3 & \left(\frac{c}{\|v\|^2} - \frac{s}{\|v\|^3}\right)v_2 v_3 & \left(\frac{c}{\|v\|^2} - \frac{s}{\|v\|^3}\right)v_3^2 + \frac{s}{\|v\|} \end{bmatrix}$ where $c = \cos \parallel v \parallel$ and $s = \sin \parallel v \parallel$ (see also [16]).

Consider now $\boldsymbol{\rho} = [w, \boldsymbol{v}]^T = [w, v_1, v_2, v_3]^T \in \mathbb{S}^3$, from the definition of $log(\boldsymbol{\rho})$ it follows that $\dfrac{\partial log(\boldsymbol{\rho})}{\partial \boldsymbol{\rho}} = \begin{bmatrix} -\frac{v_1}{\|v\|^2} + \frac{\arccos(w) w v_1}{\|v\|^{\frac{3}{2}}} & \frac{\arccos(w)}{\|v\|} & 0 & 0 \\ -\frac{v_2}{\|v\|^2} + \frac{\arccos(w) w v_2}{\|v\|^{\frac{3}{2}}} & 0 & \frac{\arccos(w)}{\|v\|} & 0 \\ -\frac{v_3}{\|v\|^2} + \frac{\arccos(w) w v_3}{\|v\|^{\frac{3}{2}}} & 0 & 0 & \frac{\arccos(w)}{\|v\|} \end{bmatrix}$.

Finally, from [21], $\mathbf{d}_1 = \dfrac{\partial(\boldsymbol{p}_{i-1}^{-1}\boldsymbol{p}_i)}{\partial \boldsymbol{p}_i} = [1, \mathbf{0}]^T$, $\mathbf{d}_2 = \dfrac{\partial(\boldsymbol{p}_i^{-1}\boldsymbol{p}_{i+1})}{\partial \boldsymbol{p}_i} = -\boldsymbol{p}_i^{-1} \cdot \mathbb{R}\{\boldsymbol{p}_i\}$.

▷ Consider the quaternion $\boldsymbol{p} = \begin{bmatrix} w \\ \boldsymbol{v} \end{bmatrix}$. From the quaternion propagation rule:

$$\dot{\boldsymbol{p}} = \begin{bmatrix} \dot{w} \\ \dot{\boldsymbol{v}} \end{bmatrix} = \begin{bmatrix} -\frac{1}{2}\boldsymbol{v}^T \\ \frac{1}{2}(w\boldsymbol{I}_3 - \boldsymbol{v}\times) \end{bmatrix} \boldsymbol{\omega} = \boldsymbol{M}_{QP}(\boldsymbol{p}) \cdot \boldsymbol{\omega}$$

where $\boldsymbol{\omega} = \boldsymbol{J}_O \dot{\boldsymbol{q}}$ is the angular velocity [22].

ACKNOWLEDGMENTS

This work was supported by the EU H2020 RoMaNS project 645582.

REFERENCES

[1] F. Abi-Farraj, N. Pedemonte, and P. Robuffo Giordano, "A visual-based shared control architecture for remote telemanipulation," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2016.

[2] R. Schilling, "Telerobots in the nuclear industry: a manufacturer's view," *Industrial Robots*, vol. 19, no. 2, pp. 3–4, 1992.

[3] J. Wright, A. Trebi-Ollennu, F. Hartman, B. Cooper, S. Maxwell, J. Yen, and J. Morrison, "Driving a rover on mars using the rover sequencing and visualization program," in *International Conference on Instrumentation, Control and Information Technology (Okayama University, Okayama 2005)*, 2005.

[4] R. R. Murphy, K. L. Dreger, S. Newsome, J. Rodocker, E. Steimle, T. Kimura, K. Makabe, F. Matsuno, S. Tadokoro, and K. Kon, "Use of remotely operated marine vehicles at minamisanriku and rikuzentakata japan for disaster recovery," in *Safety, Security, and Rescue Robotics (SSRR), 2011 IEEE International Symposium on*. IEEE, 2011, pp. 19–25.

[5] L. B. Rosenberg, "Virtual fixtures: Perceptual tools for telerobotic manipulation," in *Virtual Reality Annual International Symposium, 1993., 1993 IEEE*. IEEE, 1993, pp. 76–82.

[6] J. J. Abbott, P. Marayong, and A. M. Okamura, "Haptic virtual fixtures for robot-assisted manipulation," in *Robotics research*. Springer, 2007, pp. 49–64.

[7] C. Masone, A. Franchi, H. H. Bülthoff, and P. Robuffo Giordano, "Interactive Planning of Persistent Trajectories for Human-Assisted Navigation of Mobile Robots," in *2012 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2012, pp. 2641–2648.

[8] K. Hauser, "Recognition, prediction, and planning for assisted teleoperation of freeform tasks," *Autonomous Robots*, vol. 35, no. 4, pp. 241–254, 2013.

[9] H. Boessenkool, D. A. Abbink, C. J. Heemskerk, F. C. van der Helm, and J. G. Wildenbeest, "A task-specific analysis of the benefit of haptic shared control during telemanipulation," *Haptics, IEEE Transactions on*, vol. 6, no. 1, pp. 2–12, 2013.

[10] C. Masone, P. Robuffo Giordano, H. H. Bülthoff, and A. Franchi, "Semi-autonomous Trajectory Generation for Mobile Robots with Integral Haptic Shared Control," in *2014 IEEE Int. Conf. on Robotics and Automation*, 2014, pp. 6468–6475.

[11] T. Inagaki, "Adaptive automation: Sharing and trading of control," *Handbook of cognitive task design*, vol. 8, pp. 147–169, 2003.

[12] E. Marchand, F. Spindler, and F. Chaumette, "ViSP for visual servoing: a generic software platform with a wide class of robot control skills," *IEEE Robotics and Automation Magazine*, vol. 12, no. 4, pp. 40–52, 2005.

[13] L. Biagiotti and C. Melchiorri, *Trajectory Planning for Automatic Machines and Robots*. Springer, 2008.

[14] L. Piegl and W. Tiller, *The NURBS book*. Springer Science & Business Media, 2012.

[15] M.-J. Kim, M.-S. Kim, and S. Y. Shin, "A general construction scheme for unit quaternion curves with simple high order derivatives," in *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*. ACM, 1995, pp. 369–376.

[16] ——, "A compact differential formula for the first derivative of a unit quaternion curve," *Journal of Visualization and Computer Animation*, vol. 7, no. 1, pp. 43–57, 1996.

[17] P. F. Hokayem and M. W. Spong, "Bilateral teleoperation: An historical survey," *Automatica*, vol. 42, no. 12, pp. 2035–2057, 2006.

[18] E. Nuño, L. Basañez, and R. Ortega, "Passivity-based control for bilateral teleoperation: A tutorial," *Automatica*, vol. 47, no. 3, pp. 485–495, 2011.

[19] D. J. Lee and K. Huang, "Passive-set-position-modulation framework for interactive robotic systems," *IEEE Trans. on Robotics*, vol. 26, no. 2, pp. 354–369, 2010.

[20] D. Lee, A. Franchi, H. I. Son, H. H. Bülthoff, and P. Robuffo Giordano, "Semi-Autonomous Haptic Teleoperation Control Architecture of Multiple Unmanned Aerial Vehicles," *IEEE/ASME Trans. on Mechatronics*, vol. 4, no. 18, pp. 1334–1345, 2013.

[21] D. Xu and D. P. Mandic, "The theory of quaternion matrix derivatives," *IEEE Transactions on Signal Processing*, vol. 63, no. 6, pp. 1543–1556, 2015.

[22] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: modelling, planning and control*. Springer Science & Business Media, 2010.