Interactive Planning of Persistent Trajectories for Human-Assisted Navigation of Mobile Robots

Carlo Masone, Antonio Franchi, Heinrich H. Bülthoff, and Paolo Robuffo Giordano

Abstract— **This work extends the framework of bilateral shared control of mobile robots with the aim of increasing the robot autonomy and decreasing the operator commitment. We consider persistent autonomous behaviors where a cyclic motion must be executed by the robot. The human operator is in charge of modifying online some geometric properties of the desired path. This is then autonomously processed by the robot in order to produce an actual path guaranteeing: i) tracking feasibility, ii) collision avoidance with obstacles, iii) closeness to the desired path set by the human operator, and iv) proximity to some points of interest. A force feedback is implemented to inform the human operator of the global deformation of the path rather than using the classical mismatch between desired and executed motion commands. Physically-based simulations, with human/hardware-in-the-loop and a quadrotor UAV as robotic platform, demonstrate the feasibility of the method.**

I. INTRODUCTION

Use of robotic platforms, and in particular mobile robots, has proven to be very successful in addressing a large variety of tasks. However, up to now and for the next foreseeable future, robot autonomy is still limited by the lack of robust and reliable perception/world awareness, and of higher cognitive abilities that would allow making complex decisions in real-world unstructured scenarios. Therefore, it is usually more efficient, if not the only choice, to have human supervisors in charge of high level aspects while robots execute local and well defined tasks. In this context, the well-established framework of *robot shared control* (see, e.g., [1], [2]) represents a promising direction since it allows to merge robot (limited) autonomy and human's supervisory capabilities in an effective way

In the past literature, this idea has been often applied to mobile robots navigating in cluttered environments, see, e.g., [3], [4], [5], [6], [7], [8], [9], [2], with a particular emphasis on *bilateral shared control* architectures with *haptic feedback* for the human operator. Typically, a human operator provides with a haptic device a motion command (desired position, reference velocity) to the robot. The robot executes the command while retaining some autonomy in order, e.g., to avoid obstacles or other dangers. Finally, the loop is closed by rendering on the haptic feedback a force that is proportional to the mismatch between commanded and executed motion in order to increase the operator's situational awareness.

One limitation of this approach is that it requires a high commitment of the human who directly commands the motion, especially when the robot has low autonomy or the assignment is very complex. In this respect, we explore how to extend the framework of bilateral shared control of mobile robots to let the human command some high-level characteristics of the trajectory rather than direct motion inputs. At the same time, the robot must track the commanded path and, whenever needed, modify it to avoid collisions with obstacles or to reach a nearby target. This autonomous deformation is carried on the same parameter space in which the operator's inputs are defined, thus establishing an interaction with the user. In this sense, our deformation approach is different from other well known methods in which artificial forces are applied to a sequence of configurations [10], [11], or where the path and its modification are defined as input functions along the admissible direction of motion [12].

In this work we focus on the case of persistent autonomous tasks, i.e., tasks that require the robot to execute a cyclic motion on a closed geometric path (e.g., area monitoring, boundary patrolling, object moving). While previous studies mainly considered the problem of guaranteeing circulation on a static path using speed controllers [13], [14] or vector fields [15], [16], [17], [18], in our scenario both the human supervisor and the robot can change the shape of the path.

Finally, the bilateral component of the human-robot interaction is realized by feeding back to the human a force cue informative of the *global* deformation acting on the desired path rather than on a *local* mismatch between commanded and executed position/velocity.

Summarizing, the main contributions of this work are: i) extending previous works on autonomous persistent tasks by including a human operator in-the-loop in order to exploit of her/his superior cognitive skills, ii) broadening the classical shared control approach by endowing the mobile robot with a higher planning autonomy, iii) allowing a human operator to act at the planning level rather than at the motion control level, iv) generating a force cue informative of the global deformation of the desired path rather than of the mismatch between direct motion commands and their execution.

The rest of the paper is organized as follows. In Sec. II the trajectory is defined as a dynamical system whose states are a set of geometric parameters. The bilateral interface for human operations is described in Sec. III. Section IV introduces the online planning of alternative routes in presence of obstacles and Sec. VI contains human/hardware-in-the-loop simulations using an Unmanned Aerial Vehicle (UAV).

C. Masone, A. Franchi, and P. Robuffo Giordano are with the Max Planck Institute for Biological Cybernetics, Spemannstraße 38, 72076 Tübingen, Germany {antonio.franchi,carlo.masone, prg}@tuebingen.mpg.de.

H. H. Bülthoff is with the Max Planck Institute for Biological Cybernetics, Spemannstraße 38, 72076 Tübingen, Germany, and with the Department of Brain and Cognitive Engineering, Korea University, Seoul, 136-713 Korea. E-mail: hhb@tuebingen.mpg.de.

II. INTERACTIVE TRAJECTORY PLANNING AND CONTROL

In our shared control framework a human operator and a mobile robot have to cooperate in order to persistently sweep a cluttered environment. In particular, the robot mission is to track a planar Cartesian trajectory $\boldsymbol{p}(t) \in \mathbb{R}^2$ whose properties are interactively influenced by the human operator that, in turns, receives back a suitable haptic feedback. The first part of this framework consists of the interactive planning and control scheme that is depicted in Fig. 1 and is designed to remain still valid even if the human operator is replaced by a fully-autonomous high-level planner.

The ability to track a sufficiently smooth trajectory is a common property of almost all mobile robots within the scope of this work. This property holds, among the others, for all those systems dynamically feedback linearizable, i.e., differentially flat[1], possessing a Cartesian point in their the flat output, see, e.g., [21] for a detailed list. This also holds for the quadrotor UAV [22] used as case study in the simulations of Sec. VI.

The trajectory $\boldsymbol{p}(t)$ is specified as a geometric path together with a timing law. The geometric path is defined as

$$\boldsymbol{\gamma} : \mathbb{R}^n \times [0, L] \to \mathbb{R}^2, \quad \text{s.t.} \quad (\boldsymbol{x}, s) \mapsto \boldsymbol{\gamma}(\boldsymbol{x}, s),$$

where $\boldsymbol{x} = (x_1 \dots x_n) \in \mathbb{R}^n$ is a vector of shape parameters uniquely defining the curve, L is the curve length (depending on $\boldsymbol{x}$) and s is the arc length parameterizing the position on the curve. The motion law $s_R = s(t)$ ultimately determines how the robot travels the path. We assume $\boldsymbol{\gamma}$ to be a closed curve, i.e., $\boldsymbol{\gamma}(\boldsymbol{x}, 0) = \boldsymbol{\gamma}(\boldsymbol{x}, L)$, and sufficiently smooth everywhere, including the initial/ending point.

The vector of shape parameters $\boldsymbol{x}$ depends on the specific representation chosen for $\boldsymbol{\gamma}$ and determines how the geometric path can be interactively influenced. A large number n of parameters provides a better flexibility in the interaction, however it may exceed the number of quantities that a human operator can reasonably control at once. In order to decrease the complexity of manipulating $\boldsymbol{\gamma}$ to a level appropriate for the human capabilities, we consider a vector $\boldsymbol{y}(\boldsymbol{x}) = (y_1(\boldsymbol{x}) \dots y_m(\boldsymbol{x}))^T \in \mathbb{R}^m$, $m \leq n$, defining m degrees of freedom assigned to the human control. The time variation of $\boldsymbol{y}$ is given by

$$\dot{\boldsymbol{y}} = \left(\frac{\partial y_1}{\partial \boldsymbol{x}}^T \dots \frac{\partial y_m}{\partial \boldsymbol{x}}^T \right)^T \dot{\boldsymbol{x}} = \frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}} \dot{\boldsymbol{x}} = \boldsymbol{G}(\boldsymbol{x}) \dot{\boldsymbol{x}}, \tag{1}$$

where matrix $\boldsymbol{G}(\boldsymbol{x}) \in \mathbb{R}^{m \times n}$ is assumed to have full row-rank so that the m controlled dofs are independent.

The sought interactivity is provided by allowing the human operator to steer the desired value $\boldsymbol{x}_h \in \mathbb{R}^m$ of the shape parameters and select a desired value $\dot{s}_h \in \mathbb{R}$ for the traveling speed along the curve. This is obtained by means of the following dynamical system

$$\dot{s}_h = u_s \tag{2a}$$

$$\dot{\boldsymbol{x}}_h = \boldsymbol{G}^{\dagger} \left(\boldsymbol{x}_h \right) \boldsymbol{u}, \tag{2b}$$

where $u_s \in \mathbb{R}$ and $\boldsymbol{u} \in \mathbb{R}^m$ represent the actual human commands (whose detailed expression will be described in Sec. III), and $\boldsymbol{G}^{\dagger}$ is the pseudo-inverse of matrix $\boldsymbol{G}$. We note that the pseudo-inverse is one of the many possible strategies to map a desired $\dot{\boldsymbol{y}} = \boldsymbol{u}$ into a motion $\dot{\boldsymbol{x}}$ that realizes the given $\dot{\boldsymbol{y}}$. We also note that actual realization of a generic $\dot{\boldsymbol{y}}$ is always possible since $\boldsymbol{G}$ is assumed full row-rank.

Finally, we consider an environment populated by a set of obstacles, described by the vector of obstacle points $\boldsymbol{o} = (\boldsymbol{o}_1 \dots \boldsymbol{o}_{n_o}) \in \mathbb{R}^{2 \times n_o}$, and a set of regions of interest, described by the vector of points of interest $\boldsymbol{r} = (\boldsymbol{r}_1 \dots \boldsymbol{r}_{n_r}) \in \mathbb{R}^{2 \times n_r}$.

Given the human commands $(s_h, \boldsymbol{x}_h)$ and the environment $(\boldsymbol{o}, \, \boldsymbol{r})$, the interactive planner (see Fig. 1) modifies the trajectory $\boldsymbol{p}(t)$ as dictated by the following dynamical system:

$$\dot{s} = g \left(\boldsymbol{x}, s, \dot{s}_h \right) \tag{3a}$$

$$\dot{\boldsymbol{x}} = \boldsymbol{f} \left(\boldsymbol{x}, s, \boldsymbol{x}_h, \dot{\boldsymbol{x}}_h, \boldsymbol{o}, \boldsymbol{r} \right) \tag{3b}$$

$$\boldsymbol{p} = \boldsymbol{\gamma} \left(\boldsymbol{x}, s \right). \tag{3c}$$

Here, the scalar function g represents a control law for letting $\dot{s}$ track the commanded $\dot{s}_h$ as close as possible compatibly with the robot constraints (e.g., motor saturations, limited available energy, etc.), and $\boldsymbol{f}$ is a vector field for modifying the actual shape parameters $\boldsymbol{x}$ in order to: i) avoid obstacles $\boldsymbol{o}$, ii) travel as close as possible to the points of interest $\boldsymbol{r}$, iii) track as much as possible the desired trajectory (2a–2b) (represented by $\boldsymbol{x}_h$ and s_h), and iv) prevent that at any time the geometric properties of $\boldsymbol{\gamma}(\boldsymbol{x}, \, s_R)$ are directly influenced by the variation $\dot{\boldsymbol{x}}$ given by (3b).

Requirement iv) on $\boldsymbol{f}$ is meant to avoid that variations of shape parameters $\boldsymbol{x}$ produce sudden changes of γ at the current robot location. This is important to ease the action of the robot trajectory tracker which expects presence of some local regularity properties of the curve being tracked.

A. Dynamics of the Trajectory Parameters s and $\boldsymbol{x}$

The arc length (3a) complies with the following dynamics

$$\dot{s} = g(\boldsymbol{x}, s, \dot{s}_h) = g_1(\boldsymbol{x}, s, \dot{s}_h) \dot{s}_h, \tag{4}$$

where $g_1 \in [0, 1]$ is designed so as to cope with the robot motion capabilities. In particular, $g_1 = 1$ if the robot can travel along the path at the desired speed $\dot{s}_h$, and $g_1 \to 0$ (thus, towards a full stop) whenever the speed $\dot{s}_h$ becomes too large for the actuation capabilities of the robot (e.g., because of a too large curvature or a low energy level for the robot batteries). Another possible use of g_1 is to equally space many robots along the curve in case of a multi-robot implementation.

Concerning the vector field $\boldsymbol{f}$, we first analyze how to design it in order to meet Requirement iv). Local geometric properties of a curve are characterized by the k-th derivatives w.r.t. the arc length s, i.e.,

$$\boldsymbol{p}^{(k)}(\boldsymbol{x}, s) = \boldsymbol{\gamma}^{(k)}(\boldsymbol{x}, s) = \frac{\partial^k \boldsymbol{\gamma}}{\partial s^k}(\boldsymbol{x}, s), \quad k \in \mathbb{N}_0$$

[1] A dynamical system is differentially flat if it possible to express its state and inputs as an algebraic explicit function of a flat output and a limited number of its derivatives [19]. Output flatness is equivalent to exact dynamic feedback linearizability with the flat output taken as linearizing outputs [20].

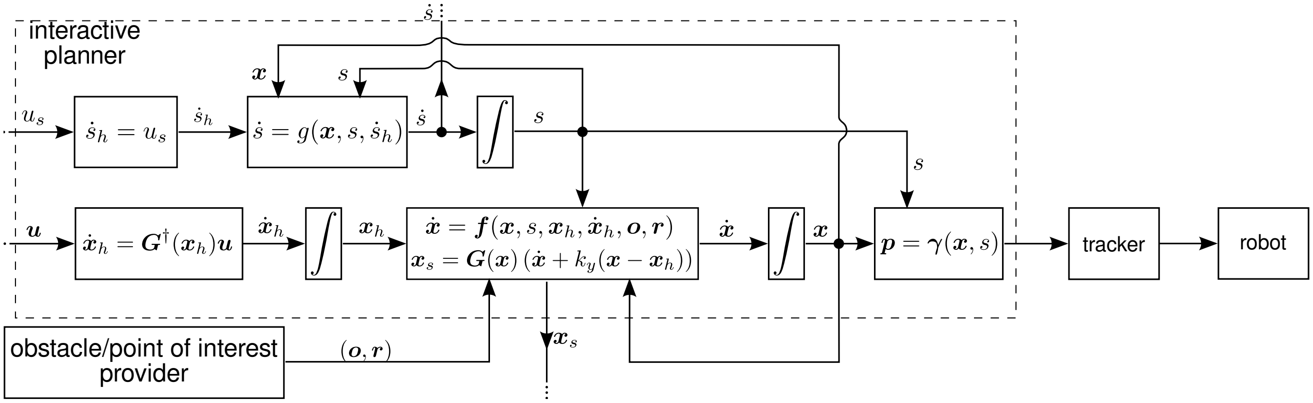

interactive planner
u_s
$\dot{s}_h = u_s$
$\dot{s}_h$
$\boldsymbol{x}$
s
$\dot{s} = g(\boldsymbol{x}, s, \dot{s}_h)$
$\dot{s}$
$\int$
s
s
$\boldsymbol{u}$
$\dot{\boldsymbol{x}}_h = \boldsymbol{G}^{\dagger}(\boldsymbol{x}_h)\boldsymbol{u}$
$\dot{\boldsymbol{x}}_h$
$\int$
$\boldsymbol{x}_h$
$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, s, \boldsymbol{x}_h, \dot{\boldsymbol{x}}_h, \boldsymbol{o}, \boldsymbol{r})$
$\boldsymbol{x}_s = \boldsymbol{G}(\boldsymbol{x})\left(\dot{\boldsymbol{x}} + k_y(\boldsymbol{x} - \boldsymbol{x}_h)\right)$
$\dot{\boldsymbol{x}}$
$\int$
$\boldsymbol{x}$
$\boldsymbol{p} = \boldsymbol{\gamma}(\boldsymbol{x}, s)$
tracker
robot
obstacle/point of interest provider
$(\boldsymbol{o}, \boldsymbol{r})$
$\boldsymbol{x}_s$

Fig. 1: Block diagram of the interactive planner and control framework.

where the operator $\cdot^{(k)}$ indicates the k-th geometric derivative. By applying the chain rule, the variation of $\boldsymbol{p}^{(k)}$ due to changes of $\boldsymbol{x}$ and s over time is given by

$$\frac{d}{dt}\left(\boldsymbol{p}^{(k)}(\boldsymbol{x}, s)\right) = \left.\frac{\partial \boldsymbol{\gamma}^{(k)}}{\partial \boldsymbol{x}}\right|_{(\boldsymbol{x},s)} \dot{\boldsymbol{x}} + \left.\frac{\partial \boldsymbol{\gamma}^{(k)}}{\partial s}\right|_{(\boldsymbol{x},s)} \dot{s}. \quad (5)$$

Stacking eq. (5) for the first k derivatives then yields

$$\dot{\boldsymbol{\Gamma}}_{0,k}(\boldsymbol{x}, s) = \boldsymbol{J}(\boldsymbol{x}, s)\dot{\boldsymbol{x}} + \boldsymbol{\Gamma}_{1,k+1}(\boldsymbol{x}, s)\dot{s}. \quad (6)$$

where $\boldsymbol{\Gamma}_{i,j} = [\boldsymbol{\gamma}^{(i)^T} \ldots \boldsymbol{\gamma}^{(j)^T}]^T \in \mathbb{R}^{2(j-i+1)}$ and $\boldsymbol{J} \in \mathbb{R}^{2(k+1)\times n}$. Equation (6) shows how the geometric properties of the curve γ at some $(\boldsymbol{x}, s)$ depend on two contributions: the first is due to the parameter change $\dot{\boldsymbol{x}}$ and the second to the longitudinal speed $\dot{s}$. Requirement iv) on $\boldsymbol{f}$ can then be interpreted as imposing that the first term in (6) is zero when evaluated at the current $(\boldsymbol{x},\, s_R)$. Therefore, the design of the vector field $\boldsymbol{f}$ must ensure that

$$\boldsymbol{J}(\boldsymbol{x}(t), s(t))\dot{\boldsymbol{x}} = \boldsymbol{0}. \quad (7)$$

This condition could be trivially satisfied by choosing $\dot{\boldsymbol{x}} = \boldsymbol{0}$. However, assuming matrix $\boldsymbol{J}$ has a non-empy null-space, an infinity of possible $\dot{\boldsymbol{x}} \in \mathcal{N}(\boldsymbol{J}) \neq \boldsymbol{0}$ would also meet the constraint. To this end, let $\boldsymbol{J}^{\dagger}(\boldsymbol{x}, s)$ represent the pseudo-inverse of $\boldsymbol{J}(\boldsymbol{x}, s)$, and $\boldsymbol{N}(\boldsymbol{x}, s) \in \mathbb{R}^{n\times n}$ a projector matrix spanning the null space $\mathcal{N}(\boldsymbol{J})$, i.e., such that $\boldsymbol{J}\boldsymbol{N} = \boldsymbol{0}$. We then design the vector field $\boldsymbol{f}$ to have the following form [2]

$$\boldsymbol{f}\left(\boldsymbol{x}, s, \boldsymbol{x}_h, \dot{\boldsymbol{x}}_h, \boldsymbol{o}, \boldsymbol{r}\right) = \boldsymbol{N}(\boldsymbol{x}, s)\boldsymbol{f}_1\left(\boldsymbol{x}, \boldsymbol{x}_h, \dot{\boldsymbol{x}}_h, \boldsymbol{o}, \boldsymbol{r}\right). \quad (8)$$

This choice will still allow the parameter vector $\boldsymbol{x}$ to change over time, but without violating the constraint (7). Assuming that matrix $\boldsymbol{J}$ has full row-rank, a well-known choice for the projector operator is $\boldsymbol{N} = (\boldsymbol{I} - \boldsymbol{J}^{\dagger}\boldsymbol{J})$.

In order to meet Requirements i), ii), and iii), we then design $\boldsymbol{f}_1$ in (8) as the composition of four terms:

$$\boldsymbol{f}_1 = \boldsymbol{f}_h(\boldsymbol{x}, \boldsymbol{x}_h, \dot{\boldsymbol{x}}_h) + \boldsymbol{f}_o(\boldsymbol{x}, \boldsymbol{o}) + \boldsymbol{f}_r(\boldsymbol{x}, \boldsymbol{r}) + \boldsymbol{f}_i(\boldsymbol{x}). \quad (9)$$

The first term implements a feedforward/proportional action

$$\boldsymbol{f}_h(\boldsymbol{x}, \boldsymbol{x}_h, \dot{\boldsymbol{x}}_h) = \dot{\boldsymbol{x}}_h + k_h(\boldsymbol{x}_h - \boldsymbol{x}), \quad (10)$$

with $k_h > 0$, in order to steer $\boldsymbol{x}$ towards the reference $\boldsymbol{x}_h$; $\boldsymbol{f}_o(\boldsymbol{x}, \boldsymbol{o})$ is a vector field that moves $\boldsymbol{\gamma}$ away from the obstacles $\boldsymbol{o}$; $\boldsymbol{f}_r(\boldsymbol{x}, \boldsymbol{r})$ is a vector field that attracts $\boldsymbol{\gamma}$ towards the points of interest $\boldsymbol{r}$; and $\boldsymbol{f}_i(\boldsymbol{x})$ is an additional vector field whose role is discussed in the following.

With regard to $\boldsymbol{f}_o$, each obstacle $\boldsymbol{o}_i$, $i = 1, \ldots, n_o$, implements a strictly monotonic and scalar potential $\varphi_{oi} : \mathbb{R}_0 \to \mathbb{R}^+$ such that, for a generic point $\boldsymbol{\gamma}(\boldsymbol{x}, s)$ on the path, $\varphi_{oi} \to \infty$ when $\|\boldsymbol{\gamma}(\boldsymbol{x}, s) - \boldsymbol{o}_i\| \to 0$, $\varphi_{oi} \to 0$ smoothly when $\|\boldsymbol{\gamma}(\boldsymbol{x}, s) - \boldsymbol{o}_i\| \to R_{oi}$ and $\varphi_{oi} \equiv 0$ when $R_{oi} > 0$ where $R_{oi} > 0$ defines the region of influence of the obstacle. The action of the anti-gradient vector field of φ_{oi} on the point $\boldsymbol{\gamma}(\boldsymbol{x}, s)$ is

$$\boldsymbol{f}^{\boldsymbol{p}}_{oi}(\boldsymbol{x}, s, \boldsymbol{o}_i) = -\frac{\partial\varphi(\|\boldsymbol{\gamma}(\boldsymbol{x}, s) - \boldsymbol{o}_i\|)}{\partial\boldsymbol{\gamma}(\boldsymbol{x}, s)}. \quad (11)$$

The overall action exerted on the curve by a single obstacle and projected on the shape parameters space is

$$\boldsymbol{f}_{oi}(\boldsymbol{x}, \boldsymbol{o}_i) = \oint_{\gamma} \left.\frac{\partial\boldsymbol{\gamma}}{\partial\boldsymbol{x}}\right|^{\dagger}_{(\boldsymbol{x},s)} \boldsymbol{f}^{\boldsymbol{p}}_{oi}(\boldsymbol{x}, s, \boldsymbol{o}_i)ds, \quad (12)$$

where the previous considerations on the existence of the pseudo-inverse still hold. Finally, the total action of the obstacles is just the sum over all the elements in $\boldsymbol{o}$:

$$\boldsymbol{f}_o = \sum_{i=1}^{n_o} \boldsymbol{f}_{oi}(\boldsymbol{x}, \boldsymbol{o}_i), \quad (13)$$

We note that, from a practical standpoint, the analytical expression of (11) can be hard to determine, so that a numerical evaluation of the integral may be needed.

By resorting to similar arguments, we define an attractive vector field $\boldsymbol{f}_{ri}(\boldsymbol{x}, \boldsymbol{r}_i)$ for each point of interest $\boldsymbol{r}_i$, and sum each contribution in order to obtain

$$\boldsymbol{f}_r = \sum_{i=1}^{n_r} \boldsymbol{f}_{ri}(\boldsymbol{x}, \boldsymbol{r}_i). \quad (14)$$

Finally, the vector field $\boldsymbol{f}_i(\boldsymbol{x})$ is left free to the engineer in order to implement some additional internal properties of the curve that may be of interest, e.g., internal elasticity, stiffness, or viscosity. The choice here clearly depends on the parameterization of γ w.r.t. $\boldsymbol{x}$. For instance, if the parameters

[2] We note that this approach could be seen as an extension of the classical Task-Priority framework developed for robot manipulators and recast to our particular needs, see [23] for more details.

$\boldsymbol{x}$ represent some way-points, $\boldsymbol{f}_i$ can be used to implement virtual springs between consecutive points similarly to what has been shown in [11].

III. HUMAN OPERATOR BILATERAL INTERFACE

The user inputs $(u_s, \boldsymbol{u}) \in \mathbb{R}^{m+1}$ are provided by making use of a $(m+1)$-DOF force-feedback device (the master side). The device is modeled as a generic mechanical system

$$M(\boldsymbol{q}_M)\ddot{\boldsymbol{q}}_M + C(\boldsymbol{q}_M, \dot{\boldsymbol{q}}_M)\dot{\boldsymbol{q}}_M = \boldsymbol{\tau}_M + \boldsymbol{\tau}_h \tag{15}$$

where $\boldsymbol{q}_M \in \mathbb{R}^{m+1}$ is the configuration vector of the device, $M(\boldsymbol{q}_M) \in \mathbb{R}^{m+1 \times m+1}$ is the positive-definite and symmetric inertia matrix, $C(\boldsymbol{q}_M, \dot{\boldsymbol{q}}_M)\dot{\boldsymbol{q}}_M \in \mathbb{R}^{m+1}$ are the Coriolis and centrifugal terms, and $\boldsymbol{\tau}_M$, $\boldsymbol{\tau}_h \in \mathbb{R}^{m+1}$ are the control and human forces respectively. In this work, we choose to couple the user inputs $(u_s, \boldsymbol{u})$ with the configuration of the force-feedback device as

$$\begin{pmatrix} u_s \\ \boldsymbol{u} \end{pmatrix} = \boldsymbol{K}_R \boldsymbol{q}_M, \tag{16}$$

where $\boldsymbol{K}_R \in \mathbb{R}^{m+1 \times m+1}$ is a positive definite diagonal matrix of scaling factors. Note that, by means of (16) and then (2), the configuration of the haptic interface results mapped into a reference tangential speed $\dot{s}_h$ and a velocity of the reference shape parameters $\dot{\boldsymbol{x}}_h$. This mapping can be thought as a *velocity-position* coupling in the bilateral teleoperation literature and is often adopted in all those cases involving a slave system with an unlimited workspace (as mobile robots), see e.g., [24].

As for the human perception, the force-feedback $\boldsymbol{\tau}_M$ is used to convey information about the motion status of the mobile robot in order to increase her/his situational awareness. Within the framework introduced in this paper, we propose to convey to the human operator the following information: i) how well the actual speed $\dot{s}$ is tracking $\dot{s}_h$, i.e., the one commanded by the human via u_s, and ii) how well the geometric path $\boldsymbol{\gamma}(\boldsymbol{x})$ (autonomously deformed by the robot) is in agreement with the desired path $\boldsymbol{\gamma}(\boldsymbol{x}_h)$ commanded by the human via $\boldsymbol{u}$.

Therefore, as a haptic cue we consider the linear combination of two errors in a 'PD-like' fashion:

$$\boldsymbol{e}_\gamma = \boldsymbol{e}_{\dot{\boldsymbol{y}}} + k_y \boldsymbol{e}_y. \tag{17}$$

Here, $\boldsymbol{e}_{\dot{\boldsymbol{y}}}$ represents the 'velocity error' part of (17) and is related to the difference between the commanded $(\dot{s}_h, \dot{\boldsymbol{x}}_h)$ and the executed $(\dot{\boldsymbol{s}}, \dot{\boldsymbol{x}})$, i.e., exploiting (1),

$$\boldsymbol{e}_{\dot{\boldsymbol{y}}} = \begin{pmatrix} \dot{s}_h - \dot{s} \\ \boldsymbol{G}(\boldsymbol{x}_h)\dot{\boldsymbol{x}}_h - \boldsymbol{G}(\boldsymbol{x})\dot{\boldsymbol{x}} \end{pmatrix} = \boldsymbol{K}_R \boldsymbol{q}_M - \begin{pmatrix} \dot{s} \\ \boldsymbol{G}(\boldsymbol{x})\dot{\boldsymbol{x}} \end{pmatrix}. \tag{18}$$

The second term $k_y \boldsymbol{e}_{\boldsymbol{y}}$ in (17) represents the 'position error' term and is associated to the mismatch between the desired shape $\boldsymbol{x}_h$ and its actual implementation $\boldsymbol{x}$, i.e.,

$$\boldsymbol{e}_{\boldsymbol{y}} = \begin{pmatrix} 0 \\ \boldsymbol{G}(\boldsymbol{x})\,(\boldsymbol{x}_h - \boldsymbol{x}) \end{pmatrix}. \tag{19}$$

We recall, for the reader's convenience, that the discrepancy between commanded and actual evolution of the shape parameters $\boldsymbol{x}$ is caused by the presence of the obstacles $\boldsymbol{o}$, of the points of interest $\boldsymbol{r}$, and of the projector operator $\boldsymbol{N}(\boldsymbol{x}, s)$ that prevents the path $\boldsymbol{\gamma}$ to be locally modified in the current robot location $\boldsymbol{\gamma}(\boldsymbol{x}(t), s(t))$.

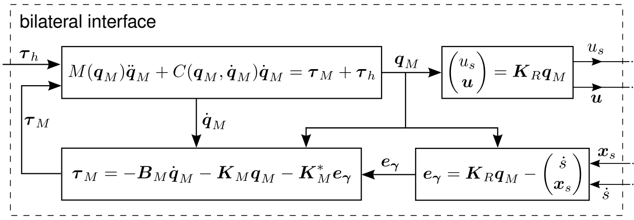

bilateral interface
$\boldsymbol{\tau}_h$
$M(\boldsymbol{q}_M)\ddot{\boldsymbol{q}}_M + C(\boldsymbol{q}_M, \dot{\boldsymbol{q}}_M)\dot{\boldsymbol{q}}_M = \boldsymbol{\tau}_M + \boldsymbol{\tau}_h$
$\boldsymbol{q}_M$
$\begin{pmatrix} u_s \\ \boldsymbol{u} \end{pmatrix} = \boldsymbol{K}_R \boldsymbol{q}_M$
u_s
$\boldsymbol{u}$
$\boldsymbol{\tau}_M$
$\dot{\boldsymbol{q}}_M$
$\boldsymbol{\tau}_M = -\boldsymbol{B}_M\dot{\boldsymbol{q}}_M - \boldsymbol{K}_M\boldsymbol{q}_M - \boldsymbol{K}_M^*\boldsymbol{e}_\gamma$
$\boldsymbol{e}_\gamma$
$\boldsymbol{e}_\gamma = \boldsymbol{K}_R\boldsymbol{q}_M - \begin{pmatrix} \dot{s} \\ \boldsymbol{x}_s \end{pmatrix}$
$\boldsymbol{x}_s$
$\dot{s}$

Fig. 2: Block diagram of the bilateral interface control.

The master control is then implemented as

$$\boldsymbol{\tau}_M = -\boldsymbol{B}_M\dot{\boldsymbol{q}}_M - \boldsymbol{K}_M\boldsymbol{q}_M - \boldsymbol{K}_M^*\boldsymbol{e}_\gamma \tag{20}$$

where $\boldsymbol{B}_M$ is a positive definite damping matrix used to stabilize the device, $\boldsymbol{K}_M$ is a diagonal non-negative matrix used to provide a perception of the distance from the zero-commanded velocity, and $\boldsymbol{K}_M^*$ a diagonal positive definite matrix of gains. The resulting scheme is depicted in Fig. 2.

As in all bilateral teleoperation applications, presence of the force feedback $\boldsymbol{\tau}_M$ may cause unstable behaviors of the haptic interface because of non-modeled dynamics, communication delays and packet losses, etc. In order to guarantee stability despite all these shortcomings, we make use of the *passive set-position modulation* (PSPM) approach, a very general and flexible framework for guaranteeing stability (passivity) of the master side and of the closed-loop teleoperation system [25]. Let $\bar{z}[k]$ be the PSPM version of the original signal

$$\boldsymbol{z} = \begin{pmatrix} \dot{s} \\ \boldsymbol{G}(\boldsymbol{x})\dot{\boldsymbol{x}} \end{pmatrix} - k_y \boldsymbol{e}_{\boldsymbol{y}}$$

sampled and sent from the mobile robot to the haptic interface through the (possibly non-ideal) communication channel. Exploiting the PSPM action, the final *passive* implementation of the master controller (20) then becomes

$$\tau_M = -\boldsymbol{B}_M\dot{\boldsymbol{q}}_M - \boldsymbol{K}_M\boldsymbol{q}_M - \boldsymbol{K}_M^*\left(\boldsymbol{K}_R\boldsymbol{q}_M - \bar{\boldsymbol{z}}[k]\right). \tag{21}$$

This is sufficient for guaranteeing stability (passivity) of the bilateral system assuming that the human operator behaves as a passive system, see [25] for more details.

IV. PLANNING ALTERNATIVE PATHS IN HIGHLY CLUTTERED ENVIRONMENTS

The planning strategy discussed in Sec. II computes a collision free trajectory by using artificial potentials φ_{oi}, whose intensity becomes infinite as the distance to the obstacles goes to zero. One drawback of this approach is that it does not allow the task path to cross over an obstacle even if this could result in a smaller error norm $e(\boldsymbol{x}, \boldsymbol{x}_h) = \|\boldsymbol{x}_h - \boldsymbol{x}\|$. In a cluttered environment, this behavior could severely limit the capability of the human operator to interact with the task, thus increasing the mental load.

This limitation is a well known problem in the reactive planning literature and it has been tackled in different ways. For instance, in the elastic strip framework [11] the planner is allowed to temporarily suspend the internal forces keeping two waypoints together and then, when the obstacle is passed, restore them to rejoin the two trajectory branches. However, this formulation is limited to paths defined as a sequences of configurations and it does not guarantee that the two disjoint branches would actually cross to the other side of the obstacle. Here we propose a procedure that, given an obstacle point $\boldsymbol{o}_i$ outside $\boldsymbol{\gamma}(\boldsymbol{x}, s)$, autonomously generates an alternative set of shape parameters $\boldsymbol{x}_{oi} \in \mathbb{R}^n$ such that $\boldsymbol{o}_i$ is inside $\boldsymbol{\gamma}(\boldsymbol{x}_{oi}, s)$. Vice-versa, if $\boldsymbol{o}_i$ is inside $\boldsymbol{\gamma}(\boldsymbol{x}, s)$ it will be $\boldsymbol{o}_i$ outside $\boldsymbol{\gamma}(\boldsymbol{x}_{oi}, s)$. The *alternative path* $\boldsymbol{\gamma}(\boldsymbol{x}_{oi}, s)$ is initialized and generated when $\boldsymbol{o}_i$ induces a big enough deformation on $\boldsymbol{\gamma}(\boldsymbol{x}, s)$. Introducing a threshold $F > 0$, this condition can be expressed in terms of the repulsive force $\boldsymbol{f}_{oi}^{\boldsymbol{p}}$ in (11) as

$$\max_{s\in[0,L]} \|\boldsymbol{f}_{oi}^{\boldsymbol{p}}(\boldsymbol{x}, s, \boldsymbol{o}_i)\| \geq F. \tag{22}$$

Note that, by definition, the maximum value of $\|\boldsymbol{f}_{oi}^{\boldsymbol{p}}\|$ is obtained on the closest point to the obstacle, let this be $\boldsymbol{\gamma}(\boldsymbol{x}, \bar{s})$, which can be computed by solving the problem

$$\bar{s} = \min_{s\in[0,L]} \|\boldsymbol{\gamma}(\boldsymbol{x}, s) - \boldsymbol{o}_i\|. \tag{23}$$

Assume that condition (22) becomes true at time instant t_0, the alternative path is generated by letting the alternative parameters follow a dynamical system of the form:

$$\begin{cases} \boldsymbol{x}_{oi}(t_0) = \boldsymbol{x}(t_0) \\ \dot{\boldsymbol{x}}_{oi} = \boldsymbol{f}_c(\boldsymbol{x}_{oi}, \boldsymbol{x}, s, \bar{s}, \hat{s}, \boldsymbol{o}_i). \end{cases} \tag{24}$$

In order to define $\boldsymbol{f}_c \in \mathbb{R}^n$, we first need to introduce another artificial vector field $\boldsymbol{f}_c^{\boldsymbol{p}} \in \mathbb{R}^2$. The role of $\boldsymbol{f}_c^{\boldsymbol{p}} \in \mathbb{R}^2$ is to apply an action that leads a single point $\boldsymbol{\gamma}(\boldsymbol{x}_{oi}, \hat{s})$ of the alternative path being generated with (24) to the other side of the obstacle. In this way, if $\boldsymbol{o}_i$ lies inside $\boldsymbol{\gamma}(\boldsymbol{x}, s)$ then the alternative path $\boldsymbol{\gamma}(\boldsymbol{x}_{oi}, s)$ will eventually not contain $\boldsymbol{o}_i$, and vice-versa.

Formally, $\boldsymbol{f}_c^{\boldsymbol{p}}$ is designed as

$$\boldsymbol{f}_c^{\boldsymbol{p}}(\boldsymbol{x}_{oi}, \boldsymbol{x}, s, \bar{s}, \hat{s}, \boldsymbol{o}_i) = \frac{d\psi(\|\boldsymbol{\gamma}(\boldsymbol{x}_{oi}, s) - \boldsymbol{o}_i\|)}{d\|\boldsymbol{\gamma}(\boldsymbol{x}_{oi}, s) - \boldsymbol{o}_i\|}\, \boldsymbol{n} \tag{25}$$

where ψ is a strictly increasing artificial potential and $\boldsymbol{n} = \frac{\boldsymbol{o}_i - \boldsymbol{\gamma}(\boldsymbol{x}, \bar{s})}{\|\boldsymbol{o}_i - \boldsymbol{\gamma}(\boldsymbol{x}, \bar{s})\|}$. Finally, the point $\boldsymbol{\gamma}(\boldsymbol{x}_{oi}, \hat{s})$ where the vector $\boldsymbol{f}_c^{\boldsymbol{p}}$ is applied is given by the intersection of the geometric path $\boldsymbol{\gamma}(\boldsymbol{x}_{oi}, s)$ with the line connecting $\boldsymbol{o}_i$ with $\boldsymbol{\gamma}(\boldsymbol{x}, \bar{s})$. With the same arguments used in Sec. II-A, the desired velocity vector $\boldsymbol{f}_c^{\boldsymbol{p}}$ for the point $\boldsymbol{\gamma}(\boldsymbol{x}_{oi}, \hat{s})$ is realized by a velocity vector in the space of the alternative shape parameters using a pseudo-inversion

$$\boldsymbol{f}_c(\boldsymbol{x}_{oi}, \boldsymbol{x}, s, \bar{s}, \hat{s}, \boldsymbol{o}_i) = \left.\frac{\partial \boldsymbol{\gamma}}{\partial \boldsymbol{x}}\right|_{(\boldsymbol{x}_{oi}, \hat{s})}^{\dagger} \boldsymbol{f}_c^{\boldsymbol{p}}. \tag{26}$$

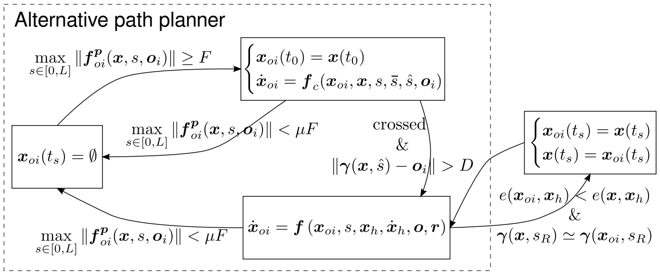

Alternative path planner
$\max_{s\in[0,L]} \|\boldsymbol{f}_{oi}^{\boldsymbol{p}}(\boldsymbol{x}, s, \boldsymbol{o}_i)\| \geq F$
$\begin{cases} \boldsymbol{x}_{oi}(t_0) = \boldsymbol{x}(t_0) \\ \dot{\boldsymbol{x}}_{oi} = \boldsymbol{f}_c(\boldsymbol{x}_{oi}, \boldsymbol{x}, s, \bar{s}, \hat{s}, \boldsymbol{o}_i) \end{cases}$
$\max_{s\in[0,L]} \|\boldsymbol{f}_{oi}^{\boldsymbol{p}}(\boldsymbol{x}, s, \boldsymbol{o}_i)\| < \mu F$
$\boldsymbol{x}_{oi}(t_s) = \emptyset$
crossed
&
$\|\boldsymbol{\gamma}(\boldsymbol{x}, \hat{s}) - \boldsymbol{o}_i\| > D$
$\begin{cases} \boldsymbol{x}_{oi}(t_s) = \boldsymbol{x}(t_s) \\ \boldsymbol{x}(t_s) = \boldsymbol{x}_{oi}(t_s) \end{cases}$
$e(\boldsymbol{x}_{oi}, \boldsymbol{x}_h) < e(\boldsymbol{x}, \boldsymbol{x}_h)$
&
$\boldsymbol{\gamma}(\boldsymbol{x}, s_R) \simeq \boldsymbol{\gamma}(\boldsymbol{x}_{oi}, s_R)$
$\max_{s\in[0,L]} \|\boldsymbol{f}_{oi}^{\boldsymbol{p}}(\boldsymbol{x}, s, \boldsymbol{o}_i)\| < \mu F$
$\dot{\boldsymbol{x}}_{oi} = \boldsymbol{f}\left(\boldsymbol{x}_{oi}, s, \boldsymbol{x}_h, \dot{\boldsymbol{x}}_h, \boldsymbol{o}, \boldsymbol{r}\right)$

Fig. 3: Block diagram of the alternative path planner

When the crossing is completed and $\boldsymbol{\gamma}(\boldsymbol{x}_{oi}, \hat{s})$ is sufficiently far from the obstacle point $\boldsymbol{o}_i$, then the dynamical evolution of the alternative shape parameters $\boldsymbol{x}_{oi}$ switches to the normal behavior

$$\dot{\boldsymbol{x}}_{oi} = \boldsymbol{f}\left(\boldsymbol{x}_{oi}, s, \boldsymbol{x}_h, \dot{\boldsymbol{x}}_h, \boldsymbol{o}, \boldsymbol{r}\right). \tag{27}$$

At this point, the alternative collision free path $\boldsymbol{\gamma}(\boldsymbol{x}_{oi}, s)$ is fully generated. If at some instant t_s it results that $e(\boldsymbol{x}_{oi}, \boldsymbol{x}_h) < e(\boldsymbol{x}, \boldsymbol{x}_h)$, then the new path becomes active, i.e.,

$$\begin{cases} \boldsymbol{x}_{oi}(t_s) = \boldsymbol{x}(t_s) \\ \boldsymbol{x}(t_s) = \boldsymbol{x}_{oi}(t_s). \end{cases} \tag{28}$$

However, the switch is allowed only if $\boldsymbol{\gamma}(\boldsymbol{x}, s_R) \simeq \boldsymbol{\gamma}(\boldsymbol{x}_{oi}, s_R)$ to avoid discontinuities in the reference for the robot.

To complete the procedure, if the obstacle $\boldsymbol{o}_i$ gets sufficiently distant from $\boldsymbol{\gamma}(\boldsymbol{x}, s)$, i.e. if $\max_{s\in[0,L]} \|\boldsymbol{f}_{oi}^{\boldsymbol{p}}(\boldsymbol{x}, s, \boldsymbol{o}_i)\| < \mu F$ with $0 < \mu < 1$, then $\boldsymbol{\gamma}(\boldsymbol{x}_{oi}, s)$ is dropped. A block representation of the overall system is depicted in Fig. 3.

The generalization of this procedure to multiple obstacles is straightforward. An alternative path is generated for every obstacle that is applying a strong enough force on the current path, and whenever the current path is switched to another one, all the other alternatives are also reset. This method is not complete at every time instant because it simultaneously considers at most $n_{\boldsymbol{o}}$ alternative paths while the family of collision free closed curves existing in the free space is partitioned in $2^{n_{\boldsymbol{o}}}$ groups, depending on the position of each obstacle (either inside or outside the curve). However, sequential switches allow to virtually extend the search to all the groups of the partition while still keeping the problem tractable, being the number of path at every instant linear w.r.t. the number of obstacles.

V. CASE OF B-SPLINES

As an illustrative example we show how the formulation proposed in the previous sections applies to B-spline curves, curves that are obtained as a linear combination of some polynomial basis functions [26]. B-splines are completely described by:

1) a vector $\boldsymbol{\kappa} = [\kappa_1, \ldots, \kappa_{n_\kappa}] \in \mathbb{R}^{n_\kappa}$, with $\kappa_j \leq \kappa_{j+1}$ (*knots* vector). The knot vector determines the pool of basis functions considered for the spline.
2) a parameter $\lambda \geq 0$ (*spline order*). It is related to the differentiability at the knots (the B-spline is $\lambda - 1$ times continuously differentiable at the knots).

3) a sequence of points $\boldsymbol{c}_j \in \mathbb{R}^2, \quad j = 1, \dots, n_c$ (*control points*) which ultimately determines the shape of the curve.

We define the vector of shape parameters as $\boldsymbol{x} = [c_1^T \ \dots \ c_{n_c}^T]^T \in \mathbb{R}^n$ with $n = 2 \times n_c$ and consider the order and the knots of the B-spline as constant parameters. The expression of the B-spline curve is

$$\boldsymbol{\gamma}(\boldsymbol{x}, s) = \sum_{j=1}^{n_c} \boldsymbol{c}_j B_j^{\lambda}(\boldsymbol{\kappa}, s) = \boldsymbol{B}(s)\boldsymbol{x} \tag{29}$$

where $s \in [\kappa_1, \kappa_{n_k}]$ is the parameter identifying a point on the curve, $\boldsymbol{B}(s) = [B_1^{\lambda}(\boldsymbol{\kappa}, s) \ \ B_1^{\lambda}(\boldsymbol{\kappa}, s) \ \ \dots \ \ B_{n_c}^{\lambda}(\boldsymbol{\kappa}, s) \ \ B_{n_c}^{\lambda}(\boldsymbol{\kappa}, s)] \in \mathbb{R}^{n \times 1}$ is the collection of basis functions and $B_j^{\lambda}(\boldsymbol{\kappa}, s)$ is the j-th basis function evaluated in s and recursively defined as

$$\begin{aligned} B_j^0(\boldsymbol{\kappa}, s) &= \begin{cases} 1, & \text{if } \kappa_j \le s < \kappa_{j+1} \\ 0, & \text{otherwise} \end{cases} \\ B_j^{\lambda}(\boldsymbol{\kappa}, s) &= \frac{s - \kappa_j}{\kappa_{j+\lambda} - \kappa_j} B_j^{\lambda-1}(\boldsymbol{\kappa}, s) + \\ &\quad \frac{\kappa_{j+\lambda+1} - s}{\kappa_{j+\lambda+1} - \kappa_{j+1}} B_{j+1}^{\lambda-1}(\boldsymbol{\kappa}, s) \end{aligned} \tag{30}$$

Since we consider closed curves that can be deformed while retaining the closure, we use a cyclic basis definition by considering the subtraction in (30) as a modulus operation on κ_{n_κ}.

Finally, we define two command maps that will be used in the simulations presented in Section VI:

Curve local/global translation: The human operator commands a translation to a subset $\mathcal{X}$ of the control points, with $|\mathcal{X}| = N$. Equations (1), (2b) specialize as

$$\dot{\boldsymbol{y}} = \begin{pmatrix} g_1 & 0 & g_2 & 0 & \dots & g_{n_c} & 0 \\ 0 & g_1 & 0 & g_2 & \dots & 0 & g_{n_c} \end{pmatrix} \dot{\boldsymbol{x}} = \boldsymbol{G}(\boldsymbol{x})\dot{\boldsymbol{x}}$$

$$\dot{\boldsymbol{x}}_h = \begin{pmatrix} g_1^{\#} & 0 & g_2^{\#} & 0 & \dots & g_{n_c}^{\#} & 0 \\ 0 & g_1^{\#} & 0 & g_2^{\#} & \dots & 0 & g_{n_c}^{\#} \end{pmatrix}^T \boldsymbol{u} = \boldsymbol{G}^{\dagger}(\boldsymbol{x})\boldsymbol{u}$$

$$g_i = \begin{cases} 1/N & \text{if } \boldsymbol{c}_i \in \mathcal{X} \\ 0 & \text{otherwise} \end{cases} \qquad g_i^{\#} = \begin{cases} 1 & \text{if } \boldsymbol{c}_i \in \mathcal{X} \\ 0 & \text{otherwise} \end{cases} \tag{31}$$

with $\boldsymbol{u} \in \mathbb{R}^2$. If $\mathcal{X}$ is the whole set of control points then the input $\boldsymbol{u}$ produces a translation of the entire curve.

Path expansion/contraction: The user commands an expansion rate to the path with respect to the control points center of mass $\bar{\boldsymbol{c}} = \frac{1}{n_c}\sum_{i=1}^{n_c} \boldsymbol{c}_i$. In particular, the command $\boldsymbol{u} \in \mathbb{R}$ is mapped exactly to the expansion rate of the link $\boldsymbol{c}_1 - \bar{\boldsymbol{c}}$. Equations (1), (2b) are written as

$$\begin{aligned} \dot{\boldsymbol{y}} &= \|\boldsymbol{c}_1 - \bar{\boldsymbol{c}}\| \frac{(\boldsymbol{x} - \bar{\boldsymbol{x}})^T}{\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|^2} \dot{\boldsymbol{x}} = \boldsymbol{G}(\boldsymbol{x})\dot{\boldsymbol{x}} \\ \dot{\boldsymbol{x}} &= \frac{(\boldsymbol{x} - \bar{\boldsymbol{x}})}{\|\boldsymbol{c}_1 - \bar{\boldsymbol{c}}\|} \boldsymbol{u} = \boldsymbol{G}^{\dagger}(\boldsymbol{x})\boldsymbol{u} \\ \bar{\boldsymbol{x}} &= \begin{pmatrix} \frac{1}{n_c} I_{2\times 2} & \dots & \frac{1}{n_c} I_{2\times 2} \\ \vdots & \ddots & \vdots \\ \frac{1}{n_c} I_{2\times 2} & \dots & \frac{1}{n_c} I_{2\times 2} \end{pmatrix} \boldsymbol{x} \end{aligned} \tag{32}$$

with $\boldsymbol{u} \in \mathbb{R}$ and $I_{2\times 2}$ is the 2×2 identity matrix.

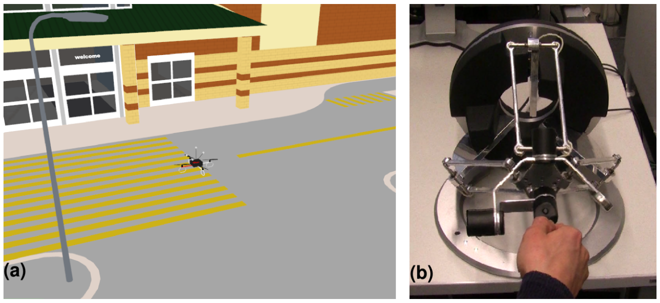


Fig. 4: simulation setup for human/hardware in the loop simulations. a): simulation environment. b): haptic device used to command the task path and feed back forces.

VI. HUMAN/HARDWARE-IN-THE-LOOP SIMULATIONS

The presented methods for interactive planning and control were tested with a human/hardware in-the-loop simulation framework composed by three parts: TeleKyb [27], SwarmSimX [28], and Matlab (see Fig. 4a). The physically simulated robot is a standard quadrotor UAV, the path $\boldsymbol{\gamma}(\boldsymbol{x}, s)$ is a fifth order B-spline (as described in Sec. V), and command maps (31) and (32) are employed for human interaction. Problem (23) has been solved with a globally convergent modification of Newton's method. Finally, the commands $(u_s, \boldsymbol{u})$ are provided by the user through an Omega.6 haptic device (Fig. 4b) with 3 actuated degrees of freedom that are used to implement the force feedback $\boldsymbol{\tau}_M$ as in (21). A video of the three simulations discussed hereinafter is attached to provide a better understanding.

In the first simulation we evaluate the effects introduced by the null space projector in (8), in particular using $\boldsymbol{J} = \partial\gamma/\partial\boldsymbol{x}|_{(\boldsymbol{x},s)}$ in (7) to keep the local position in the point tracked by the robot invariant to $\dot{\boldsymbol{x}}$. The experiment consists in commanding a planar sinusoidal translation (Fig. 5b) to the task curve in an obstacle-free environment. The case with the null-space projector (Fig. 5a-2) is compared against the case without it (Fig. 5a-1). Figure 5c shows the tracking error $\boldsymbol{\gamma}(\boldsymbol{x}, s) - \boldsymbol{w}$, where $\boldsymbol{w} \in \mathbb{R}^2$ is the planar position of the UAV. When the null space projection is applied (solid lines) the tracking error is visibly smaller than when no projection is used (dashed lines). This confirms the beneficial effects of keeping the local geometric properties of the curve in the point to be tracked. On the other hand, the deformation introduced by the projector $\boldsymbol{N}$ (see Fig. 5a-2) causes a mismatch between $\boldsymbol{x}$ and the reference $\boldsymbol{x}_h$, even in free space. This effect is reflected on the force feedback $\boldsymbol{\tau}_M$ (Fig. 5d), that becomes informative of the inertia of the path to these local changes.

The second simulation offers an example of how the human operator can interact with the path using different command maps $\boldsymbol{G}(\boldsymbol{x})$, specifically the command maps (31) and (32). Figure 6a shows two snapshots of the simulation when 1) a translation is commanded to a subset of control points, and 2) a contraction is commanded to shrink the path. The commands are plotted in Fig. 6b, in particular they are i) collective translation of the path, ii) translation of a subset of control points, and iii) expansion/dilation w.r.t. the center of mass of the control points. Vertical black lines indicate the

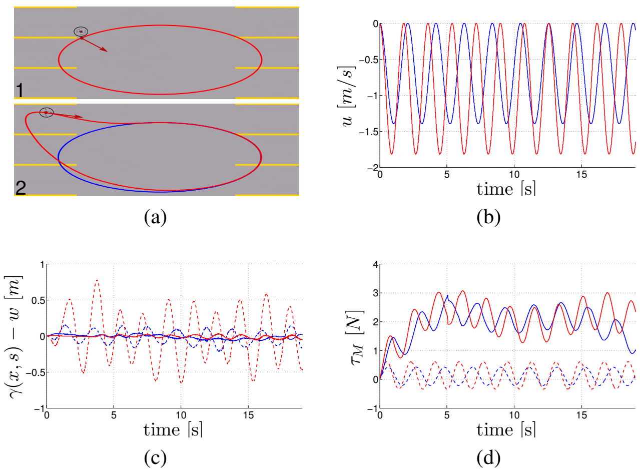


Fig. 5: (a) Snapshots of the first simulation showing 1) the case of no null space projection and 2) when $\dot{\boldsymbol{x}}$ is projected in the null space of $\boldsymbol{J} = \partial\gamma/\partial\boldsymbol{x}|_{(\boldsymbol{x},s)}$. The red curve is the one actually tracked by the UAV while the blue one is the task path. (b) Translational command $\boldsymbol{u}$ along x (blue) and y (red) axes. (c) Tracking error $\gamma(\boldsymbol{x}, s) - \boldsymbol{w}$ along x (blue) and y (red) directions, when the null space projection is off (dashed lines) and on (solid lines). (d) force feedback $\boldsymbol{\tau}_M$ along x (blue) and y (red) directions, when the null space projection is off (dashed lines) and on (solid lines).

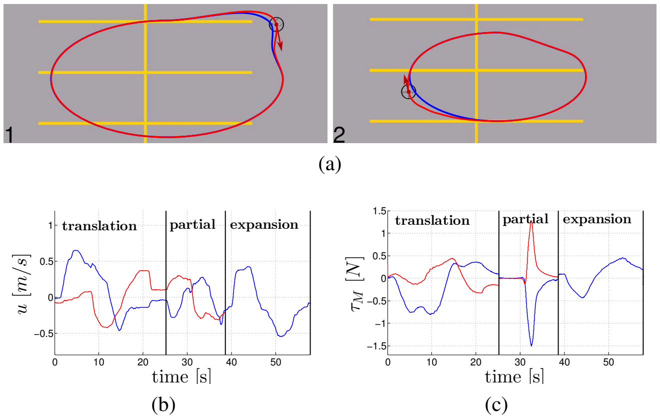


Fig. 6: (a) Snapshots of the second simulation showing 1) a translation command to a subset of control points 2) a contraction command to shrink the path. The red curve is the one actually tracked by the UAV while the blue one is the task path. (b) Command $\boldsymbol{u}$: collective translation of all the control points and translation of some control points along x (blue line) and y (red line) directions; expansion rate w.r.t. the center of mass of the control points (blue line). The vertical black lines indicate a switch to a different command map. (c) Corresponding force feedback $\boldsymbol{\tau}_M$ along x (blue) and y (red) directions.

change to a different command map. Notice how the force feedback (Fig. 6c) when using the map ii) (from $t = 25$ s to $t = 38$ s) is not null only for a short period. This is due to the fact that the map produces a local modification of the path, therefore the null space projection produces a mismatch only when the robot is traveling on this part.

Finally, the third simulation demonstrates the behavior of the planner in presence of two obstacles and a point of interest, while the human operator commands a collective translation of the task path. Figure 7a shows the following snapshots of the simulation: 1) the path is deformed by the two obstacles, $\boldsymbol{o}_1$ and $\boldsymbol{o}_2$, and attracted by the target $\boldsymbol{r}_1$; 2) condition (22) is met for $\boldsymbol{o}_1$ and a new path (green line) is being generated according to (26); 3) the new path is created and evolves according to (24), however the system cannot switch to it because $\gamma(\boldsymbol{x}, s_R) \neq \gamma(\boldsymbol{x}_{o1}, s_R)$; 4) the system has now switched to the new path, and another alternative has been generated by $\boldsymbol{o}_2$. All the switches in the replanning are denoted by solid vertical black lines in the plots of Figs. 7b-7e, while a dashed vertical line indicates that the point of interest is detected. Figure 7b shows the evolution of the average error $e(\boldsymbol{x}, \boldsymbol{x}_h) = \sum_{i=1}^{n} |\boldsymbol{x}_{h,i} - \boldsymbol{x}_i\|$. After every path switch the error becomes smaller, confirming the usefulness of the replanning procedure introduced in Sec. IV.

Figure 7c shows the evolution of the minimum distance between the path and the obstacles and between the path and the point of interest. After every path switch there is an increase in the distance from the corresponding obstacle. On the other hand, the minimum distance from the point of interest decreases when this enters the radius of attraction of the potential function that defines $\boldsymbol{f}_r$ (3 m). The evolution of the artificial forces $\boldsymbol{f}_o$ and $\boldsymbol{f}_r$ is summarized in Fig. 7d. The intensity of the obstacle forces evidently increases when the path gets closer to them, while the intensity of the attractive force decreases when the path gets closer to the point of interest.

Finally, Fig. 7e shows the evolution of the force feedback $\boldsymbol{\tau}_M$. As expected, the force feedback becomes stronger when the path is deformed by the obstacle. Note also that the discontinuities in the feedback when a path switch occurs is helpful to inform the user that the task has been directed past an obstacle and can therefore continue more easily. Similarly, the sudden force that is felt when a point of interest is within range naturally guides the human operator in directing the task towards it.

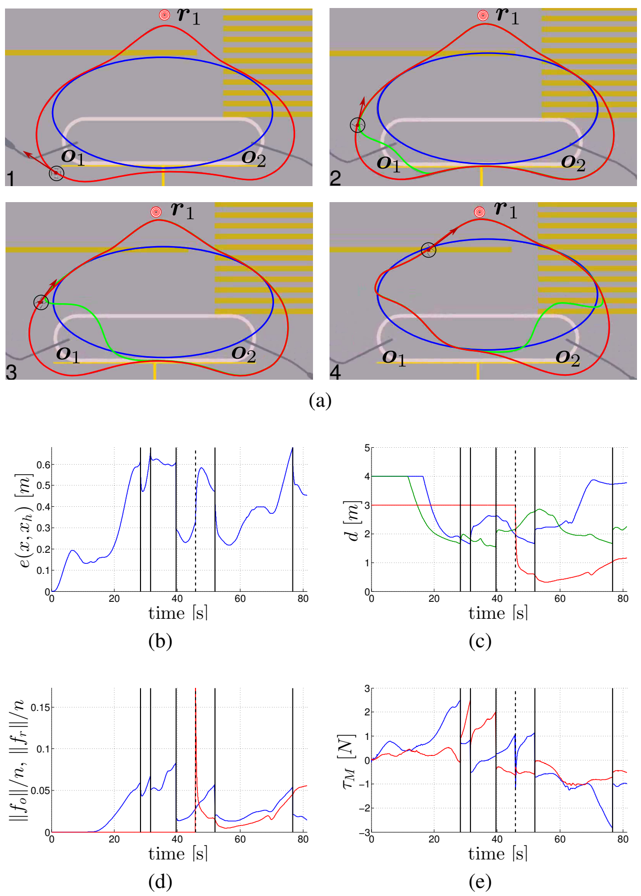
VII. CONCLUSION AND FUTURE WORK

In this work we extended classical persistent-task algorithms where a cyclic motion is executed by the robot, by adding an interaction with a human operator in order to modify online some properties of the path. In this way the autonomy of the robot has been better exploited and the overall human commitment decreased. We also considered the presence of an haptic feedback informative of the global deformation acting on the desired path rather than of a local mismatch between commanded and executed position/velocity commands (as in the the previous literature). The feasibility of the proposed approach has been demonstrated by means of a physically-simulated quadrotor UAV with human/hardware in-the-loop and a real haptic interface.

We are working on the implementation of this framework in a experimental scenario with a real UAV and we also plan to extend the theoretical and experimental results to a multi-robot scenario.

VIII. ACKNOWLEDGEMENTS

This research was partly supported by WCU (World Class University) program funded by the Ministry of Education,



(a)

(b)

(c)

(d)

(e)

Fig. 7: (a) Snapshots of the third simulation, the red path is the current one, the blue indicates the reference path and the green ones are the alternative generated with the replanning. (b-e) Solid black lines indicate a path switch and the dashed black line indicates when the point of interest is detected. (b) Error $e(\boldsymbol{x}, \boldsymbol{x}_h) = \sum_{i=1}^{n} |\boldsymbol{x}_{h,i} - \boldsymbol{x}_i|/n$. (c) Minimum distance from left obstacle (blue), right obstacle (red) and point of interest (red) (d) $\|\boldsymbol{f}_o\|/n$ (blue) and $\|\boldsymbol{f}_r\|/n$ (red). (e) Force feedback $\boldsymbol{\tau}_M$ along x (blue) and y (red) directions.

Science and Technology through the National Research Foundation of Korea (R31-10008). The authors also wish to thank Martin Riedel and Johannes Lächele for their help in the simulative setup.

References

[1] C. P. G. Niemeyer and G. Hirzinger, "Telerobotics," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Springer, 2008, pp. 741–757.

[2] A. Franchi, C. Secchi, M. Ryll, H. H. Bülthoff, and P. Robuffo Giordano, "Shared control: Balancing autonomy and human assistance with a group of quadrotor UAVs." *IEEE Robotics & Automation Magazine*, vol. 19, no. 3, 2012.

[3] N. Diolaiti and C. Melchiorri, "Teleoperation of a mobile robot through haptic feedback," in *IEEE Int. Work. on Haptic Virtual Environments and Their Applications*, Ottawa, Canada, Nov. 2002, pp. 62–72.

[4] D. J. Lee, O. Martinez-Palafox, and M. W. Spong, "Bilateral teleoperation of a wheeled mobile robot over delayed communication network," in *2006 IEEE Int. Conf. on Robotics and Automation*, Orlando, FL, May 2006, pp. 3298–3303.

[5] A. Franchi, P. Robuffo Giordano, C. Secchi, H. I. Son, and H. H. Bülthoff, "A passivity-based decentralized approach for the bilateral teleoperation of a group of UAVs with switching topology," in *2011 IEEE Int. Conf. on Robotics and Automation*, Shanghai, China, May 2011, pp. 898–905.

[6] D. Lee, A. Franchi, P. Robuffo Giordano, H. I. Son, and H. H. Bülthoff, "Haptic teleoperation of multiple unmanned aerial vehicles over the internet," in *2011 IEEE Int. Conf. on Robotics and Automation*, Shanghai, China, May 2011, pp. 1341–1347.

[7] P. Robuffo Giordano, A. Franchi, C. Secchi, and H. H. Bülthoff, "Passivity-based decentralized connectivity maintenance in the bilateral teleoperation of multiple UAVs," in *2011 Robotics: Science and Systems*, Los Angeles, CA, Jun. 2011.

[8] A. Franchi, C. Masone, H. H. Bülthoff, and P. Robuffo Giordano, "Bilateral teleoperation of multiple UAVs with decentralized bearing-only formation control," in *2011 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, San Francisco, CA, Sep. 2011, pp. 2215–2222.

[9] A. Franchi, C. Secchi, H. I. Son, H. H. Bülthoff, and P. Robuffo Giordano, "Bilateral teleoperation of groups of mobile robots with time-varying topology," *IEEE Trans. on Robotics*, In Press, Electronically published at http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6199993.

[10] S. Quinlan and O. Khatib, "Elastic bands: Connecting path planning and control," in *1993 IEEE Int. Conf. on Robotics and Automation*, Atlanta, GA, May 1993, pp. 802–807.

[11] O. Brock and O. Khatib, "Elastic strips: A framework for motion generation in human environments," *International Journal of Robotics Research*, vol. 21, no. 12, pp. 1031–1052, 2002.

[12] F. Lamiraux, D. Bonnafous, and O. Lefebvre, "Reactive path deformation for nonholonomic mobile robots," *IEEE Trans. on Robotics*, vol. 20, no. 6, pp. 967–977, 2004.

[13] S. L. Smith, M. Schwager, and D. Rus, "Persistent monitoring of changing environments using a robot with limited range sensing," in *2011 IEEE Int. Conf. on Robotics and Automation*, Shanghai, China, May 2011, pp. 5448–5455.

[14] D. E. Soltero, S. L. Smith, and D. Rus, "Collision avoidance for persistent monitoring in multi-robot systems with intersecting trajectories," in *2011 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, San Francisco, CA, Sep. 2011, pp. 3645–3652.

[15] E. W. Frew, D. A. Lawrence, C. Dixon, J. Elston, and W. J. Pisano, "Lyapunov guidance vector fields for unmanned aircraft applications," in *2007 American Control Conference*, New York, USA, Jul. 2007, pp. 371–376.

[16] H. Chen, K. Chang, and C. S. Agate, "Tracking with UAV using tangent-plus-lyapunov vector field guidance," in *12th Int. Conf. on Information Fusion*, Seattle, WA, Jul. 2009, pp. 363–372.

[17] M. A. Hsieh, S. Loizou, and V. Kumar, "Stabilization of multiple robots on stable orbits via local sensing," in *2007 IEEE Int. Conf. on Robotics and Automation*, Rome, Italy, Apr. 2007, pp. 2312–2317.

[18] L. Sabattini, C. Secchi, C. Fantuzzi, and D. de Macedo Possamai, "Tracking of closed-curve trajectories for multi-robot systems," in *2010 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Taipei, Taiwan, Oct. 2010, pp. 6089–6094.

[19] M. Fliess, J. Lévine, P. Martin, and P. Rouchon, "Flatness and defect of nonlinear systems: Introductory theory and examples," *International Journal of Control*, vol. 61, no. 6, pp. 1327–1361, 1995.

[20] A. Isidori, *Nonlinear Control Systems, 3rd edition*. Springer, 1995.

[21] R. M. Murray, M. Rathinam, and W. Sluis, "Differential flatness of mechanical control systems: A catalog of prototype systems," in *ASME Int. Mechanical Eng. Congress and Exposition*, San Francisco, CA, Nov. 1995.

[22] V. Mistler, A. Benallegue, and N. K. M'Sirdi, "Exact linearization and noninteracting control of a 4 rotors helicopter via dynamic feedback," in *10th IEEE Int. Symp. on Robots and Human Interactive Communications*, Bordeaux, Paris, France, Sep. 2001, pp. 586–593.

[23] B. Siciliano and O. Khatib, *Handbook of Robotics*. Springer, 2008.

[24] D. J. Lee and D. Xu, "Feedback r-passivity of lagrangian systems for mobile robot teleoperation," in *2011 IEEE Int. Conf. on Robotics and Automation*, Shanghai, China, May 2011, pp. 2118–2123.

[25] 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.

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

[27] M. Riedel, "Telekyb: A modular software framework for bilateral teleoperation scenarios and its applications in robotics research," Master Thesis, Eberhard Karls Universität Tübingen, 2012.

[28] J. Lächele, "Development of a real-time simulation environment for multiple robot systems," Master Thesis, Eberhard Karls Universität Tübingen, http://laechele.eu/SwarmSimX/, 2012.