Shared Control in Robot Teleoperation With Improved Potential Fields

Alberto Gottardi, Stefano Tortora, Elisa Tosello, and Emanuele Menegatti

***Abstract*—In shared control teleoperation, the robot assists the user in accomplishing the desired task. Rather than simply executing the user's command, the robot attempts to integrate it with information from the environment, such as obstacle and/or goal locations, and it modifies its behavior accordingly. In this article, we propose a real-time shared control teleoperation framework based on an artificial potential field approach improved by the dynamic generation of escape points around the obstacles. These escape points are virtual attractive points in the potential field that the robot can follow to overcome the obstacles more easily. The selection of which escape point to follow is done in real time by solving a soft-constrained problem optimizing the reaching of the most probable goal, estimated from the user's action. Our proposal has been extensively compared with two state-of-the-art approaches in a static cluttered environment and a dynamic setup with randomly moving objects. Experimental results showed the efficacy of our method in terms of quantitative and qualitative metrics. For example, it significantly decreases the time to complete the tasks and the user's intervention, and it helps reduce the failure rate. Moreover, we received positive feedback from the users that tested our proposal. Finally, the proposed framework is compatible with both mobile and manipulator robots.**

***Index Terms*—Artificial potential fields (APFs) le2, collision avoidance, human–robot interaction, shared control, soft constraint satisfaction problem (CSP), teleoperation.**

I. Introduction

TELEOPERATION allows a human user to control a robotic device through a human–machine interface (HMI) (e.g., a joystick). The ability to interact with remote or inaccessible environments made this technique widespread and effective in several applications. Examples include underwater and space exploration, telepresence, and telesurgery [1]. In direct teleoperation, the user fully controls the robot via HMI: commands are directly mapped to robot actions with no processing. However, the interface may have fewer degrees of freedom (DoFs) than the controlled robot [2], [3]. Moreover, user commands may be unreliable due to noise or disturbances, particularly in cognitive HMIs such as electromyographic interface or brain–computer interface (BCI) [4]. The robot workspace may be partially visible to the user, e.g., due to camera occlusions. Finally, user skills may be compromised by motor and/or neurological disabilities. In all these cases, the difficulty of controlling the robotic device increases the user workload and generates a sense of frustration and irritation.

To alleviate these limitations, shared control [5] allows the teleoperated robot to assist its human guide by contextualizing the delivered commands in terms of robot state in the environment. The aim is to let users focus on their intended goal, while robotic intelligence handles low-level control problems, such as adjusting the robot's trajectory or avoiding collisions with obstacles. Shared control is effective in several applications that require precise operations, e.g., in assembly or feeding tasks [6]. It also improves the independence of people with disabilities in both daily-living mobility [7] and manipulation [8]. Several approaches, with different assistance types [9], have been proposed to achieve for shared control teleoperation of robotic devices. From these works and our own experience, we can deduce the following requirements.

1) *Intent recognition:* If we assume a complete robot knowledge of the intended user task, we can optimize assistance to complete that specific assignment [10]. However, users may want to achieve multiple goals, and the robotic system may not know *a priori* users' intentions. In this context, an implicit interpretation of users' intention from their actions should exist to guarantee a natural and effective human–robot interaction [11]. Moreover, since the user intention prediction is generally uncertain, the shared-controlled robot should assist the motion toward the whole goal distribution [12].
2) *Safe motion*: Controlling a robotic device without damaging it and its surrounding is essential to ensure safe robot behavior in real-world applications. With no goal knowledge, shared control should precisely reflect the user's commands: only minimal robot's trajectory alterations are allowed to avoid collisions. If the goal is precisely estimated, instead, teleoperation can benefit from collision avoidance to optimize the robot's control in following the shortest path to the target [13].
3) *Seamless interaction*: Goodrich and Schultz [14] identified ten autonomy levels for a shared-controlled robot,

Manuscript received March 31, 2021; revised October 30, 2021 and January 6, 2022; accepted February 18, 2022. This work was supported in part by the Italian Ministry of Education, University and Research under the initiative "Departments of Excellence (Law 232/2016)" and in part by Fondazione Cariverona under the Project "Collaborazione Uomo-Robot per Assemblaggi Manuali Intelligenti." This article was recommended by Associate Editor Emilia I. Barakova. *(Stefano Tortora and Elisa Tosello contributed equally to this work.) (Corresponding author: Stefano Tortora.)*

The authors are with the Intelligent Autonomous Systems Laboratory, Department of Information Engineering, University of Padova, 35131 Padova, Italy (e-mail: alberto.gottardi@studenti.unipd.it; stefano.tortora@unipd.it; elisa.tosello@unipd.it; emanuele.menegatti@unipd.it).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/THMS.2022.3155716.

Digital Object Identifier 10.1109/THMS.2022.3155716

going from direct teleoperation to fully autonomous task completion. Selecting the best control level is critical to reducing the users' workload while keeping them feeling in control of the robotic device. Ideally, we should allow smooth and dynamic transitions between autonomy levels. In this way, at each time instant, robot control depends on the weighted cooperation between the two agents based on current contextual data [15].

A. Contribution

To address the requirements outlined above, we propose a framework for real-time shared control teleoperation that integrates the prediction of user's intention with collision avoidance. Collision avoidance is based on an improved artificial potential field (APF) [16] method that includes the dynamic generation of attractive points around the obstacles, called escape points. Escape points generate pathways that the robot may follow to avoid collisions. We estimate the best escape point sequence dynamically during teleoperation by solving a constraint satisfaction problem (CSP) [17] that optimizes both obstacle overcoming and goal achievement. Our contributions are as follows.

1) a novel APF approach with improved performance compared with the state-of-the-art;
2) a full framework for real-time shared control teleoperation that is compatible with both mobile and manipulator robots, both in static and dynamic environments;
3) an exhaustive and extensive comparison of our proposal with two baseline approaches toward performance and users' satisfaction. Our evaluation comprises both a simulated and a complex real-world setup. Moreover, we test our system in both a static and a dynamic environment, the latter populated with moving and movable obstacles.

The rest of this article is organized as follows. Section II provides an overview of the state-of-the-art for shared control teleoperation. Section III depicts our proposal, fully detailing its collision avoidance system based on APFs and escape points. Section IV compares our framework with the state-of-the-art within both a static and a dynamic environment. Section V analyzes obtained results in the static setup, while Section VI shows the ones obtained in the dynamic scenario. Finally, Section VII concludes this article.

II. Related Work

In shared control, the robot assists the user to accomplish the desired task. To simplify the intent inference problem, Losey *et al.* [18] embed the robot's actions into a low-dimensional latent space, while Quere *et al.* [6] propose a constraint-based shared control scheme to define skills, which provide support during task execution. In general, many works focus on predefined tasks or assume that the robot knows the user's intent [10]. However, in real-world applications, the user's goal may be unknown; thus, target prediction should exploit the user's commands. A *hidden Markov model* (HMM) is one of the most widely used predictors for interpreting human intention while executing a task [19]: it predicts the intent treating it as the latent state of the model. Other techniques are based on a *Bayesian network*. Tahboub [20], for example, models the intention–action–state scenario through a *dynamical Bayesian network* (DBN) to facilitate probabilistic intention recognition. Recent solutions [12], [21], [22] combine the prediction of user's intention and the generation of robot assistance into a unique framework based on inverse reinforcement learning. For example, in [22], the shared control problem is modeled as a *partially observable Markov decision process*, while the prediction of user's intention is based on the *maximum entropy inverse optimal control* (MaxEnt IOC) [23]. This approach showed good performance in clutter environments, where many possible targets exist, one next to the other.

Other works dealt with shared control by focusing on collision avoidance. In literature, many motion planning algorithms exist that are particularly useful in high-dimensional spaces, e.g., *probabilistic roadmap* and *rapidly exploring random tree* (RRT). Some works exploit RRT for the control of a semiautonomous robotic manipulator [24]. However, such sampling-based algorithms show their full potential in offline motion planning. APFs [25], instead, due to their low computational cost, are one of the most widely used techniques for dynamic motion planning in teleoperation scenarios. Conventional APF approaches have some well-known limitations, such as getting trapped in local minima, movement oscillations, and unreachable goals in very clutter environments. In literature, several APF variations exist that try to overcome these limitations. For example, navigation functions [26] and harmonic potentials [27] have been proposed to overcome the local minima problem. However, they significantly increase the computational cost. In [28], movement oscillations are reduced by computing the gradient of the potential function. In [29] and [30], collision avoidance is obtained through a combination of potential fields and model-predictive control. You and Hauser [10] implemented and combined a reactive potential field, an inverse kinematics with predictive safety filter, and a real-time sample-based motion planner based on the RRT method.

The above works generally consider only one aspect of shared control, either the prediction and assistance to the goal or collision avoidance. On the other hand, only a few solutions optimize goal-reaching and collision overcoming within the same framework. In [31], an HMM predictor identifies user intention, while obstacles are avoided thanks to potential fields. More recently, in [32], a DBN for user motion prediction is combined with a tree-based search of collision-free paths. In this article, we present a complete shared control framework for teleoperation, integrating prediction and assistance to the user's intended goal with collision avoidance. We implicitly infer target prediction from the user's actions through a reinforcement learning approach. Moreover, we propose an improved potential field method for dynamic robot motion planning that exploits the user's goal estimation to identify the best path to overcome the obstacles and reach the goal. Our simulated analyses and real-time experiments demonstrate the efficacy of our approach in assisting the users in teleoperation while reaching some targets in dynamic clutter environments.

In our experiments, we suppose continuous human teleoperation, while the robot continuously assists the user in reaching

the target while avoiding static and dynamic obstacles. Other works focus on other dynamicity aspects. Broad *et al.* [33] dynamically allocate control authority to the user based on a learned model of the joint human–robot system. Reddy *et al.* [34] propose an algorithm for shared autonomy that uses model-free reinforcement learning to help human users with tasks with unknown robot dynamics, user policies, and goal representation. Such features can be integrated into our system to face both system dynamics and workspace changes.

III. Our Framework

This section depicts our shared control framework.[1] Briefly, it integrates a goal predictor based on MaxEnt IOC with an improved version of APF. In particular, we generate a group of escape points around the obstacles to prevent local minima and improve target achievement. A detailed description of the adopted predictor follows, together with our technique for collision avoidance, escape point generation, and selection.

A. Goal Prediction

We assume to have a set of N possible destinations and that each teleoperation task conduces the robot to one and only one of them. Only the user knows the target point, and this goal does not change during the whole task execution. Thus, given the set of N goals, we aim at providing an efficient assistance to complete the task by inferring, at first, the most probable target according to the user's actions. We also assume that the user is aware that the robotic intelligence handles collisions, making the user's commands addressed toward the target. Given this assumption, we model the human's behavior toward a certain goal as a Markov decision process [35] $\mathcal{M}$ with policy $\pi(u|\boldsymbol{p},\boldsymbol{p}_g)$, where each state of $\mathcal{M}$ is a robot position $\boldsymbol{p}\in\mathcal{W}$ in the workspace, and $u\in\mathcal{U}$ is the user's action. The goal destination is a robot position $\boldsymbol{p}_g\in\mathcal{W}$, and the state transition function is $T(\boldsymbol{p}'|\boldsymbol{p},u)$, with $\boldsymbol{p}'$ being the next robot position. The cost function $C_g^u(\boldsymbol{p},u)$ is defined as

$$C_g^u(\boldsymbol{p},u)=\begin{cases}\alpha, & d>\delta\\ \alpha\frac{d}{\delta}, & d\le\delta\end{cases} \tag{1}$$

where d is the distance between $\boldsymbol{p}'=T(\boldsymbol{p},u)$ and $\boldsymbol{p}_g$. This model can be solved as a reinforcement learning problem through the well-known value function method [36]. In particular, we compute the optimal policy of $\mathcal{M}$ through MaxEnt IOC, and we solve it via dynamic programming, as suggested in [22]. Knowing the optimal user's policy and given the sequence of robot states and user's commands $\xi^{0\to t}=\{\boldsymbol{p}_0,u_0,\dot{sc},\boldsymbol{p}_t,u_t\}$ from the beginning of the task to time t, the probability that the human user has chosen one specific target location $\boldsymbol{p}_g$ (i.e., $\boldsymbol{p}_{g_n}$) among a set $\mathcal{G}=\{\boldsymbol{p}_{g_1},\ldots,\boldsymbol{p}_{g_N}\}$ of available goals is estimated as

$$P(\boldsymbol{p}_g)=p(\boldsymbol{p}_g|\xi^{0\to t})=\frac{p(\xi^{0\to t}|\boldsymbol{p}_g)p(\boldsymbol{p}_g)}{\sum_{\boldsymbol{p}'_g}p(\xi^{0\to t}|\boldsymbol{p}'_g)p(\boldsymbol{p}'_g)}$$

$$\text{with } p(\xi^{0\to t}|\boldsymbol{p}_g)=\prod_t\pi_t^u(u_t|\boldsymbol{p}_t,\boldsymbol{p}_g) \tag{2}$$

where $p(\boldsymbol{p}_g)$ is the prior probability of $\boldsymbol{p}_g$. $\boldsymbol{p}'_g$, instead, is the sum iterator at the denominator; it normalizes the product of the probabilities at the numerator over the whole set of goals (Bayes formula).

B. Artificial Potential Fields

APFs are generated in the Cartesian space of $\mathcal{W}\in\mathbb{R}^3$ and operate as a gradient descent search to minimize the potential function. Assuming to know the position and geometry of the objects in the workspace, we follow the method proposed in [16], and we generate a repulsive field around the obstacles according to the *FIRAS function*

$$U_r(\boldsymbol{p})=\begin{cases}\frac{\eta}{2}\left(\frac{1}{\rho(\boldsymbol{p})}-\frac{1}{\rho_0}\right)^2, & \text{if}\rho(\boldsymbol{p})\le\rho_0\\ 0, & \text{if}\rho(\boldsymbol{p})>\rho_0\end{cases} \tag{3}$$

where $\rho(\boldsymbol{p})$ represents the distance between $\boldsymbol{p}\in\mathcal{W}$ and the closest point on the obstacle, while ρ_0 is the influence ray of the obstacle. The robot kinematic is, thus, determined by applying a force equal to the negative gradient of the potential

$$\boldsymbol{f}_r(\boldsymbol{p})=-\nabla U_r(\boldsymbol{p})=\frac{\eta}{\rho^2(\boldsymbol{p})}\left(\frac{1}{\rho(\boldsymbol{p})}-\frac{1}{\rho_0}\right)\nabla\rho(\boldsymbol{p}). \tag{4}$$

The global minimum of the gradient descent search is the goal position $\boldsymbol{p}_g$. To produce this global minimum, an attractive field is generated around $\boldsymbol{p}_g$ as a conical well

$$U_a^g(\boldsymbol{p})=k_a\|\boldsymbol{p}_g-\boldsymbol{p}\| \tag{5}$$

where $k_a>0$ is the attraction constant. The following attraction force results:

$$\boldsymbol{f}_a(\boldsymbol{p})=-\nabla U_a^g(\boldsymbol{p})=k_a\frac{\boldsymbol{p}_g-\boldsymbol{p}}{\|\boldsymbol{p}_g-\boldsymbol{p}\|}. \tag{6}$$

The robot velocity due to the effect of APF is then the combination of the repulsive and attractive forces. In case of shared control teleoperation, this velocity modifies the input velocity $\boldsymbol{v}_{\text{in}}$ provided by the user to reach the desired goal

$$\boldsymbol{v}_r=\boldsymbol{v}_{\text{in}}-(\nabla U_a^g(\boldsymbol{p})+\nabla U_r(\boldsymbol{p})). \tag{7}$$

Given multiple goals and the predictor of Section III-A, we can update the potential field function accounting for the probability distribution over these goals

$$\boldsymbol{v}_r=\boldsymbol{v}_{in}-\left(\sum_{\boldsymbol{p}_g\in\mathcal{W}}P(\boldsymbol{p}_g)\nabla U_a^g(\boldsymbol{p})+\nabla U_r(\boldsymbol{p})\right). \tag{8}$$

This means that the robot will be attracted more toward that most probable destination of the teleoperation task.

C. Improved APF With Escape Points

Despite its diffusion among the robotic community, APF suffers from some well-known limitations. First, the robot risks getting stuck where the potential gradient becomes zero, i.e., in

[1]The GitHub repository of our framework is available at https://github.com/Shared-control/improved-apf.git

local minima. As highlighted in Section II, variations of APF partially solve this problem through more advanced potential functions that ensure the gradient to be zero only in a global minimum (e.g., the goal). The second problem of APF is that collision avoidance is purely reactive: the robot velocity is determined only by the distance between the robot and the obstacles. Thus, the robot modifies its trajectory to overcome the obstacle only when it is close to it and not in advance. We can increase ρ_0, the influence ray of the obstacles. However, this solution may prevent the robot from reaching the goal in cluttered environments with multiple nearby obstacles.

To overcome these limitations, we enhance APF by introducing escape points, i.e., points appropriately created around obstacles to help their smooth overcoming while avoiding local minima. We took inspiration from [37], extending their proposal to a 3-D environment. In detail, given the position $\boldsymbol{p}_o \in \mathcal{W}$ of an obstacle and its geometry, we identify its convex vertices and the corresponding convex edges, and we sample escape points close to these edges with a certain density α. Then, we choose the escape point $\boldsymbol{p}_e \in \mathcal{W}$ that lets the robot bypass the obstacle better while approaching the most likely target. To perform this selection, we solve a CSP composed of one *hard* and three *soft* constraints. Their description is as follows.

1) **C0***(hard):* Given the tuple $< \boldsymbol{p}, \boldsymbol{p}_e >$, $\boldsymbol{p}$ and $\boldsymbol{p}_e$ must be mutually visible and the line through them must poke into $\mathcal{C}_{\text{free}}$ when extended outward from each of these two points, with $\mathcal{C}_{\text{free}}$ that portion of the robot configuration space free from obstacles. Formally, $\vec{\boldsymbol{p}\boldsymbol{p}}_e \in \mathcal{C}_{\text{free}} \forall < \boldsymbol{p}, \boldsymbol{p}_e >$.
2) **C1***(soft):* Given the set of tuples $< \boldsymbol{p}_g, P(\boldsymbol{p}_g) >$ relating a goal to its probability, this constraint aims to favor the most likely target at the current time.
3) **C2***(soft):* This constraint considers the set of tuples $< \boldsymbol{p}_e, \frac{1}{d(\boldsymbol{p}_e, \boldsymbol{p})} >$, where $\frac{1}{d(\boldsymbol{p}_e, \boldsymbol{p})}$ is the reciprocal of the distance between the escape point and the robot position mapped in the interval [0, 1]. The goal is to privilege the nearest (visible) escape point. The distance definition can be adapted according to the use case. In our experiments, we consider the Euclidean distance in the Cartesian space.
4) **C3***(soft):* We have the set of tuples $< \boldsymbol{p}_e, \frac{1}{d(\boldsymbol{p}_e, \boldsymbol{p}_g)} >$, where $\frac{1}{d(\boldsymbol{p}_e, \boldsymbol{p}_g)}$ is the reciprocal of the distance between the escape point and the goal mapped in [0, 1]. We should minimize the distance between the selected escape point and the selected goal. In addition, in this case, the distance can be arbitrarily defined, and we exploit the Euclidean distance as a metric.

We solve the CSP in real time. Once selected the set of escape points that satisfy the hard constraint **C0**, we solve the CSP by using a fuzzy c-semiring structure $\langle [0, 1], \max, \min, \mathbf{0}, \mathbf{1} \rangle$ [17]: we give a preference level in [0,1] to the partial solution of each constraint, where we consider a higher level better than the others. Then, we obtain the preference of a global solution as the minimal preference on all constraints, where this preference is the optimal escape point. We generate an attractive field at the selected escape point $\boldsymbol{p}_e$ as a parabolic well

$$U_a^e(\boldsymbol{p}) = \frac{1}{2} k_e \|\boldsymbol{p}_e - \boldsymbol{p}\|^2 \tag{9}$$

where k_e is the attraction constant. The following attraction force results:

$$\boldsymbol{f}_a^e(\boldsymbol{p}) = -\nabla U_a^e(\boldsymbol{p}) = k_e(\boldsymbol{p}_e - \boldsymbol{p}). \tag{10}$$

Adding this force to (7), we obtain the final robot velocity

$$\boldsymbol{v}_r = \boldsymbol{v}_{\text{in}} - \left(\sum_{\boldsymbol{p}_g \in \mathcal{W}} P(\boldsymbol{p}_g) \nabla U_a^g(\boldsymbol{p}) + \nabla U_a^e(\boldsymbol{p}) + \nabla U_r(\boldsymbol{p}) \right). \tag{11}$$

To illustrate our improved-APF method, we refer to a simple 2-D scenario, and we propose four case studies where both the robot and the goal are modeled as dots in a bidimensional space with $v_{\text{in}} = 0$. Fig. 1 shows obtained results: (a) and (b) do not contain local minima, and the robot successfully reaches the goal (red dot) both via APF and through our enhanced proposal. However, escape points (blue dots) let achieve the target more quickly by guiding the robot around the obstacle in advance. In Fig. 1(c), the obstacles form a closed aisle: APF causes the robot to get stuck in a local minimum. Our method, instead, lets the robot avoid this local minimum and successfully reach the goal. In detail, initially, the robot moves from its start configuration to point A of Fig. 1(c) as this point is closer to the robot (in terms of the Euclidean distance). While approaching A, B becomes the best escape point as it is relatively close to A (thus, to the new robot position) but closer to the target. This escape point change lets the robot accomplish the assignment. Finally, in Fig. 1(d), the starting point is already at a local minimum. The classical APF does not allow the robot to exit from it, while our method successfully reaches the goal thanks to the attraction to the closest escape point.

D. Shared Control Framework

Algorithm 1 depicts our resulting shared control framework. It receives as input the initial position $\boldsymbol{p}$ of the robot and the set $\mathcal{G}$ of available target destinations that it can reach, with $\mathcal{G} = \{\boldsymbol{p}_{g_1}, \ldots, \boldsymbol{p}_{g_N}\}$. Given a not null user's velocity command, it computes the probability distribution of the targets, as described in Section III-A. This procedure lets predict the most probable target destination (line 7 in Algorithm 1). Having the robot initial position, goals, and goal probability distribution, the algorithm initializes $\boldsymbol{v}_{\text{APF}}$, the robot velocity due to the improved-APF (line 8 in Algorithm 1). As in this phase, no obstacle is known, only the attractive force of the most probable goal will act. Then, the procedure searches for obstacles $\mathcal{O} = \{\boldsymbol{p}_{o_1}, \ldots, \boldsymbol{p}_{o_M}\}$ in the scene (line 9 in Algorithm 1). If obstacles exist that occlude the goal, namely $\overline{\mathcal{O}}$, an escape point $\boldsymbol{p}_e$ should be found (line 13 in Algorithm 1). Algorithm 2 details escape points generation and selection. Briefly, we generate the set $\mathcal{V}$ of convex vertices of the occluding obstacles (line 1 in Algorithm 2), filter the visible ones (line 2 in Algorithm 2), and choose the best escape point $\boldsymbol{p}_e \in \bar{\mathcal{V}}$, as depicted in Section III-C (line 3 in Algorithm 2). Once having $\boldsymbol{p}_e$, the velocity imprinted by our improved-APF can be updated taking into consideration the repulsive forces of existing obstacles and occluding obstacles. This means that all obstacles, not only the occluding ones, contribute to the generation of this repulsive force. Moreover, our escape point

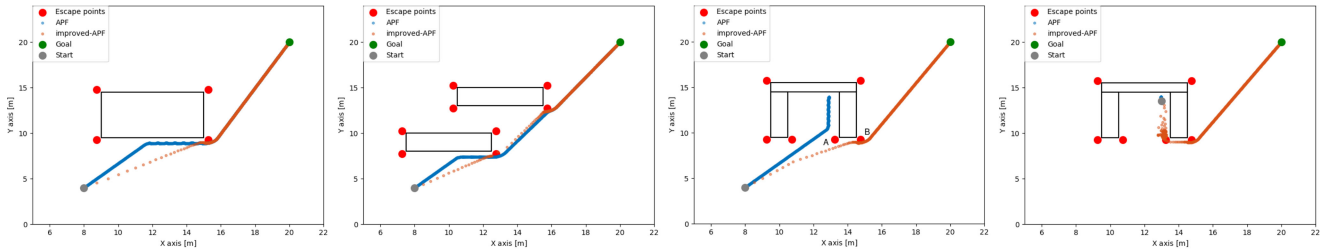


Fig. 1. 2-D simulated examples. (a) Rectangular obstacle. (b) Two rectangular obstacles. (c) and (d) Closed aisle. For every setup, figures show the robot start position (gray dot) and the goal one (green dot). APF lets the robot reach the goal by following the blue trajectory. Our improved-APF version, instead, computes the red trajectory thanks to the generated escape points (red dots). Our method introduces a visible improvement with respect to the state-of-the-art.

Algorithm 1: SharedControl.

Input: $\boldsymbol{p}$, $\mathcal{G}$
1: $g_{\text{reached}} = \textbf{false}$
2: $\boldsymbol{v}_{\text{in}} = 0$
3: **while** $!g_{\text{reached}}$ **do**
4: **while** $\boldsymbol{v}_{\text{in}} = 0$ **do**
5: $\boldsymbol{v}_{\text{in}} = \text{takeUserInput}()$
6: **end while**
7: $P(\boldsymbol{p}_g) = \text{predictGoal}(\boldsymbol{v}_{\text{in}}, \boldsymbol{p}, \mathcal{G})$
8: $\boldsymbol{v}_{\text{APF}} = \text{improvedAPF}(\boldsymbol{p}, \mathcal{G}, P(\boldsymbol{p}_g))$
9: $\mathcal{O} = \text{findObstaclesInScene}()$
10: **if** $\mathcal{O} \neq \emptyset$ **then**
11: $\overline{\mathcal{O}} = \text{findOccludingObstacles}(\boldsymbol{p}, \mathcal{G}, \mathcal{O}, P(\boldsymbol{p}_g))$
12: **if** $\overline{\mathcal{O}} \neq \emptyset$ **then**
13: $\boldsymbol{p}_e = \text{findEscapePoint}(\boldsymbol{p}, \mathcal{G}, P(\boldsymbol{p}_g), \overline{\mathcal{O}})$
14: $\boldsymbol{v}_{\text{APF}} = \text{improvedAPF}(\boldsymbol{p}, \mathcal{G}, P(\boldsymbol{p}_g), \mathcal{O}, \overline{\mathcal{O}}, \boldsymbol{p}_e)$
15: **end if**
16: **end if**
17: $\boldsymbol{v}_r = \boldsymbol{v}_{\text{in}} - \boldsymbol{v}_{\text{APF}}$
18: $\boldsymbol{p} = \text{move}(\boldsymbol{v}_r)$
19: $g_{\text{reached}} = \text{isGoalReached}(\boldsymbol{p}, \mathcal{G}, P(\boldsymbol{p}_g))$
20: **end while**

Algorithm 2: FindEscapePoint.

Input: $\boldsymbol{p}$, $\mathcal{G}$, $P(\boldsymbol{p}_g)$, $\overline{\mathcal{O}}$
Output: $\boldsymbol{p}_e$
1: $\mathcal{V} = \text{generateConvexVertices}(\overline{\mathcal{O}})$
2: $\bar{\mathcal{V}} = \text{hardCSP(p, } \mathcal{V})$
3: $\boldsymbol{p}_e = \text{softCSP}(\boldsymbol{p}, \mathcal{G}, P(\boldsymbol{p}_g), \bar{\mathcal{V}})$
4: return $\boldsymbol{p}_e$

is added as an attractive force, together with the attractive force of the goal (line 17 in Algorithm 1). Finally, the robot velocity $\boldsymbol{v}_r$ is computed according to (11), and the robot moves toward the goal according to this velocity (line 18 in Algorithm 1). The process iterates until the goal is reached.

IV. Experiments

To exhaustively evaluate our proposal, we provide a comprehensive set of experiments that compare our shared control framework, named *improved-APF* in this article, with two baselines. The first baseline is a fully teleoperated approach called *Teleoperation*. The other is *APF*, i.e., the shared control approach of Section III-B. It adopts APF to push users away from obstacles, but it does not exploit any escape point. In the experiments, a human user is asked to accomplish a picking task by remotely controlling a robotic manipulator. We selected this task as it appears broadly in teleoperation. Indeed, its simplicity of use guarantees a fair comparison of the adopted control strategies both when accomplishing relatively easy tasks and when performing complex manipulations in the clutter. We tested the systems in both a static and a dynamic environment. In the former, goals and obstacles do not move during task execution. In the latter, instead, targets are fixed (but can change at each trial), while two external human users interfere friendly or adversary with the environment by randomly moving the obstacles. A detailed description of the two experimental setups follows, together with the metrics and procedure used for evaluation.

A. Evaluation Setups

In both the static and the dynamic setups, a table is in front of a manipulator robot. On it, some red cubes represent the objects to be picked (goals fixed in the scene), while other objects of different color and shape serve as obstacles (fixed and/or movable). Every object is equipped with an Apriltag fiducial marker [38] that lets its detection. A Microsoft Kinect One [39] perceives the workspace, detects the objects, and recognizes them. Once detected, a picking point is computed on top of each goal. Thus, when a target is selected, the human user guides the robot to the picking location on its top. And once the user lets the arm reach this location, the robot automatically concludes the picking routine. The scene (e.g., the location of obstacles and their escape points) is updated at 30 Hz during task execution. The robotic agent is a six-DoF Universal Robots UR5[2] manipulator with a Robotiq 3-Finger[3] Adaptive Gripper attached to its end-effector. The repulsive force generated by each obstacle acts both on the end-effector and on all six joints—if they are within the area of influence of the obstacle itself. The attractive forces, instead, act only on the end-effector. The human user sees the workspace through a camera mounted on the top back of the robot (Fig. 2

[2]See [Online]. Available: https://www.universal-robots.com
[3]See [Online]. Available: https://robotiq.com

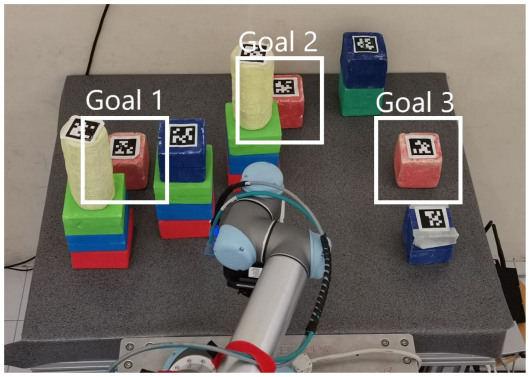


Fig. 2. Scene available to the human user while testing the static setup. *Goal 3* is free from obstacles. *Goal 1* is reachable only by passing the obstacle barrier. *Goal 2* introduces the possibility of choosing the best path: going around the central yellow cylinder from the left or the right.

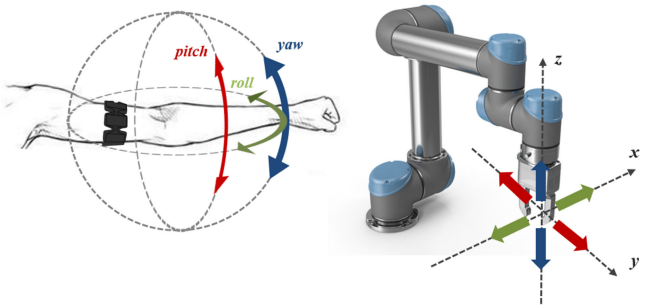


Fig. 3. Myo Armband gesture control: a *roll* of the human arm corresponds to a robot end-effector displacement along the *x*-axis, a *pitch* is a motion along the *y*-axis, and a *yaw* is a *z* movement.

shows the view available to the user). This 2-D representation of the scenario makes the task more challenging due to missing depth perception and aims to better pinpoint the efficacy of shared control assistance in a nonideal working condition. From this view, the user imprints a velocity control to the robot end-effector via Joystick or through a Myo Armband [40]. The former lets the human user easily move the end-effector along the three axes. Using the latter, instead, is more challenging. The Myo is equipped with an inertial sensor measuring linear accelerations and angular velocities. Robot velocity commands depend on the Myo orientation when moved by the user. As shown in Fig. 3, a *roll* of the human arm corresponds to an end-effector displacement along the *x*-axis, a *pitch* is a motion along the *y*-axis, and a *yaw* is a *z* movement. Myo Armband adds complexity to the teleoperation process; this should emphasize the benefits of shared control.

A detailed description of the location of goals and obstacles within both the static and the dynamic setup follows, together with the control interface used to assist the robotic agent.

1) Static Setup: On the table, three red cubes, namely, *Goal 1*, *Goal 2*, and *Goal 3*, represent the objects to be picked (see Fig. 2). Other objects of different color and shape, instead, represent the obstacles to be avoided. We selected specific positions for the goals, which are paradigmatic of common situations during teleoperation. *Goal 3* is free from obstacles. Its aim is to prove that our proposal does not add complexity to the state-of-the-art in terms of both performance and user satisfaction (see Section IV-B). *Goal 1* is hidden by an obstacle barrier. The robot reaches it only if it steps over the obstacles. Escape points should help this agent shorten its trajectory while guaranteeing its smoothness (i.e., these points should encourage the transition between the two cylinders). Finally, *Goal 2* introduces the possibility of choice for the user: it lets decide whether to go around the obstacle (central yellow cylinder of Fig. 2) from the left or the right. Depending on the agent's direction, different escape points will or will not be activated. In this context, the human user can control the robotic agent via Joystick or through the Myo Armband.

2) Dynamic Setup: We subdivide the table into three areas (left, center, and right), and we randomly place two targets (i.e., two red cubes) in one of them. In the same area, two human operators randomly add, move, or remove two obstacles (i.e., two yellow cylinders) by blocking or not blocking the targets. The human user helps the robot reach the assigned goal through the Myo Armband.

B. Evaluation Metrics

We intend to meticulously evaluate both the effectiveness of our proposal and users' satisfaction. To this aim, we adopt the following objective and subjective metrics.

1) Objective Metrics: Inspired by [22], five objective metrics measure the efficiency of our proposal as reported in the next pages. The *failure rate* identifies the degree of failure, i.e., the percentage of incomplete trials, where users do not achieve the intended target destination. The *average execution time* measures the time, on average, participants take to accomplish a correct task. The *average trajectory path* analyzes the path performed by the robotic agent to reach the target. The *total user inputs* measures the number of movements performed during each successful trial. Finally, the *number of direction changes* identifies how many times participants have to change the direction of the robot during task execution. The first three metrics measure how effectively the participants can reach the goals, whereas the last two evaluate their workload.

2) Subjective Metrics: We evaluate both perceived workload and user satisfaction while using *Teleoperation*, *APF*, and *improved-APF*. We assess workload through the Hart and Staveland's NASA Task Load Index (TLX) [41]. It evaluates *mental*, *physical*, and *temporal demands*. Moreover, it analyzes the *performance*, *effort*, and *frustration* of every participant. Ten different answer options are available to an agreement that would be distinct enough for the respondents. It ranges from *Very low* to *Very high*, passing through a neutral midpoint, thus offering an accurate Likert scale that does not throw testers into confusion. Despite the popularity of the NASA TLX, this assessment tool has some criticisms that may limit its effectiveness in the current application [42]. For example, the NASA TLX is affected by prior task load, where a high workload (e.g., pure teleoperation) appears higher after an easy task (e.g., shared control) and *vice versa*. To overcome this limitation, we combine the NASA TLX with an ad-hoc survey that aims to mitigate the differences

TABLE I
SEVEN-POINT LIKERT SCALE SURVEY

	Question
Confidence	How *confident* were you in your ability to move the robot's arm to the desired location?
Control	I felt in *control*.
Expectation	The robot did what I *wanted*.
Speed	I was able to accomplish the tasks *quickly*.
Accuracy	My goals were perceived *accurately*.
Usability	If I was going to teleoperate a robotic arm, I would *like* to use the system.
Satisfaction	I was *satisfied* with my performance.

between objective and subjective measurements of performance. Such a survey is the seven-point Likert scale survey of Table I. It analyzes if, when testing, users are *confident* in using the systems and if they feel in *control*. We also investigate if the robot does what they *want* and, in this case, if it accomplishes the assigned tasks *quickly* and *accurately*. Finally, we inspect whether users *like* the system and if they are *satisfied* with their performance. The survey assesses satisfaction on seven-point scales, from *Very low* to *Very high*.

C. Evaluation Procedure

To compare our system with the state-of-the-art, we asked to a set of volunteers to test the three systems: *Teleoperation*, *APF*, and *improved-APF*. Details about the static and dynamic environments are as follows.

1) Static Setup: In the static setup, we involved 12 volunteers (four women and eight men) aged between 28 and 58, each with full cognitive abilities. The group consists of four nonexpert and eight expert users, the latter with experience in robotics but no prior exposure to our system. Based on their availability and preference, eight volunteers performed the experiments with both control devices, three of them used only the Joystick, and one subject experimented only the Myo Armband. Thus, we tested in total 11 subjects with the Joystick and nine subjects with the Myo Armband. For each subject, we tested each control method in random order. For each control method, participants have to successfully pick each target of Section IV-A1 for three times in random order. In case of nonsuccess, they repeated the trial. Thus, the total number of attempts for each goal is three plus the sum of failures.

2) Dynamic Setup: In the dynamic setup, we involved nine volunteers (two women and seven men) aged between 22 and 34, each with full cognitive abilities. The group consists of three nonexpert and six expert users. Three of them already tested our proposal in the static environment. Volunteers have to perform a total of 15 trials for each of the three control methods, regardless of whether they are successful or not. All volunteers used only the Myo Armband to control the robotic agent.

In both the static and dynamic setups, we consider one trial successful if the robot picks the target and if, during its trajectory, its end-effector, links, and joints do not collide with the obstacles. We use successful trials to compute the objective metrics of Section IV-B. To evaluate the failure rate, instead, we consider the number of times the robot collides with obstacles or picks the wrong item due to a misidentification of the goal predictor. We discard failures due to hardware errors or user misunderstandings, e.g., when the user misinterprets the target. It is meaningful to consider the robot's collisions with objects in the environment even when using shared control methods. Indeed, collisions may occur when both the user's command and goal attraction lead toward an obstacle, and the repulsive forces are not strong enough to avoid the impact [see (7)]. Our proposal should mitigate this limitation as the escape points attraction forces influence the output velocity [see (11)]. Such attraction positively contributes to the overall velocity component bringing the robot away from collision paths.

Before testing, we inform the users if they will use teleoperation or shared control. No explanation is given about the difference between the two shared control systems. We only explain that shared control helps users in collision avoidance over xyz. Besides seeing the scene from the camera, no feedback on robot movements is provided to the user. This choice is intentional since we want to figure out if the robot behavior is perceived as natural or not. Before recording the trials, users perform a couple of training runs to get accustomed with both control systems and devices. This is done because we want to minimize the learning effect during task execution.

Upon completing all trials for one control method, users complete the short surveys of Section IV-B. Then, they test the other systems. Volunteers can review their answers if the current system makes them change their opinion about the previous ones. They can also add comments and explanations.

D. Statistical Analysis

We applied a Kolmogorov–Smirnov test to check if the results of the objective and subjective metrics come from a normal distribution. Since we found that the results are not normally distributed (Kolmogorov–Smirnov test, $p < 0.05$), we applied a nonparametric Friedman's test, with repeated measures between the subjects, to analyze the significant impact of the control methods on the objective metrics. Through a nonparametric Kruskal–Wallis test, instead, we evaluated a significant impact of the control methods on the subjective metrics. Upon significance, we performed a multiple post-hoc comparison between the control methods with two-sided Mann–Whitney test and Bonferroni correction. Effects are deemed significant if the p-value is less than or equal to 0.05.

V. RESULTS OBTAINED IN THE STATIC SETUP

This section illustrates the results obtained for each of the objective and subjective metrics detailed in Section IV in the

TABLE II
FAILURE RATE OBTAINED IN THE STATIC SETUP

Goal	Joystick (11 subjects)						Myo Armband (9 subjects)					
	Teleoperation		APF		Improved-APF		Teleoperation		APF		Improved-APF	
	# failures	fail rate [%]	# failures	fail rate [%]	# failures	fail rate [%]	# failures	fail rate [%]	# failures	fail rate [%]	# failures	fail rate [%]
1	5	13.2	2	5.7	2	5.7	8	22.9	1	3.6	2	6.9
2	3	8.3	5	13.2	1	2.9	8	22.9	8	22.9	4	12.9
3	0	0	0	0	0	0	2	6.9	0	0	0	0
Total	8	7.5	7	6.6	3	2.3	18	18.2	9	10.0	6	6.9

For each control device, the first column depicts the total failures occurred during the experiments, divided by goal. The second column shows the corresponding failure rate.

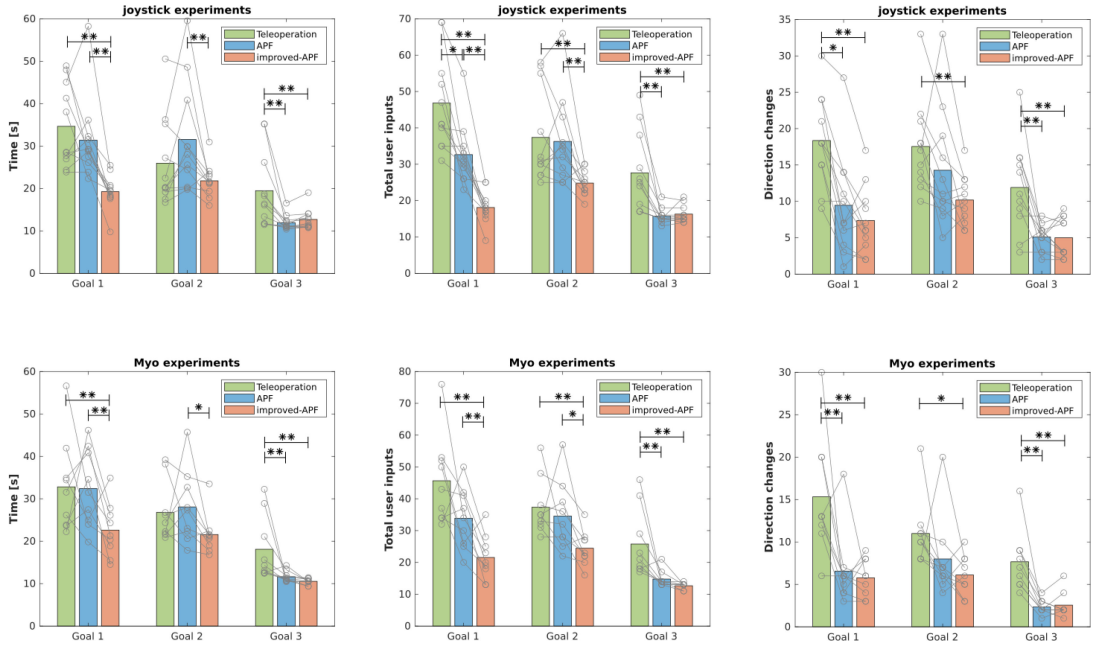


Fig. 4. Average execution time (left column), total user inputs (center column), and direction changes (right column) needed to reach each goal in the static setup. The first row depicts the results obtained when using the Joystick interface. The second row collects performance obtained through the Myo Armband. Gray circles represent the average results of each of the tested subjects. * indicates that $p < 0.05$; ** is for $p < 0.01$.

static evaluation setup (videos of performed experiments are available online[4]).

A. Objective Metrics

All the objective metrics results are grouped based on the goal.

1) *Failure rate:* Table II showcases the total number of failures that occurred during the experimental evaluation for each goal, together with their corresponding failure rate. Both for the joystick and the Myo Armband, our framework outperforms both *Teleoperation* and *APF*. In particular, when using the Joystick, our system reduces the failure rate to about a third compared with its counterparts (showing a failure rate of 2.3% only). As expected, the number of failures increases for the Myo Armband, particularly for *Goal 2*. Nevertheless, also in this more challenging condition, our proposal halved the number of failures, reducing the failure rate from 18.2% of the *Teleoperation* to 6.9% with *improved-APF*. With respect to the participants, the failures are generally equally distributed with an average failure rate across subjects of $11.5 \pm 11.2\%$ for *Teleoperation*, $8.0 \pm 6.9\%$ for *APF*, and $5.1 \pm 4.8\%$ for *improved-APF*, again showing that our proposal allows a safer control of the robotic device.
2) *Average execution time (see Fig. 4, left):* A significant impact of the control method on the execution time is found for all the goals with both the Joystick (*Goal 1* $p = 0.000$, *Goal 2* $p = 0.017$, *Goal 3* $p = 0.000$) and the Myo Armband (*Goal 1* $p = 0.000$, *Goal 2* $p = 0.005$, *Goal 3* $p = 0.000$). As shown in Fig. 4, our framework resulted to be generally faster than the other two systems for all the goals. A statistically significant improvement in the completion time is obtained by the *improved-APF* when reaching *Goal 1* with respect to both *Teleoperation* (Joystick $p = 0.000$, Myo $p = 0.002$) and *APF* (Joystick $p = 0.000$, Myo $p = 0.004$). Indeed, it does not get trapped in a local minimum, and escape points help the robot to overcome the barrier in less time than the other methods. In addition,

[4]Videos at https://www.youtube.com/watch?v=WkMB-4fMR3M

for *Goal 2*, the *improved-APF* is on average faster than the competitors, with a statistically significant difference with respect to *APF* (Joystick $p = 0.007$, Myo $p = 0.018$), but not with respect to *Teleoperation*. Nevertheless, most of the failures obtained with *Teleoperation* and *APF* happens exactly when reaching this target. Thus, even if execution times are similar, our framework is shown to be safer. Finally, to reach *Goal 3*, no escape point is generated, and the performance of *improved-APF* and *APF* is equivalent (Joystick $p = 1.000$, Myo $p = 1.000$). This outcome highlights the important consideration that our proposal does not add complexity to the state-of-the-art.

3) *Total user inputs (see Fig. 4, center):* The second column of Fig. 4 shows the total number of user inputs obtained in both the Joystick (upper figure) and Myo experiments (lower figure). A significant impact of the control method on total user inputs is found for all the goals with both the Joystick (*Goal 1* $p = 0.000$, *Goal 2* $p = 0.000$, *Goal 3* $p = 0.000$) and the Myo Armband (*Goal 1* $p = 0.000$, *Goal 2* $p = 0.000$, *Goal 3* $p = 0.000$). The *improved-APF* outperformed the *Teleoperation* in reaching *Goal 1* (Joystick $p = 0.000$, Myo $p = 0.000$), *Goal 2* (Joystick $p = 0.009$, Myo $p = 0.003$), and *Goal 3* (Joystick $p = 0.003$, Myo $p = 0.000$). Significantly better performance are obtained by *improved-APF* also with respect to *APF* overall for *Goal 1* (Joystick $p = 0.000$, Myo $p = 0.002$) and *Goal 2* (Joystick $p = 0.000$, Myo $p = 0.003$). We can infer that this metric is strictly correlated with the average execution time: a trial lasted longer when the user sent many commands. This also means that more robot assistance results in fewer total user inputs, possibly reducing the workload.
4) *Number of direction changes (see Fig. 4, right):* Regarding the number of direction changes, again a significant impact of the control method is found for all the goals with both the Joystick (*Goal 1* $p = 0.000$, *Goal 2* $p = 0.000$, *Goal 3* $p = 0.000$) and the Myo Armband (*Goal 1* $p = 0.000$, *Goal 2* $p = 0.000$, *Goal 3* $p = 0.000$). Data are very illustrative: when assisted, users modify the direction of the robot a fewer number of times, trusting the intervention of the robot assistance. Thus, more robot assistance results not only in fewer user inputs but also in fewer direction changes. For example, focusing on data obtained using the Myo interface when reaching *Goal 1*, users change the robot direction for 15 times in *Teleoperation*, while *improved-APF* needs not much more than five direction changes ($p = 0.000$). Indeed, during the tests, users were more cautious when in full teleoperation and imprinted small discrete commands to the robot for fear of damaging it and its surroundings. In shared control, instead, users' control was continuous as they knew they had collision avoidance support. However, even if shown to be better than *Teleoperation*, *APF* does not improve the performance significantly, while our framework does for all the goals and control methods. These results prove the goodness of *improved-APF:* users can aim straight at the goal without worrying about the obstacles in the environment. This outcome should make users less stressed when using our system.
5) *Average trajectory path:* Fig. 5 shows both the 2-D and 3-D average trajectories obtained with both the Joystick and the Myo Armband. When using *Teleoperation*, all users adopted the same strategy with both control interfaces: they preferred to take a very safe path away from obstacles. For example, to reach *Goal 1*, they usually raised up the robot at the beginning of the task and then moved toward the target. To reach *Goal 2*, they first moved far away from the occluding yellow cylinder, and then, they reached the destination. This may explain the higher completion time and the number of commands required. On the other hand, the keypoint that distinguishes our proposal with respect to *APF* is visible in the 3-D graphs: when reaching *Goal 1*, *improved-APF* performs a trajectory that is closer to the barrier as it correctly exploits the generated escape points, similarly to what is shown in Fig. 1. *APF* does not employ escape points, and it needs to stay farther away from the barrier because of the repulsive forces of the obstacles.

Given a static environment, it is worth mentioning that we could have forced the users to follow some predefined trajectories to reach each goal and see how the users follow these specific paths assisted by our proposal. Nevertheless, such a limitation would have limited one of the requirements for an ideal shared control system: the seamless interaction (see Section I). We want our users to be and feel in control as much as possible, leaving them the high-level decision of which path to take to reach the destination point. We believe that forcing the movement along a predefined trajectory would reduce this feeling. In addition, not imposing restrictions makes the system more suitable for applications in dynamic environments with movable obstacles.

B. Subjective Metrics

Upon completing all trials for one control method, users filled out the two questionnaires of Section IV-B. Fig. 6 shows obtained results. For both control interfaces, users perceive *APF* as the worst system, particularly when using the joystick. They do not feel in *Control* (versus *Teleoperation* $p = 0.008$) because the robot does not act as they *Want* (versus *Teleoperation* $p = 0.002$, versus *improved-APF* $p = 0.025$) and they are not able to reach the target *Quickly* (versus *improved-APF* $p = 0.005$). Indeed, potential fields push the robot away from obstacles, and users perceive this behavior as an unplanned motion. This outcome is confirmed by the analysis of the *Performance* (versus *improved-APF* $p = 0.033$) and *Frustration* (versus *improved-APF* $p = 0.018$): users feel on average three or four times more frustrated than with our proposal. Focusing again on *Control* and *Expectation*, *Teleoperation* obtained in general the best evaluation, since the user is fully controlling the robot. Nevertheless, our proposal is perceived similarly, with differences that are not statistically significant. Interestingly, some users commented that they preferred the *Teleoperation* at the beginning, by sending small discrete commands slowly conducing the robot away from obstacles, than relying on a system that they did not know. However, thanks to the training trials performed before the

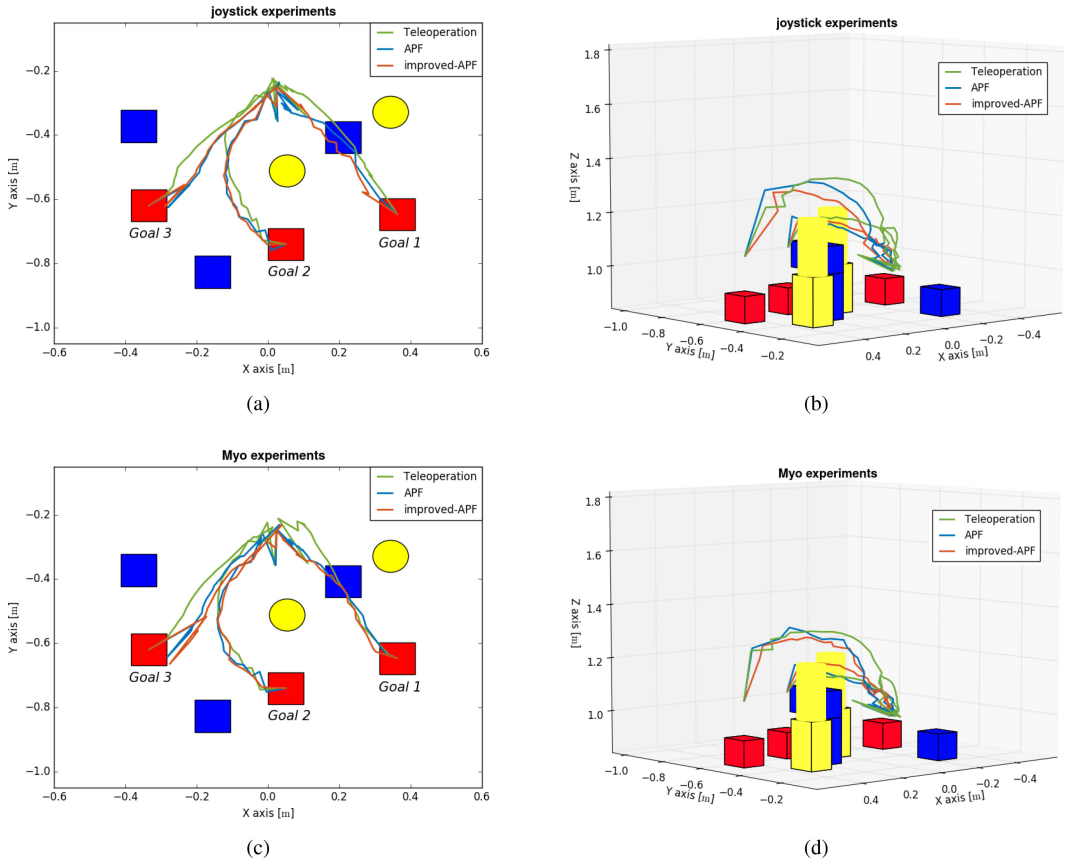


Fig. 5. Average 2-D and 3-D trajectories obtained with (a, b) Joystick and (c, d) Myo in the static setup.

experiment, users understood how shared control works and begin to trust the system, particularly in the more challenging control condition with the Myo. Their levels of *Effort* (Joystick, *Teleoperation* versus *improved-APF* $p = 0.022$) and *Frustration* (Myo, *Teleoperation* versus *improved-APF* $p = 0.047$) decrease accordingly. In terms of *Accuracy* and *Speed*, our proposal is perceived better than the *Teleoperation*, and significantly better than *APF*. Generally, users are more confident on the usage of the joystick, and this attitude makes them more confident also when performing the teleoperation tasks. On the other hand, the Myo Armband is less known. As a result, *improved-APF* is essential to speed up the task execution while maintaining a good level of accuracy. The NASA *Performance* outcome supports these results (Joystick, *APF* versus *improved-APF* $p = 0.033$). Nevertheless, *improved-APF* significantly reduces the workload even in a simple control condition, like when using the joystick, as highlighted by the reduced *Mental Demand* required to the users while using our framework (versus *Teleoperation* $p = 0.023$).

VI. Results Obtained in the Dynamic Setup

This section summarizes the outcomes obtained in the dynamic evaluation setup. Figs. 7 and 8 show the results of the objective and subjective metrics, respectively, obtained by the participants with the three control methods. To deal with the variability introduced by random changes of the scene in the dynamic setup, all the objective metrics for each subject are normalized by the sum of their values in all the successful trials.

In line with the results of the evaluation in the static environment, the proposed approach generally improved the task performance. A significant impact of the control method is found for the execution time ($p = 0.001$) and the number of direction changes ($p = 0.000$). In particular, *improved-APF* significantly helps the subjects decreasing the execution time with respect to *Teleoperation* ($p = 0.001$), while no significant reduction is found using the state-of-the-art *APF* approach ($p = 0.097$). When in pure teleoperation, almost all the participants tend to stop controlling the robot for fear of collisions, while the objects are moved by the operators, increasing the task completion time. In addition, they have to manually replan the robot trajectory to reach the desired target. On the other hand, when assisted by a shared control system, minimal adjustments of the robot motion are required since the collision avoidance is autonomously handled by the robotic system, significantly reducing the number of direction changes (*Teleoperation* versus *APF* $p = 0.000$, versus *improved-APF* $p = 0.000$) and the overall user's workload. However, due to the purely reactive characteristic of the conventional *APF*, it requires more user intervention to complete the task in case of complex objects configurations (e.g., obstacles placed nearby the target or close to the robotic gripper). On the other hand, the real-time generation and selection of the escape points allow the robot to modify in advance its trajectory to pass over the obstacles without any

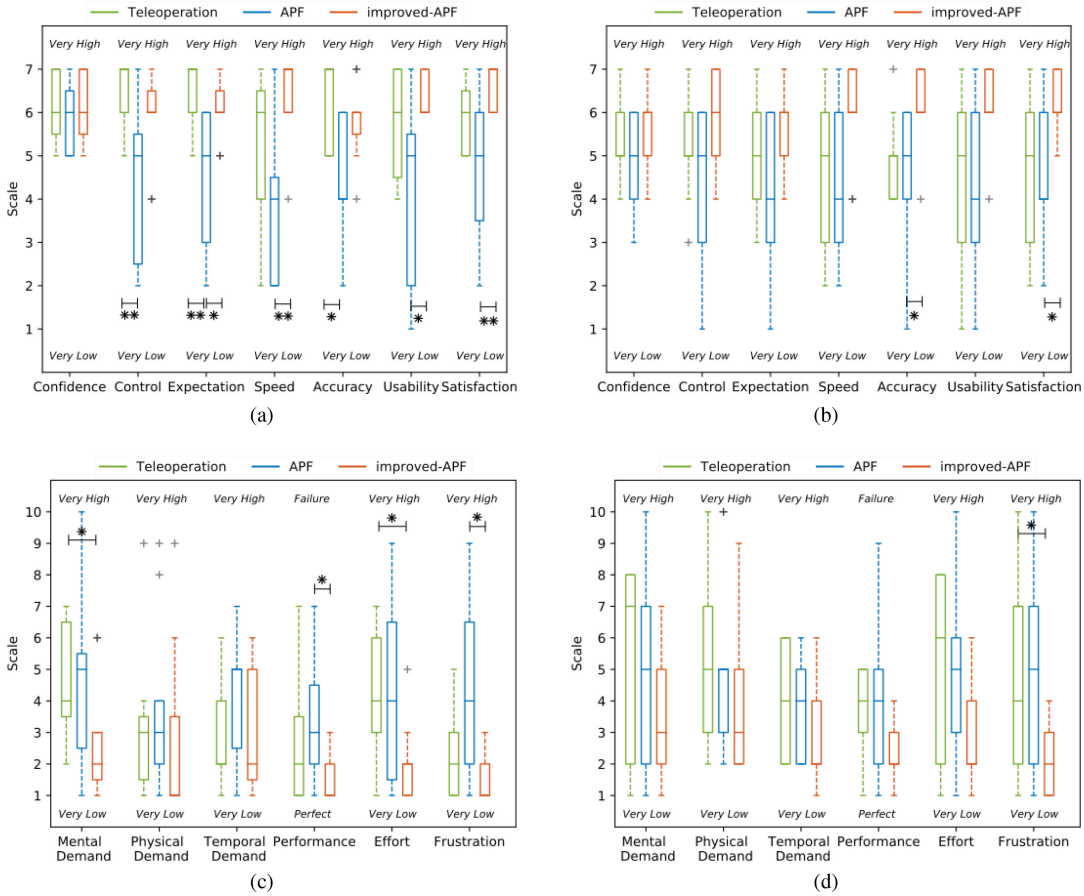


Fig. 6. Seven-point Likert scale and NASA TLX results survey for (a, b) Joystick and (c, d) Myo in the static setup. * indicates that $p < 0.05$; **, instead, is for $p < 0.01$.

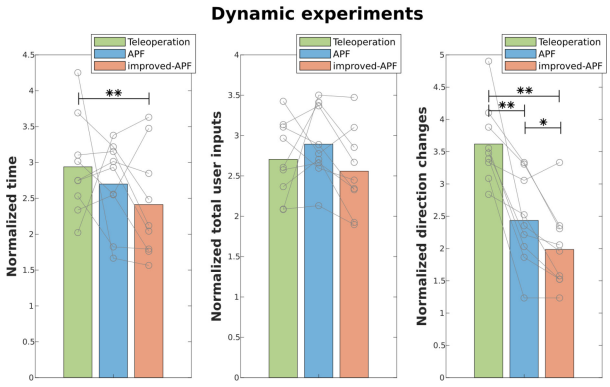


Fig. 7. Normalized average execution time, user inputs, and direction changes obtained in the dynamic setup. Gray circles represent the average results of each of the tested subjects. * indicates that $p < 0.05$; ** is for $p < 0.01$.

user intervention, significantly reducing the number of required direction changes (versus *APF* $p = 0.015$) and contributing to increasing users' *Satisfaction* (versus *APF* $p = 0.023$). These results are paralleled by the positive perception of the proposed *improved-APF* by the users: they report a significant reduction of both *mental demand* (versus *Teleoperation* $p = 0.006$) and *physical demand* (versus *Teleoperation* $p = 0.046$), which instead is not obtained with *APF*.

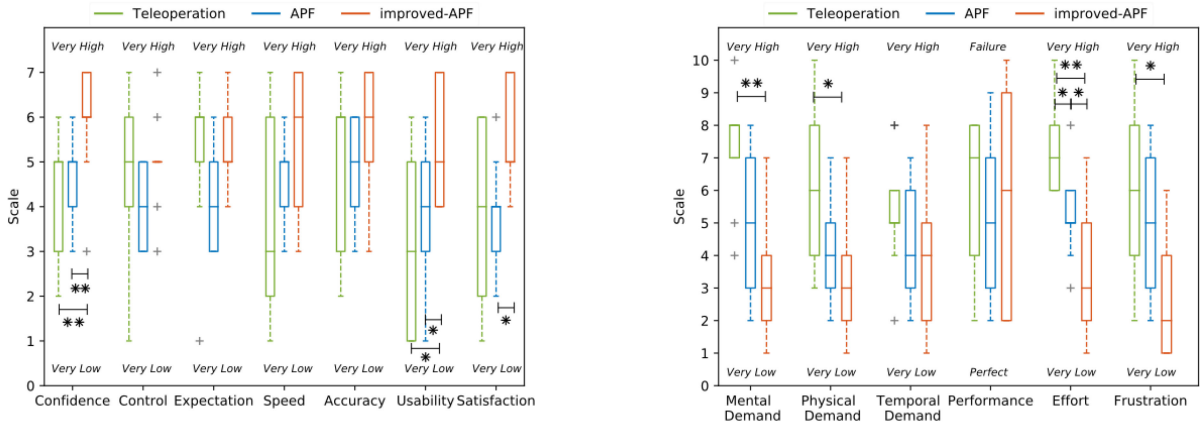
TABLE III
FAILURE RATE OBTAINED IN THE DYNAMIC SETUP

	Myo Armband (9 subjects)					
	Teleoperation		APF		Improved-APF	
	# failures	fail rate [%]	# failures	fail rate [%]	# failures	fail rate [%]
Total	47	34.8	33	24.4	22	16.3

The first column depicts the total failures occurred during the experiments, while the second column shows the corresponding failure rate.

As might be expected, the execution of the task in a dynamically changing environment represents a more challenging task with respect to the experiments in the static setup. This aspect is highlighted by the higher number of failures. As shown in Table III, the participants failed more than one-third of the trials when in pure teleoperation, with a failure rate that is double the one obtained in a static scenario. The introduction of repulsive forces from the objects helps reducing the failures due to obstacle collisions from 34.8% to 24.4%. Nevertheless, also, in these experiments, the *improved-APF* further reduced the failure rate to 16.3%, which may explain the significant reduction of the users' sense of *frustration* while using our system (versus *Teleoperation* $p = 0.019$), but not with the *APF* (versus *Teleoperation* $p = 0.424$).

In addition to the above results, *improved-APF* proves to be overall preferred by users: in a dynamic environment, users



(a) (b)

Fig. 8. (a) and (b) Seven-point Likert scale and NASA TLX results survey for Myo in the dynamic setup. * indicates that $p < 0.05$; **, instead, is for $p < 0.01$.

feel more *Confident* with our system. (versus *Teleoperation* $p = 0.007$, versus *APF* $p = 0.007$) and require a significantly lower *Effort* to complete the task (versus *Teleoperation* $p = 0.001$, versus *APF* $p = 0.007$). We believe that the advances proposed in this article significantly contribute to increasing the *Usability* (versus *Teleoperation* $p = 0.011$, versus *APF* $p = 0.021$) of our shared control framework with respect to both pure teleoperation and state-of-the-art approach.

VII. Conclusion

In this article, we proposed a framework for real-time shared control teleoperation that integrates the prediction of users' intention with a collision avoidance system. The latter is an improved version of APF that involves the dynamic generation of attractive points around the obstacles, named *escape points*. They help the robotic agent achieving the desired goal while preventing getting stuck on local minima. The best sequence of escape points is dynamically estimated during the teleoperation routine by solving a constrained problem that optimizes obstacle avoidance and target achievement. We demonstrated the goodness of our proposal when compared with a fully teleoperated approach and a classical APF. In a static setup, adopted objective metrics state that our proposal is faster and has a lower failure rate. Moreover, users need fewer input commands to reach intended targets as well as fewer changes of robot direction. Although users objectively performed better in all tests when using our system, a few of them felt little in control. In their opinion, the robot moved as they did not expect. This aspect can be explained by the intervention of potential fields pushing the robot away from obstacles. In these cases, the robot movements were sometimes perceived as incoherent with the user's command, even if they were following the optimal trajectory to safely reach the goal. Interestingly, similar outcomes were obtained by Javdani *et al.* [22]. In their work, subjects started preferring the shared control over direct teleoperation after they learnt and understood what the robot was doing to help them achieving the goal. Thus, we believe that more practice using our system and/or including a proper real-time feedback of the robot behavior would greatly enhance the system's usability and user's satisfaction. Finally, in a dynamic setup, the robotic agent proved to adapt to environment changes and faced unexpected interferences. Significant improvement of task performance, reduced users' effort, and higher satisfaction when using our collision avoidance system in a randomly changing scenario confirm that our approach is independent of the setup and can be easily exploited in unknown environments. As follow-up, we will investigate the performance of our proposal to handle collisions with continuously moving objects (e.g., agents in motion), which were not considered in the present experimental setup. Moreover, we will further explore uncertainties in the scene and human movements. In the first case, we will investigate online replanning techniques while acting in the belief space. Such methods are well known for partially observable task and motion problems, as proposed by Garrett *et al.* [43]. In the second case, we will exploit a human motion predictor [11].

In the current implementation and evaluation, we considered only 2-D and 3-D translations of the robotic device. Without loss of generality, the proposed method can be extended to include rotations, e.g., the orientation of the end-effector. If the user can control only the robot position, the APF will be computed in the configuration space of the manipulator, expanding (11) so that the forces are applied at the robot joints through both linear and angular Jacobian matrices [44]. If the user has control over the robot orientation, the cost function in (1) should also consider the minimization of the angular distance between the gripper orientation and the goal orientation (e.g., grasping pose). We will also let the robot choose the best grasping point while manipulating an object. To this aim, we will provide it with a knowledge base collecting datasets of grasping points as depicted in [45]. Finally, we will add further generality to our manipulation routine by considering the Riemannian motion policies [46]. Other future works will include tests with other control interfaces, such as BCI, and other robotic devices, such as mobile robots or powered wheelchairs. We aim to make our framework usable by any application in which a human has to control a robot to complete a task.

REFERENCES

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

[2] E. Tosello, S. Michieletto, A. Bisson, E. Pagello, and E. Menegatti, "A learning from demonstration framework for manipulation tasks," in *Proc. 41st Int. Symp. Robot.*, 2014, pp. 114–120.

[3] S. Jain, A. Farshchiansadegh, A. Broad, F. Abdollahi, F. Mussa-Ivaldi, and B. Argall, "Assistive robotic manipulation through shared autonomy and a body-machine interface," in *Proc. IEEE Int. Conf. Rehabil. Robot.*, 2015, pp. 526–531.

[4] A. Vallabhaneni, T. Wang, and B. He, *Brain–Computer Interface*. Boston, MA, USA: Springer, 2005, pp. 85–121.

[5] K. H. Goodrich, P. C. Schutte, F. O. Flemisch, and R. A. Williams, "Application of the H-mode, a design and interaction concept for highly automated vehicles, to aircraft," in *Proc. IEEE/AIAA 25TH Digit. Avionics Syst. Conf.*, 2006, pp. 1–13.

[6] G. Quere *et al.*, "Shared control templates for assistive robotics," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2020, pp. 1956–1962.

[7] E. Demeester, A. Hüntemann, D. Vanhooydonck, G. Vanacker, H. Van Brussel, and M. Nuttin, "User-adapted plan recognition and user-adapted shared control: A Bayesian approach to semi-autonomous wheelchair driving," *Auton. Robots*, vol. 24, no. 2, pp. 193–211, 2008.

[8] K. Muelling *et al.*, "Autonomy infused teleoperation with application to brain computer interface controlled manipulation," *Auton. Robots*, vol. 41, no. 6, pp. 1401–1422, 2017.

[9] D. P. Losey, C. G. McDonald, E. Battaglia, and M. K. O'Malley, "A review of intent detection, arbitration, and communication aspects of shared control for physical human-robot interaction," *Appl. Mech. Rev.*, vol. 70, no. 1, 2018, Art. no. 010804.

[10] E. You and K. Hauser, "Assisted teleoperation strategies for aggressively controlling a robot arm with 2D input," in *Proc. Robot.: Sci. Syst. Conf.*, 2012, pp. 354–361.

[11] S. Tortora, S. Michieletto, F. Stival, and E. Menegatti, "Fast human motion prediction for human-robot collaboration with wearable interface," in *Proc. IEEE Int. Conf. Cybern. Intell. Syst./IEEE Conf. Robot., Autom. Mechatronics*, 2019, pp. 457–462.

[12] S. Javdani, S. S. Srinivasa, and J. A. Bagnell, "Shared autonomy via hindsight optimization," in *Proc. Robot. Sci. Syst. Conf.*, 2015, pp. 1–24.

[13] G. Beraldo, M. Antonello, A. Cimolato, E. Menegatti, and L. Tonin, "Brain-computer interface meets ROS: A robotic approach to mentally drive telepresence robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2018, pp. 4459–4464.

[14] M. A. Goodrich and A. C. Schultz, *Human-Robot Interaction: A Survey*. Delft, The Netherlands: Now Publishers Inc., 2008.

[15] M. E. Bratman, "Shared cooperative activity," *Philos. Rev.*, vol. 101, no. 2, pp. 327–341, 1992.

[16] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Autonomous Robot Vehicles*. New York, NY, USA: Springer, 1986, pp. 396–404.

[17] S. Bistarelli, U. Montanari, and F. Rossi, "Semiring-based constraint satisfaction and optimization," *J. ACM*, vol. 44, no. 2, pp. 201–236, 1997.

[18] D. P. Losey, K. Srinivasan, A. Mandlekar, A. Garg, and D. Sadigh, "Controlling assistive robots with learned latent actions," in *Proc. Int. Conf. Robot. Autom.*, 2020, pp. 378–384.

[19] D. Aarno and D. Kragic, "Motion intention recognition in robot assisted applications," *Robot. Auton. Syst.*, vol. 56, no. 8, pp. 692–705, 2008.

[20] K. A. Tahboub, "Intelligent human-machine interaction based on dynamic Bayesian networks probabilistic intention recognition," *J. Intell. Robot. Syst.*, vol. 45, no. 1, pp. 31–52, 2006.

[21] A. D. Dragan and S. S. Srinivasa, "A policy-blending formalism for shared control," *Int. J. Robot. Res.*, vol. 32, no. 7, pp. 790–805, 2013.

[22] S. Javdani, H. Admoni, S. Pellegrinelli, S. S. Srinivasa, and J. A. Bagnell, "Shared autonomy via hindsight optimization for teleoperation and teaming," *Int. J. Robot. Res.*, vol. 37, no. 7, pp. 717–742, 2018.

[23] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning," in *Proc. AAAI Conf. Artif. Intell.*, 2008, vol. 8, pp. 1433–1438.

[24] W. Merkt, Y. Yang, T. Stouraitis, C. E. Mower, M. Fallon, and S. Vijayakumar, "Robust shared autonomy for mobile manipulation with continuous scene monitoring," in *Proc. 13th IEEE Conf. Autom. Sci. Eng.*, 2017, pp. 130–137.

[25] H. Reimann, I. Iossifidis, and G. Schöner, "Generating collision free reaching movements for redundant manipulators using dynamical systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2010, pp. 5372–5379.

[26] E. Rimon and D. E. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Trans. Robot. Autom.*, vol. 8, no. 5, pp. 501–518, Oct. 1992.

[27] R. Daily and D. M. Bevly, "Harmonic potential field path planning for high speed vehicles," in *Proc. Amer. Control Conf.*, 2008, pp. 4609–4614.

[28] J. Ren, K. A. McIsaac, and R. V. Patel, "Modified Newton's method applied to potential field-based navigation for mobile robots," *IEEE Trans. Robot.*, vol. 22, no. 2, pp. 384–391, Apr. 2006.

[29] R. Quirynen, K. Berntorp, and S. Di Cairano, "Embedded optimization algorithms for steering in autonomous vehicles based on nonlinear model predictive control," in *Proc. Annu. Amer. Control Conf.*, 2018, pp. 3251–3256.

[30] J.-C. Kim, D.-S. Pae, and M.-T. Lim, "Obstacle avoidance path planning based on output constrained model predictive control," *Int. J. Control, Autom. Syst.*, vol. 17, no. 11, pp. 2850–2861, 2019.

[31] W. Yu, R. Alqasemi, R. Dubey, and N. Pernalete, "Telemanipulation assistance based on motion intention recognition," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2005, pp. 1121–1126.

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

[33] A. Broad, T. D. Murphey, and B. D. Argall, "Learning models for shared control of human-machine systems with unknown dynamics," 2018, *arXiv:1808.08268*.

[34] S. Reddy, S. Levine, and A. D. Dragan, "Shared autonomy via deep reinforcement learning," 2018, *arXiv:1802.01744*.

[35] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Hoboken, NJ, USA: Wiley, 2014.

[36] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1238–1274, 2013.

[37] G. Li, A. Yamashita, H. Asama, and Y. Tamura, "An efficient improved artificial potential field based regression search method for robot path planning," in *Proc. IEEE Int. Conf. Mechatronics Autom.*, 2012, pp. 1227–1232.

[38] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2011, pp. 3400–3407.

[39] J. Vargas, C. Mariño, C. Aldas, L. Morales, and R. Toasa, "Kinect sensor performance for windows V2 through graphical processing," in *Proc. 10th Int. Conf. Mach. Learn. Comput.*, 2018, pp. 263–268.

[40] S. Tortora, M. Moro, and E. Menegatti, "Dual-myo real-time control of a humanoid arm for teleoperation," in *Proc. 14th ACM/IEEE Int. Conf. Human-Robot Interact.*, 2019, pp. 624–625.

[41] S. G. Hart and L. E. Staveland, "Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research," *Hum. Ment. Workload*, vol. 1, no. 3, pp. 139–183, 1988.

[42] R. D. McKendrick and E. Cherry, "A deeper look at the NASA TLX and where it falls short," in *Proc. Hum. Factors Ergonom. Soc. Annu. Meeting*, 2018, pp. 44–48.

[43] C. R. Garrett, C. Paxton, T. Lozano-Pérez, L. P. Kaelbling, and D. Fox, "Online replanning in belief space for partially observable task and motion problems," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2020, pp. 5678–5684.

[44] C. C. B. Viturino, U. de Melo Pinto Junior, A. G. S. Conceição, and L. Schnitman, "Adaptive artificial potential fields with orientation control applied to robotic manipulators," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 9924–9929, 2020.

[45] E. Tosello, Z. Fan, A. Castro, and E. Pagello, "Cloud-based task planning for smart robots," *Adv. Intell. Syst. Comput.*, vol. 531, pp. 285–300, 2017.

[46] N. D. Ratliff, J. Issac, and D. Kappler, "Riemannian motion policies," 2018, *arXiv:1801.02854*.