A Model Predictive Control Approach To Blending In Shared Control

Elio Jabbour, Margot Vulliez, Celestin Preault, Vincent Padois

A Model Predictive Control Approach To Blending In Shared Control

Elio Jabbour[1], Margot Vulliez[1], Celestin Preault[2], Vincent Padois[1]

***Abstract*— Shared control methods distribute control between human operators and robots in demanding tasks, enabling collaboration that leverages their respective strengths and expertise. Sharing the task typically involves blending algorithms that combine human control inputs to (pre)planed assistance trajectories. Conventional blending techniques, such as Linear Blending, compute a combined output but neither guarantee feasibility of this shared motion, nor ensure compliance with safety or task-related constraints.**

This paper proposes to tackle feasibility and safety by formulating the blending strategy as the solution of a constrained optimal control problem, that enforces environment limits, task requirements, and physical capabilities. A Model Predictive Control approach is used to solve the optimization problem and anticipate constraints by predicting the robot motion over a receding time horizon. We evaluate this approach in simulated and real-world pick-and-place teleoperation experiments. The experimental study compares the Model Predictive Control approach to Linear Blending and full Teleoperation. The results show that the new framework offers significant improvements, as it provides a safer, more accurate, and repeatable response.

I. Introduction

Autonomous robots, although highly effective in speed and precision, often fall short in handling complex and dynamic scenarios [1]. In these contexts, teleoperation provides a mean for human experts to perform tasks remotely in dangerous or inaccessible environments, such as underwater [2] or in nuclear decommissioning [3]. Despite its advantages, teleoperation can be complex and lead to cognitive overload, as the operator must continuously process and respond to various sensory inputs - visual, haptic, and auditory [4], [5]. Thus, improving teleoperation requires a teaming approach that leverages the strengths of both the human and robot.

Shared control frameworks [6] aim at addressing this challenge by dynamically allocating the control authority between the human and the robot, based on the task and environment, so that they collaborate. Sheridan [7] first conceptualized this dynamic allocation of authority as a ten-level system, where control shifts between full human operation to full automation.

Combining control inputs from human operators with those generated by autonomous systems typically involves blending algorithms [8]. Blending methods range from applying weighted combinations of the inputs from each source [9], [10] to probabilistic distributions [11]. Another commonly used method is to model shared control as a random process described by a joint distribution over the operator, the autonomous system, and the environment [12]. Similar to the game-theoretic approach in [13], where both agents act independently with their own objectives and strategies, this blending method dynamically adapts control inputs based on ongoing interactions, without prior modeling or training.

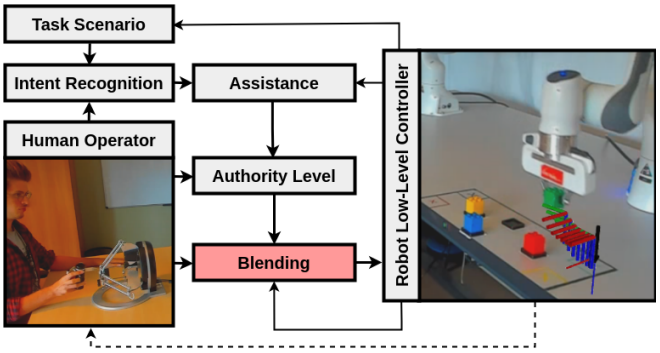


Fig. 1: Overview of a general shared control block diagram and experimental setup. The diagram represents the interaction between the human input, intent recognition, and assistance input modules with control authority. The control input is blended and sent to the robot controller. Here, the operator uses a Omega 7 haptic device to remotely control a Panda Franka 7dof manipulator. Control is shared between the human and the assistance to complete a pick-and-place task. The blending effect is visualized through the horizon of intermediate Cartesian poses. It reflects the predictive nature of the MPC-B approach (described in section III-B), continuously guiding the robot toward the target.

One of the main challenges in blending human and autonomous control signals is to ensure the feasibility of the combined output. While each input may individually respect constraints related to the environment and robot capabilities, there is no guarantee that their combination will. Existing blending methods lack of a generic solution to this issue. Although robot control safety mechanisms, such as saturation strategies, can address this, handling constraints upfront is likely to improve system performance. This highlights the need for blending approaches formulated as constrained optimization problems. These approaches are becoming increasingly standard in robotics control [14], [15], especially when they are formulated over an horizon such as in Model Predictive Control (MPC) [16], [17]. MPC can indeed solve the optimal control problem over a receding horizon. In addition to enforcing constraints and optimizing performance, MPC has filtering properties that smooth out abrupt variations that may be caused by sudden authority shifts, errors in the human intent detection, or operator tremors. A shared steering control system for automated driving lane-keeping has demonstrated that dynamic constraints and stage

This work was supported by the ANR programm ASAP-HRC (ANR-21-CE10-0001).

[1]Elio Jabbour, Margot Vulliez, and Vincent Padois are with Inria, Auctus team, Bordeaux, France. firstname.name@inria.fr

[2]Celestin Preault is with CESI Lineact, France cpreault@cesi.fr

costs can ensure smooth control allocation [18]. Another recent MPC-based haptic shared steering controller has shown the benefit of the prediction model in optimal control by accounting for the cognitive behavior and neuromuscular dynamics of the driver and the vehicle-steering dynamics [19].

This work investigates the formulation of the blending problem using a MPC approach. To demonstrate its effectiveness, the proposed method is compared to standard teleoperation and linear blending shared control in a remote pick-and-place scenario. Through experiments and simulations, this proof-of-concept shows that the MPC-based approach outperforms traditional methods in terms of safety, feasibility, repeatability, and human effort reduction.

The paper is organized as follows: an overview of shared control architecture is presented in section II, the formulation of the proposed MPC-based blending controller is introduced in section III, the experimental set-up for our evaluation is described in section IV, the results and their analysis are then given in section V. Conclusions and future works directions end the paper in section VI.

II. Shared Control architecture

Shared control in teleoperation consists of several key components as illustrated in figure 1: human input, intent recognition, authority level distribution, and blending controller. These components are briefly described in this section. Their specific implementation for the experiments is described in section IV-C. This section also describes performance metrics generally considered to qualify a shared controller.

A. Main components

In teleoperation, the user interacts with a joystick which captures the human position ${}^{\mathbf{H}}\mathbf{x}_h \in \mathbb{R}^3$ for the robot to track, which is measured in the device base frame $\{\mathrm{H}\}$. The transformation of the user input position to the robot frame is given by:

$$ {}^{R}\mathbf{x}_h = s\mathbf{R}_{RH}({}^{H}\mathbf{x}_h - {}^{H}\mathbf{x}_{hi}) + {}^{R}\mathbf{x}_{ri} \tag{1}$$

Here, H and R denote the teleoperation device and robot frames, respectively. ${}^{H}\mathbf{x}_{hi}$ and ${}^{R}\mathbf{x}_{ri} \in \mathbb{R}^3$ are the positions of the device and robot workspace centers. The scalar $s \in \mathbb{R}^+$ is a scaling factor, and $\mathbf{R}_{RH} \in \mathbb{R}^{3\times3}$ is the rotation matrix expressing the orientation of the haptic frame in the robot frame. The same transformation can be applied to the human velocity ${}^{\mathbf{H}}\dot{\mathbf{x}}_h \in \mathbb{R}^3$. The position and velocity are the human inputs used in the shared controller. The following control models are all expressed in robot space. For reasons of readability, we do not indicate the R-frame upper index in the equations for the rest of the section.

The authority level, defined as a scalar $\beta \in [0, 1]$, allocates the control between the operator ($\beta = 1$) and the assistance system ($\beta = 0$). It can be modulated by specific factors such as the operator fatigue [20] or the robot fidelity [21]. The expertise level of the operator can be learned, to provide less assistance as expertise grows [22]. Trust in the system's performance, especially in safety-critical tasks, can also be used to determine the authority distribution [23]. Task proximity is another metric that can be used to shift the authority when precision is demanded [24].

The assistance trajectory $[\mathbf{x}_a, \dot{\mathbf{x}}_a] \in \mathbb{R}^3$ is often learned by demonstrations [25] or preplaned [26]. It reflects the motion the robot would do if performing the task fully autonomously.

Detecting the human's intention to align the assistance with the operator's goals is crucial, to avoid conflicting behaviors. Among others, techniques such as neural networks or Hidden Markov Models (HMMs) can be used to predict human intention from their motion [27], [28]. A review on intent detection in shared control is given in [29].

The objective in the subsequent sections is to compute the blending position and velocity, denoted as $[\mathbf{x}_b, \dot{\mathbf{x}}_b]$, using the function $f(\mathbf{x}_a, \mathbf{x}_h, \dot{\mathbf{x}}_a, \dot{\mathbf{x}}_h, \beta)$. This combined output will then be used as the input to the robot.

B. Evaluation metrics

The impact of a shared control system can be evaluated using various objective metrics. Common ones, like **execution time** and **Cartesian position error**, though not specific to shared control, remain relevant [30]. However, shared control-specific performance metrics offer deeper insights. The **level of assistance**, sometimes called *interference*, quantifies how much the assistive system intervenes, indirectly indicating the user's expertise [31]. Another key metric is the **agreement**, measuring the alignment between human input and system autonomy, providing a clearer sense of assistance quality [32].

Human comfort metrics include **motion (dis)continuity**, evaluated through jerk, as motion discontinuity can undermine perceived safety [33]. **Human effort**, reflecting the physical demand, can be measured by the power exerted on the teleoperation device, with shared control systems often reducing this burden [34].

Lastly, system safety is assessed by monitoring **violations** of workspace boundaries, control limits, or task-specific constraints.

III. Blending Strategies

This section describes the MPC-based blending strategy proposed in this work with a prior description of the standard linear blending approach.

A. Linear Blending (LB)

In the literature, a conventional approach to blend both human operator and autonomous agents control inputs is linear blending, here illustrated at the position level:

$$\mathbf{x}_b = \beta\mathbf{x}_h + (1 - \beta)\mathbf{x}_a \tag{2}$$

Linear blending presents significant safety challenges, particularly when the human and assistance inputs diverge while they have a similar authority weight, as highlighted by [9]. In such scenarios, LB is unable to effectively resolve conflicts, leading to meaningless median trajectory that can be unsafe, which is especially concerning in critical environments. Indeed, LB is not designed to consider a priori constraints on the task or robot workspace, or physical capabilities.

Moreover, LB lacks a built-in mechanism to ensure smooth transitions between control inputs. This is particularly problematic when the authority weighting is poorly tuned or when the human operator's movements are erratic, as can be the case with less experienced users.

B. Predictive Blending (MPC-B)

These limitations can be addressed by employing control approaches that explicitly account for system constraints. One such method is Model Predictive Control (MPC), which solves an optimal control problem over a receding horizon while adhering to predefined constraints. By utilizing a model of the system's dynamics, MPC is able to predict the effects of control actions over multiple future time steps, enabling the computation of a locally optimal control strategy at each step. This predictive capability allows MPC to make informed and constraint-compliant decisions that improve system performance and safety.

Let $\mathbf{X}_b = \{\mathbf{x_b}, \dot{\mathbf{x}}_\mathbf{b}\} \in \mathbb{R}^6$ be the desired robot end-effector state resulting from blending. And let the current state $\mathbf{X}_r = \{\mathbf{x}_r, \dot{\mathbf{x}}_r\} \in \mathbb{R}^6$ be the current position and velocity of the robot[1].

Extending equation (2), blending can be formulated as a minimization problem over an horizon with two, potentially conflicting, objectives related to the human operator input $\mathbf{X}_h$ and autonomous agent input $\mathbf{X}_a$. Considering a control variable $\mathbf{u}$ expressed at the acceleration level, the MPC cost function can be formulated as:

$$J = \sum_{k=1}^{H} [\|(\mathbf{X}_{b_k} - \mathbf{X}_h)\|_{\mathbf{Q}_h} + \|(\mathbf{X}_{b_k} - \mathbf{X}_a)\|_{\mathbf{Q}_a} + \|\mathbf{u}_k\|_{\mathbf{R}} \quad (3)$$

where $\mathbf{Q}_h$ and $\mathbf{Q}_a \in \mathbb{R}^{6\times6}$ are symmetric positive-definite diagonal matrices defined as

$$\mathbf{Q}_h = \begin{bmatrix} \beta\mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & k_{vh}\beta\mathbf{I}_{3\times3} \end{bmatrix}$$

and

$$\mathbf{Q}_a = \begin{bmatrix} (1-\beta)\mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & k_{va}(1-\beta)\mathbf{I}_{3\times3} \end{bmatrix}$$

These weights reflect the distribution of the control authority as well as the priority given to tracking the reference velocity with respect to the reference position through the k_{vh} and k_{va} weighting factors. A higher value for these parameters prioritizes faster responses to trajectory changes, enabling the system to quickly adapt to deviations in either the human or assistance velocities. However, increasing these weights may also lead to higher tracking errors at the position level, as the system could become overly sensitive to minor fluctuations, potentially compromising stability and precision in favor of responsiveness. $\mathbf{R} = diag(\gamma) \in \mathbb{R}^{3\times3}$ with $\gamma \geq 0$ is a symmetric positive definite diagonal matrix weighting the cost associated to the control input $\mathbf{u}$.

The relationship between the current state and the control input at time step k, is described by the discrete-time linear system:

$$\mathbf{X_b}_{k+1} = \mathbf{A}\mathbf{X_b}_k + \mathbf{B}\mathbf{u}_k \quad (4)$$

where $\mathbf{A}$ and $\mathbf{B}$ are the state space and input matrices.

For a receding time horizon of H steps and a time step δt, the total horizon time is $T = H\delta t$. Defining state and input vectors over the horizon, with $k = 0, 1, ..., H$, we have:

$$\bar{\mathbf{U}} = \begin{bmatrix} \mathbf{u_k} \\ \mathbf{u_{k+1}} \\ \cdots \\ \mathbf{u_{k+H-1}} \end{bmatrix} \in \mathbb{R}^{3\times(H-1)} \quad \bar{\mathbf{X}} = \begin{bmatrix} \mathbf{X_k} \\ \mathbf{X_{k+1}} \\ \cdots \\ \mathbf{X_{k+H}} \end{bmatrix} \in \mathbb{R}^{6\times H}$$

Propagating equation (4) over the horizon yields:

$$\bar{\mathbf{X}} = \bar{\mathbf{A}}\mathbf{X_0} + \bar{\mathbf{B}}\bar{\mathbf{U}} \quad (5)$$

where $\bar{\mathbf{A}}$ and $\bar{\mathbf{B}}$ are the propagated state and input matrices.

Reformulating equation (3) using equation (5), we obtain the state vector $\bar{\mathbf{X}}_\mathbf{b}$ which expresses the target state over the horizon. Constraints from the environment (workspace limits), robot capabilities (Cartesian velocity and acceleration), human motion (motor capabilities in terms of Cartesian velocity and acceleration) can be thus considered and written over the horizon under the generic form:

$$\bar{\mathbf{X}}_{min} \leq \bar{\mathbf{X}}_b \leq \bar{\mathbf{X}}_{max} \quad \bar{\mathbf{U}}_{min} \leq \bar{\mathbf{U}} \leq \bar{\mathbf{U}}_{max} \quad (6)$$

This leads to the constrained quadratic program:

$$\bar{\mathbf{U}}^\star = \underset{\bar{\mathbf{U}}}{\operatorname{argmin}} \frac{1}{2}\bar{\mathbf{U}}^T\mathbf{H}\bar{\mathbf{U}} + \mathbf{F}^T\bar{\mathbf{U}} \quad (7)$$

subject to equation (6). The first term of the optimal receding control horizon $\bar{\mathbf{U}}^\star$ is the blended acceleration command to which a blended velocity and position $[\mathbf{x}_b, \dot{\mathbf{x}}_b]$ is associated through equation (4).

To ensure proper convergence towards the target position, the terminal cost, *i.e.* the cost of the last step of the horizon in the cost function, can be modulated by tuning k_{vh} and k_{va} differently for that terminal step. The same applies to the action cost which can be modulated by tuning γ differently for the terminal step of the horizon.

IV. Experimental set-up

To objectively assess the performance of the proposed blending approach, we propose a series of simulations and a real-world shared-controlled teleoperation experiment where Teleoperation, Linear Blending (LB) and MPC-based blending (MPC-B) are compared. The evaluation task is a pick-and-place scenario, in the teleoperation of a Franka Emika Panda robot.

A. Simulation

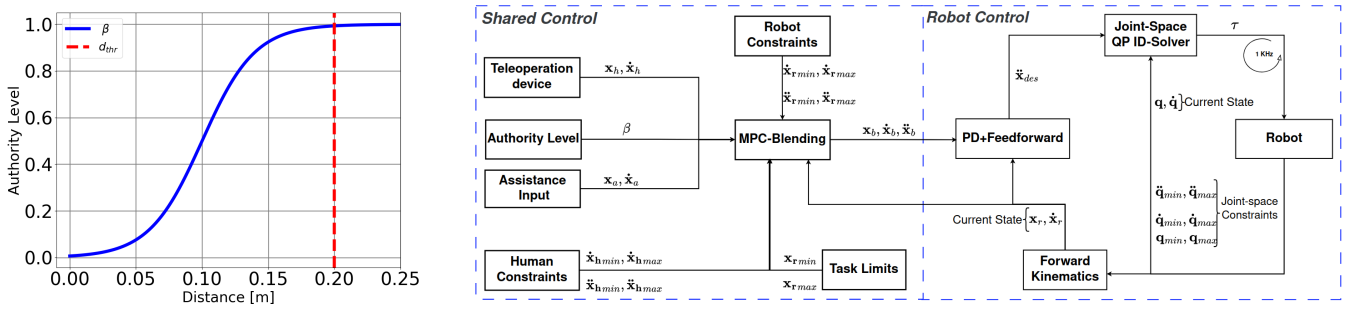
The simulation scenario is divided into three distinct phases to compare MPC-B and LB. To ensure repeatability and eliminate variability of human teleoperation, the human input $\mathbf{x}_h$ is automatically generated as an oscillating position given by:

$$\mathbf{x}_h = \mathbf{x}_r + K(\mathbf{x}_t - \mathbf{x}_r) + c\sin(2\pi ft + P) \quad (8)$$

$\mathbf{x}_t$ is the target position, K is a proportional gain, c is the amplitude, f is the frequency, and P is the phase shift of the sinusoidal component. The human velocity $\dot{\mathbf{x}}_h$ is derived from equation (8). The assistance input $\mathbf{x}_a$ is predefined as a step response to the target position. The simulation is constrained to motion along a single axis for simplicity.

Phase 1: The robot starts in full assistance mode ($\beta = 0$). $\mathbf{x}_a$ is instantaneously set to the target position. This step tests the controller's ability to accurately track sudden changes in the target position.

[1]Without loss of generality and considering the pick-and-place scenario retained for this work, the controller derivation is formulated at the translation level. Extension to SE(3) is possible [35] but out of scope for this paper.



(a) (b)

Fig. 2: (a) The authority level β shifts from full assistance $\beta = 0$ to full human control $\beta = 1$ with respect to the distance to the target $||\mathbf{x}_t - \mathbf{x}_r||$. (b) Control architecture for MPC-based blending.

Phase 2: The authority level is shifted to full human control ($\beta = 1$). $\mathbf{x}_h$ goes toward a new target position following equation (8) with the oscillation parameters $c = 10.0, f = 1.2, P = 1.6$. This phase evaluates the controllers' ability to cope with fluctuations in the human input and a discontinuous change in the authority level.

Phase 3: The authority is equally distributed ($\beta = 0.5$). $\mathbf{x}_a$ is set to a new target position, that lies outside the defined workspace. $\mathbf{x}_h$ goes toward this target position without oscillation ($c = 0.0$). This phase tests the controller's capability to predict and prevent violations of workspace boundaries while maintaining smooth control response.

B. Real-World Teleoperation Scenario

The pick-and-place scenario is performed on a real robot, where Lego pieces must be moved from some fixed areas to delimited zones, as shown in figure 1. The workspace has predefined safety limits, from which the placement zones are close to test the controller's ability to handle constraints.

In this experimental study, a pure teleoperation mode serves as a baseline for comparison to LB and MPC-B shared control modalities. The human operator provides its motion input through a Force Dimension Omega.7 haptic device.

The user must pick-and-place four different-colored objects in a specific order. The displacement phases, toward the object to pick or the area where to drop it, are controlled by the human, either in full teleoperation or assisted through shared control. When at the correct pick/place position, with a maximum error of $e = 0.02m$, the user presses the device switch and the robot maintains its current position while grabbing/releasing the object. These grab/release steps are completed after a given time delay $t = 1.5s$ and, then, an automatic vertical motion of the robot and the haptic device is performed. This timed strategy is designed to avoid collisions in the critical release and grab phases.

This task is performed 6 times for each control mode. The individual who conducted the experiment is an expert in teleoperation with a background in robotics, but is not a contributor to this work.

C. Implementation

The detailed shared control architecture is given on figure 2b. It implements the MPC-B controller that combines the human and assistance inputs, while considering several constraints related to the human, robot, and task.

For this comparison study, and to not rely on the accuracy of an intent detection, the assistance reference is successively set to the ordered four target objects' positions.

In this experiment, the control authority distribution is based on the distance to the target, which is a well-established method from the literature [36]. As shown in figure 2a and equation (9), it computes β from a distance function $f(x)$ where $\mathbf{x}_t$ is the target position. $d_{thr} = 0.2m$ is the distance threshold at which the human has full remote control. $s = 0.6$ is a scalar that shifts the authority curve to be equal to zero when the distance to the target is zero. $z = 0.1$ is a parameters that tunes the slope of the authority curve.

$$\beta = \frac{1}{1 + e^{-\frac{f(x)-s}{z}}} \text{ and } f(x) = \frac{||\mathbf{x}_t - \mathbf{x}_r||}{d_{thr}} \tag{9}$$

The MPC-B controller is implemented according to the equations written in section III-B. The MPC-B parameters for this study are given in table I and the position limits enforced for the MPC can be seen in figure 4. A systematic study of the influence of the main MPC parameters was initially performed in simulation using the NSGA-3 multi-objective genetic algorithm implemented in pymoo[2]. All parameters were then fine-tuned empirically using the real robot. The tuning process aimed to balance controller responsiveness with motion accuracy. Likewise, the step length and number of steps in the MPC were selected to ensure that the associated QP could be solved at 1 kHz, striking an optimal balance between computational efficiency and control performance.

The MPC-B control input $\mathbf{u}$ is computed at the time step $\delta t = 40$ms, while the robot's control loop operates at 1ms. Interpolation is applied to the calculated acceleration input to get the desired trajectory at the faster control rate. This is achieved through the control-to-state linear equation (4), with matrices $\mathbf{A}$ and $\mathbf{B}$ set to the control step time.

The robot local controller implemented in our experiments is depicted in figure 2b. The control input to the low-level task-space inverse dynamics of the robot control architecture is expressed at the acceleration level. Thus the output acceleration

[2]`https://pymoo.org/`

TABLE I: Control Parameters for the System

Parameter	Value	Parameter	Value
$K_p/(s^{-2})$	250	$K_d/(s^{-1})$	30
H (–)	10	δ/ms	40
K_{va} (–)	1×10^{-2}	K_{vh} (–)	1×10^{-3}
K_{va} (Terminal Cost) (–)	1000	K_{vh} (Terminal Cost) (–)	1000
γ (–)	1×10^{-5}	γ (Terminal Cost) (–)	1×10^{-9}
$\{\dot{x}_{bmin}, \dot{x}_{bmax}\}/(m/s)$	$\{-0.3, 0.3\}$	$\{\ddot{x}_{bmin}, \ddot{x}_{bmax}\}/(m/s^2)$	$\{-7.0, 7.0\}$

$\mathbf{u} = \ddot{\mathbf{x}}_b$ of the MPC-B controller is fed to a PD + feedforward local controller, together with the corresponding desired position $\mathbf{x}_{des}$ and velocity $\dot{\mathbf{x}}_{des}$, computed from the MPC-B acceleration output with equation equation (5):

$$\ddot{\mathbf{x}}_{\mathbf{des}} = K_p(\mathbf{x}_{des} - \mathbf{x}_r) + K_d(\dot{\mathbf{x}}_{des} - \dot{\mathbf{x}}_r) + \mathbf{u} \qquad (10)$$

with K_p and K_d the proportional and damping gain matrices. This allows to close the loop at the Cartesian level and reject, to some extent, modelling errors to ensure correct tracking of the desired trajectory.

Unlike MPC, the teleoperation and linear blending modes cannot compute a reference acceleration. Therefore, in these two cases, the local robot controller is restricted to a PD without feedforward.

Task-space inverse dynamics is implemented as a constrained Quadratric Program [37], [38] and computes joint torques references $\boldsymbol{\tau} \in \mathbb{R}^{7\times 1}$ at 1 kHz, given the desired Cartesian acceleration $\ddot{\mathbf{x}}_{\mathbf{des}}$. Constraints provide both safety guarantees on the joint position, velocity and torque limits of the robot, as well as guarantees on the Cartesian level velocities and acceleration (generally prescribed by the task and user requirements as safety measures).

All the control parameters for this implementation are summarized in table I for replicability.

D. Evaluation Metrics in practice

The evaluation metrics described in section II-B are evaluated through different physical quantities described in table II. In this table, t_f and t_0 are the ending and starting time of the experiments respectively. N is the number of time samples over the duration of an experiment. n_t is the number of place targets in the experiment, $\mathbf{x}_{t_i}$ the position of each target and $\mathbf{x}_{r,N_i}$ the position reached by the operator when releasing the part at target i. $\mathbf{F}_k$ is the force exerted by the human on the haptic device (here estimated through the dynamics model of the device). A constraints violation metric is computed independently for position, velocity and acceleration and $t_{p,v,a\notin\mathcal{F}}$ is the time during which a constraint was violated (with $\mathcal{F}$ the corresponding feasible interval for the considered quantity).

V. Results

This section provides both the results and their analysis for the simulation and experimental study.

A. Simulation Results

Figure 3 presents a comparison of the trajectories resulting from MPC-based Blending (MPC-B) and Linear Blending (LB), across the three phases of the simulation study (see section IV-A).

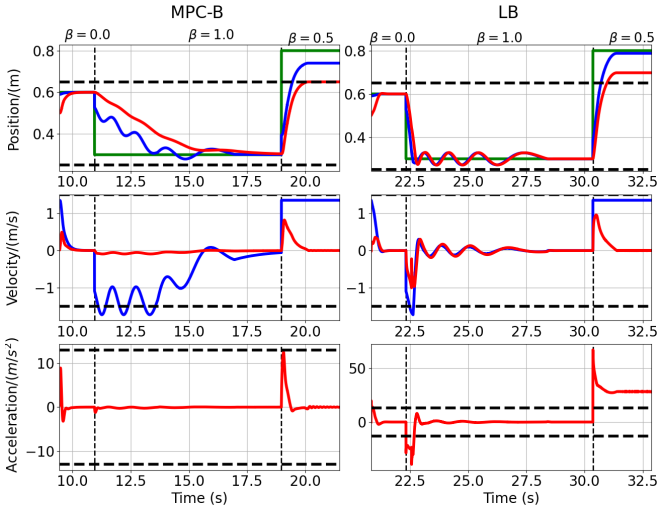


Fig. 3: Comparison of the simulated trajectories for the MPC-B (left) and LB (right), across the three phases ($\beta = 0$, $\beta = 1$, $\beta = 0.5$). The top, middle, and bottom rows respectively show the position, velocity, and acceleration responses. The red curves are the MPC-B and LB trajectories, the blue curves are the human input motion, and the green curves are the assistance input. Horizontal black dashed lines show the pose, velocity, and acceleration limits. Overall, MPC-B maintains smoother responses and respects constraints in all conditions, whereas LB results in high acceleration spikes above the limit and does not filter the human input oscillations.

In Phase 1, where the authority is to full assistance, MPC-B outputs a smooth motion toward the assistance step input, while LB exhibits abrupt transitions and violates the acceleration constraint. It shows the MPC-B ability to predict and locally plan a motion that respect given physical limits. Phase 2 simulates the controllers' responses when tracking an oscillating human input trajectory without assistance. By enforcing dynamic constraints on the motion, MPC-B filters the input fluctuations, while LB results in an oscillating robot motion that, again, violates the acceleration constraint. In Phase 3, the assistance and human inputs together try to reach a position outside the workspace. MPC-B tracks the inputs until it predicts the proximity of the position limit and smoothly decelerates to stop before it, whereas LB continues its movement and breaches both position and acceleration constraints.

This simulation study demonstrates the MPC-B ability to

TABLE II: Specific formulation retained for the evaluation metrics

Metric / unit	Symbol	Formula
Execution time $/s$	T	$t_f - t_0$
Position error $/mm$	P	$\frac{1}{n_t}\sum_{i=1}^{n_t} \|\|\mathbf{x}_{t_i} - \mathbf{x}_{r,N_i}\|\|$
Motion continuity $/(m/s^3)$	M	$\frac{1}{N}\sum_{k=0}^{N} \|\dddot{\mathbf{x}}_{r,k}\|$
Human effort $/W$	E	$\frac{1}{N}\sum_{k=0}^{N} \mathbf{F}_k^T \dot{\mathbf{x}}_{h,k}$
Agreement $/(m^2/s^2)$	A	$\frac{1}{N}\sum_{k=0}^{N} \dot{\mathbf{x}}_{a,k}^T \dot{\mathbf{x}}_{h,k}$
Level of Assistance %	L	$\frac{100}{N}\sum_{k=0}^{N} 1 - \beta_k$
Constraints Violation %	$V_{p,v,a}$	$100\frac{t_{p,v,a\notin\mathcal{F}}}{t_f}$

TABLE III: Evaluation metrics for the three shared control modalities

Control Modality	T/s	P/mm	$A/(m^2/s^2)$	E/W	$M/(m/s^3)$	$L\%$	$V_p\%$	$V_v\%$	$V_a\%$
Teleoperation	45.02 ± 0.85	15.2 ± 6.2	0.1508 ± 0.0128	3.773 ± 0.699	212.767 ± 15.817	Not applicable	16.58 ± 3.96	16.72 ± 2.71	15.85 ± 4.10
Linear Blending	37.20 ± 0.88	8.2 ± 0.8	0.2606 ± 0.0444	1.586 ± 0.465	306.425 ± 39.249	40.92 ± 1.76	12.53 ± 3.45	36.76 ± 3.59	31.54 ± 6.08
MPC-B	$\mathbf{35.97 \pm 0.65}$	$\mathbf{3.8 \pm 2.4}$	$\mathbf{0.3056 \pm 0.0248}$	$\mathbf{0.990 \pm 0.136}$	$\mathbf{181.926 \pm 7.340}$	$\mathbf{45.95 \pm 0.84}$	$\mathbf{1.25 \pm 1.07}$	$\mathbf{4.29 \pm 2.19}$	$\mathbf{0.00 \pm 0.00}$

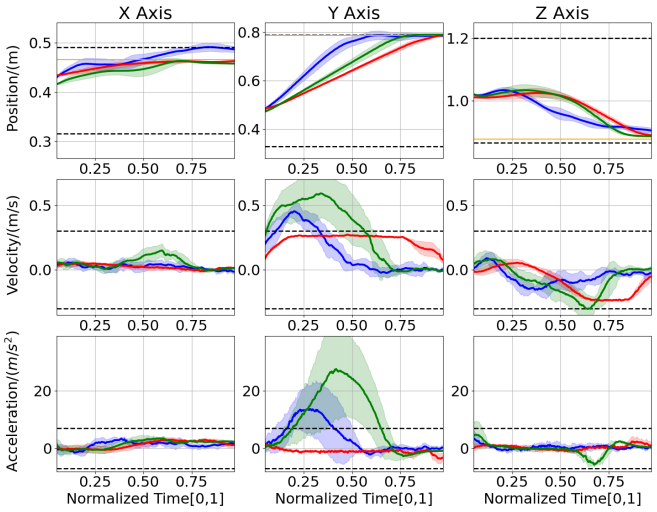


Fig. 4: This figure presents a comparative analysis of three control modalities in the case of one of the four placing motions of the retained scenario: Teleoperation (blue), LB (green) and MPC-B (red). The plots depict the real robot's position and the desired robot's velocity, and acceleration fed to the low level robot controller in the X, Y, and Z directions over a normalized time. Shaded regions around each line show the standard deviation across 6 trials. Dashed black lines indicate the respective constraints on position, velocity, and acceleration. The yellow line represent the center of the target zone where the brick should be placed.

enforce position, velocity, and acceleration constraints by modeling and planing the robot motion in a near future. The MPC-B trajectory is, therefore, smoother than the one produced by LB, even in the presence of sudden changes or oscillations in the input human-assistance motion.

B. User Experiment Results

The results for the user experiment are exemplified in figure 4 which exhibits the average robot's end-effector trajectory for one placing motion among the four induced by the scenario. This graph provides an interesting exemplification of the repeatability of the produced motion as well as of the constrains violations of all three modes. These observations are generalized by table III which gathers the evaluation metrics for all three modes over the whole scenario. In these results, the real robot position $\mathbf{x}_r$ is used while the velocity and acceleration signal are respectively the desired velocity $\dot{\mathbf{x}}_{des}$ (coming from the device for teleoperation and blended for the other two modes) and the acceleration signal $\ddot{\mathbf{x}}_{des}$ sent to the low-level controller. The low-level controller saturates the velocity and the acceleration to the prescribed constraints in a strictly reactive way so that the robot only performs feasible motions. Accounting for these constraints at the lowest level only is sub-optimal and it is thus interesting to show the unconstrained signals coming from the teleoperation and linear blending modes.

Constraints violation and motion properties: As expected, the consideration of constraints *a priori* in MPC-B leads to minor violations ($\leq 5\%$ for $V_{p,v,a}$). This is a major feature of MPC-B with respect to teleoperation and linear blending. Also, the standard deviations both on the evaluation metrics and on the trajectories are illustrative of the repeatability of the task induced by MPC-B with respect to teleoperation and linear blending. Finally, the average jerk metric (M) indicates that the resulting motion is more continuous which is a desired property in critical tasks.

Performance: Both linear blending and MPC-B lead to improved performance in terms accuracy (P) and execution time (T) with respect to teleoperation. This is also supported by the agreement metric (A) which illustrates the quality of the human control input towards the different target zones.

Assistance: The effect of assistance on the effort (E) put by the human in achieving the task advocates for the use of shared control rather than pure teleoperation. Here again MPC-B outperforms linear blending which can be explained by the optimality of the produced blended control when considering the control problem over an horizon rather than reactively. As a consequence, the user resort more to assistance (L) with MPC-B than with linear blending. This provides some insight on the increased trust MPC-B might induce.

VI. CONCLUSION

In this paper, we proposed a model predictive control-based blending (MPC-B) approach for shared control, comparing it to standard teleoperation and linear blending in both simulations and a proof-of-concept experimental study using a pick-and-place task as a general application scenario. To enable objective evaluation, a set of quantitative metrics are developed, building on existing literature.

The results demonstrate that MPC-B outperforms both teleoperation and linear blending, offering a safer, more efficient, and repeatable solution for shared control. The findings also suggest an improvement in user trust and comfort.

Future work will focus on developing adaptive models that dynamically adjust the authority level based on contextual factors. Additional experiments with a larger and more diverse user population will be conducted to further validate the controller's effectiveness, particularly regarding user trust and comfort. Finally, while the assistance system operates with reliable knowledge of the environment in this study, future research will explore scenarios with higher levels of uncertainty, where degraded assistance may require the user to assume greater control.

REFERENCES

[1] S. H. Tang, F. Kamil, W. Khaksar, N. Zulkifli, and S. Ahmad, "Robotic motion planning in unknown dynamic environments: Existing approaches and challenges," in *Proceeddings of the IEEE International Symposium on Robotics and Intelligent Sensors*, pp. 288–294, 2015.

[2] F. Rydén, A. Stewart, and H. J. Chizeck, "Advanced telerobotic underwater manipulation using virtual fixtures and haptic rendering," in *OCEANS - San Diego*, pp. 1–8, 2013.

[3] N. Marturi, A. Rastegarpanah, C. Takahashi, M. Adjigble, R. Stolkin, S. Zurek, M. Kopicki, M. Talha, J. A. Kuo, and Y. Bekiroglu, "Towards advanced robotic manipulation for nuclear decommissioning: A pilot study on tele-operation and autonomy," in *Proceedings of the International Conference on Robotics and Automation for Humanitarian Applications*, pp. 1–8, IEEE, 2016.

[4] J. M. Hillis, M. O. Ernst, M. S. Banks, and M. S. Landy, "Combining sensory information: mandatory fusion within, but not between, senses," *Science*, vol. 298, no. 5598, pp. 1627–1630, 2002.

[5] M. Goodrich and A. Schultz, "Human-robot interaction: A survey," *Foundations and Trends in Human-Computer Interaction*, vol. 1, pp. 203–275, 01 2007.

[6] D. A. Abbink, T. Carlson, M. Mulder, J. C. F. de Winter, F. Aminravan, T. L. Gibo, and E. R. Boer, "A topology of shared control systems—finding common ground in diversity," *IEEE Transactions on Human-Machine Systems*, vol. 48, no. 5, pp. 509–525, 2018.

[7] T. B. Sheridan, *Telerobotics, automation, and human supervisory control*. Cambridge, MA, USA: MIT Press, 1992.

[8] S. Musić and S. Hirche, "Control sharing in human-robot team interaction," *Annual Reviews in Control*, vol. 44, pp. 342–354, 2017.

[9] A. D. Dragan and S. S. Srinivasa, "A policy-blending formalism for shared control," *The International Journal of Robotics Research*, vol. 32, no. 7, pp. 790–805, 2013.

[10] D. Gopinath, S. Jain, and B. D. Argall, "Human-in-the-loop optimization of shared autonomy in assistive robotics," *IEEE robotics and automation letters*, vol. 2, no. 1, pp. 247–254, 2016.

[11] C. Ezeh, P. Trautman, L. Devigne, V. Bureau, M. Babel, and T. Carlson, "Probabilistic vs linear blending approaches to shared control for wheelchair driving," in *Proceedings of the International Conference on Rehabilitation Robotics*, pp. 835–840, IEEE, 2017.

[12] P. Trautman, "Assistive planning in complex, dynamic environments: A probabilistic approach," in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, pp. 3072–3078, 2015.

[13] S. Musić and S. Hirche, "Haptic shared control for human-robot collaboration: A game-theoretical approach," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 10216–10222, 2020.

[14] A. Escande, N. Mansard, and P.-B. Wieber, "Hierarchical quadratic programming: Fast online humanoid-robot motion generation," *The International Journal of Robotics Research*, vol. 33, no. 7, pp. 1006–1028, 2014.

[15] M. Liu, Y. Tan, and V. Padois, "Generalized hierarchical control," *Autonomous Robots*, vol. 40, pp. 17–31, 2016.

[16] J. B. Rawlings, D. Q. Mayne, M. Diehl, *et al.*, *Model predictive control: theory, computation, and design*, vol. 2. Nob Hill Publishing Madison, WI, 2017.

[17] S. Kleff, A. Meduri, R. Budhiraja, N. Mansard, and L. Righetti, "High-frequency nonlinear model predictive control of a manipulator," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 7330–7336, 2021.

[18] C. Guo, C. Sentouh, J.-C. Popieul, and J.-B. Haué, "MPC-based shared steering control for automated driving systems," in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, pp. 129–134, 2017.

[19] A. M. R. Lazcano, T. Niu, X. C. Akutain, D. Cole, and B. Shyrokau, "MPC-based haptic shared steering system: A driver modeling approach for symbiotic driving," *IEEE/ASME Transactions on Mechatronics*, vol. 26, no. 3, pp. 1201–1211, 2021.

[20] L. Peternel, N. Tsagarakis, D. Caldwell, and A. Ajoudani, "Robot adaptation to human physical fatigue in human–robot co-manipulation," *Autonomous Robots*, vol. 42, pp. 1011–1021, 2018.

[21] R. Balachandran, H. Mishra, M. Cappelli, B. Weber, C. Secchi, C. Ott, and A. Albu-Schaeffer, "Adaptive authority allocation in shared control of robots using bayesian filters," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 11298–11304, 2020.

[22] L. Milliken and G. A. Hollinger, "Modeling user expertise for choosing levels of shared autonomy," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 2285–2291, 2017.

[23] P. A. Hancock, D. R. Billings, K. E. Schaefer, J. Y. C. Chen, E. J. de Visser, and R. Parasuraman, "A meta-analysis of factors affecting trust in human-robot interaction," *Human Factors*, vol. 53, no. 5, pp. 517–527, 2011. PMID: 22046724.

[24] C. Mower, J. Moura, and S. Vijayakumar, "Skill-based shared control," in *Robotics: Science and Systems XVII*, The Robotics: Science and Systems Foundation, 2021.

[25] M. J. Zeestraten, I. Havoutis, and S. Calinon, "Programming by demonstration for shared control with an application in teleoperation," *Robotics and Automation Letters*, vol. 3, no. 3, pp. 1848–1855, 2018.

[26] A. Hansson and M. Servin, "Semi-autonomous shared control of large-scale manipulator arms," *Control Engineering Practice*, vol. 18, no. 9, pp. 1069–1076, 2010.

[27] Y. Li and S. S. Ge, "Human–robot collaboration based on motion intention estimation," *IEEE/ASME Transactions on Mechatronics*, vol. 19, no. 3, pp. 1007–1014, 2014.

[28] M. Li and A. Okamura, "Recognition of operator motions for real-time assistance using virtual fixtures," in *Proceedings of the 11th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pp. 125–131, 2003.

[29] 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," *Applied Mechanics Reviews*, vol. 70, no. 1, p. 010804, 2018.

[30] P. Damacharla, A. Y. Javaid, J. J. Gallimore, and V. K. Devabhaktuni, "Common metrics to benchmark human-machine teams (hmt): A review," *IEEE Access*, vol. 6, pp. 38637–38655, 2018.

[31] M. Itoh, F. Flemisch, and D. Abbink, "A hierarchical framework to analyze shared control conflicts between human and machine," vol. 49, no. 19, pp. 96–101, 2016. 13th IFAC Symposium on Analysis, Design, and Evaluation of Human-Machine Systems HMS.

[32] D. Zhang, R. Tron, and R. P. Khurshid, "Haptic feedback improves human-robot agreement and user satisfaction in shared-autonomy teleoperation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 3306–3312, 2021.

[33] B. Pano, P. Chevrel, F. Claveau, C. Sentouh, and F. Mars, "Obstacle avoidance in highly automated cars: Can progressive haptic shared control make it safer and smoother?," *IEEE Transactions on Human-Machine Systems*, vol. 52, no. 4, pp. 547–556, 2022.

[34] T.-C. Lin, A. Unni Krishnan, and Z. Li, "Shared autonomous interface for reducing physical effort in robot teleoperation via human motion mapping," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 9157–9163, 2020.

[35] N. Torres Alberto, A. Skuric, L. Joseph, V. Padois, and D. Daney, "Linear Model Predictive Control in SE(3) for online trajectory planning in dynamic workspaces." Technical Report, Sept. 2022.

[36] A. Erdogan and B. D. Argall, "The effect of robotic wheelchair control paradigm and interface on user performance, effort and preference: An experimental assessment," *Robotics and Autonomous Systems*, vol. 94, pp. 282–297, 2017.

[37] A. Del Prete, "Joint position and velocity bounds in discrete-time acceleration/torque control of robot manipulators," *Robotics and Automation Letters*, vol. 3, no. 1, pp. 281–288, 2017.

[38] L. Joseph, V. Padois, and G. Morel, "Towards x-ray medical imaging with robots in the open: safety without compromising performances," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 6604–6610, 2018.