Modulating Human Input for Shared Autonomy in Dynamic Environments

Christopher E. Mower[1], João Moura[1], Aled Davies[2], and Sethu Vijayakumar[1]

***Abstract*— Many robotic tasks require human interaction through teleoperation to achieve high performance. However, in industrial applications these methods often require high levels of concentration and manual dexterity leading to high cognitive loads and dangerous working conditions. Shared autonomy attempts to address these issues by blending human and autonomous reasoning, relieving the burden of precise motor control, tracking, and localization. In this paper we propose an optimization-based representation for shared autonomy in dynamic environments. We ensure real-time tractability by modulating the human input with the information of the changing environment in the same task space, instead of adding it to the optimization cost or constraints. We illustrate the method with two real world applications: grasping objects in a cluttered environment, and a spraying task requiring sprayed linings with greater homogeneity. Finally we use a 7 degree of freedom KUKA LWR arm to simulate the grasping and spraying experiments.**

I. INTRODUCTION

Satisfactory completion of complex robotic tasks often requires human intervention, via teleoperation, due to high-risk and unpredictability of the task and environment. However, there are a number of factors that negatively impact direct teleoperation, such as:

(L1) inadequate or unintuitive interfaces,
(L2) coarse and highly variant input commands from the operator,
(L3) poor observability of the task by the operator,
(L4) a limited supply of skilled workers,
(L5) deficient fidelity of the link between operator and the robot (for instance caused by network latency), and
(L6) operator fatigue due to high levels of concentration for prolonged periods of time.

These limitations often lead to excessive cognitive loads and consequently dangerous working environments. An example from construction, and our main application example, is concrete spraying shown in Fig. 1. The operator here maneuvers the spraying unit by commanding it's individual actuators using a number of joysticks, while simultaneously ensuring job-site safety, achieving high task performance, and minimizing wastage.

There are numerous shared autonomy methods that address some of the above limitations of teleoperation, such as: predict-then-act [1], next-best viewpoint for an external camera-in-hand robot [2], learning from demonstration approaches [3], and sliding autonomy [4]. Shared autonomy is defined as the blend between teleoperation and autonomous reasoning that realizes robot actions [5]. The aim of shared autonomy is to reduce the cognitive load on the human operator by leveraging autonomous capabilities informed by available sensory information and task heuristics addressing, therefore, the limitations of direct teleoperation. Determining an appropriate balance between human input and autonomous assistance that is intuitive for the operator and reliable for the completion of complex tasks is a key challenge for the robotics community.

Fig. 1. Concrete spraying in a freshly excavated tunnel, using a 5-DoF concrete spraying unit. A skilled operator controls the device on a per-actuator level via a number of joysticks. Image courtesy of Costain Laing O'Rourke Joint Venture.

A common approach when blending human input and autonomous reasoning is to allocate specific sub task-space dimensions as either human or autonomously regulated. For example, Abi-Farraj et al. [6] proposes a shared control architecture for a pick-and-place task, that allocates one translation and three rotational dimensions to the human and the additional two translation dimensions to a fully autonomous system. Sub-task allocation is often favorable since it avoids the difficulty of dealing with conflicting interests, but fails to address limitation (L2).

Additionally, even though there is literature, e.g. [7], suggesting a "*bounded rationality*" on the cognitive capabilities of the human, most shared autonomy methods either consider a static environment or implicitly assume that the operator keeps track of all the changes in environment [8], [6], [9], [10]. Therefore, some of these methods might still lead to heightened fatigue, highlighted in limitation (L6). The level of trust between human and the autonomous reasoning is also evidently a key factor when addressing limitations (L1)

[1]Christopher E. Mower, João Moura, and Sethu Vijayakumar are with the School of Informatics, University of Edinburgh, UK {chris.mower, joao.moura, sethu.vijayakumar}@ed.ac.uk
[2]Aled Davies is with the Costain Group PLC., UK aled.davies@costain.com

and (L6) [1], [11], however this is out of the scope of our discussion in this paper.

In our previous work [12] we investigated how various control spaces (e.g. joint space or task space), in which unskilled operators submit commands, and their dimensionality affect overall task performance for target acquisition tasks, addressing the limitations (L1) and (L4). A key conclusion was that unskilled human operators achieve higher performance by commanding the robot in a lower dimensional task space. We leverage this knowledge here in order to design a more effective shared autonomous system.

In this work we build on the work of [1] and blend the human and autonomous input within the same task space, rather than the common approach of allocating specific task spaces for each to operate within [13]. We assume that the human gives coarse input, i.e. we assume unskilled operators, and our goal is to modulate that input to produce optimal robot motion with respect to some objective function and motion constraints, effectively addressing the limitation (L2). Additionally, we address limitation (L6) by incorporating information about the environment changes in the objective function. Thus, the human acts as a guide and the autonomous reasoning is responsible for the environment and precise motor control in order to reduce the cognitive load. In summary, in this paper

- we propose an optimization-based representation for shared autonomy that accounts for the environment changes while remaining computationally tractable, by modulating the human input with the task/environment information within the same task space,
- we describe how to apply our method to the following two case studies, by defining various objective terms,
  - a grasping task where the human guides the robot and is assisted in collision avoidance and, by a prediction scheme, choosing appropriate grasp orientations, and
  - a spraying task that demonstrates how to modulate the human input with changes in the environment introduced by the sprayed material, and
- finally, we implement our method and the different objective terms for the previous case studies using a simulated 7 degrees' of freedom (DoF) KUKA LWR arm.

The paper is organized as follows. Section II details a problem formulation for the type of problems our method applies to in the scope of numerical optimization, and Section III describes the related work. In Section IV we describe the proposed method and in Section V we expand on how to realize it for a shared autonomous pick-and-place and spraying tasks. Section VI covers the experiments and results and Section VIII covers the discussion. Finally, in Section VIII we conclude our work and discuss future avenues.

II. PROBLEM FORMULATION

Let us model motion generation as a discrete time state-dependent policy function

$$x_{t+1} = \pi(x_t, u_t, e_{0:t}), \tag{1}$$

where x_t represents the robotic system state at the current time t, u_t the control input, and e_t the state of the environment. We assume the $(t+1)$th order Markovian property for the policy π and that the initial state x_0 is known.

There are, in the literature, multiple approaches for describing a policy π. From simple analytical expressions for trivial examples to various heuristic algorithms purposely suited for a given application. Recent years have popularized descriptions based on numerical optimization, since they are able to account for multiple motion constraints whilst minimizing a given objective function. Moreover, an optimization-based problem description allows for the use of readily available robust optimization algorithms and libraries (such as SNOPT and IPOPT), instead of ad-hoc implementations.

In this work we formulate π as the optimization problem of finding the next-best state with respect to an objective function describing task goals that consider the full history of the environment, and motion constraints as

$$x_{t+1} = \arg\min_{x} \; f(x; x_t, u_t, e_{0:t}) \tag{2a}$$

$$\text{subject to}$$

$$c_{eq}(x) = 0 \quad \text{and} \quad c_{ineq}(x) \leq 0 \tag{2b}$$

where f is some real-valued objective function, and c_{eq},c_{ineq} describe the equality and inequality constraints respectively.

For complex tasks in dynamic environments it is often very difficult to find robust methods that specify appropriate control u_t to autonomously complete the task. Complexities in tasks often derive from poor situational and contextual awareness of the part of the autonomous system, often translating as a poor representation or handling of e_t. Additionally, common methods such as trajectory optimization suffer from large computation times rendering these methods inappropriate for use in real-time systems.

As an alternative, teleoperation methods express the control input u_t as a function of the human input h_t (often $u_t \propto h_t$ and projected in the joint or task space). These systems typically consider the human as an observer of the environment and assume that e_t is a parameter of the humans inherent model of the task, rather than a parameter of (2). We can therefore think of those control schemes as an one-to-one mapping between the human input and robot motion. Shared autonomy attempts to relax this assumption and redefine the control input as a blend of human and autonomous reasoning.

III. RELATED WORK

A. Motion planning

Various methods allow robots to operate in dynamic environments. For example, sampling based planners use sensory data in the current time frame [14]. Whilst typically robust, this method is generally unable to ensure accuracy and efficiency.

Another approach for handling dynamic environments is to track the changes using some representation, e.g. an occupancy grid, and to continuously check for the intersection between the predicted robot motion or pre-defined

plan and the collision map [15]. Our method is similar in approach to these methods, however, we consider tasks such as spraying where the full history of the task affects the future motion of the robot, as opposed to a snapshot of the current environment state.

When demonstrations are available, Dynamic Movement Primitives (DMP) [16] can encode the demonstrated trajectories as a set of differential equations, being able to adapt the robot motion and to handle perturbations during execution. Motivated by the dissimilarities between job sites in the concrete spraying application, we assume no access to demonstrated trajectories.

A dynamical system approach [17] also allows the adaptation of the robot motion on-the-fly with respect to some original plan while ensuring collision-free motions with multiple convex shaped objects. The approach only considers collisions with the end-effector.

The potential field approach [18] generates collision-free motion within some static environment by summing attractive and repulsive virtual forces defined by some known goal position and obstacle positions, respectively. In order to produce more robust plans, [19] extends it to handle dynamic obstacles. However, the main drawback of the potential field approach is its propensity to local minima, especially in high dimensions.

A number of methods in the literature handle dynamic environments by projecting task goals into alternative representations. For example, [20] uses a relative distance space representation for real-time null-space motion adaptation to avoid self-collisions and collisions with known objects. Topological-based representations using a combination of writhe and interaction mesh space, as in [21], can handle complex tasks such as wrapping. Another method [22] leverages a computation scheme typically used to estimate electric flux in the field of electro-dynamics for coverage tasks involving wrapping.

The methods discussed above propose various approaches to deal with dynamic environment without incorporating human-level intervention, thus, being unable to handle complex tasks that require situational and/or contextual awareness.

B. Shared autonomy

There exists a body of research based on shared autonomy that addresses the limitations of direct control. A subset of these works develop arbitration frameworks utilizing operator intent prediction that assumes a probability distribution over the possible goals of the operator. These methods typically attempt to address problems (L1, L2). Hauser [8] and Javdani et al. [23] develop intent recognition systems that blend assistance based on a prediction of the operators goal. These works enable assistance even when the confidence in the prediction is low and are important when it is difficult to predict a single goal from multiple possibilities. Dragan and Srinivasa [1] propose a policy blending formalism that relies on the concept of arbitration for blending operator input with a prediction of the operators intent. From their user study, they verify the importance of the confidence in the prediction for moderating the level of arbitration. In our work we assume a given intention/goal and show how to integrate it within our shared autonomy framework.

To address (L1, L2, L5) and to prevent unsafe robot motions, forbidden regions can be defined by specifying virtual fixtures in the task space. These can be pre-specified by an expert [9], or computed on-the-fly with respect to sensory data [24]. However, in both cases the misplacement of such virtual fixtures due to human error or sensor noise can still lead to unsafe motions.

Rakita et al. [2] address problem (L3) by continuously providing an effective viewpoint to a remote user using an additional camera-in-hand robot arm. Abi-Farraj addresses problem (L4) by encoding tasks as trajectory distributions demonstrated by expert operators using direct control. The proposed system assists an unskilled operator in manipulation tasks by providing force cues to the user via a haptic interface based on the demonstrated trajectory distributions.

Several works, demonstrated at the DARPA Robotics Challenge Finals, address problems (L5) and (L6). For example, Marion et al. [10] proposes a piloting system, called DIRECTOR for the ATLAS robot, where the pilot specifies task sequences using high-level motion primitives, and the shared autonomous system incorporates perception and optimization-based trajectory motion planning.

IV. PROPOSED METHOD

In this section we detail our shared autonomy optimization-based approach for obtaining the robot policy, using the formulation (2). In order to achieve real-time tractability when solving this optimization, we must take special care in setting up our shared autonomy problem. For instance, providing an analytical Jacobian of the cost function f to the optimizer, as opposed to some estimation method, considerably speeds up solving times. In the next section, we will extend on the specification of f via various objective terms.

However, for complex and dynamic environments, the analytical Jacobians for the collision map are typically unavailable or slow to compute, easily rendering the problem intractable for real-time purposes. In this work, instead of neglecting the dynamic environment from (2) altogether, we offload its representation from the users inherent model to the autonomous system by incorporating it in the control input function as

$$u_t(h_t, e_{0:t}) := g_H(h_t) + g_A(e_{0:t}), \tag{3}$$

where g_H and g_A respectively map the human input h_t and the full task history $e_{0:t}$ to a control action. Therefore, within this formulation, the environment information is a parameter of the control, rather than incorporated in the optimization.

We assume that e_t appropriately describes the task and the environment, and that at each time-step we can keep track of all $e_{0:t}$. For example, in a shared autonomous spraying system, e_t can be the current surface location being sprayed,

easily computed by the forward kinematics and a model of the surface.

Blending the human input and task/environment information in the same space by means of a summation as in (3) raises questions about conflicting input. We resolve that conflict by treating the human input h_t as a goal that attracts the robot into some particular task position, and the task/environment information $e_{0:t}$ either as an attractor or a repeller conditioning the robot task motion. Because both human input and task goals/environment vary with time, we can think of this blending as essentially building a different task potential field for each time iteration, which determines the corresponding control action.

We express g_H as

$$g_H(h_t) := \alpha_H \big(h_t - \phi_t\big) \tag{4}$$

where $0 < \alpha_H \in \mathbb{R}$ is some scalar parameter representing the strength of the human part, and $\phi_t = \phi(x_t)$ maps the system state space to the task space of interest, e.g. ϕ typically maps the robot joint positions to the translation and rotational components of the end-effector. In a spraying task, ϕ could represent the two dimensional position on the spraying surface that the robot end-effector is pointing.

We express g_A as

$$g_A(e_{0:t}) := \alpha_A \sum_{\tau=0}^{t} \beta_\tau (e_\tau - \phi_t) \tag{5}$$

where $0 < \alpha_A \in \mathbb{R}$ is some scalar parameter representing the strength of the autonomous part, and $\beta_\tau \in \mathbb{R}$ is a scalar parameter tuned with respect to every point e_τ. We assume that the choice of each β_τ can be either constant or updated by some strategy during the task. If $\beta_\tau > 0$ then this implies e_τ is a goal, otherwise if $\beta_\tau < 0$ then e_τ is an obstacle.

V. CASE STUDIES

We can apply the proposed method to various tasks such as grasping, welding, wiping, and spraying. In this section, we illustrate how to use our optimization-based approach for a shared autonomous grasping task and a shared autonomous spraying task. We model the goals of each task as a weighted objective function

$$f(x; x_t, u_t, e_{0:t}) = \sum_i \rho_i f_i(x; x_t, u_t, e_{0:t}) \tag{6}$$

where $0 \neq \rho_i \in \mathbb{R}$ is a weighting term that reflects the importance of each sub-task, and each f_i models a specific sub-task. In the following sub-sections we specify possible objective terms f_i for the grasping and spraying tasks. For brevity, we omit their analytical Jacobians which we use for computational efficiency.

A. Shared autonomous grasping

Scenarios such as assistive robots in every day tasks [1] and nuclear waste disposal [6] motivate the use of teleoperated grasping. One typical example is clearing, where the robot has to grasp a number of static or dynamic objects in a particular order and in a changing and cluttered environment, whilst avoiding collisions. Such tasks require contextual and situational awareness, as shown in Fig. 2.

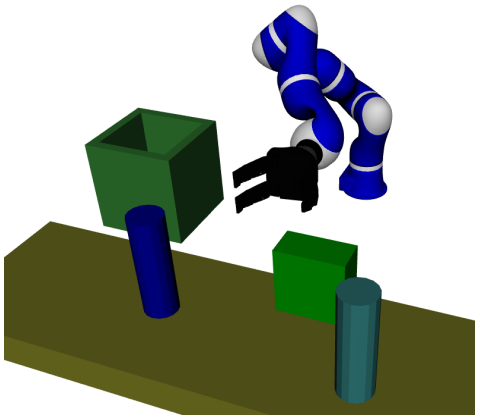


Fig. 2. Example of a shared autonomous grasping task. We assume that the objects poses and grasp locations are known via appropriate sensory data.

Under direct control, in order to complete this task, the operator must decide the ordering in which to grasp the objects, track the changes in environment, and maneuver the grasping tool to pick and place the items at a goal position (e.g. in the bin) whilst avoiding collisions. The combination of planning motion for multiple goals, tracking and localization, and ensuring precise collision-free robot motion undoubtedly results in high cognitive loads for the operator.

Let us assume the human provides commands in the three-dimensional end-effector velocity space. We can represent collision-free end-effector motion using

$$f_{EffPos} := \|\phi_{pos}(x) - (\phi_{pos}(x_t) + u_t(h_t, e_{0:t}))\|^2 \tag{7}$$

where ϕ_{pos} is the forward kinematics mapping that computes the position of the end-effector, and u_t is the modulated input (3). In this case the user is assisted to produce collision-free motions by merging the operators goal and the environment model.

Moreover, we can incorporate a goal orientation g_t, i.e. a function of the predicted item the operator intends to grasp, by using

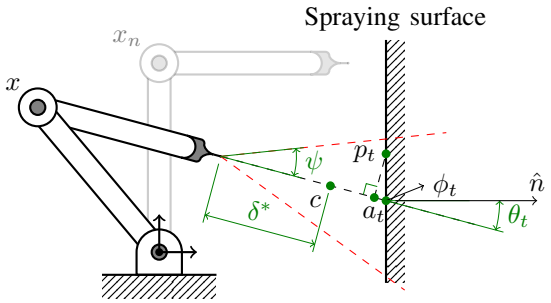
$$f_{EffOri} := \|\phi_{ori}(x) - g_t(h_t)\|^2 \tag{8}$$

where ϕ_{ori} is the forward kinematics mapping that computes the orientation of the end-effector. For g_t we use a simple prediction rule that chooses the appropriate orientation according to the closest object to the end-effector. We refer to [1] for more elaborate prediction rules.

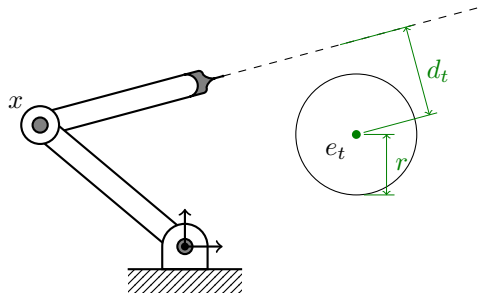
Another goal for the shared autonomous grasping task is to have collision free null-space motion, which we can achieve by introducing an additional cost term (e.g. see [20]) or constraint equation (e.g. see [25]).

B. Shared autonomous spraying

In this example, we take inspiration from our main application source, concrete spraying. A criteria for high job



(a) Representation of a robot in some state x, some nominal state x_n, and points of interest with respect to the spraying surface. The dotted red line indicates an estimation of the spraying cone.



(b) Depiction of the $AvoidLookAtSphere$ task map.

Fig. 3. Schematic diagrams showing main points of interest for the shared autonomous spraying task.

quality concrete spraying is to ensure the sprayed lining is as homogeneous as possible [26]. This requires the operator to track all the positions along the spraying surface previously visited in order to ensure minimal overlap. We aim to use our system to aid the operator to perform more accurate spraying.

For spraying tasks, there are five main task objectives of interest: (i) given a position on the wall to spray, orient the end-effector to "look at" this position, (ii) given the normal to the wall at this point, align end-effector axis with this normal, (iii) maintain a given standoff distance, (iv) ensure smooth motion synthesis, (v) help the operator to avoid over-spraying. Note, all points of interest discussed in the rest of this section are shown in Fig. 3.

Our $LookAt$ task relies on the cost term

$$f_{LookAt} := \|p_t - a_t\|^2 \tag{9}$$

where $p_t := \phi_t + u_t$ is the target point, a function of the current sprayed position on the surface ϕ_t and control u_t, and a_t is the orthogonal projection of p_t onto the end-effector approach axis.

The $EffAxisAlign$ task consists in aligning the end-effector with the spraying surface using the cost term

$$f_{EffAxisAlign} := \|\cos(\theta_t) - 1\|^2 \tag{10}$$

where θ_t denotes the angle between the surface normal $\hat{n}$ and the end-effector approach axis.

We assume a given optimal standoff distance $0 < \delta^* \in \mathbb{R}$ specified by an operator [27]. In this case we model the $StandoffDist$ task as

$$f_{StandoffDist} := \|c - p_t\|^2 \tag{11}$$

where c is a constant point in the end-effector frame such that its distance away is δ^*.

In order to draw the robot to a more desirable configuration, for example a configuration which keeps an elbow joint up, we can specify an additional cost term

$$f_{Nominal} := \|x - x_n\|^2 \tag{12}$$

where x_n is some given manipulator specific nominal configuration.

To ensure smooth joint motion we minimize joint velocity $\dot{x}$, acceleration $\ddot{x}$, and jerk $\dddot{x}$ using

$$f_{MotionSmooth} := \omega_1\|\dot{x}_t\|^2 + \omega_2\|\ddot{x}_t\|^2 + \omega_3\|\dddot{x}_t\|^2 \tag{13}$$

where $0 < \omega_1, \omega_2, \omega_3 \in \mathbb{R}$ are weighting terms, and $\dot{x}, \ddot{x}$, and $\dddot{x}$ are estimated using backward differencing using a window of the previous three solutions. Smoothing joint space motion up to the third derivative is generally beneficial in terms of wear and tear on the robot.

Let $e_{0:t}$ represent all points previously sprayed. We describe two potential approaches for avoiding over-spray. The first uses an inequality constraint, and the second modulates the input using the sprayed locations information.

a) Inequality constraint approach: We can define the constraint for every e_t as

$$c_{AvoidLookAtSphere} := r^2 - d_t^2, \tag{14}$$

which ensures that the distance d_t between the end-effector approach axis and the point e_t is always greater than a given radius r. Of course, over the duration of the task the number of points e_t grows.

b) Input modulation approach: In this approach we remove the previous inequality constraint and instead directly use a control input u_t that modulates the human input with the environment model $e_{0:t}$ using (3).

Finally, additional inequality constraints enforce practical motion constraints such as joint limits. To ensure that the target point p_t is within a spraying cone we can also represent this as an inequality constraint. Let ψ be the angle of the spraying cone. We can then use the equation of a cone to derive an additional inequality constraint

$$c_{GazeAt} := \begin{pmatrix} p_t^{x2} + p_t^{y2} - \tan(\theta)^2 p_t^{z2} \\ -p_t^z \end{pmatrix} \tag{15}$$

where $p_t = (p_t^x, p_t^y, p_t^z)^T$ is the target point defined with respect to the end-effector frame. Note, the bottom term in (15) ensures p_t is in-front of the end-effector.

VI. EXPERIMENTS AND RESULTS

The experiments described in this section consist in two different tasks inspired by real world industrial applications: grasping and spraying. Our experiments use a simulated 7-DoF KUKA LWR arm, and additionally the grasping task uses a simulated Robotiq 3-finger gripper. In the grasping task, the user supplies commands to the system using a Logitech F710 gamepad which streams motion commands to the robot at a sampling frequency of 50Hz. For the spraying experiments we use synthetic human input. All experiments use the EXOTica optimization framework [28].

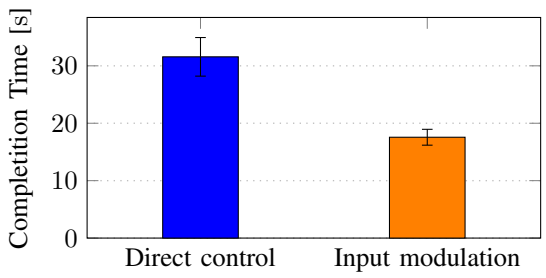


Fig. 4. Completion time for the direct control mode versus the input modulation mode for a clearing task.

A. Shared autonomous grasping

We illustrate the potential benefits of our formulation in a grasping scenario, as described in Section V-A. The goal of the task consists of clearing a number of items from the table in a particular order by grasping each item and placing it into the bin without collisions and as timely as possible, as depicted in Fig. 2.

We tested two different modes: in the first, *direct control* mode, the human operator regulates the three translation dimensions of the end-effector; the second, *input modulation* mode, is our proposed method. The system has access to a collision map of the scene, appropriate grasp orientations for each item, and it is able to keep track of objects. The operator has to place the objects in a given order according to their color, and this contextual knowledge is only known by the human in both modes. The environment is static, however this need not be the case if tracking is available via perception.

To compare each mode we collected the completion time for five trials under each mode. The direct control mode resulted in an average time of 31.6 ± 3.0 seconds, whilst the input modulation mode resulted in an average time of 17.6 ± 1.2 seconds, as shown in Fig. 4.

B. Shared autonomous spraying

1) Direct control versus input modulation: In this section, the task comprises spraying a surface with minimal sprayed overlap. Similarly to the grasping experiment, we compare two modes, a direct control mode, and an input modulation mode based on our proposed method. In both cases the human defines a two-dimensional trajectory for the robot to follow along the spraying surface. The direct control method tracks this trajectory as best as possible. The input modulation method uses our proposed method to modulate this input and avoid sprayed areas.

Fig. 5 shows our experimental setup. For each mode, we assume the same kinematic model, human-defined trajectory, and environment model. We synthesize 469 data points corresponding to the input from the human operator in the task space. At each iteration in time, the environment variable e_t stores the currently sprayed position ϕ_t. However, we introduce a delay when logging e_t to the environment

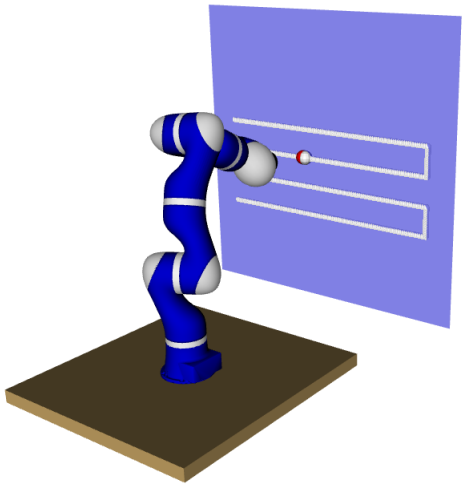


Fig. 5. Experimental setup for a mock spraying task. The blue surface represents the spraying surface and the white line indicates synthesized human input.

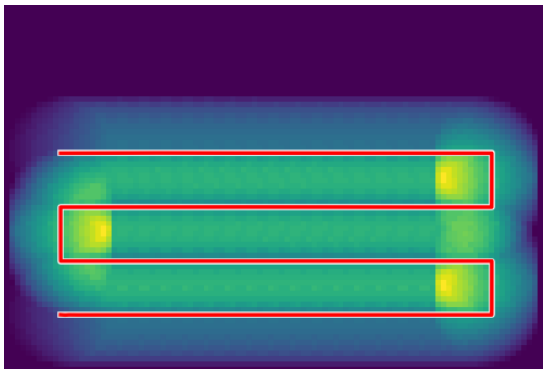
model $e_{0:t}$ in order to avoid large and unrealistic repulsive potentials.

Fig. 6 shows a color-map representing the sprayed regions, where brighter colors correspond to over-spraying. We can qualitatively observe that the precision tracking of the direct control mode results in a high overlap and minimal coverage, whereas the input modulation mode results in an increased coverage and less overlap. However, overlap is still high in the corners.

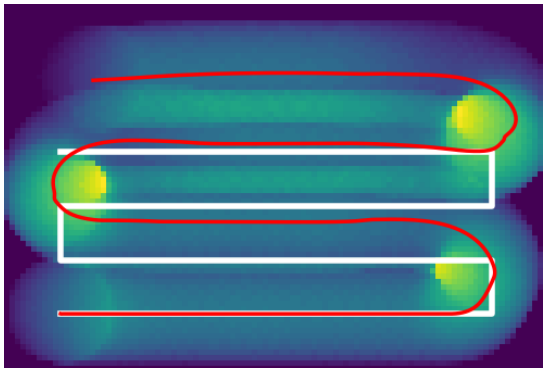
2) Inequality constraints versus input modulation: We compare two strategies for shared autonomy applied to the spraying task. The sub-task under consideration is to assist the user to avoid re-spraying. The setup for this experiment is the same as in the previous section. The first method encodes $e_{0:t}$ as avoidance regions using the $AvoidLookAtSphere$ constraint (14) in the optimization formulation (2b). The second, based on our proposed method, modulates the human input using (3). The goal of this experiment is to demonstrate the computational savings our method can have, rendering it applicable for a real-time system.

We split each control cycle into two phases: a setup phase and a solver phase. The setup phase initializes a problem by specifying the target p_t. The solver phase parses the problem to an optimization solver; here we use SNOPT [29]. For the first formulation, the setup phase consists of specifying target position p_t. For the second formulation, the setup phase uses our proposed method to modulate the human input. We repeated each experiment 20 times for each mode and present the average computation times in Fig. 7.

For the setup phase (Fig. 7a), the completion time for the inequality constraint mode is negligible whereas we can clearly observe a positive correlation between the CPU time and the number of points in the scene. For the solver phase, Fig. 7b shows that the input modulation mode scales better for high numbers of points in the scene as opposed to the inequality constraint mode. We additionally report

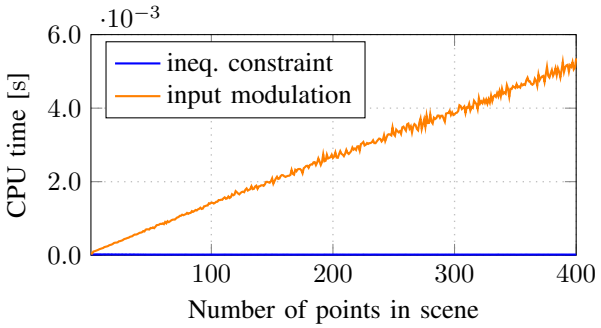


(a) Direct control.

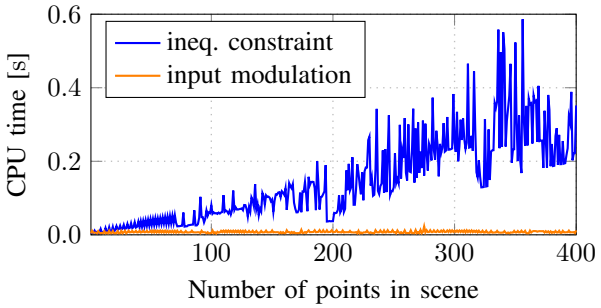


(b) Input modulation.

Fig. 6. Comparison between direct control mode and the input modulation mode for a mock spraying task. The white line represents the synthesized human input. The red line represents the spraying trajectory.



(a) Setup phase.



(b) Solver phase.

Fig. 7. Completion time results plotted against number of points in the scene for (a) the setup phase, and (b) the solver phase for the shared autonomous spraying task.

for each mode the average number of iterations for the solver phase and the mean-variance for the setup and solver phases. The average number of iterations for the inequality constraint mode and input modulation mode is 12.73 ± 5.90 and 12.82 ± 5.72 iterations respectively. For the inequality constraint mode, the mean-variance for the setup and solver time was 5.58×10^{-12} and 3.57×10^{-6} respectively. For the input modulation mode, the mean-variance for the setup and solver time was 6.11×10^{-8} and 5.84×10^{-7} respectively.

VII. DISCUSSION

We compared a direct control mode versus our proposed input modulation mode for a grasping task. The experiments demonstrate that the input modulation mode leads to significantly faster completion times. The completion time for the direct control mode is nearly double that of our method. This is unsurprising, since we modulate the operator input in order to produce collision-free motion whereas for the direct control this is a responsibility of the human. This modulation removes the need for precise input relegating it to the autonomous system relaxing, therefore, the burden on the operator. However, this is just an illustrative experiment. More conclusive results require further user-study experiments.

We also compare a direct control mode against our proposed method for a mock spraying task. Fig. 6a shows the spraying profile when using the direct control mode. We clearly observe there is a large amount of overlap between the sprayed layers. There is also a significant portion left to spray in the top part of the plane thus requiring additional, and consequently, wasted material. Fig. 6b shows a more homogeneous color distribution. Fig. 6 demonstrates how our method is able to regulate the robot motion in order to produce greater homogeneity in the sprayed lining. In Fig. 6b, we can see the resulting curved edges of the spraying trajectory which is a desirable motion for obtaining smoothness of the laid material. Additionally, we can observe in both modes a high amount of overlap in the corners. In the experiments we tune every β_τ, as in (5), to a constant value. We could potentially use different β_τ at different points in the trajectory to minimize this overlap effect.

Finally, we compared setup and solver phase completion times for two different shared autonomy formulations, an inequality constraint mode and input modulation mode. Fig. 7b shows that the inequality constraint mode requires larger solver times, whilst Fig. 7a shows the opposite relation for the setup time. However, given the difference in their order of magnitude, we can essentially neglect the effect of the setup time in the total computation time. Therefore, the total computation time scales better for the input modulation mode This happens because the input modulation mode only requires an additional small computation in the task space per additional point in the scene as in (5), whereas the inequality constraint mode requires the computation of an additional Jacobian per additional point in the scene, which is more costly. We also observe that the number of solver iterations

remains roughly constant when increasing the number of points in the scene, which suggests that the time per iteration increases for every additional point in the scene.

VIII. CONCLUSIONS

In this work, we present an optimization based method that blends human and autonomous reasoning within the same task space. We achieve computational tractability by modulating the human input with respect to changes in the environment rather than including within the optimization itself. We describe two realizations of our method for two different case studies and the respective experiments. The experiments illustrate the effectiveness of the method and demonstrate its relevance to real world applications. Our experiments show that modulating the human input, as opposed to representing task objectives in the constraints of the optimization formulation, results in large computational savings.

Future work includes investigating whether this method significantly improves user performance and experience while reducing cognitive load, by conducting human-robot interaction studies. Furthermore, we intend to explore how to handle non-planar terrain for spraying tasks.

ACKNOWLEDGMENT

Christopher E. Mower is partially funded by the Costain Group PLC. This research is supported by the EPSRC UK RAI Hub in Future AI and Robotics for Space (FAIR-SPACE, project ID: EP/R026092/1), the EPSRC UK RAI Hub in Extreme and Challenging Environments (ORCA, project ID: EP/R026173/1), and the UK India Education and Research Initiative (UKIERI, project ID: DST 2016-17-0152).

REFERENCES

[1] 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. [Online]. Available: https://doi.org/10.1177/0278364913490324

[2] D. Rakita, B. Mutlu, and M. Gleicher, "An autonomous dynamic camera method for effective remote teleoperation," in *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, ser. HRI '18. New York, NY, USA: ACM, 2018, pp. 325–333. [Online]. Available: http://doi.acm.org/10.1145/3171221.3171279

[3] F. Abi-Farraj, T. Osa *et al.*, "A learning-based shared control architecture for interactive task execution," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 329–335.

[4] W. Merkt, Y. Yang *et al.*, "Robust shared autonomy for mobile manipulation with continuous scene monitoring," in *2017 13th IEEE Conference on Automation Science and Engineering (CASE)*, Aug 2017, pp. 130–137.

[5] T. B. Sheridan, *Telerobotics, Automation, and Human Supervisory Control*. MIT Press, 1992.

[6] F. Abi-Farraj, N. Pedemonte, and P. Robuffo Giordano, "A visual-based shared control architecture for remote telemanipulation," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 4266–4273.

[7] H. A. Simon, "Rational decision making in business organizations," *The American Economic Review*, vol. 69, no. 4, pp. 493–513, 1979. [Online]. Available: http://www.jstor.org/stable/1808698

[8] K. Hauser, "Recognition, prediction, and planning for assisted teleoperation of freeform tasks," *Autonomous Robots*, vol. 35, no. 4, pp. 241–254, Nov 2013. [Online]. Available: https://doi.org/10.1007/s10514-013-9350-3

[9] L. B. Rosenberg, "Virtual fixtures: Perceptual tools for telerobotic manipulation," in *Proceedings of the 1993 IEEE Virtual Reality Annual International Symposium*, ser. VRAIS '93. Washington, DC, USA: IEEE Computer Society, 1993, pp. 76–82. [Online]. Available: http://dx.doi.org/10.1109/VRAIS.1993.380795

[10] P. Marion, M. Fallon *et al.*, "Director: A user interface designed for robot operation with shared autonomy," *Journal of Field Robotics*, vol. 34, no. 2, pp. 262–280, 2017.

[11] S. Nikolaidis, Y. X. Zhu *et al.*, "Human-robot mutual adaptation in shared autonomy," in *2017 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI*, March 2017, pp. 294–302.

[12] C. E. Mower, W. Merkt *et al.*, "Comparing alternate modes of teleoperation for constrained tasks," in *[to appear] 2019 15th IEEE Conference on Automation Science and Engineering (CASE)*, Aug 2019.

[13] T. Inagaki, "Adaptive automation: Sharing and trading of control," in *Handbook of cognitive task design*. CRC Press, 2003, pp. 171–194.

[14] S. Karaman, M. R. Walter *et al.*, "Anytime motion planning using the rrt*," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 1478–1483.

[15] A. Hermann, F. Mauch *et al.*, "Anticipate your surroundings: Predictive collision detection between dynamic obstacles and planned robot trajectories on the gpu," in *2015 European Conference on Mobile Robots (ECMR)*, Sep. 2015, pp. 1–8.

[16] S. Schaal, *Dynamic Movement Primitives -A Framework for Motor Control in Humans and Humanoid Robotics*. Tokyo: Springer Tokyo, 2006, pp. 261–280. [Online]. Available: https://doi.org/10.1007/4-431-31381-8_23

[17] S. M. Khansari-Zadeh and A. Billard, "A dynamical system approach to realtime obstacle avoidance," *Autonomous Robots*, vol. 32, no. 4, pp. 433–454, May 2012. [Online]. Available: https://doi.org/10.1007/s10514-012-9287-y

[18] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986. [Online]. Available: https://doi.org/10.1177/027836498600500106

[19] D. Park, H. Hoffmann *et al.*, "Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields," in *Humanoids 2008 - 8th IEEE-RAS International Conference on Humanoid Robots*, Dec 2008, pp. 91–98.

[20] Y. Yang, V. Ivan, and S. Vijayakumar, "Real-time motion adaptation using relative distance space representation," in *2015 International Conference on Advanced Robotics (ICAR)*, July 2015, pp. 21–27.

[21] D. Zarubin, V. Ivan *et al.*, "Hierarchical motion planning in topological representations," *Proceedings of Robotics: Science and Systems VIII*, 2012.

[22] V. Ivan and S. Vijayakumar, "Space-time area coverage control for robot motion synthesis," in *2015 International Conference on Advanced Robotics (ICAR)*, July 2015, pp. 207–212.

[23] S. Javdani, H. Admoni *et al.*, "Shared autonomy via hindsight optimization for teleoperation and teaming," *The International Journal of Robotics Research*, vol. 37, no. 7, pp. 717–742, 2018. [Online]. Available: https://doi.org/10.1177/0278364918776060

[24] F. Rydn and H. J. Chizeck, "Forbidden-region virtual fixtures from streaming point clouds: Remotely touching and protecting a beating heart," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2012, pp. 3308–3313.

[25] F. Kanehiro, F. Lamiraux *et al.*, "A local collision avoidance method for non-strictly convex polyhedra," *Proceedings of robotics: science and systems IV*, 2008.

[26] M. Ballou, "Shotcrete rebound – how much is enough?" *Shotcrete magazine, American Shotcrete Association*, 2003.

[27] M. Honegger and A. Codourey, "Redundancy resolution of a cartesian space operated heavy industrial manipulator," in *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*, vol. 3, May 1998, pp. 2094–2098 vol.3.

[28] V. Ivan, Y. Yang *et al.*, *EXOTica: An Extensible Optimization Toolset for Prototyping and Benchmarking Motion Planning and Control*. Cham: Springer International Publishing, 2019, pp. 211–240. [Online]. Available: https://doi.org/10.1007/978-3-319-91590-6_7

[29] P. E. Gill, W. Murray, and M. A. Saunders, "Snopt: An sqp algorithm for large-scale constrained optimization," *SIAM J. on Optimization*, vol. 12, no. 4, pp. 979–1006, Apr. 2002. [Online]. Available: http://dx.doi.org/10.1137/S1052623499350013