Baseline Policy Adapting and Abstraction of Shared Autonomy for High-Level Robot Operations

Ehsan Yousefi, Mo Chen, and Inna Sharf, *Member, IEEE*

***Abstract*—This article presents a novel shared autonomy and baseline policy adapting framework for human–robot interactions in high-level context-aware robotic tasks. With a unique methodology that leverages hierarchies in decision-making as well as variational analysis of human policy, we propose a mathematical model of shared autonomy policy. The framework aims at interpretable high-level decision-making for efficient robot operation with human in the loop. We modeled the decision-making process using hierarchical Markov decision processes in an algorithm we called *policy adapting*, where the autonomous system policy is adapted, and hence shaped by incorporating design variables contextual to the robot, human, task, and pretraining. By integrating deep reinforcement learning within a multiagent hierarchical context, we present an end-to-end algorithm to train a baseline policy designed for shared autonomy. We showcase the effectiveness of our framework, and particularly the interplay between different design elements and human's skill level, in a pilot study with a human user in a simulated sequence of high-level pick-and-place tasks. The proposed framework advances the state of the art in shared autonomy for robotic tasks, but can also be applied to other domains of autonomous operation.**

***Index Terms*—Shared autonomy, policy adapting, human–robot interactions, hierarchical markov decision processes, deep reinforcement learning, variational auto-encoder, robot planning, human-in-the-loop systems, autonomous systems.**

I. Introduction

A. Background and Motivation

SHARED autonomy is a framework to enable humans and robots to interact in a shared manner in order to accomplish certain goals. Shared autonomy has been utilized in a wide range of applications, from autonomous driving [1] to assistive robots [2], in order to extend and enhance human capabilities [3]. Indeed, the wide range of its applications attests to the importance of efficient human–robot interactions, as well as the current state of coexistence between humans and increasingly more intelligent robots. Our interest in shared autonomy is motivated by its potential applications in the context of articulated robots/machines, which are commonly operated by a human operator physically located in the machine. Operating such a machine involves multiple levels of hierarchy in the operator's decision-making. These encompass the high-level strategical decision-making, such as path planning for the machine, all the way to the low-level decision-making related to individual joint control for arm manipulation. In essence, this hierarchy is comparable to human decision-making when driving a car [4]. In addition, articulated machines, such as excavators and feller-bunchers, require extensive domain-specific expertise, which is why becoming a highly skilled operator can take years [5].

One of the main challenges in the multilevel, real-world field robotic applications is that complete knowledge of the relevant application domain needed to achieve a fully autonomous system cannot be assumed. However, this should not prevent us from incorporating whatever knowledge we have into a versatile framework. Moreover, the type of applications targeted here, involving large, extremely powerful machines, does not allow for hazardous trial-and-error experimentation in the field. This motivates the central research question addressed in this work, that of how to design a baseline shared autonomy architecture that allows different levels of autonomy in a human-in-the-loop (HITL) framework for hierarchical robotic decision-making tasks.

B. Related Work

In shared autonomy, *arbitration* of human and autonomous action commands, which jointly form the input to the robot/machine system, is of prominent importance and our primary focus in this work. In this regard, the available schemes in literature can be categorized into two main groups. The first is referred to as *policy blending*, where the human action and autonomous action are treated as two separate signals, and an *arbitration function* is used to decide how to blend these two signals [2], [6]. Despite wide application due to its simplicity and efficacy, the main limitation is the inherent "predict-then-go" nature of the system architecture to implement the policy blending approach [7].

The second group of methods is what we will classify as *policy adapting*, where the autonomous policy (and its action) is

Received 18 November 2024; revised 10 April 2025; accepted 26 June 2025. Date of publication 11 July 2025; date of current version 30 July 2025. This work was supported in part by the National Sciences and Engineering Research Council (NSERC) Canadian Robotics Network (NCRN) and in part by MITACS Accelerate Award. This article was recommended for publication by Associate Editor N. Bezzo and Editor J. Kober upon evaluation of the reviewers' comments. *(Corresponding author: Ehsan Yousefi.)*

This work involved human subjects or animals in its research. Approval of all ethical and experimental procedures and protocols was granted by (Research Ethics Board 1 (REB-1), Research Ethics Board Office, McGill University under Application No. 21-07-07, and performed in accordance with the requirements of the McGill University Policy on the Ethical Conduct of Research In volving Human Participants and the Tri-Council Policy Statement: Ethical Conduct for Research Involving Humans.

Ehsan Yousefi and Inna Sharf are with the Department of Mechanical Engineering, McGill University, Montreal, QC H3A 0G4, Canada (e-mail: ehsan.yousefi@mail.mcgill.ca; inna.sharf@mcgill.ca).

Mo Chen is with the School of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada (e-mail: mochen@cs.sfu.ca).

Digital Object Identifier 10.1109/TRO.2025.3588455

adapted, and hence shaped by taking into account human action, as well as other available information, and it is the *only* input to the robot. In other words, unlike policy blending, where the agents' inputs (i.e., human and autonomous) are combined in parallel, in policy adapting, they are in series. This approach does not suffer from the abovediscussed drawback of policy blending; however, it is computationally expensive and users report less comfort using it, despite having better performance in certain scenarios [7], [8]. One of the strategies in this group of schemes involves conditioning the autonomous action on the human signal. Javdani et al. [7] defined an augmented state consisting of the state of the robot and human's (partially observable) goal. They introduced a maximum entropy inverse optimal control framework that the autonomy acts based on the augmented state assuming modeled and known human policy. Reddy et al. [8] developed a deep reinforcement learning (RL) algorithm to learn a model-free policy that maps the augmented state of the robot to the (autonomous) action. The purpose in [8] was to find an optimal autonomous action close to the human action to deliver high performance, while keeping the human as a high-quality input source in the loop.

Research on how to define the *state* in shared autonomy is quite extensive and is also application-specific, as illustrated by the work in [2], [6], and [8]. It could be argued that there is neither a unique formulation nor a methodology to define the state, as there is no unique way of performing the same task. We will demonstrate the significance of this choice further in this work. In tasks involving pick-and-place operations, using information about the goal space in defining the state has been shown a good choice [2]. Moreover, there is substantial literature on conceptualizing the human aspect in the context of shared autonomy, whether through explicit model assumptions (for example, [6]) or model-free approaches (for example, [8]). In the literature to date, inferred human's belief over the goals of the specific task is often used as one of the human internal states [9], [10], [11], [12], [13], even though inference of specific goals may not always be feasible.

C. Case Study Application

To demonstrate the effectiveness of our proposed methodologies, we look at an abstract implementation of shared autonomy decision-making in operating a feller-buncher machine. These machines are utilized in timber harvesting, and their operation involves multiple levels of hierarchy in the operator's decision-making [5], [14]. The productivity of the operations is significantly affected by human performance [15], and thus motivating the case for autonomy in these machines. We suggest that shared autonomy can provide the way forward to address both the issue of productivity and operator training. Our focus in this article is on the high-level abstract decision-making.

D. Contributions and Organization

This work introduces a novel shared autonomy and baseline policy adapting framework for human–robot interactions. The framework is particularly useful for applications, which rely heavily on a skilled human to operate the robot/machine, when the operations involve a hierarchy of decision-making, and where safety is important. With this perspective, the main contributions of this article are as follows.

1) Proposition of a design strategy for a shared autonomy architecture that allows variable levels of autonomy in an HITL setup for hierarchical, robot planning, and decision-making tasks, with human interpretability and shared control in mind (see Section II).
2) Proposition of a novel mathematical model of human–robot interaction and policy adapting designed for shared autonomy and hierarchical robot planning (see Section III).
3) Formulation of Markov decision processes (MDPs) and Options framework to enable deep RL for shared autonomy, and its novel application to a case study of task-oriented robot planning (see Section V).
4) Design and results of HITL experiments to investigate the effects of architecture design and its core elements on training as well as model performance (see Sections VI-B–IX).

For a successful deployment of a policy in real-life, especially in shared autonomy for safety-critical applications, one important stage is a prerequisite: a functional baseline policy designed for shared autonomy with generalizable and tunable design elements and variables. For this purpose, we first develop a general task-oriented hierarchical planning framework for the robot/machine. Our framework is from the outset designed with human and AI interactions in mind and it extends beyond the standard robotic planning techniques. A shared autonomy policy is then trained under various adversarial conditions with the goal of serving as a baseline for subsequent fine-tuning and deployment.

Our work builds on prior literature but diverges in key assumptions and methodologies. First, our approach is in line with the body of work that does not assume direct/explicit access to human goals, either initially or subsequently (see, for example, [6] and [8]). Moreover, similarly to [8], we neither assume knowledge of the environment, nor human behavior dynamics. However, to mitigate performance issues reported in [8], we assume that the goal space is represented, which provides the necessary structure to the problem. We share the perspective discussed in, for example, [7] and [8] that a successful shared autonomy should deliver a near-optimal action while adhering to human intent. However, one of the key differentiators of our work is the human input condition: we explicitly address scenarios involving noncooperative human agents who provide intermittent or noisy inputs. Our method is therefore designed to robustly handle real-world conditions where human input is unpredictable. Another differentiator of our approach is that we target high-level hierarchical decision-making. By incorporating human high-level latent states into the policy adaptation, our system integrates richer contextual information. This allows us to tackle complex planning for robotic tasks, rather than refining low-level control.

It is important to highlight that in this work, the agents—human and autonomous— jointly make decisions for one entity, the robot/machine. This bears similarities to autonomous

driving scenarios [16], and is unlike many other human–robot interaction scenarios where the two agents act on/through two separate entities [17], [18]. Moreover, it should be noted that our framework is not necessarily a "human-knowledge-based" method [19].

This rest of this article is organized as follows. In Section II, we discuss the foundational concepts of our framework design, followed by problem statement in Section III and relevant methods in Section IV. Then, in Section V, we discuss the case study application with details. In Section VI, we present our design of experiment. Then, in Sections VII–IX, we present our results. Finally, Section X concludes this article by reiterating the main ideas presented and suggesting directions for future work.

II. Conceptual Framework

We consider the problem of designing a shared autonomy architecture for robot planning. More specifically, we focus on designing a baseline hierarchical structure for shared autonomy in complex robotic tasks in a generalizable and tunable formulation. There are three important aspects in designing such a system: 1) hierarchy in robot planning, 2) hierarchy in *shared* autonomy, and 3) multiagent aspect of human–robot interactions in shared autonomy.

For robots to achieve higher levels of autonomy, the planning must occur at the *situational understanding and task-planning* level, what is referred to as *cognitive autonomy* [20]. We, therefore, consider the robot planning problem in terms of tasks. In our work, a task is an abstract goal that an agent needs to achieve in its environment, and we adopt a task-oriented formulation of robot planning. A task-oriented approach has been shown to lead to effective interactions in human–robot interaction literature (see, for example, [21]). As will be demonstrated in Section VIII, this approach is also advantageous from the framework implementation perspective.

A. Hierarchy in Robot Planning

The conceptual representation of our task-oriented robot planning is hierarchical. The overarching policy, i.e., the master policy, is decomposed spatiotemporally into high-level (HL) and low-level (LL) policies. HL policies provide abstract, cognitive, and task-related options (e.g., selecting a goal object in a pick-and-place operation), whereas LL policies handle specific tasks, such as point-to-point trajectory planning. We draw an analogy between an *option* in the Options framework and a task in our robot planning: just as a task may involve multiple subtasks, an option comprises multiple actions and extends the MDP to cover overlapping action time scales [22], [23].

In this article, our focus is on the planning problem at the high-level tasks ($\pi_{\text{RP}}^{\text{HL}}$) within an HITL shared autonomy framework. It involves high-level tasks and policies in which interaction with a human is critical, beyond the lower level arm manipulations. As we will see throughout this article, analysis of the high-level policies for shared autonomy is complex and thus warrants this one-pointed focus. Note that for simplicity of notation, we will use π_{RP} throughout this article to refer to $\pi_{\text{RP}}^{\text{HL}}$.

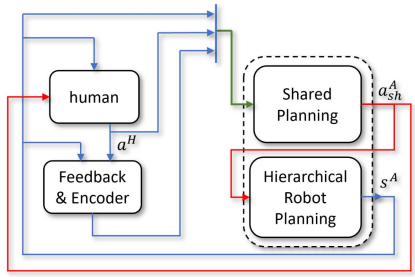

Fig. 1. Representation of hierarchy in human–robot interactions in our shared autonomy architecture and how it relates to hierarchy in robot planning. s^A denotes the designed autonomous state defined in (1), and a^H and a^A_{sh} denote action of human agent and that of autonomous agent under shared policy, respectively.

B. Hierarchy in Shared Autonomy

The task-oriented approach makes it possible to accommodate the inherent hierarchy of tasks of robot planning, and consequently, the hierarchy in human–robot interactions [4]. This is in part related to the *arbitration* issue in human–robot interactions, as discussed in Section I. Moreover, it has been shown that the behavior of a human agent operating an articulated machine/robot can be described by a *well-structured sequence* of repetitive (sub)tasks [5]. We, therefore, take into account the *spatiotemporal* aspects of a *shared* policy in providing the abstraction of shared autonomy.

As illustrated in Fig. 1, we, therefore, build our framework on two intertwined aspects of hierarchy: 1) hierarchy in robot planning and 2) hierarchy in shared autonomy.

C. Multiagent Aspect of Shared Autonomy and Decision-Making Model

We leverage the task-oriented approach and the two aforementioned hierarchies to interface the human and the autonomous agents in the context of shared autonomy, as well as to allow for sliding levels of autonomy. Our view of the overall system involving a human and an autonomous agent is that of a multiagent system with gamified interactions. From a graphical probabilistic model point of view, the policy structure can be depicted, as in Fig. 2. In this figure, s^{RP} is the state of the robotic system, and $o^{\text{RP},H}$ and $o^{\text{RP},A}$ denote option/action of the human and autonomous agents in the robot planning context, respectively. The flow of information to/from a human agent is shown with blue arrows. The dashed lines are related to the human analysis process that will be discussed in Section IV-A. Information gained through the pretraining enters the model via the green arrows.

We next introduce our shared autonomy design variables z_0, z_1, and z_2. These three categories of variables can be considered as pillars of how humans learn to perform a task: we bring in our past knowledge and experiences (z_2), fine-tune those for a particular series of tasks toward optimality, based on the task requirements (z_0), and personalize how we proceed with taking actions (z_1). In a shared autonomy context, this understanding of design variables is helpful as it allows the system to be mutually

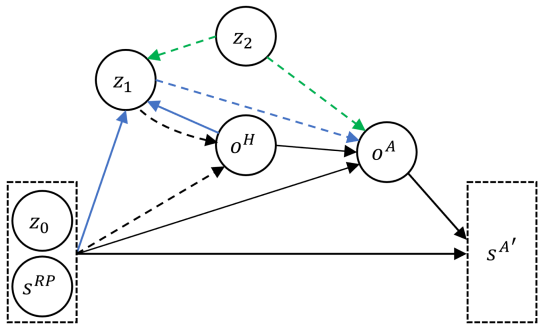

Fig. 2. Graphical probabilistic model for shared autonomy. The dashed black arrows are related to cVAE and human analysis discussed in Section IV-A. Green (dashed) arrows are related to including information from pretrained model (z_2) in shared autonomy operation. Blue arrows designate the flow of signals to human and our inferred internal state variable, z_1. As noted in (1), the autonomous state s^A is comprised of task-related (z_0) as well as robot planning-related (s^{RP}) elements. Note that o^H and o^A denote the option (or action) of the human and autonomous agents, respectively.

understandable to both the human and the autonomous agents. Our proposed shared autonomy framework can, therefore, be considered as an instance of computational human–robot interaction [24], [25]. In this regard, based on our analysis of prior work in Section I regarding the design elements in a shared autonomy setting, and in-line with our earlier work [26], we unify the existing literature and define the three categories of design variables for shared autonomy as follows.

1) z_0: Task/operation specific state variables, representing the domain knowledge, encoding the *task/function* specific state variables. It augments the classic robot-related states as follows:

$$s^A \triangleq s^{\text{RP}} \vee (s^{\text{RP}} \cup z_0) \tag{1}$$

where s^A is the *designed* (autonomous) state. For example, in Section V, z_0 will be associated with the goal space of possible task-related targets within the robot reach. One advantage of our framework is that z_0 is defined for shared autonomy by design, which makes the robot operation interpretable as well as efficient. The mutual interpretability attribute is particularly important for tasks involving a human agent in the loop and for those with limited domain knowledge.

2) z_1: Human-related (internal) states that can encode information about the HITL. As will be discussed extensively in Section IV-A, this includes abstract variables that can, for example, encode human's history of actions as well as explicit descriptive measures, such as trust and comfort.

3) z_2: Pretraining state variables to ensure that a shared autonomy architecture is built on a functional, autonomous foundation to serve as a starting point for a shared autonomy policy structure. In our framework, we refer to pretraining as the process of training a fully autonomous agent using a model similar to Fig. 2. This is the first and necessary step in the shared autonomy design, ensuring that the state definition and the model are capable of performing a task autonomously. As well, the pretrained model will allow us to look into the human's internal state (z_1), their task-related internal fine-tuning, and/or their noise through the lens of a structured task.

III. Problem Statement

We now present the mathematical model of our shared autonomy architecture in a compact form. We first introduce a central proposition that contains the core of our novel formulation of human and autonomous policies, their interdependence, and our model of shared autonomy.

Proposition 3.1: Let π_H and π_A be human and autonomous policies, respectively, at any time-step t, and s_t^A, a_t^H, and a_t^A denote the defined state, the human action, and autonomous action, respectively. Let T denote the time horizon for the task at hand. Given a typical trajectory of sequential state–actions in the context of human–robot interaction, τ, represented as

$$\tau = \{s_t^A, a_t^H, a_t^A, \ldots, s_T^A, a_T^H, a_T^A\} \tag{2}$$

the probability distribution of a trajectory τ is given by

$$p(\tau) = p(s_1^A) \prod_{t=1}^{T} \pi_H(a_t^H|\overline{s}_t^A) \times \pi_A(a_t^A|a_t^H, s_t^A) p(s_{t+1}^A|s_t^A, a_t^A) \tag{3}$$

where we introduced the state variable $\overline{s}_t^A = \{s_t^A, \ldots, s_{t-n_h}^A\}$ assembling the current and n_h steps of history of the state trajectory, with $n_h \leq t$. (See Appendix A for proof.)

It is noted that the actions in Proposition 3.1 can be extended to options, as needed. In this model, we do not impose a Markovian constraint on the human action, and thus include a history of state–actions in the human policy. The problems we tackle in this article are: 1) how to formulate and obtain the human-conditioned policy of the autonomous agent π_A, and 2) how to setup a baseline shared autonomy policy update. These should be accomplished within the constraints and necessities of the conceptual framework introduced in Section II.

IV. Methods

In this section, we first discuss how to model and provide a formulation of the autonomous policy introduced in (3). Then, we discuss how this methodology incorporates the human agent and their policy. Finally, we setup our baseline shared autonomy policy.

To model the autonomous policy π_A, we incorporate a human-related latent variable z_1. The policy can be expressed as

$$\pi_A(a_t^A \mid a_t^H, s_t^A) = \int_{z_1} \pi_A(a_t^A \mid a_t^H, s_t^A, z_{1,t}) p(z_{1,t} \mid a_t^H, s_t^A) dz_1. \tag{4}$$

This formulation utilizes the second term in (4) as a variational analysis of the human policy, incorporating the human-related latent variable z_1. By employing this methodology, we facilitate a nuanced analysis of the human policy through the lens of variational inference.

Algorithm 1: Algorithm for Baseline Policy Adapting Shared Autonomy.

Data: pre-trained autonomous policy π_0^A, human state-action trajectory data size L_H
Output: Updated shared autonomy policy π_{sh}^A

`baseline training`(π_0^A, L_H):
  $t_H \leftarrow 0$
  while *step* $t_H \leq L_H$ **do**
    $\mathcal{D}_H \leftarrow \mathcal{D}_H \cup (s_{t_H}^H, a_{t_H}^H)$ `// collect the HITL dataset` $\mathcal{D}_H$
  end
  `cVAE`($\mathcal{D}_H$, π_0^A):
    Initialize parameters θ_H, ϕ_H for decoder and encoder respectively
    while *not converged* **do**
      foreach *datapoint* i **do**
        Encode inputs $\overline{e}_i^H, \overline{a}_i^H, \overline{s}_i^A$ into latent space z_1 using encoder $q_{\phi_H}(z_1|\overline{e}_i^H, \overline{a}_i^H, \overline{s}_i^A)$
        Sample z_1 from the latent space
        Decode z_1 to reconstruct inputs using decoder $p_{\theta_H}(\overline{e}_i^H, \overline{a}_i^H|\overline{s}_i^A, z_1)$
        Calculate the cost function $\mathcal{L}_{i,H}$ for datapoint i:

$$\mathcal{L}_{i,H} = \mathbb{E}_{z_1 \sim q_{\phi_H}(z_1|\overline{e}_i^H, \overline{a}_i^H, \overline{s}_i^A)}[\log p_{\theta_H}(\overline{e}_i^H, \overline{a}_i^H|\overline{s}_i^A, z_1)] - D_{KL}(q_{\phi_H}(z_1|\overline{e}_i^H, \overline{a}_i^H, \overline{s}_i^A) \parallel p(z_1))$$

        Update θ_H, ϕ_H to minimize $\mathcal{L}_{i,H}$
      end
    end
  `Train`(π_0^A, $\mathcal{D}_H$, q_{ϕ_H}):
    Initialize policy π_{sh}^A parameters θ, value function parameters ϕ
    while *not converged* **do**
      for *actor* $= 1, 2, \ldots, N$ **do**
        Observe initial robot state s_t
        for $t = 1, 2, \ldots, T$ **do**
          Query a_t^H from D_H
          Query $z_{2,t} = a^*$ from π_0
          Sample $z_{1,t}$ from q_{Φ_H}
          Form HITL autonomous state using a_t^H, $z_{1,t}$, and s_t discussed in (4)
          Sample action a_t^A from $\pi_A(a_t^A|a_t^H, z_{1,t}, s_t; \theta)$
          Execute action a_t^A and observe reward R_t (discussed in (6)) and new state
        end
        Compute advantage estimates $\hat{A}_t$
      end
      Update policy π_{sh}^A by maximizing PPO objective:

$$\theta \leftarrow \arg\max_\theta \frac{1}{NT} \sum_{n=1}^{N} \sum_{t=1}^{T} \min\left(\frac{\pi_A(a_t^A|.;\theta)}{\pi_A(a_t^A|.;\theta_{\text{old}})} \hat{A}_t, \text{clip}\left(\frac{\pi_A(a_t^A|.;\theta)}{\pi_A(a_t^A|.;\theta_{\text{old}})}, 1-\epsilon, 1+\epsilon \right) \hat{A}_t \right)$$

    end

Algorithm 1 shows the complete end-to-end algorithm starting from data collection to shared autonomy policy update for our framework.

A. Analysis of Human Agent and Policy π_H

In analyzing human behavior, it is important to note that, in general, assuming that the state definition is the same between the human and the autonomous agent is not valid. The reason for this lies in the fact that we do not have direct access to the internal perception and state definition of a human. Therefore, to assume that a human policy maps from s_t^A to a_t^H is conceptually inaccurate, in general. In this article, we assume that a human has an internal, latent state $z_{1,t}$; hence, $s_t^H \triangleq (\overline{s}_t^A, z_{1,t})$.

The explicit allowance for human's internal state becomes even more important when we have a hierarchy of tasks. The notion of including the human signal in the augmented state definition, as proposed in [8], makes sense in this light. If the human internal signal is defined to be the low-level actions, such as the joystick inputs, then this implicitly enforces the Markovian assumption, i.e., the history is ignored. In contrast, it can be argued that conditioning the autonomous *shared* policy π_{sh}^A on a rich signal from the human, such as the goal space and, if feasible, the intended goal, is critical, as it effectively reflects the human history of actions. This argument is also supported by the results in [8] for an unstructured user input, where raw (i.e., unconditioned) low-level human inputs were used and poor performance was reported. Consequently, the performance of a collaboration scheme highly depends on a relatively successful encoding of human's internal state variable(s) z_1. It is worth noting that goal inference algorithms, in general, are based on a history of human inputs. The goal/intent inference requires knowledge of goal space, which, in turn, requires domain knowledge. We encode the latter in z_0 without assuming direct knowledge of the intended goal, but only as a measure of goal space.

From another point of view, using the human's internal signal as input to the autonomous policy provides a mechanism to *synchronize* human actions and the resultant autonomous actions over a finite horizon that ends with reaching a goal. Otherwise, the performance of collaboration will be poor, as was the case in the results reported in [8] when low-level human input was used in the autonomous policy adapting.

Based on the above considerations, we do not assume equivalence between state definitions of the human and autonomous agents. To address this knowledge gap and to analyze the

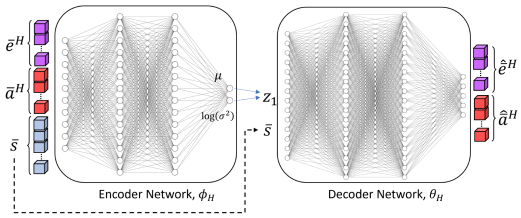

Fig. 3. cVAE architecture to encode a measure of human behavior in latent variable z_1. Inputs to encoder network Φ_H are augmented state, augmented human action, and its error with respect to surrogate optimal action denoted by $\bar{e}^H$, $\bar{a}^H$, and $\bar{s}$, respectively. Decoder θ_H outputs estimates of augmented human action and its error denoted by $\hat{\bar{a}}^H$ and $\hat{\bar{e}}^H$, respectively.

$\pi_H(a_t^H|\bar{s}_t^A)$ term in (3), we propose to explicitly encode the human's internal state variable z_1, as shown in (4). To encode z_1, we use conditional variational autoencoder (cVAE) due to its strength in providing structured latent space and its flexibility in modeling complex distributions [27]. It also facilitates robustness of the model against human *noise* levels, as will be demonstrated with results in Section VIII.

Let n_s denote the dimension of autonomous state $\overline{S} \subset \mathbb{R}^{(n_h+1)\times n_s}$. We train an encoder $\phi_H : \{\overline{E}, \overline{A}, \overline{S}\} \to \mathbf{Z}_1$, with human's latent state space $\mathbf{Z}_1 \subset \mathbb{R}^{d_{z_1}}, d_{z_1} < (n_h \times n_s)$, from $\overline{S}$ conditioned on human's n_h steps of history of actions $\overline{a}^H \subset \overline{A}$, as well as history of deviations (i.e., errors) of their actions $\overline{e}^H \subset \overline{E}$. The latter are defined with respect to the actions of a known surrogate optimal agent, which might be another human or a pretrained model. Therefore, we opt to incorporate the variable z_2 in Fig. 2 in the form of the surrogate optimal action, i.e., $z_2 := a^*$ (see details of its deployment in Algorithm 1). The deviation is defined as the normalized deviation of a human action from the surrogate optimal action $e^H = |a^H - a^*|/N$, where $|a^H - a^*|$ represents the absolute deviation between the human action and the surrogate optimal action, and N is a normalization factor that depends on the context and could, for example, represent the angular distance in case of 2-D spatial actions. Moreover, we learn a decoder $\theta_H : \mathbf{Z}_1 \times \overline{S} \to \{\overline{A}, \overline{E}\}$, with the following reward function of the optimization process in cVAE:

$$\mathcal{L}_{i,H} = E_{z_1 \sim q_{\phi_H}(z_1|\bar{e}_i^H, \bar{a}_i^H, \bar{s}_i^A)}(\log p_{\theta_H}(\bar{e}_i^H, \bar{a}_i^H|\bar{s}_i^A, z_1)) - D_{\mathrm{KL}}(q_{\phi_H}(z_1|\bar{e}_i^H, \bar{a}_i^H, \bar{s}_i^A) \parallel p(z_1)) \quad (5)$$

where ϕ_H and θ_H are the encoder and decoder networks, as shown in Fig. 3. The two terms in (5) are the reconstruction error and KL-divergence, respectively [27].

It is worth noting that pretraining is a necessary step since its policy is used as a surrogate optimal model and as a reference to encode human agent behavior. Since in this article, our focus is on providing a baseline shared autonomy policy, it is important to start with an optimally trained model to set the stage for effective human encoding, as well as for future fine-tuning of the baseline policy.

Finally, lines 6–15 in Algorithm 1 show the implementation of this step of our framework. Note that D_H denotes the collected HITL dataset of size L_H. The procedure for experiments, data collection, and further details are discussed in Sections VI and VIII-A.

B. Shared Autonomy Setup

Based on the above discussions, we set up our shared autonomy framework, as shown in Figs. 1 and 2. It is worth noting that the degree to which a human agent participates in sharing the operation of the system depends on: 1) the level of autonomy desired for the system, 2) the task itself and its requirements, and 3) the extent of human presence/engagement. Our shared autonomy setup is designed to provide a near-optimal input to the robot with respect to a reward function comprised of the following two contributions.

1) R^{RP}: Reward from robot planning, which includes rewards related to, most notably, task execution and environment accommodations.
2) R^H: Reward related to closeness to human input and its optimality.

Hence, the general form of the reward function is

$$R = c^{\mathrm{RP}} R^{\mathrm{RP}}(s^A, s^{A'}, a^A) + c^H R^H(a^H, a^*) \quad (6)$$

where $c^{\mathrm{RP}}, c^H \in \mathcal{R}^+$. Our specific choice of the reward function as defined in (6) is motivated by several considerations. Since shared autonomy is inherently about multiagent decision-making and interaction, the state of cooperation or competition between agents depends on task-related (as the overarching objective of the multiagent system) and human-related elements (analogous to individual agent's objective). Furthermore, this inherent need for compatibility is related to the gamified human–robot interaction and game theoretic aspects needed for the shared autonomy framework [28]. With the proposed reward function design in (6) and by strategic assignment of the coefficients $\mathbf{c}$, we partially resolve possible conflicts between human and autonomous agents and implicitly decide on the degree of cooperation or competition in their interactions within our novel policy adapting shared autonomy architecture. This assignment also represents how much autonomy is required, or the level of autonomy. This is a novel perspective on the problem of multiagent shared autonomy with human as an agent in the loop. The unique design of R^H can be realized in the context of reward shaping and constraint satisfaction in optimal control and RL. This specific design of the reward function measures how optimally the human agent is acting at any state.

Finally, lines 16–32 in Algorithm 1 show the implementation of this concluding step of our framework.

V. Case Study

A. Application Background

As mentioned in Section I, the motivation for our research on shared autonomy stems from its potential applications to machines employed in the timber harvesting industry.

In our previous work [14], we considered the planning problem for operating the manipulator (a.k.a., the crane) of a feller-buncher machine for cutting, grabbing, and placing trees in

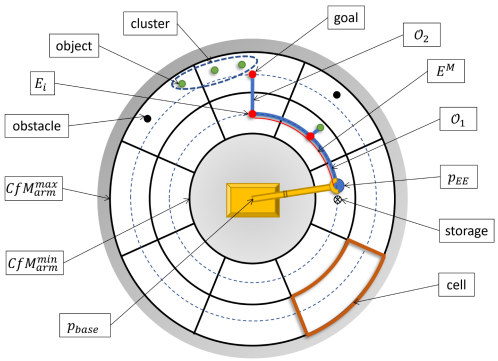

Fig. 4. Design setup schematic for robot task planning for machine working in a forest site. p_{EE} shows position of end-effector within each "cell"; "objects" are shown with green dots, which can be grouped into clusters shown in blue dashed oval; "goals" are shown with red dots; and "obstacles" are shown with black dots. Blue curves show combination of options $\mathcal{O}_1$ and $\mathcal{O}_2$ from key point E_i to another. CfM_{arm} denotes robot arm's capacity for maneuverability.

a storage location. In this context, a human-inspired planning algorithm was proposed by using a concept we called the *Envelope of Manipulation* E^M: a curve connecting *key points* E_i associated with clusters of trees, a.k.a., objects (see Fig. 4), with the key points assembled in a set $\mathcal{E}$. The envelope E^M can take any shape; however, based on our field observations of the tree-harvesting machines, it is well approximated by a circular arc. Also, based on our observations of human operators in the field, we identified two high-level options in the options space $\mathcal{O}$: 1) $O_1 \in \mathcal{O}$ that encapsulates the motion of the crane end-effector along the envelope between two key points, and 2) $O_2 \in \mathcal{O}$ that encodes the operations around each key point. The operator may group several objects into a *cluster* in order to cut and grab several trees together, before moving on. It is thus possible to encode a sequence of operations as a sequence of the two aforementioned options. The problem of robot planning, therefore, turns into optimizing this sequence of options. The overall task of robot planning includes a hierarchy of subtasks, namely, envelope options (i.e., O_1 and O_2), and moving the arm or crane along the specified trajectories [move arm (MA)]. We will discuss these further in the subsequent sections.

B. Robot Planning: Conceptual Framework

Following the hierarchy introduced in Section II, we decompose operator tasks into a hierarchical planning scheme: the high-level policy $\pi_{\mathcal{E}}$ (an instance of $\pi_{\text{RP}}^{\text{HL}}$) encapsulates envelope of manipulation (E^M) actions, and the low-level policy π_{MA} (an instance of $\pi_{\text{RP}}^{\text{LL}}$) handles arm movement. As noted earlier, in the current scheme of operations, a human operator uses the crane to manipulate objects (e.g., cut, grab, and deposit) in the operational region of a specific base location.

1) $\pi_{\mathcal{E}}$—Policy for Envelope of Manipulation (E^M): The definition of this level in our hierarchy shares the structure with our previous work [14]. However, here, we reformulate the planning into a generalizable framework. As our focus in this article is on high-level policies, for simplicity of notation, we will refer to this policy as π_{RP}. As we will discuss extensively in Section V-C, it is at this level that high-level planning and human–robot interaction happens.

2) π_{MA}—Policy to MA: This is the lowest-level policy in our hierarchy, and it directly interacts with the environment. This policy takes the destination key point as its goal, plans a smooth trajectory for the end-effector to reach it, and executes the motion of the arm. Standard robotic tools can be employed to execute these subtasks. We assume access to adequate policies for trajectory planning, as well as for trajectory execution.

C. Model of High-Level Robot Planning and Decision-Making

To find the optimal policy for robot planning, π_{RP}^*, we build on our previous proof-of-concept formulation in [14]. For that purpose, we construct the more general and versatile, compared to [14], MDP framework as described next.

Environment or The World: We have tailored a grid world to our specific robotic application, for a generalizable MDP backbone. As schematically shown in Fig. 4, this world is comprised of $n_c \times n_r$ cells, circumferentially arranged around the robot base; n_c and n_r are the circular and radial dimensions of the grid, respectively, and these are chosen to accommodate the desired resolution. An important element of environment definition, which directly affects π_{RP}, is how to handle the objects surrounding the robot. These objects, depending on the state of the system, can be either obstacles or subgoals, and this categorization changes *dynamically*, which are encoded by the task-related state variable z_0 (see Fig. 2). This makes for an important novelty of our work, which ties situational awareness and environment uncertainty into task-oriented hierarchical planning of robot operation. We next discuss how we utilize this formulation (see Section VII for further details). We first construct the relevant *Spaces* as follows.

1) *Objects space $\mathcal{S}_{oj}$:* includes all objects.
2) *Subgoals space $\mathcal{S}_{sg}$:* a subset of $\mathcal{S}_{\text{oj}}$ and it includes all objects accessible to the robot (and hence, not blocked by other objects from robot's reach). The augmented subgoal space, $\hat{\mathcal{S}}_{\text{sg}}$, is constructed by adding the storage point, as conditioned on CfM_{ee} with the following logic:
   a) if $\text{CfM}_{\text{ee}} = 0$, $\hat{\mathcal{S}}_{\text{sg}} = \{E_n\}$;
   b) if $0 < \text{CfM}_{\text{ee}} < 1$, $\hat{\mathcal{S}}_{\text{sg}} = \{\mathcal{S}_{\text{sg}}, E_n\}$;
   c) if $\text{CfM}_{\text{ee}} = 1$, $\hat{\mathcal{S}}_{\text{sg}} = \{\mathcal{S}_{\text{sg}}\}$;
   where capacity for maneuverability ($\text{CfM}_{\text{ee}} \in [0, 1]$) is the remaining holding capacity in the end-effector for a human intervention [29].
3) *Obstacles space $\mathcal{S}_{ot}$:* defined by negating the augmented subgoal space from the objects space. Therefore, the three object spaces are related through $\mathcal{S}_{\text{oj}} = \mathcal{S}_{\text{sg}} \cup \mathcal{S}_{\text{ot}}$.

Constraints: In general, the workspace of the robot is limited by 1) boundaries of the grid, which are in turn defined by the minimum and maximum values of the reachability of the robot, CfM_{arm}, and 2) obstacles located next to the end-effector as well as those obstructing its arm movement. This environment, therefore, includes the following constraints built-in and dynamically updated.

1) Robot workspace and related constraints, such as the capacity for maneuverability of the arm and the end-effector,

i.e., CfM_{arm} and CfM_{ee}, respectively. Other constraints, for example, stability related constraints, can also be accommodated.

2) Path planning-related constraints, such as obstacles.

With the definition of the world, the categorization of object spaces, and the constraints, we are now ready to define the main MDP elements, which in turn encode the path planning problem with obstacle avoidance.

State (or Observation) Space: Following our design variables introduced in Section II and shown in Fig. 2, the observation space is a discrete space with two types of observations.

1) *Robot-related variables:*
   a) s_1^{RP}: discrete 2-D position of the robot end-effector, (angular position and radial position), in the range $0, \ldots, n_c - 1$ and $0, \ldots, n_r - 1$, respectively,
   b) s_2^{RP}: payload indicator $0, \ldots, p_{\max}$; it is related to CfM_{ee} through $\text{CfM}_{\text{ee}} = (p_{\max} - s_2^{\text{RP}})/p_{\max}$, where $p_{\max}$ is the maximum number of trees/objects that the end-effector is able to carry.
2) *Task-related variable:*
   a) z_0: contains the circular distance from the current cluster/key point to all subgoals with respect to the robot end-effector in the CCW direction. If no subgoal is present at a location, -1 is returned. Therefore, z_0 effectively augments the state of the robot (comprised of s_1^{RP} and s_2^{RP}) with the information related to the subgoals space.

Therefore, we denote the (augmented) state in this level with s^A defined as follows (see Figs. 1 and 2):

$$s^A \triangleq (s_1^{\text{RP}},\, s_2^{\text{RP}},\, z_0). \tag{7}$$

Together with the objects, subgoals, and obstacles spaces defined above, a state s^A encodes three features, depending on the following scenario.

1) Obstacle, if $s^A \in \mathcal{S}_{\text{ot}}$.
2) Subgoal, if $s^A \in \hat{\mathcal{S}}_{\text{sg}}$. If the agent is *done* with the operation overall, reaching the storage point is the *final goal* and the episode is *done*.
3) Normal, otherwise.

Action Space: The action space is discrete, consisting of four actions: left, right, front, and back, encoded by 0–3, respectively. Note that the directions of these actions are defined with respect to each cell, and not in an absolute sense.

Rewards: Rewards are defined as follows.

$R_1^{\text{RP}} = -2$: All transitions except the transition to the "subgoal" or "goal" state.
$R_2^{\text{RP}} = 20n_{\text{cut}}$ or $20n_{\text{store}}$: Transition to one of the "subgoal" states for the cases of cutting or storing.
$R_3^{\text{RP}} = 400$: Transition to the "goal" state. This ends an episode and resets the environment.
$R_4^{\text{RP}} = -20$: Collision with an obstacle.
$R_5^{\text{RP}} = -20$: Out of boundaries action.
$R_6^{\text{RP}} = -5\, s_2^{\text{RP}}$: Cost of carrying an object.
$R_7^{\text{RP}} = -400$: Getting trapped. This also ends an episode and resets the environment.

In R_2^{RP}, n_{cut} and n_{store} denote the number of objects cut/picked and stored, respectively. Note that the combined value of the above reward elements form R^{RP} in (6).

Policy: With the architecture shown in Fig. 2, we denote the policy for this high-level decision-making for robot planning with $\pi_{\text{RP}} = \pi_{\text{RP}}(o^A|s^A)$.

D. Model of Shared Autonomy Decision-Making

As shown in Fig. 1, we define the *shared planning* as the highest level in the hierarchy structure and model it as an instance of MDP with the following elements.

Environment or The World: We designed a higher level environment for the shared autonomy for a generalizable MDP backbone, the attributes of which are defined next.

State Space: This is defined based on the state space of the MDP^{RP} introduced in Section V-C, expanding it to include z_1 and a^H. Note that a^H is recorded as -1 if no action is taken by the human agent to accommodate such instances.

Action Space: This is defined similarly to that of MDP^{RP}.

Policy: The policy π_{sh}^A considering the model presented in Fig. 2 and discussions in Section IV.

Rewards: Rewards, as discussed earlier, take the form of $\mathbf{c}^\top \mathbf{R}$ in (6), where $\mathbf{c} : -(c^{RP}, c^H)$.

From another perspective, since the planning model and formulation are human-inspired by design, with a shared autonomy mindset, the autonomous actions are comprehensible to the human agent and vice versa. Hence, this shared mental model [13], [30] is not only necessary for proper collaboration, but more importantly provides a road map to the design of any modern framework involving humans and (semi)autonomous agents.

VI. Design of Experiment

A. Experiment Design Objectives

It should be highlighted that the purpose of this study and the experiments within is to validate a set of specific hypotheses about the performance of baseline policy adapting for shared autonomy, prior to carrying out a full-fledged user study. We, therefore, aim to understand the effects of key elements of our design formulation in training performance as well as model behavior. In particular, we are concerned with: 1) whether or not a shared autonomy agent can be trained under the discussed uncertainties and subtleties given the proposed MDP structure, 2) to what extent the human-tuned variable z_1 affects the training process, 3) how the reward terms and their weights in (6) affect the training process, and 4) how a trained model under different conditions performs when deployed with humans of different expertise or noise levels. In all cases, if no human action is available, the human agent is considered to be *noncollaborative*. No action is an action itself. Moreover, the human actions may not be optimal. This adds another layer of complexity and uncertainty to the problem.

B. Experiments Details and Ablation Study

In order to study the abovementioned effects, we conduct a total of eight experiments, details of which are given in Table I.

TABLE I
DETAILS OF ABLATION STUDY AND EXPERIMENTS

#	Exp. ID	data	z_1 type	c^{RP}, c^H
1	EI-1	D^E	z_1^E	$c^{\text{RP}} < c^H$
2	EI-2	D^E	z_1^E	$c^{\text{RP}} > c^H$
3	EII-1	D^E	z_1^N	$c^{\text{RP}} < c^H$
4	EII-2	D^E	z_1^N	$c^{\text{RP}} > c^H$
5	NI-1	D^N	z_1^E	$c^{\text{RP}} < c^H$
6	NI-2	D^N	z_1^E	$c^{\text{RP}} > c^H$
7	NII-1	D^N	z_1^N	$c^{\text{RP}} < c^H$
8	NII-2	D^N	z_1^N	$c^{\text{RP}} > c^H$

Our first descriptor of the experiments, E or N, reflects the expertise level of the human, as follows.

1) *Expert Human:* A human agent who is familiar with our setup. The data corresponding to this expertise level are denoted by D^E.
2) *Noisy Human:* We deliberately perturb the human's action in *real time* with a probability of 0.3 to emulate a noisy human after they register their input in using the input device and before enacting in the environment. The data corresponding to this human expertise level are denoted by D^N.

The second ablation category, denoted with I or II, indicates what types of human data were employed to encode z_1. The third variation indicates the relationship between the weights c^{RP} and c^H in the reward function. These weights are considered to be constant design elements to be studied. For example, EI-1 means that the corresponding model was obtained with expert data D^E, with z_1^E encoded with that same data and with $c^H > c^{\text{RP}}$.

C. Metrics of Study

As discussed in Section VI-A, this study aims to study shared autonomy policy training performance as well as the trained model performance under various conditions. In this regard, we introduce the following measure of the training performance.

1) *Sample Processing Rate (SPR):* The number of processed samples per second taking into account the feedforward and backpropagation computations. In a shared autonomy, the speed of the platform is of crucial importance, since it needs to train an agent for a shared autonomy while the task progresses with HITL. SPR is defined as follows:

$$\text{SPR}(k) = \sum_{j=0}^{k} n_{\text{tr}}^j/(t - t_0) \tag{8}$$

where t and t_0 are the current and initial wall-time in seconds, and n_{tr}^k is the number of training samples processed at time-step k. This also includes the time required by the stochastic gradient descent. This measure enables us to quantitatively measure the effects of introducing different variables in the HITL model training process.

Moreover, for the shared autonomy policy evaluation, since we are concerned with the challenging case of shared autonomy with a nonpersistent (noisy) human agent, we look at the following measures carefully designed for shared autonomy.

1) *Similarity Ratio σ_H^A:* The ratio of similarity of the autonomous agent action, a^A, to that of the human agent, a^H, over an entire episode where the human participated, defined as follows:

$$\sigma_H^A = \frac{\sum_{t=0}^{t=T} \mathbf{1}(a_t^A = a_t^H)}{\sum_{t=0}^{t=T} \mathbf{1}(a_t^H \neq \varnothing)} \tag{9}$$

where T is the episode horizon and $\mathbf{1}(.)$ is an indicator function that evaluates to 1 when the condition inside is satisfied.
2) *Failure rate f^A:* The number of times the autonomous agent fails to complete the task in the shared autonomy setting.
3) *Return $\overline{R}^A$:* The cumulative reward R^{RP} over the entire episode. We include this measure in our evaluation, since the ultimate goal of the system is to accomplish the task as best as possible.

D. Hypotheses

Based on our experiment objectives, we lay out our hypotheses as follows.

Hypothesis 1: This hypothesis is regarding the training performance and how it is related to the decision-making model and state definition, particularly the human-tuned variable z_1. Specifically, we hypothesize that the following holds.

H1-a) A shared autonomy policy is trainable using our formulation, with random uncertainty in environment, uncertainty in human's degree of collaborativeness and expertise level, and the tuned model of z_1.
H1-b) All other conditions being equal, training with the expert human, i.e., D^E, is more efficient than training with a noisy human, i.e., D^N.
H1-c) Related to the above, we expect that the state augmented with the z_1 signal *matching* with human in terms of expertise level, e.g., z_1^E with D^E, does not degrade training efficiency.

Hypothesis 2: This hypothesis is regarding the trained models' performance based on the *shared autonomy-related* metrics defined in Section VI-C. Specifically, we hypothesize that the following holds.

H2-a) There is a correlation between human data and z_1 type pairs and, the key performance metrics.
H2-b) The selection of the coefficients of the reward function, i.e., c^H and c^{RP}, affects the performance metrics of the trained models.

VII. RESULTS FOR PRETRAINED MODEL

In this section, we showcase the training of a fully autonomous robot agent based on the formulation presented in Section V-C. This is an important step as it showcases that the task-oriented robot state definition in (7) is sufficient to train an autonomous model, given the inherent stochasticity of the environment and the challenges of the application.

In the context of our shared autonomy framework, this will represent a *pretrained* agent, to be used as the baseline for

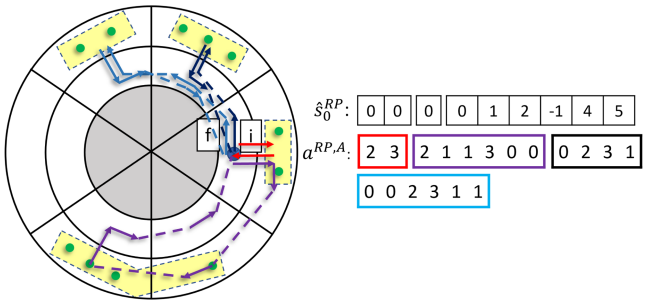

Fig. 5. Pretrained agent example for a crowded scenario. Green circles are objects (trees), yellow rectangles are clusters, blue circle is the initial agent location, solid arrows indicate actions, and dashed lines conceptualize the path executed by low-level policy (simplified to straight lines for illustration). Sequence starts at storage location "i" and continues until last action "f." CFM_{ee} is set to 4. The four subsequences end at storage location "i" (red, purple, black, and blue in that order).

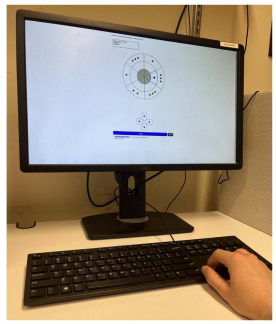

Fig. 6. Setup to gather human data in simulated environment. We use a visualization layer to show the environment to the user, who can move with four basic inputs in four directions in our robot planning introduced in Sections V-C and V-D.

computing the human agent's error. For *uncertainty-aware* generalizable results, we sample the objects in the environment from a Gaussian distribution in order to account for different possible variations of the object spaces. Specifically, for object randomization process, we draw a sample for each cell (see Fig. 4) that represents the number of objects in that cell. We assume that the objects are from a truncated Gaussian distribution in the interval $[0,\ 4]$ with the mean and standard deviation of 2 and 1 per unit circular ring around the robot base. We employ the proximal policy optimization (PPO) algorithm [31] notably with a batch size of 32, learning rate of 1×10^{-3}, and γ of 0.99 for a multilayer-perceptron policy with 2 layers of 64 nodes. We use Adam as our optimizer [32].

In Fig. 5, we provide an example of the output from the trained autonomous policy for a scenario with a relatively crowded environment configuration. This figure includes information on the initial state and the output action sequences of the policy. We observe that these are logical and intuitive.

VIII. Experimental Results for Shared Autonomy Training

In this section, we present the training results for our proposed novel shared autonomy framework. First, we present the results supporting the encoding of the human agent's internal/latent variable z_1. Then, we build up a shared autonomy platform that, at its core, is comprised of our hierarchical MDPs. In-line with our experiment design and hypotheses, we assess Hypothesis 1 and discuss the parameter selection process as we present the results.

A. Data Collection Setup Details and Procedure

We collect human user data using a simulated environment but with real human user within our proposed shared autonomy framework.[1] Fig. 6 shows our setup for a test. During a test, a human user is presented with a random initialization of the environment. We record the actual human data using similar object randomization and environment configurations as in Section VII. The four basic discrete inputs in the action space are mapped to four direction buttons on a regular keyboard. The human task is the same as that of the pretrained agent, namely, to collect all objects in the environment and place them in designated storage location, subject to the robot and environment constraints. Using these conditions, we collected two separate datasets: 1) D^E (expert): a total of 400 complete episodes with a user familiar with the setup, and 2) D^N (noisy): a total of 400 complete episodes with a user familiar with the setup but their input *perturbed* in real time as described in Section VI-B. For the training and testing purposes, we randomly divide each dataset into three folds: one fold to be used for human encoding (i.e., cVAE training), a fold for the shared autonomy policy training with the HITL data, and a fold for the analysis of model/policy performance (i.e., hold-out set).

B. Human Encoding Results

Following the formulation presented in Section IV-A, we train our cVAEs for a 15-D latent variable z_1, using two history steps ($n_h = 2$) separately for expert and noisy human data, resulting in models for z_1^E and z_1^N, respectively. These models are then utilized in training different shared autonomy models using the design of experiment presented in Table I. It is noted that we randomly divide the utilized data (i.e., human encoding fold) with a ratio of 0.7 between training and validation of encoding. Notably, the learning rate and batch size are 1×10^{-3} and 20, respectively. To compute the input error/deviation, e^H, we use the pretrained model from Section VII as the surrogate optimal model. From a practical point of view, we used one-hot transformation for our discrete variables, such as the state s^{RP}, and introduced white noise for better training. Fig. 7 shows the training and validation processes of our cVAE model for expert human data. It is worth noting that the encoding for a human is fast, taking 17 s for the expert case and 23 s for the noisy case.[2]

[1] Ethics approval obtained from Research Ethics Office, McGill University, under REB# 21-07-071.

[2] All experiments were conducted on a laptop with 12th Gen Intel Core i7-12700H, 2.30 GHz, with 16.0 GB of RAM and NVIDIA GeForce RTX-4050 Laptop GPU.

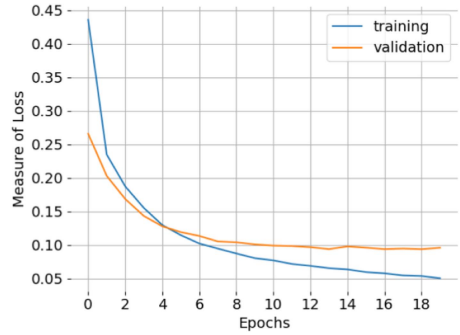

Fig. 7. Training versus validation for cVAE model for expert human data.

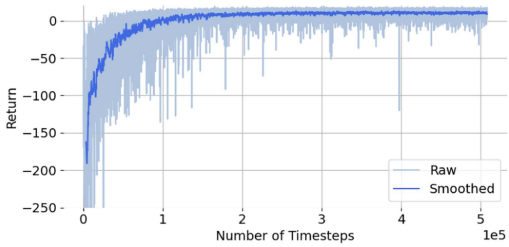

Fig. 8. Sample of shared policy training process for EI-1. Dark blue is the smoothed, average reward with a window of 50 time-steps, and light blue shows raw training rewards.

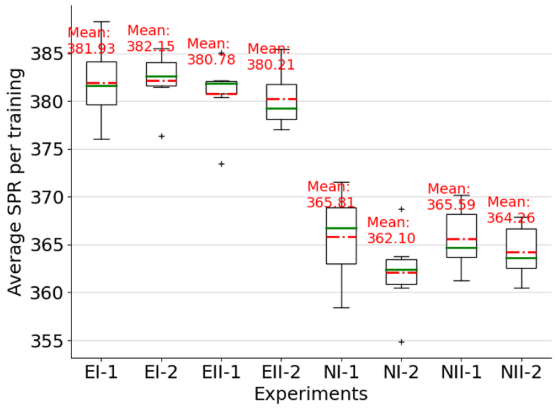

Fig. 9. SPR for each experiment.

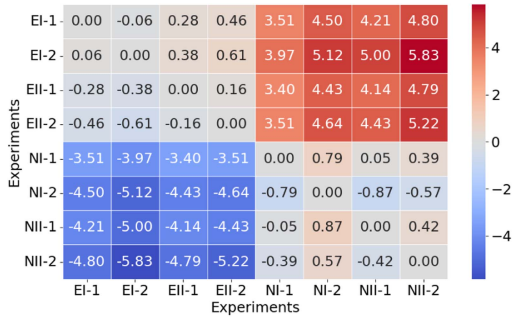

Fig. 10. Comparison of SPR values for experiments in Table I using Cohen's d corresponding to Fig. 9.

C. Shared Autonomy Policy Training Results

We train eight different models within eight designed experiments under the conditions listed in Table I. We repeat policy training for each experiment six times and report the SPR, defined in (8). To compare the results, in reporting the effect sizes, we use Cohen's d as our metric, which is the difference between two means divided by a measure of standard deviation for the data [33]. We consider 0.2, 0.5, and 0.8 as the thresholds for small, medium, and large effect sizes, respectively.

We begin the presentation of shared autonomy results by demonstrating the training process for expert human data D^E under the conditions of experiment EI-1 for $\mathbf{c} = (5, 10)$. Arguably, no HITL test can cover the complete state space, and therefore, as discussed in Section VI-A, we will treat the human agent in the unseen states as a noncooperative agent who takes no action. From Fig. 8, we observe that a shared autonomy policy can be trained using our formulation for an expert human who is alternating between being cooperative and noncooperative, with the inherent stochasticity of the environment. These results demonstrate the success of our algorithm in training a shared autonomy under the abovementioned conditions. Therefore, from the example training Fig. 8 and cumulative results reported in Fig. 9, we found support for Hypothesis H1-a in the success of our framework to train a shared autonomy policy under different conditions.

Regarding Hypothesis H1-b, Figs. 9 and 10 present the average SPR metric for the eight experiments in Table I: these clearly show that the training of models with expert HITL is more efficient than with a noisy human, thus supporting Hypothesis H1-b.

From Fig. 10, in relation to the effect of variable z_1, considering the related pairs differentiated only by z_1, i.e., (EI-1, EII-1) and (NII-1, NI-1), we report a Cohen's d of +0.28 and -0.05 indicating a small and negligible effect size, respectively. These results indicate that a matching z_1 and human data noise levels result in more efficient training only for expert HITL, while there is no significant effect in the noisy human case. Therefore, we found evidence in support of Hypothesis H1-c.

In addition, we next compare the pairs of experiments with $c^{\text{RP}} < c^H$ to those with $c^{\text{RP}} > c^H$: For the expert human, in the pairs (EI-1, EI-2) and (EII-1, EII-2), we report a Cohen's d of -0.06 and +0.16 indicating no significant effect size. For the noisy human, in the pairs (NI-1, NI-2) and (NII-1, NII-2), we report a Cohen's d of +0.79 and +0.42 indicating a large and small effect size. These results indicate that in the case of expert human, the training performance is invariant to the chosen coefficients. However, in the case of a noisy human, higher c^{RP} results in lower performance with different effect sizes. In other words, noisy human is a source of disturbance to the autonomy, although z_1^E seems to contribute positively.

IX. Shared Autonomy Policy Evaluation

Next, we discuss the models' performance for a human in the loop test scenarios. In reporting the results in this section, we group the eight experiments according to the reward weights c^H versus c^{RP} relationship and analyze them separately: Group 1 is

TABLE II
POSTHOC P-VALUES FOR SUBCASE 1-A: GROUP 1 MODELS DEPLOYED WITH (D^E, z_1^E)

metric	value	EI-1 - EII-1	EI-1 - NI-1	EI-1 - NII-1	EII-1 - NI-1	EII-1 - NII-1	NI-1 - NII-1
σ_H^A	p-value Mean	**0.041** (0.884, 0.829)	**<0.001** (0.884, 0.649)	**0.005** (0.884, 0.818)	**<0.001** (0.829, 0.649)	NS (0.829, 0.818)	**<0.001** (0.649, 0.818)
f^A	p-value Mean	**<0.001** (0.0, 0.0)	**<0.001** (0.0, 0.41)	**<0.001** (0.0, 0.21)	**<0.001** (0.0, 0.41)	**<0.001** (0.0, 0.21)	**0.006** (0.41, 0.21)
$\overline{R}^A$	p-value Mean	**<0.001** (541.5, 572.0)	**<0.001** (541.5, 357.7)	**<0.001** (541.5, 329.4)	**<0.001** (572.0, 357.7)	**<0.001** (572.0, 329.4)	NS (357.7, 329.4)

See Table I for list of experiments and Section VI-C for definition of metrics. "NS" stands for not significant.

TABLE III
POSTHOC P-VALUES FOR SUBCASE 1-B: GROUP 1 MODELS DEPLOYED WITH (D^N, z_1^N)

metric	value	EI-1 - EII-1	EI-1 - NI-1	EI-1 - NII-1	EII-1 - NI-1	EII-1 - NII-1	NI-1 - NII-1
σ_H^A	p-value Mean	**<0.001** (0.926, 0.860)	NS (0.926, 0.922)	NS (0.926, 0.911)	**<0.001** (0.860, 0.922)	**<0.001** (0.860, 0.911)	NS (0.922, 0.911)
f^A	p-value Mean	**0.014** (0.0, 0.09)	NS (0.0, 0.01)	NS (0.0, 0.0)	**0.043** (0.09, 0.01)	**0.014** (0.09, 0.0)	NS (0.01, 0.0)
$\overline{R}^A$	p-value Mean	NS (509.0, 458.1)	NS (509.0, 502.0)	NS (509.0, 512.9)	NS (458.1, 502.0)	NS (458.1, 512.9)	NS (502.0, 512.9)

comprised of experiments, where $c^H > c^{\text{RP}}$, that is, EI-1, EII-1, NI-1, and NII-1. Group 2 is comprised of experiments EI-2, EII-2, NI-2, and NII-2, where $c^H < c^{\text{RP}}$. For better readability, we report the results under two cases: Case 1 for Group 1 and Case 2 for Group 2. In each case, we interface the four models in their associated group to two human data and cVAE model pairs: 1) (D^E, z_1^E), and 2) (D^N, z_1^N). Therefore, in total, we present four sets of results. It is also noted that we test each model for 100 full episodes using the hold-out dataset, of each dataset type. Procedurally, in each set, we first report repeated measures ANOVA results for the three performance metrics introduced in Section VI-C, and in case of statistical significance, we then report posthoc analysis with Holm corrections for the important pairs of our experiments. We use 0.05 as the statistical significance threshold. Beyond the statistical analysis, we discuss the key findings and evaluation of Hypothesis 2, focusing on the results of most significance.

A. Case 1: Group 1 With Expert or Noisy Human

Subcase 1-a:

In this subcase, the models in Group 1 are deployed with the expert HITL and human encoding based on expert data, that is, the expert pair (D^E, z_1^E). Repeated measures ANOVA shows $p < 0.001$ for all metrics, indicating statistical significance. Results of posthoc analysis for this subcase are presented in Table II . In this regard, we can make the following observations.

Looking at Column 3 of Table II (color-coded with light blue), i.e., for the pair (EI-1, EII-1), we observe that EI-1 shows higher σ_H^A (similarity ratio) metric. Both EI-1 and EII-1 show equally perfect performance for the metric f^A (failure rate). However, we observe that EII-1 shows higher $\overline{R}^A$ (return). These results show that the system is able to attune to the (albeit slight) relative imperfection in expert human behavior. In Column 5 of the same table (color-coded with green), i.e., for the pair (NI-1, NII-1), we are interested to see how the models trained with noisy human data will perform. We observe that NII-1 shows higher σ_H^A metric and lower f^A. Therefore, although NII-1 is trained on noisy human data, the use of *matching* z_1^N during training improves its performance. However, we do not observe a significant difference between NI-1 and NII-1 in terms of $\overline{R}^A$. Overall, the results in Table II indicate that the performance of models trained with noisy human data, regardless of the encoding, is inferior when deployed with expert human/encoding pair. We, therefore, have found evidence for Hypothesis H2-a.

Subcase 1-b:

In this subcase, the same models are deployed with noisy pair, i.e., (D^N, z_1^N). In this case, repeated measures ANOVA shows $p < 0.001$, $p < 0.001$, and $p = 0.021$ for metrics σ_H^A, f^A, and $\overline{R}^A$, respectively, which indicates that they are of statistical significance. Results of posthoc analysis for this case are presented in Table III, which lead to the following observations.

Looking at Column 3 of Table III (color-coded with light red), for the pair (EI-1, EII-1), we observe that EI-1 shows higher σ_H^A and lower f^A. This indicates that the model in EI-1 (*matching* z_1^E) is robust against noisy human data. This result is opposite of what we observed in the previous subcase for the models trained on noisy data deployed on expert pair, and reveals the importance of human expertise level in training the models and on their performance during deployment. We did not observe significant difference in $\overline{R}^A$. In Column 7 of the same table (color-coded with light amber), for the pair (EII-1, NII-1), we observe that NII-1 (*matching* z_1^N) shows higher σ_H^A and lower f^A. We did not observe significant difference in $\overline{R}^A$. Therefore, again we find evidence for Hypothesis H2-a.

B. Case 2: Group 2 With Expert or Noisy Human

Subcase 2-a:

In this subcase, the models in Group 2 are deployed with expert pairing, i.e., (D^E, z_1^E). Repeated measures ANOVA

TABLE IV
POSTHOC P-VALUES FOR SUBCASE 2-A: GROUP 2 MODELS DEPLOYED WITH (D^E, z_1^E)

metric	value	EI-2 - EII-2	EI-2 - NI-2	EI-2 - NII-2	EII-2 - NI-2	EII-2 - NII-2	NI-2 - NII-2
σ_H^A	p-value Mean	NS (0.704, 0.640)	NS (0.704, 0.733)	NS (0.704, 0.638)	NS (0.640, 0.733)	NS (0.640, 0.640)	**0.014** (0.733, 0.640)
f^A	p-value Mean	NS (0.0, 0.01)	NS (0.0, 0.02)	**<0.001** (0.0, 0.32)	NS (0.01, 0.02)	**<0.001** (0.01, 0.32)	**<0.001** (0.02, 0.32)
$\overline{R}^A$	p-value Mean	NS (587.0, 587.0)	**<0.001** (587.0, 533.7)	**<0.001** (587.0, 421.8)	**0.001** (587.0, 533.7)	**<0.001** (587.0, 421.8)	**<0.001** (533.7 421.8)

TABLE V
POSTHOC P-VALUES FOR SUBCASE 2-B: GROUP 2 MODELS DEPLOYED WITH (D^N, z_1^N)

metric	value	EI-2 - EII-2	EI-2 - NI-2	EI-2 - NII-2	EII-2 - NI-2	EII-2 - NII-2	NI-2 - NII-2
σ_H^A	p-value Mean	**<0.001** (0.822, 0.692)	NS (0.822, 0.8)	NS (0.822, 0.828)	**<0.001** (0.692, 0.8)	**<0.001** (0.692, 0.828)	NS (0.8, 0.828)
f^A	p-value Mean	NS (0.0, 0.03)	NS (0.0, 0.0)	NS (0.0, 0.0)	NS (0.03, 0.0)	NS (0.03, 0.0)	NS (0.0, 0.0)
$\overline{R}^A$	p-value Mean	NS (570.2, 536.3)	NS (570.2, 557.4)	NS (570.2, 573.5)	NS (536.2, 557.4)	NS (536.3, 573.5)	**0.0369** (557.4, 573.5)

shows $p = 0.005$, $p < 0.001$, and $p < 0.001$ for the metrics σ_H^A, f^A, and $\overline{R}^A$, respectively, which indicates that they are of statistical significance. Results of posthoc analysis for this case are presented in Table IV, leading to the following observations.

Looking at Table IV, for most of the model comparison results, we observe no significance for any of the metrics, notably except for $\overline{R}^A$ that shows higher performance of models trained with expert data compared to those trained with noisy data (color-coded with orange). Moreover, in the last column of the same table (color-coded with cyan), we observed degraded performance in NII-2 across all metrics. We, therefore, have found evidence for Hypothesis H2-a. It is also noted that we observe lower values of σ_H^A (on average) and higher $\overline{R}^A$ compared to those of Table II, in accordance with the relative weights on the closeness to human and task execution rewards. This confirms that selection of the reward coefficients affects the performance metrics. However, we observe a similar trend to that of Subcase 1-a where the models trained on noisy data tend to show degraded performance against expert pair, and moreover, the particular selection of the reward weights does not affect this trend noticeably.

Subcase 2-b:

In this subcase, the same models are deployed with the noisy pair (D^N, z_1^N). Repeated measures ANOVA shows $p < 0.001$, $p = 0.029$, and $p = 0.037$, for σ_H^A, f^A, and $\overline{R}^A$, respectively, indicating statistical significance. Results of posthoc analysis for this case are presented in Table V: these do not show significant differences between the models. We, therefore, have not found evidence for Hypothesis H2-a in this subcase, which shows that models in Group 2 (i.e., models trained with $c^H < c^{\text{RP}}$) when deployed with noisy pair (D^N, z_1^N) perform equally well in the defined performance metrics. Moreover, considering both subcases, we have found evidence for Hypothesis H2-b.

X. CONCLUSION AND FUTURE WORK

In this work, we proposed a novel shared autonomy and baseline policy adapting framework for robot operations. Building on a foundation of a task-oriented hierarchical approach, we first presented a novel methodology for variational analysis of human policy to encode the human internal state beyond the designed state variables. We employed a cVAE architecture to find the human's latent embedding through the lens of a structured task.

Next, we proposed a novel shared autonomy framework to facilitate interactions between the human and the autonomy—the two participating agents in this system. We modeled the decision-making process with hierarchical MDPs and Options in a policy adapting algorithm. The autonomous system policy is adapted by incorporating design variables contextual to the task, the human internal state as per encoding model, and pretraining, these combined with the human input.

To evaluate the proposed baseline shared autonomy policy, we carefully designed a testing procedure that allowed a comprehensive assessment of eight models trained under different human input and model parameter conditions. We compared their training performance and the performance of the trained models with respect to specific hypotheses by using relevant metrics. Following accepted statistical analysis norms, we evaluated our results for statistical significance in deploying different models in various conditions by using a virtual environment for a pick-and-place task. The results showcase how human's noise level and how encoding and incorporating human's latent variable affect the training process and the resultant model performance. Our results confirm the validity of the proposed architecture and assumptions, as well as the inherently complex dynamics of a shared autonomy setting with an HITL.

Since what is formulated in this article is a baseline shared autonomy policy, the future directions are numerous given the potential of this framework. The next logical step is to provide a fine-tuning of the baseline policy for online application, for example, as described in [26]. Applying the framework to real-life scenarios, such as forestry machine operations, is our next major focus.

APPENDIX A

Proof of Proposition 3.1 for three time-steps: Starting from (2), the probability over trajectory can be written as

$$\begin{aligned} p_\tau &= p(s_1^A, a_1^A, a_1^H, \ldots, s_3^A, a_3^A, a_3^H) \\ &= p(a_1^A, a_1^H, \ldots, s_3^A, a_3^A, a_3^H | s_1^A) p(s_1^A). \end{aligned} \tag{10}$$

Next, we write

$$\begin{aligned} p_\tau &= p(s_3^A | a_2^A, s_2^A) p(a_2^A, a_2^H, s_2^A, a_1^A, a_1^H | s_1^A) p(s_1^A) \\ &= p(s_3^A | a_2^A, s_2^A) p(a_2^A, a_2^H | s_2^A, a_1^A, a_1^H, s_1^A) \\ &\quad \times p(s_2^A, a_1^A, a_1^H | s_1^A) p(s_1^A). \end{aligned} \tag{11}$$

Next, we have

$$\begin{aligned} p_\tau &= p(s_3^A | a_2^A, s_2^A) p(s_2^A | a_1^A, s_1^A) p(a_2^A | a_2^H, s_2^A) \\ &\quad \times p(a_2^H | s_2^A, s_1^A) p(a_1^A | a_1^H, s_1^A) p(a_1^H | s_1^A) p(s_1^A) \end{aligned} \tag{12}$$

which simplifies to

$$\begin{aligned} p(\tau) &= p(s_1^A) \prod_{t=1}^{3} \pi_H(a_t^H | \bar{s}_t^A) \\ &\quad \times \pi_A(a_t^A | a_t^H, s_t^A) p(s_{t+1}^A | s_t^A, a_t^A). \end{aligned} \tag{13}$$

REFERENCES

[1] B. R. Kiran et al., "Deep reinforcement learning for autonomous driving: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 6, pp. 4909–4926, Jun. 2022.

[2] D. P. Losey et al., "Learning latent actions to control assistive robots," *Auton. Robots*, vol. 46, pp. 115–147, 2022.

[3] A. Annaswamy, K. Johansson, and G. Pappas, eds., *Control for Societal-scale Challenges: Road Map 2030*. New York, NY, USA: IEEE Control Syst. Soc., May 2023.

[4] C. Guo, C. Sentouh, J. C. Popieul, and J. B. Haué, "Predictive shared steering control for driver override in automated driving: A simulator study," *Transp. Res. Part F: Traffic Psychol. Behav.*, vol. 61, pp. 326–336, 2019.

[5] S. Westerberg, "Semi-automating forestry machines," Ph.D. thesis, Umeå Univ., Umeå, Sweden, 2014.

[6] 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.

[7] 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, pp. 717–742, Jun. 2018.

[8] S. Reddy, A. Dragan, and S. Levine, "Shared autonomy via deep reinforcement learning," in *Proc. Robot.: Sci. Syst.*, Pittsburgh, Pennsylvania, Jun. 2018.

[9] S. Jain and B. Argall, "Probabilistic human intent recognition for shared autonomy in assistive robotics," *ACM Trans. Hum.-Robot Interaction*, vol. 9, no. 1, pp. 1–23, 2020.

[10] C. Liu et al., "Goal inference improves objective and perceived performance in human-robot collaboration," in *Proc. 2016 Int. Conf. Autonom. Agents Multiagent Syst.*, Richland, SC, USA, 2016, pp. 940–948.

[11] S. Arora and P. Doshi, "A survey of inverse reinforcement learning: Challenges, methods and progress," *Artif. Intell.*, vol. 297, 2021, Art. no. 103500.

[12] A. Bobu, D. R. R. Scobee, J. F. Fisac, S. S. Sastry, and A. D. Dragan, "Less is more: Rethinking probabilistic models of human behavior," in *Proc. 2020 ACM/IEEE Int. Conf. Hum.-Robot Interaction*, 2020, pp. 429–437.

[13] C. L. Baker, J. B. Tenenbaum, and R. R. Saxe, "Goal inference as inverse planning," in *Proc. Annu. Meeting Cogn. Sci. Soc.*, vol. 29, 2007.

[14] E. Yousefi, D. P. Losey, and I. Sharf, "Assisting operators of articulated machinery with optimal planning and goal inference," in *Proc. 2022 Int. Conf. Robot. Autom.*, 2022, pp. 2832–2838.

[15] B. Löfgren, "Kinematic control of redundant with automatic path-following functions," PhD thesis, KTH, Mechatronics, 2009.

[16] M. R. Amini, I. Kolmanovsky, and J. Sun, "Hierarchical MPC for robust eco-cooling of connected and automated vehicles and its application to electric vehicle battery thermal management," *IEEE Trans. Control Syst. Technol.*, vol. 29, no. 1, pp. 316–328, Jan. 2021.

[17] J. Hong, A. Dragan, and S. Levine, "Learning to influence human behavior with offline reinforcement learning," in *Proc. Adv. Neural Inform. Process. Syst.*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., 2023, vol. 36, pp. 36094–36105.

[18] A. D. Dragan, "Robot planning with mathematical models of human state and action," 2017, *arXiv:1705.04226*.

[19] R. Sutton, "The bitter lesson," 2019. Accessed: Jun. 27, 2025. [Online]. Available: http://www.incompleteideas.net/IncIdeas/BitterLesson.html

[20] M. Jorda, M. Vulliez, and O. Khatib, "Local autonomy-based haptic-robot interaction with dual-proxy model," *IEEE Trans. Robot.*, vol. 38, no. 5, pp. 2943–2961, Oct. 2022.

[21] R. Zhang, Q. Lv, J. Li, J. Bao, T. Liu, and S. Liu, "A reinforcement learning method for human-robot collaboration in assembly tasks," *Robot. Comput.- Integr. Manuf.*, vol. 73, 2022, Art. no. 102227.

[22] R. S. Sutton, D. Precup, and S. Singh, "Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning," *Artif. Intell.*, vol. 112, pp. 181–211, 1999.

[23] S. Pateria, B. Subagdja, A.-H. Tan, and C. Quek, "Hierarchical reinforcement learning: A comprehensive survey," *ACM Comput. Surveys*, vol. 54, pp. 1–35, Jun. 2021.

[24] A. Thomaz, G. Hoffman, and M. Cakmak, "Computational human-robot interaction," *Foundations Trends Robot.*, vol. 4, no. 2-3, pp. 105–223, 2016.

[25] S. Griffith, K. Subramanian, J. Scholz, C. L. Isbell, and A. L. Thomaz, "Policy Shaping: Integrating human feedback with reinforcement learning," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst.*, 2013, pp. 2625–2633.

[26] E. Yousefi, M. Chen, and I. Sharf, "Shared autonomy policy fine-tuning and alignment for robotic tasks," *Int. J. Robot. Res.*, 2025. [Online]. Available: https://journals.sagepub.com/doi/full/10.1177/02783649241312699

[27] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," 2013, *arXiv:1312.6114*.

[28] Y. Shoham and K. Leyton-Brown, *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge, U.K.: Cambridge Univ. Press, 2008.

[29] E. Eraslan, Y. Yildiz, and A. M. Annaswamy, "Shared control between pilots and autopilots: Illustration of a cyber-physical human system," *IEEE Control Syst. Mag.*, vol. 40, no. 6, pp. 77–97, 2020.

[30] T. Yucelen, Y. Yildiz, R. Sipahi, E. Yousefi, and N. Nguyen, "Stability limit of human-in-the-loop model reference adaptive control architectures," *Int. J. Control*, vol. 91, pp. 2314–2331, Oct. 2018.

[31] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.

[32] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017, *arXiv:1412.6980*.

[33] J. Cohen, "Statistical power analysis," *Curr. Directions Psychol. Sci.*, vol. 1, no. 3, pp. 98–101, 1992.