Reachability-based Human-in-the-Loop Control with Uncertain Specifications

Yulong Gao [*] **Frank J. Jiang** [*] **Xiaoqiang Ren** [**] **Lihua Xie** [***]
Karl H. Johansson [*]

[*] *School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, Stockholm 10044, Sweden (e-mail: {yulongg, frankji, kallej}@kth.se)*
[**] *School of Mechatronic Engineering and Automation, Shanghai University, Shanghai, 200072, China (e-mail: xqren@shu.edu.cn)*
[***] *School of Electrical and Electronic Engineering, Nanyang Technological University, 639798, Singapore (e-mail: elhxie@ntu.edu.sg)*

Abstract: We propose a shared autonomy approach for implementing human operator decisions onto an automated system during multi-objective missions, while guaranteeing safety and mission completion. A mission is specified as a set of linear temporal logic (LTL) formulae. Then, using a novel correspondence between LTL and reachability analysis, we synthesize a set of controllers for assisting the human operator to complete the mission, while guaranteeing that the system maintains specified spatial and temporal properties. We assume the human operator's exact preference of how to complete the mission is unknown. Instead, we use a data-driven approach to infer and update the automated system's internal belief of which specified objective the human intends to complete. If, while the human is operating the system, she provides inputs that violate any of the invariances prescribed by the LTL formula, our verified controller will use its internal belief of the human operator's intended objective to guide the operator back on track. Moreover, we show that as long as the specifications are initially feasible, our controller will stay feasible and can guide the human to complete the mission despite some unexpected human errors. We illustrate our approach with a simple, but practical, experimental setup where a remote operator is parking a vehicle in a parking lot with multiple parking options. In these experiments, we show that our approach is able to infer the human operator's preference over parking spots in real-time and guarantee that the human will park in the spot safely.

Keywords: shared autonomy, linear temporal logic, reachability analysis, robotic missions, safety, automated vehicles

1. INTRODUCTION

With the rapid advancement of automation technology, there is an increasing interest in the trade-off between consistent performance of automated systems and the human situational awareness. In particular, researchers have proposed approaches for designing control systems that appropriately respect both the automated control inputs and decision-making of human operators, e.g., McRuer (1980); Cao et al. (2008); Li et al. (2014).

In this paper, we propose a solution to this problem, which we illustrate by the block diagram in Fig. 1. Namely, we design a control approach that allows a human operator ($\mathcal{H}$) to make decisions and provide inputs to a verification system corresponding to a guiding controller ($\mathcal{GC}$) that infers the human's intended task and computes a verified input to implement on the plant ($\mathcal{P}$). We break down the presentation of our solution into two main parts: (1) the synthesis of control sets that guarantee the mission is completed, and (2) the development of a guiding controller based on the control sets that allows a human operator to freely make decisions while the system maintains the specified invariances.

To synthesize our control sets, first, we use linear temporal logic (LTL) to specify missions. As shown by Huth and Ryan (2004) and Fainekos et al. (2005), using LTL formula allows us to conveniently express time-related invariances for automated systems. Furthermore, the work presented in Guo et al. (2018) exemplifies the advantage of using temporal tasks for human-in-the-loop mixed-initiative control. However, with LTL specified missions, Tabuada and Pappas (2006); Tabuada (2009); Belta et al. (2017); Kloetzer and Belta (2008) show that synthesizing controls that guarantee that some specification is met is nontrivial. Chen et al. (2018b) details a correspondence between

[⋆] The work of Y. Gao, F. J. Jiang, and K. H. Johansson is supported in part by the Swedish Strategic Research Foundation, the Swedish Research Council, and the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. The work of X. Ren is funded in part by Shanghai Key Laboratory of Power Station Automation Technology, and by National Key R&D Program of China (No. 2018AAA0102800, No. 2018AAA0102804).

reachable sets and signal temporal logic (STL) that allows for control synthesis directly from STL specifications with guarantees that the controller will satisfy the invariances given by the STL formula. We propose a similar approach for synthesizing control sets from LTL specifications.

There are several proposals for how to design guiding controllers. In Alshiekh et al. (2018), the authors propose an approach to learn optimal policies via reinforcement learning while enforcing LTL specifications. They utilize a shield, a similar notion to the guiding controller in our paper, to monitor the actions from the learner and corrects them only if the chosen action causes a violation of the specification. We remark that the systems studied in Alshiekh et al. (2018) are finite-transition systems, whereas in our work we consider discrete-time dynamical systems, leading to different control synthesis approaches. Another notable approach is given in Inoue and Gupta (2018), which proposes one of the first frameworks where humans are given a higher priority than the automated system in the decision making process whereas the human's *direct control* of the automated system is "weakened". The designed controller provides a set of admissible control inputs with enough degrees of freedom to allow the human operator to easily complete her task. We take inspiration from this approach for the design of our guiding controller.

The main contribution of this paper is to propose a guiding controller that allows a human operator to provide control inputs to a verification system that infers an LTL specified objective the human intends to complete. To compute the verified control input, we provide a result similar to Chen et al. (2018b), but introduce an equivalent transition system for LTL formulae that allows us to do control synthesis using reachability analysis, giving us guarantees that the system will follow the LTL specifications. Using these equivalent transition systems, we are able to define verified control sets that tell us what a human operator is allowed and not allowed to do. Then, with the verified control sets, we improve the approach in Guo et al. (2018) by allowing the human to freely make decisions as long as they do not violate invariances specified by the LTL formula.

The remainder of the paper is organized as follows. In Section 2, we outline our plant model and provide some preliminaries on LTL. In Section 3, we introduce a motivating example that we refer to throughout the paper and provide the problem statement. In Section 4, we describe a control set synthesis approach for LTL formula. In Section 5, we formulate the guiding controller. In Section 6, we illustrate the effectiveness of our approach with an experiment. In Section 7, we conclude the paper with a discussion about our work and future directions.

Notation. Let $\mathbb{N}$ denote the set of nonnegative integers and $\mathbb{R}$ denote the set of real numbers. For some $q, s \in \mathbb{N}$ and $q < s$, let $\mathbb{N}_{\geq q}$ and $\mathbb{N}_{[q,s]}$ denote the sets $\{r \in \mathbb{N} \mid r \geq q\}$ and $\{r \in \mathbb{N} \mid q \leq r \leq s\}$, respectively. When $\leq$, $\geq$, $<$, and $>$ are applied to vectors, they are interpreted elementwise. The indicator function of a set $\mathbb{X}$ is denoted by $\mathbf{1}_{\mathbb{X}}(x)$, i.e., if $x \in \mathbb{X}$, $\mathbf{1}_{\mathbb{X}}(x) = 1$ and otherwise, $\mathbf{1}_{\mathbb{X}}(x) = 0$.

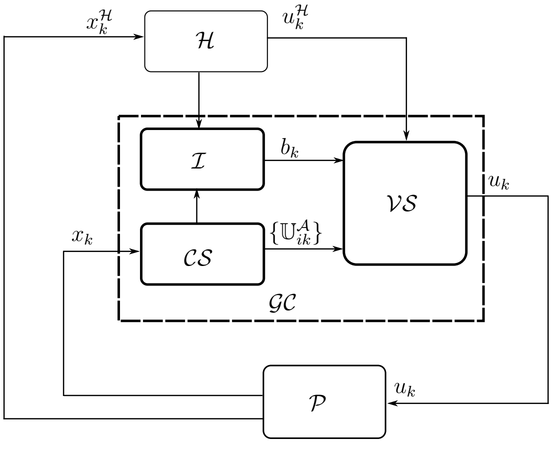


Fig. 1. Guiding control framework. $\mathcal{H}$: human decision-maker; $\mathcal{P}$: plant; $\mathcal{GC}$: guiding controller; $\mathcal{I}$: inferring; $\mathcal{CS}$: control set synthesis; $\mathcal{VS}$: verification synthesis.

2. PRELIMINARIES

2.1 Plant model

Consider a discrete-time dynamic control system

$$x_{k+1} = f(x_k, u_k, w_k), \tag{1}$$

where $x_k \in \mathbb{R}^{n_x}$, $u_k \in \mathbb{R}^{n_u}$, $w_k \in \mathbb{R}^{n_w}$, and $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_w} \to \mathbb{R}^{n_x}$. At each time instant k, the control input u_k is constrained by a set $\mathbb{U} \subset \mathbb{R}^{n_u}$ and the disturbance w_k belongs to a compact set $\mathbb{W} \subset \mathbb{R}^{n_w}$.An infinite path $\boldsymbol{s}$ starting from x_0 is a sequence of states $\boldsymbol{s} = x_0 x_1 \ldots x_k x_{k+1} \ldots$ such that $\forall k \in \mathbb{N}$, $x_{k+1} = f(x_k, u_k, w_k)$ for some $u_k \in \mathbb{U}$ and $w_k \in \mathbb{W}$.

For a path $\boldsymbol{s}$, the k-th state is denoted by $\boldsymbol{s}[k]$, i.e., $\boldsymbol{s}[k] = x_k$, the k-th prefix is denoted by $\boldsymbol{s}[..k]$, i.e., $\boldsymbol{s}[..k] = x_0 \ldots x_k$, and the k-th suffix is denoted by $\boldsymbol{s}[k..]$, i.e., $\boldsymbol{s}[k..] = x_k x_{k+1} \ldots$.

Each atomic proposition p_i is defined as a linear inequality in $\mathbb{R}^{n_x}$:

$$[p_i] \triangleq \{x \in \mathbb{R}^{n_x} \mid C_i^T x + d_i \leq 0\}, C_i \in \mathbb{R}^{n_x \times n_i}, d_i \in \mathbb{R}^{n_i},$$

where n_i is the number of inequalities in the ith atomic proposition. $\mathcal{AP}$ is a finite set of atomic propositions, i.e., $\mathcal{AP} = \{p_i\}_{i=1}^{N_{\mathcal{A}}}$.

Given a path $\boldsymbol{s} = x_0 x_1 \ldots x_k x_{k+1} \ldots$, a trace is a sequence of sets $\boldsymbol{P} = P_{x_0} P_{x_1} \ldots P_{x_k} P_{x_{k+1}} \ldots$, where each set $P_{x_k} \subseteq \mathcal{AP}$ is defined as $P_{x_k} = \{p_i \in \mathcal{AP} \mid x_k \in [p_i]\}$.

2.2 Linear temporal logic

An LTL formula is defined over a finite set of atomic propositions $\mathcal{AP}$ and both logic and temporal operators. The syntax of LTL can be described as:

$$\varphi ::= \text{true} \mid p \in \mathcal{AP} \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \bigcirc\varphi \mid \varphi_1 \mathsf{U} \varphi_2,$$

where $\bigcirc$ and U denote the "next" and "until" operators, respectively. By using the negation operator and the conjunction operator, we can define disjunction, $\varphi_1 \vee \varphi_2 = \neg(\neg\varphi_1 \wedge \neg\varphi_2)$. And by employing the until operator, we can define: (1) eventually, $\Diamond\varphi = \text{true} \cup \varphi$ and (2) always, $\Box\varphi = \neg\Diamond\neg\varphi$.

Definition 2.1. (LTL semantics) For an LTL formula φ and a path $\boldsymbol{s}$, the satisfaction relation $\boldsymbol{s} \vDash \varphi$ is defined as

$$
\begin{aligned}
&\boldsymbol{s} \vDash p \in \mathcal{AP} \Leftrightarrow p \in P_{x_0},\\
&\boldsymbol{s} \vDash \neg\varphi \Leftrightarrow \boldsymbol{s} \nvDash \varphi,\\
&\boldsymbol{s} \vDash \varphi_1 \wedge \varphi_2 \Leftrightarrow \boldsymbol{s} \vDash \varphi_1 \wedge \boldsymbol{s} \vDash \varphi_2,\\
&\boldsymbol{s} \vDash \varphi_1 \vee \varphi_2 \Leftrightarrow \boldsymbol{s} \vDash \varphi_1 \vee \boldsymbol{s} \vDash \varphi_2,\\
&\boldsymbol{s} \vDash \bigcirc\varphi \Leftrightarrow \boldsymbol{s}[1..] \vDash \varphi,\\
&\boldsymbol{s} \vDash \varphi_1 \mathsf{U} \varphi_2 \Leftrightarrow \exists j \in \mathbb{N} \text{ s.t. } \begin{cases} \boldsymbol{s}[j..] \vDash \varphi_2, \\ \forall i \in \mathbb{N}_{[0,j-1]}, \boldsymbol{s}[i..] \vDash \varphi_1, \end{cases}\\
&\boldsymbol{s} \vDash \Diamond\varphi \Leftrightarrow \exists j \in \mathbb{N}, \text{ s.t. } \boldsymbol{s}[j..] \vDash \varphi,\\
&\boldsymbol{s} \vDash \Box\varphi \Leftrightarrow \forall j \in \mathbb{N}, \text{ s.t. } \boldsymbol{s}[j..] \vDash \varphi,
\end{aligned}
$$

where P_{x_0} is the first element in the trace of the path $\boldsymbol{s}$.

Definition 2.2. (Robust feasibility) Consider the system (1). An LTL formula φ is robustly feasible from the initial state x_0 if there exists a feedback control law $u(x_k, k)$ mapping the pairs (x_k, k) into $\mathbb{U}$ such that the path $\boldsymbol{s} = x_0 x_1 \ldots$ generated from the closed-loop system

$$x_{k+1} = f(x_k, u(x_k, k), w_k)$$

satisfies φ for all possible disturbances $w_k \in \mathbb{W}, k \in \mathbb{N}$.

3. PROBLEM AND MOTIVATING EXAMPLE

3.1 Problem statement

Let us recall the shared autonomy scenario in Fig. 1, where the plant $\mathcal{P}$ is described by the dynamics (1). We consider a specification group consisting of a finite number of LTL specifications, denoted by $\{\varphi_i\}_{i=1}^{N_s}$, for the plant $\mathcal{P}$. Here, N_s denotes the number of specifications, which are defined *a priori* as a description of the tasks at hand. We assume that the human's preference over the specification group is uncertain, e.g., time-varying or random. In Fig. 1, we distinguish the state x_k that is measured by the sensor and transmitted to the guiding controller with the state $x_k^{\mathcal{H}}$ that the human operator perceives by herself. According to the state $x_k^{\mathcal{H}}$ at time instant k, the human operator $\mathcal{H}$ can make decisions and provide inputs $u_k^{\mathcal{H}}$ to a guiding controller, denoted by $\mathcal{GC}$. This guiding controller filters the human's decision $u_k^{\mathcal{H}}$ to a verified control command u_k and send it for implementation at the plant $\mathcal{P}$.

The main objective of this paper is to design the guiding controller $\mathcal{GC}$. More specifically, we will design three submodules for $\mathcal{GC}$ as shown in Fig. 1: (1) a control set synthesis module $\mathcal{CS}$ which provides a group of control sets, i.e., $\{\mathbb{U}_{ik}^{\mathcal{A}}\}_{i=1}^{N_s}$; (2) an inferring module $\mathcal{I}$ which updates the automated system's belief b_k of which specified objective the human intends to complete; and (3) a verification synthesis module $\mathcal{VS}$ which provides a verified control command u_k for satisfying the LTL specified task whenever the human's decision does not satisfy the specification. The problem to be solved is stated as follows.

Problem 3.1. Consider a plant $\mathcal{P}$ with dynamics (1) and a group of LTL specifications $\{\varphi_i\}_{i=1}^{N_s}$. Design a guiding controller $\mathcal{GC}$ in which

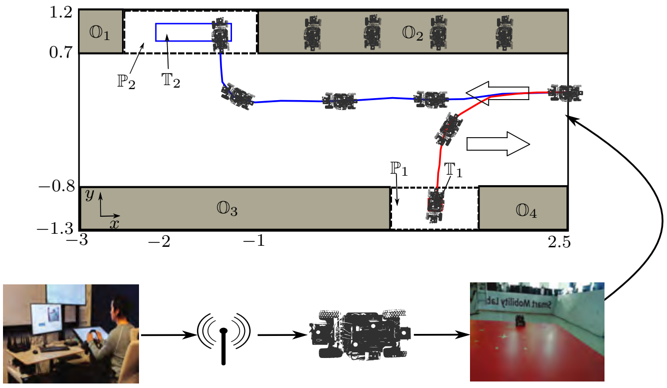


Fig. 2. A parking situation where a remote human operator would like to drive a vehicle to a narrow parking space $\mathbb{P}_1$ or a broad parking space $\mathbb{P}_2$.

(i) if φ_i is robustly feasible from x_k, the control set synthesis module $\mathcal{CS}$ can design a nonempty control set $\mathbb{U}_{ik}^{\mathcal{A}} \subseteq \mathbb{U}$ such that φ_i is robustly feasible from $x_{k+1} = f(x_k, u_k, w_k)$, $\forall u_k \in \mathbb{U}_{ik}^{\mathcal{A}}$ and $\forall w_k \in \mathbb{W}$; and

(ii) the inferring module $\mathcal{I}$ and the verification synthesis module $\mathcal{VS}$ can guarantee recursive feasibility regardless of the human's decisions.

3.2 Remote parking example

In this subsection, we will present an example that motivates our work and allows us to illustrate the approach. We consider a remote human operator parking example as shown in Fig. 2, where a human operator would like to drive a vehicle to a narrow parking space $\mathbb{P}_1$ or a broad parking space $\mathbb{P}_2$ in a parking lot. This remote human operator and the vehicle correspond to $\mathcal{H}$ and $\mathcal{P}$ in Fig. 1, respectively.

The vehicle is modeled as a two-dimensional double-integrator affected by a bounded disturbance. After discretizing the model with a sampling period of 0.2 second, it follows that

$$x_{k+1} = Ax_k + Bu_k + w_k,$$

where $x_k = [p_k^x, p_k^y]^T$, $u_k = [v_k^x, v_k^y]^T$, p_k^x and p_k^y, v_k^x and v_k^y denote the longitudinal and lateral position and velocity, respectively. The control input u_k is bounded by $\mathbb{U} = \{u \in \mathbb{R}^2 \mid [-0.3, -0.3]^T \leq u \leq [0.3, 0.3]^T\}$ and the disturbance w_k is bounded by $\mathbb{W} = \{w \in \mathbb{R}^2 \mid [-0.01, -0.01]^T \leq w \leq [0.01, 0.01]^T\}$. We consider the following atomic propositions, where we have written the expressions in an implicit form based on the notation in Fig. 2:

$[p_1] = \{x \in < ParkingLot >\}$, $[p_2] = \{x \in \mathbb{O}_1\}$, $[p_3] = \{x \in \mathbb{O}_2\}$, $[p_4] = \{x \in \mathbb{O}_3\}$, $[p_5] = \{x \in \mathbb{O}_4\}$, $[p_6] = \{x \in \mathbb{P}_1\}$, $[p_7] = \{x \in \mathbb{P}_2\}$, $[p_8] = \{x \in \mathbb{T}_1\}$, $[p_9] = \{x \in \mathbb{T}_2\}$.

We consider two specifications, which can be defined by LTL formulae:

$$
\begin{aligned}
\varphi_1 &= \Box p_1 \wedge \Box(\neg p_2 \wedge \neg p_3 \wedge \neg p_4 \wedge \neg p_5) \wedge \Diamond\Box p_6 \wedge \Diamond p_8,\\
\varphi_2 &= \Box p_1 \wedge \Box(\neg p_2 \wedge \neg p_3 \wedge \neg p_4 \wedge \neg p_5) \wedge \Diamond\Box p_7 \wedge \Diamond p_9.
\end{aligned}
$$

The specification φ_1 (or φ_2) requires that the vehicle always stays within the set $[p_1]$ without colliding into any obstacles and eventually enters the set $[p_6]$ (or $[p_7]$). After entering $[p_6]$ (or $[p_7]$), the vehicle stays there and eventually enters the set $[p_8]$ (or $[p_9]$). The set $\{\varphi_1, \varphi_2\}$ is the specification group.

The objective in this example is to design a guiding controller $\mathcal{GC}$ to *online* assist the human operator $\mathcal{H}$ to complete φ_1 or φ_2. More specifically, we will design (1) a control synthesis module $\mathcal{CS}$ which can synthesize a control set $\mathbb{U}_{ik}^{\mathcal{A}}$ such that the vehicle can be eventually parked to $\mathbb{P}_i$ if φ_i is robustly feasible; (2) an inferring module $\mathcal{I}$ which infers the parking space the human operator prefers; and (3) a verification synthesis module $\mathcal{CS}$ which corrects the human's decision $u_k^{\mathcal{H}}$, if $u_k^{\mathcal{H}}$ makes both parking specifications φ_1 and φ_2 infeasible.

4. CONTROL SET SYNTHESIS

This section focuses on handling part (i) of Problem 3.1. We first review some basic results of reachability analysis and then provide a correspondence between temporal operators and reachability analysis. Based on this, we finally present control set synthesis under an LTL formula.

4.1 Reachability analysis

This subsection recalls the computation of backward reachable sets and robust controlled invariant set for the control system (1).

Definition 4.1. Consider two sets $\Omega_1, \Omega_2 \subseteq \mathbb{R}^{n_x}$ and the system (1). The reachable set from Ω_1 to Ω_2 in N steps is defined as

$$\mathcal{R}(\Omega_1, \Omega_2, N) = \Big\{x_0 \in \mathbb{R}^{n_x} \mid \exists u_k \in \mathbb{U}, \forall k \in \mathbb{N}_{[0,N-1]}, \text{ s.t.}, \\ x_k \in \Omega_1, x_N \in \Omega_2, \forall w_k \in \mathbb{W}, \forall k \in \mathbb{N}_{[0,N-1]}\Big\}.$$

The reachable set from Ω_1 to Ω_2 is defined as

$$\mathcal{R}(\Omega_1, \Omega_2) = \bigcup_{N \in \mathbb{N}} \mathcal{R}(\Omega_1, \Omega_2, N).$$

For a set $\mathbb{X} \subseteq \mathbb{R}^{n_x}$, define the map $\mathcal{BR} : 2^{\mathbb{R}^{n_x}} \to 2^{\mathbb{R}^{n_x}}$:

$$\mathcal{BR}(\mathbb{X}) = \Big\{x \in \mathbb{R}^{n_x} \mid \exists u \in \mathbb{U}, \text{ s.t. } f(x, u, \mathbb{W}) \subseteq \mathbb{X}\Big\},$$

where $f(x, u, \mathbb{W}) = \{f(x, u, w) \mid w \in \mathbb{W}\}$. The set $\mathcal{BR}(\mathbb{X})$ collects all states from which the set $\mathbb{X}$ is reachable for any disturbance $w \in \mathbb{W}$. As shown in Bertsekas (1972), the reachable set from Ω_1 to Ω_2 evolves as

$$\mathcal{R}(\Omega_1, \Omega_2, N) = \mathcal{BR}(\mathcal{R}(\Omega_1, \Omega_2, N-1)) \cap \Omega_1, \\ \mathcal{R}(\Omega_1, \Omega_2, 0) = \Omega_2.$$

Definition 4.2. A set $\Omega_f \subseteq \mathbb{R}^{n_x}$ is said to be a robust controlled invariant set (RCIS) of the system (1) if for any $x \in \Omega_f$, there exists a control input $u \in \mathbb{U}$ such that $f(x, u, w) \in \Omega_f$, $\forall w \in \mathbb{W}$.

Definition 4.3. For a set $\mathbb{X} \subseteq \mathbb{R}^{n_x}$, a set $\mathcal{RI}(\mathbb{X}) \subseteq \mathbb{X}$ is said to be the maximal RCIS in $\mathbb{X}$ if each RCIS $\Omega_f \subseteq \mathbb{X}$ satisfies $\Omega_f \subseteq \mathcal{RI}(\mathbb{X})$.

For a set $\mathbb{X} \subseteq \mathbb{R}^{n_x}$, define

$$\mathbb{Q}_{k+1} = \mathcal{BR}(\mathbb{Q}_k) \cap \mathbb{Q}_k, \ \mathbb{Q}_0 = \mathbb{X}.$$

Then, it is shown in Blanchini and Miani (2007) that $\mathcal{RI}(\mathbb{X}) = \bigcap_{k \in \mathbb{N}} \mathbb{Q}_k$.

Remark 4.1. There are many methods for computing reachable sets, e.g., Chen et al. (2018a); Raković et al. (2006), or inner approximations of reachable sets, e.g., Althoff and Krogh (2014); Mitchell (2011). We remark that inner approximations are also applicable for the algorithms in this paper.

Next we propose a correspondence between temporal operators and reachability analysis. Given an LTL formula φ, let us denote by $\mathbb{S}_\varphi \subseteq \mathbb{R}^{n_x}$ the set of the initial states from which φ is robustly feasible.

Proposition 4.1. Consider the LTL formulae φ, φ_1, and φ_2. The following statements hold: (i) "next": $\mathbb{S}_{\bigcirc\varphi} = \mathcal{BR}(\mathbb{S}_\varphi)$; (ii) "until": $\mathbb{S}_{\varphi_1 \mathsf{U} \varphi_2} \subseteq \mathcal{R}(\mathbb{S}_{\varphi_1}, \mathbb{S}_{\varphi_2})$; (iii) "eventually": $\mathbb{S}_{\Diamond\varphi} = \mathcal{R}(\mathbb{R}^{n_x}, \mathbb{S}_\varphi)$; (iv) "always": $\mathbb{S}_{\Box\varphi} = \mathcal{RI}(\mathbb{S}_\varphi)$.

The proof of the above proposition follows the definitions of reachability analysis and temporal operators, see Chen et al. (2018b) for similar derivations. Due to limitation of space, we omit them here.

4.2 Control set synthesis under LTL

Before providing the procedure of control set synthesis, let us recall the correspondence between Boolean operators and set operators: (i) "negation": $\mathbb{S}_{\neg\varphi} \subseteq \bar{\mathbb{S}}_\varphi$; (ii) "conjunction": $\mathbb{S}_{\varphi_1 \wedge \varphi_2} \subseteq \mathbb{S}_{\varphi_1} \cap \mathbb{S}_{\varphi_2}$; (iii) "disjunction": $\mathbb{S}_{\varphi_1 \vee \varphi_2} \subseteq \mathbb{S}_{\varphi_1} \cup \mathbb{S}_{\varphi_2}$.

Definition 4.4. A temporal labeled transition (TLT) of the system (1) is a quadruple $(\mathcal{X}, \mathcal{T}, \to, N)$ with

- a sequence of sets: $\mathcal{X} = \mathbb{X}_0 \dots \mathbb{X}_l \dots \mathbb{X}_N$ with $\mathbb{X}_l \subseteq \mathbb{R}^{n_x}, \forall l \in \mathbb{N}_{[0,N]}$;
- a sequence of temporal operators $\mathcal{T} = \tau_0 \dots \tau_l \dots \tau_{N-1}$ with $\tau_l \in \{\bigcirc, \mathsf{U}, \Diamond, \Box\}$;
- a sequence of transitions $\mathbb{X}_l \xrightarrow{\tau_l} \mathbb{X}_{l+1}$:
  (1) $\tau_l = \bigcirc$ if $\forall x_0 \in \mathbb{X}_l$, $\exists u_0 \in \mathbb{U}$, such that $f(x_0, u_0, w_0) \in \mathbb{X}_{l+1}$, $\forall w_0 \in \mathbb{W}$;
  (2) $\tau_l = \mathsf{U}$ if $\forall x_0 \in \mathbb{X}_l$, $\exists j \in \mathbb{N}$, such that $\forall k \in \mathbb{N}_{[0,j-1]}$, $\exists u_k \in \mathbb{U}$, $x_k \in \mathbb{X}_l$, and $x_j \in \mathbb{X}_{l+1}$, $\forall w_k \in \mathbb{W}$;
  (3) $\tau_l = \Diamond$ if $\forall x_0 \in \mathbb{X}_l$, $\exists j \in \mathbb{N}$, such that $\forall k \in \mathbb{N}_{[0,j-1]}$, $\exists u_k \in \mathbb{U}$, $x_j \in \mathbb{X}_{l+1}$, $\forall w_k \in \mathbb{W}$;
  (4) $\tau_l = \Box$ if $\mathbb{X}_l = \mathbb{X}_{l+1}$ and $\forall x_0 \in \mathbb{X}_l$, $\exists u_0 \in \mathbb{U}$, such that $f(x_0, u_0, w_0) \in \mathbb{X}_{l+1}$, $\forall w_0 \in \mathbb{W}$.

We show how to employ the reachability analysis to construct an equivalent TLT for an LTL formula with finite length through an example.

Example 4.1. Let us continue the remote parking example in Section 3.2. The specification φ_1 can be transformed as an equivalent TLT, denoted instead as $(\mathcal{X}^{\varphi_1}, \mathcal{T}^{\varphi_1}, N^{\varphi_1}) = (\mathbb{X}_0^{\varphi_1}\mathbb{X}_1^{\varphi_1}\mathbb{X}_2^{\varphi_1}, \Diamond\Diamond, \to, 2)$, where

$$\mathbb{X}_2^{\varphi_1} = [p_8], \mathbb{X}_1^{\varphi_1} = \mathcal{R}(\mathcal{RI}([p_6]), \mathbb{X}_2^{\varphi_1}), \\ \mathbb{X}_0^{\varphi_1} = \mathcal{R}([p_1] \setminus (\cup_{i=2}^5 [p_i]), \mathbb{X}_1^{\varphi_1}).$$

Similarly, φ_2 can also be transformed as an equivalent TLT $(\mathcal{X}^{\varphi_2}, \mathcal{T}^{\varphi_2}, N^{\varphi_2}) = (\mathbb{X}_0^{\varphi_2}\mathbb{X}_1^{\varphi_2}\mathbb{X}_2^{\varphi_2}, \Diamond\Diamond, \to, 2)$, where

$$\mathbb{X}_2^{\varphi_2} = [p_9], \mathbb{X}_1^{\varphi_2} = \mathcal{R}(\mathcal{RI}([p_7]), \mathbb{X}_2^{\varphi_2}),$$
$$\mathbb{X}_0^{\varphi_2} = \mathcal{R}([p_1] \setminus (\cup_{i=2}^{5}[p_i]), \mathbb{X}_1^{\varphi_2}).$$

Lemma 4.1. Consider the control system (1). Assume that a finite-length LTL formula φ and a TLT $(\mathcal{X}^{\varphi}, \mathcal{T}^{\varphi}, \rightarrow, N^{\varphi})$ are equivalent in the sense of Definition 4.4. Given an initial state x_0, the formula φ is robustly feasible from x_0 if and only if $x_0 \in \mathbb{X}_0^{\varphi}$.

Proof. This result follows from the definitions of reachable sets and RCISs, the correspondence between reachability analysis and temporal operators, and the correspondence between Boolean operators and set operators, as described above.

Remark 4.2. Note that with reachability analysis, we can find the equivalent TLT for a *class* of LTL formulae. This equivalence does not hold for all LTL formulae due to limitations with the Boolean operations.

Assumption 4.1. Each LTL specification φ_i from the specification group has an equivalent TLT $(\mathcal{X}^{\varphi_i}, \mathcal{T}^{\varphi_i}, \rightarrow, N^{\varphi_i})$, $\forall i = 1, \ldots, N_s$.

At time instant k, the measured state path is $\boldsymbol{s}[..k] = x_0 \ldots x_k$. For the specification φ_i, we use l_{ik} to denote the position of $\boldsymbol{s}[..k]$ in the sequence $\mathcal{X}^{\varphi_i}$. With the initialization $l_{i0} = 0$, l_{ik} evolves as

$$l_{ik} = \begin{cases} l_{i,k-1} + 1, & \text{if } x_k \in \mathbb{X}_{l_{i,k-1}+1}^{\varphi_i}, \\ -1, & \text{if } x_k \notin \mathbb{X}_l^{\varphi_i}, \forall l \text{ or } l_{i,k-1} = -1, \\ l_{i,k-1} & \text{otherwise.} \end{cases}$$

If $l_{ik} = -1$, it means that the specification φ_i becomes infeasible based on the current measured state x_k. We can understand the dynamics of l_{ik} as follows: if the measured state x_k moves forward along the sequence $\mathcal{X}^{\varphi_i}$, the position l_{ik} is updated to $l_{i,k-1} + 1$; if x_k no longer belongs to any set of $\mathcal{X}^{\varphi_i}$, l_{ik} becomes -1; if x_k still belongs to the same set as x_{k-1}, then l_{ik} equals to $l_{i,k-1}$.

We implement Algorithm 1 to synthesize the control set $\mathbb{U}_{ik}^{\mathcal{A}}$ for each specification φ_i. If φ_i is infeasible, the synthesized control set is empty (line 2). We use $l_{ik} = N^{\varphi_i}$ to determine if φ_i is completed or not. If $l_{ik} = N^{\varphi_i}$, we have two cases: if x_k is driven from the temporal operator $\Box$, we set $\mathbb{U}_{ik}^{\mathcal{A}} = \{u \in \mathbb{U} \mid f(x_k, u, \mathbb{W}) \subseteq \mathbb{X}_{l_{ik}}^{\varphi_i}\}$ (line 6); otherwise, we set $\mathbb{U}_{ik}^{\mathcal{A}} = \mathbb{U}$ (line 8). If $l_{ik} \neq N^{\varphi_i}$, we also have two cases: if x_k is driven by the temporal operator $\bigcirc$, we set $\mathbb{U}_{ik}^{\mathcal{A}} = \{u \in \mathbb{U} \mid f(x_k, u, \mathbb{W}) \subseteq \mathbb{X}_{l_{ik}+1}^{\varphi_i}\}$ (line 14); otherwise, we set $\mathbb{U}_{ik}^{\mathcal{A}} = \{u \in \mathbb{U} \mid f(x_k, u, \mathbb{W}) \subseteq \mathbb{X}_{l_{ik}}^{\varphi_i}\}$ (line 12). In practice, the computation of the control set $\mathbb{U}_{ik}^{\mathcal{A}}$ is manageable. The set $\mathbb{U}_{ik}^{\mathcal{A}}$ can be expressed in an implicit form if the system is nonlinear or in an explicit form if the system is linear, where the constraint sets are expressed by polyhedra.

5. GUIDING CONTROLLER

This section will address the second part (ii) of Problem 3.1 based on the synthesized control sets. We do not detail how a human *actually* performs a decision-making process, but only assume that the human can synthesize a control input $u_k^{\mathcal{H}}$ at each time instant k. Next, we will show how to design the inferring module $\mathcal{I}$ and the verification synthesis $\mathcal{VS}$, and then outline the algorithm for our guiding controller $\mathcal{GC}$.

Algorithm 1 Control Set Synthesis

```
    Input: $x_k$, $l_{ik}$, $\varphi_i$, and its corresponding TLT
    $(\mathcal{X}^{\varphi_i}, \mathcal{T}^{\varphi_i}, \rightarrow, N^{\varphi_i})$
    Output: $\mathbb{U}_{ik}^{\mathcal{A}}$
 1: if $l_{ik} = -1$ then
 2:     $\mathbb{U}_{ik}^{\mathcal{A}} = \emptyset$;
 3: else
 4:     if $l_{ik} = N^{\varphi_i}$ then
 5:         if $\tau_{l_{ik}-1} \neq \Box$ then
 6:             $\mathbb{U}_{ik}^{\mathcal{A}} = \mathbb{U}$;
 7:         else
 8:             $\mathbb{U}_{ik}^{\mathcal{A}} = \{u \in \mathbb{U} \mid f(x_k, u, \mathbb{W}) \subseteq \mathbb{X}_{l_{ik}}^{\varphi_i}\}$;
 9:         end if
10:     else
11:         if $\tau_{l_{ik}} \neq \bigcirc$ then
12:             $\mathbb{U}_{ik}^{\mathcal{A}} = \{u \in \mathbb{U} \mid f(x_k, u, \mathbb{W}) \subseteq \mathbb{X}_{l_{ik}}^{\varphi_i}\}$;
13:         else
14:             $\mathbb{U}_{ik}^{\mathcal{A}} = \{u \in \mathbb{U} \mid f(x_k, u, \mathbb{W}) \subseteq \mathbb{X}_{l_{ik}+1}^{\varphi_i}\}$;
15:         end if
16:     end if
17: end if
```

5.1 *Inferring module $\mathcal{I}$*

As mentioned before, we assume that the human's preference is unknown for the guiding controller $\mathcal{GC}$. We introduce a specification belief b_k, which is a probability distribution vector over the specification group. Each element $b_k(i)$ quantifies the preference of the human on the specification φ_i. The inferring module $\mathcal{I}$ is to update this belief b_k in a data-driven manner. If the decision of the human $u_k^{\mathcal{H}}$ satisfies the specification φ_i, i.e., $u_k^{\mathcal{H}} \in \mathbb{U}_{ik}^{\mathcal{A}}$, we justify that the human has preference to choose this specification at time instant k. We denote by a $0-1$ vector $o_k \in \mathbb{R}^{N_s}$ the observation vector: if $u_k^{\mathcal{H}} \in \mathbb{U}_{ik}^{\mathcal{A}}$, $o_k(i) = 1$; otherwise, $o_k(i) = 0$. According to the Bayesian rule, the specification belief is updated as

$$b_{k+1}(i) = \frac{o_k(i) b_k(i) (\text{vol}(\mathbb{U}_{ik}^{\mathcal{A}}) + \epsilon)}{\sum_{i=1}^{N_s} o_k(i) b_k(i) (\text{vol}(\mathbb{U}_{ik}^{\mathcal{A}}) + \epsilon)}. \tag{2}$$

Here, $\text{vol}(\cdot)$ denotes the set volume. We define $\text{vol}(\emptyset) = -\infty$ and $0 \times (-\infty) = 0$. In addition, ϵ is a positive constant to avoid the singular case when $\text{vol}(\mathbb{U}_{ik}^{\mathcal{A}}) \leq 0$, $\forall i$. Intuitively, the larger the volume of $\mathbb{U}_{ik}^{\mathcal{A}}$ is, the easier for the operator to complete the specification φ_i, which in turn means that the more likely the human chooses φ_i.

5.2 *Verification synthesis module $\mathcal{VS}$*

After synthesizing the control sets $\{\mathbb{U}_{ik}^{\mathcal{A}}\}_{i=1}^{N_s}$ for all the specifications, we use a verification synthesis scheme to filter the human decision. If the decision of the human satisfies some specification, the decision will be respected. Otherwise, it will be corrected based on the specification belief b_k and the control sets $\{\mathbb{U}_{ik}^{\mathcal{A}}\}_{i=1}^{N_s}$. Mathematically, the control input u_k after verification synthesis is derived as

$$u_k = \begin{cases} u_k^{\mathcal{H}}, & \text{if } \exists i \text{ s.t. } u_k^{\mathcal{H}} \in \mathbb{U}_{ik}^{\mathcal{A}}, \\ \underset{u\in\mathbb{U}_{ik}^{\mathcal{A}}, i=1,\ldots,N_s}{\operatorname{argmin}} \dfrac{\|u-u_k^{\mathcal{H}}\|}{b_k(i)}, & \text{otherwise,} \end{cases} \tag{3}$$

where $u_k^{\mathcal{H}}$ is the original human decision. In (3), the belief $b_k(i)$ plays the role of weighing the distance between $u_k^{\mathcal{H}}$ and $\mathbb{U}_{ik}^{\mathcal{A}}$. Larger $b_k(i)$'s increase the possibility of choosing the projected control input of $u_k^{\mathcal{H}}$ on the set $\mathbb{U}_{ik}^{\mathcal{A}}$.

5.3 Guiding controller $\mathcal{GC}$

Next we develop the algorithm for the guiding controller $\mathcal{GC}$.

Definition 5.1. The terminal conditions are a set of states that are consistent with the specification group, i.e., each state satisfies at least one specification φ_i and each specification has at least one state in this set. We denote the terminal conditions by $h(x) \leq 0$, where $h : \mathbb{R}^{n_x} \to \mathbb{R}^{N_t}$ and N_t denotes the number of terminal conditions.

Example 5.1. For the remote parking example in Section 3.2, the terminal condition corresponds to that the state of the vehicle reaches $[p_8]$ or $[p_9]$, because then the parking task is completed. Thus, we can write $h(x) = 1 - \mathbf{1}_{[p_8]\cup[p_9]}(x)$.

Due to the presence of disturbances w_k, we implement the robust guiding controller in a closed-loop manner. As shown in Algorithm 2, at each time instant k, if all the synthesized control sets $\mathbb{U}_{ik}^{\mathcal{A}}$ are empty, i.e., all specifications are infeasible, the algorithm ends up with output Infeasible (lines 11–13). Otherwise, the guiding controller will mix the decision of the human $u_k^{\mathcal{H}}$ and the synthesized control sets $\mathbb{U}_{ik}^{\mathcal{A}}$ to synthesize the control input u_k (lines 8, 9, and 15). Meanwhile, the specification belief b_k is updated (line 16). If the terminal conditions $h(x_k) \leq 0$ hold, the algorithm ends up with output Successful (lines 4–7).

Algorithm 2 Guiding Controller Algorithm

```
 1: Initialization: Set $k = 0$ and TerInd $= 1$;
 2: while TerInd do
 3:     Measure $x_k$;
 4:     if $h(x_k) \leq 0$ then ▷ Terminal conditions
 5:         TerInd $= 0$;
 6:         Output: Successful;
 7:     else
 8:         Human makes a decision $u_k^{\mathcal{H}}$;
 9:         Update $l_{ik}$ and synthesize $\mathbb{U}_{ik}^{\mathcal{A}}$ for each $\varphi_i$;
10:         ▷ Algorithm 1
11:         if $\mathbb{U}_{ik}^{\mathcal{A}} = \emptyset$, $\forall i \in \mathbb{N}_{[1,N_s]}$ then
12:             TerInd $= 0$;
13:             Output: Infeasible;
14:         else
15:             Synthesize controller $u_k$ by (3);
16:             Update specification belief $b_k$ by (2);
17:             ▷ Guiding controller
18:             Implement $u_k$;
19:             Update $k = k + 1$;
20:         end if
21:     end if
22: end while
```

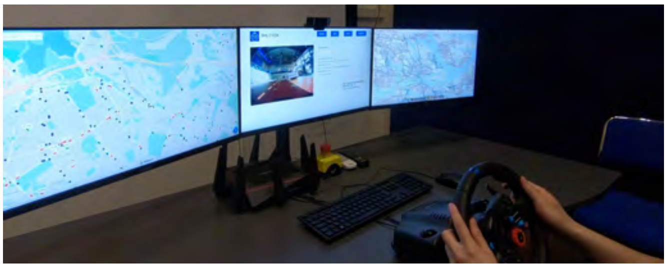


Fig. 3. The teleoperation station where a human operator can control a remotely connected vehicle

The following theorem shows that Algorithm 2 stays feasible.

Theorem 5.1. Consider the control system (1) and an initial state x_0. Suppose Assumption 4.1 holds and $x_0 \in \mathbb{X}_0^{\varphi_i}$, $\forall i \in \mathbb{N}_{[1,N_s]}$. Then, Algorithm 2 is feasible for all $k \in \mathbb{N}$.

Proof. Algorithm 2 is feasible for all $k \in \mathbb{N}$ if and only if there exists at least one feasible specification at each time instant k. Let us define a sequence of sets $\{\mathbb{F}_k\}_{k\in\mathbb{N}}$, of which each set $\mathbb{F}_k$ collects the indexes of feasible specifications at time instant k. If $x_0 \in \mathbb{X}_0^{\varphi_i}$, $\forall i \in \mathbb{N}_{[1,N_s]}$, $\mathbb{F}_0 = \{1, 2, \ldots, N_s\}$. From Algorithm 2, the sets $\mathbb{F}_k$ satisfy $\mathbb{F}_{k+1} \subseteq \mathbb{F}_k$, i.e., the sequence of sets $\{\mathbb{F}_k\}_{k\in\mathbb{N}}$ is nonincreasing. Furthermore, if the cardinality of $\mathbb{F}_k$ is 1 at some time instant k, it follows from Algorithm 1 that the cardinality of $\mathbb{F}_j$ is 1 for all $j \geq k$. Thus, each set of the sequence $\{\mathbb{F}_k\}_{k\in\mathbb{N}}$ is nonempty, by which we complete the proof.

6. EXPERIMENTS

In this section, we detail our experimental setup and report experimental results based on the remote parking example described in Section 3.2.

6.1 Experimental setup

The experimental setup consists of three components: the ego vehicle, a human operator interface, and the parking lot environment, see Fig. 2.

The ego vehicle is represented by the Small-Vehicles-for-Autonomy (SVEA) platform, which is a small robotic car platform designed to evaluate automated vehicle-related software stacks. For our experiment, we equip the SVEA car with an ELP fish-eye camera to provide a wide-angle

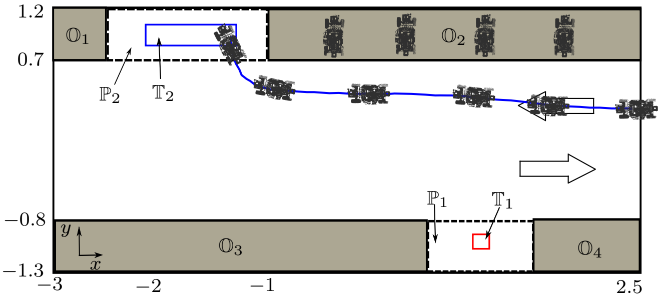
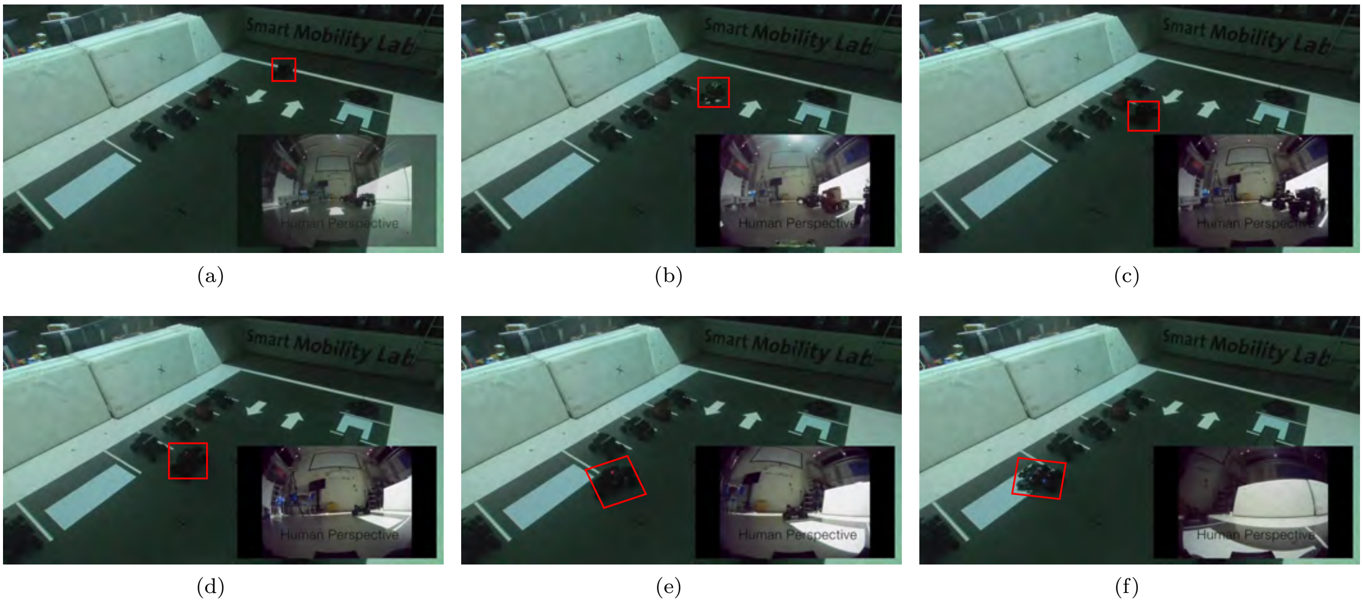


Fig. 4. Position trajectory when the human drives the vehicle to the parking region $\mathbb{P}_2$.



(a) (b) (c)

(d) (e) (f)

Fig. 5. An example where a human remotely drives the vehicle to the parking region $\mathbb{P}_2$ of Fig. 4. We highlight the position of the vehicle by the red box and show in the bottom right corner of each snapshot the view of the human operator.

view for the human operator and a TP-Link 4G LT modem for streaming both the camera data to the huma operator and the control from the human operator bac to the SVEA car.

For the human operator interface, we place a human a a teleoperation desk built to support the management c remotely connected vehicles, see Fig. 3. A computer a the teleoperation desk is connected to the internet an is running a WebRTC-based app that handles the dat transmission between the teleoperation station and th SVEA car over a peer-to-peer connection. The human ca provide input to the control system with a Logitech G2 steering wheel and pedals. This interface subsumes the $\mathcal{GC}$ block in Fig. 1.

The parking lot environment corresponds to the environment defined in Section 3.2, see Fig. 4. The free parking spots and obstacles are all in the coordinate frame of our Qualisys motion capture system.

6.2 Experimental results

The human operator is parking the vehicle in parking region $\mathbb{P}_2$, corresponding to specification φ_2 derived as a TLT in Example 4.1. The video of the experiment is available at `https://youtu.be/WhFNleymOJ8`.

We show snapshots of the vehicle's position in Fig. 5 and the corresponding trajectories in Fig. 4. We can see that during the parking process, there is no collision between the vehicle and the obstacles. Fig. 6 shows the control inputs, where the dashed lines denote the control bounds. The red and cyan regions represent the synthesized control sets for φ_1 and φ_2, respectively. The blue lines are the decision trajectories of the human driver while the black lines are the implemented control trajectories under Algorithm 1.

Note that at some time instants, the human's decision cannot satisfy any specification, thus the input is corrected according to the synthesized control sets. After 4.6 seconds (at which p_k^x is about 1 m), the synthesized control set for φ_1 is empty since this specification becomes infeasible. This can also be observed from Fig. 7, which shows the belief update. Note that the beliefs in φ_1 and φ_2 oscillate from 1.2 seconds to 2.6 seconds since the volume of the

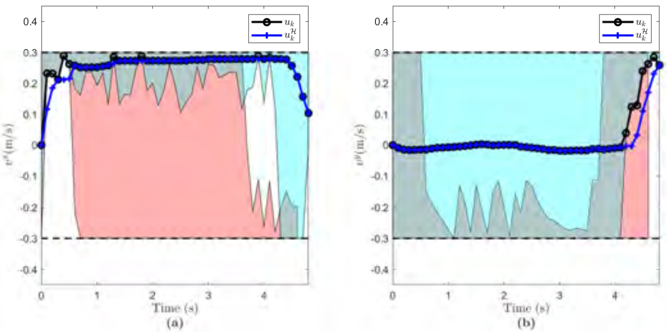


Fig. 6. Velocity trajectory when the human drives the vehicle to the parking region 2.

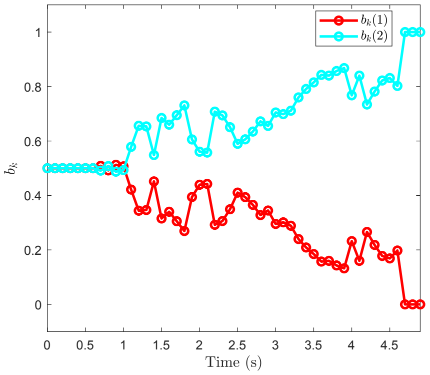


Fig. 7. Belief update when the human drives the vehicle to the parking region 2.

control sets changes significantly during this time interval. After that, the belief in φ_2 increases since the vehicle passes the parking region $\mathbb{P}_1$ and approaches the parking region $\mathbb{P}_2$, which then becomes more likely.

In this example, we can observe the capabilities of our approach. Even though the system's initial belief is neutral, as the human operates the vehicle, the system updates its belief appropriately. The guiding controller works together with the human operator to complete the parking maneuver.

7. CONCLUSION

In this paper, we presented a solution for robust human-in-the-loop learning and control under uncertain temporal specifications. With our framework, we give priority to the human operator's decision, allowing her to complete one of several possible tasks. Our framework makes no assumptions about the operator's preference over the tasks. Our system updates a data-driven belief of the operator's intent. We proposed a new method for synthesizing the control sets for LTL formulae based on a correspondence between LTL and reachability analysis. We proved recursive feasibility of the method, showing that the controller is always feasible and able to guarantee that the human will not be able to drive the system to violate invariances, despite her freedom to control the system. We illustrated the effectiveness of the proposed method on a remote parking example.

Future work includes the extension of TLTs to handle general LTL formulae and more detailed experimental evaluation of our approach.

REFERENCES

Alshiekh, M., Bloem, R., Ehlers, R., Knighofer, B., Niekum, S., and Topcu, U. (2018). Safe reinforcement learning via shielding. In *Proceedings of 32rd AAAI Conference on Artificial Intelligence*.

Althoff, M. and Krogh, B. (2014). Reachability analysis of nonlinear differential-algebraic systems. *IEEE Transactions on Automatic Control*, 59(2), 371–383.

Belta, C., Yordanov, B., and Gol, E. (2017). *Formal methods for discrete-time dynamical systems*. Springer.

Bertsekas, D. (1972). Infinite time reachability of state-space regions by using feedback control. *IEEE Transactions on Automatic Control*, 17(5), 604–613.

Blanchini, F. and Miani, S. (2007). *Set-theoretic methods in control*. Springer.

Cao, M., Stewart, A., and Leonard, N. (2008). Integrating human and robot decision-making dynamics with feedback: models and convergence analysis. In *Proceedings of 47th IEEE Conference on Decision and Control*, 1127–1132.

Chen, M., Herbert, S., Vashishtha, M., Bansal, S., and Tomlin, C. (2018a). Decomposition of reachable sets and tubes for a class of nonlinear systems. *IEEE Transactions on Automatic Control*, 63(11), 3675–3688.

Chen, M., Tam, Q., Livingston, S., and Pavone., M. (2018b). Signal temporal logic meets Hamilton-Jacobi reachability: connections and applications. In *Proceedings of International Workshop on the Algorithmic Foundations of Robotics*.

Fainekos, G., Kress-Gazit, H., and Pappas., G. (2005). Temporal logic motion planning for mobile robots. In *Proceedings of IEEE International Conference on Robotics and Automation*, 2020–2025.

Guo, M., Andersson, S., and Dimarogonas, D. (2018). Human-in-the-loop mixed-initiative control under temporal tasks. In *Proceedings of IEEE International Conference on Robotics and Automation*, 6395–6400.

Huth, M. and Ryan, M. (2004). *Logic in computer science: modelling and reasoning about systems*. Cambridge University Press.

Inoue, M. and Gupta, V. (2018). Weak control for human-in-the-loop systems. *IEEE Control Systems Letters*, 3(2), 440–445.

Kloetzer, M. and Belta, C. (2008). A fully automated framework for control of linear systems from temporal logic specifications. *IEEE Transactions on Automatic Control*, 53(1), 287–297.

Li, W., Sadigh, D., Sastry, S., , and Seshia, S. (2014). Synthesis for human-in-the-loop control systems. In *Proceedings of International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, 470–484.

McRuer, D. (1980). Human dynamics in man-machine systems. *Automatica*, 16(3), 237–253.

Mitchell, I. (2011). Scalable calculation of reach sets and tubes for nonlinear systems with terminal integrators: a mixed implicit explicit formulation. In *Proceedings of 14th ACM International Conference on Hybrid Systems: Computation and Control*, 103–112.

Raković, S., Kerrigan, E., Mayne, D., and Lygeros, J. (2006). Reachability analysis of discrete-time systems with disturbances. *IEEE Transactions on Automatic Control*, 51(4), 546–561.

Tabuada, P. (2009). *Verification and control of hybrid systems: a symbolic approach*. Springer.

Tabuada, P. and Pappas, G. (2006). Linear time logic control of discrete-time linear systems. *IEEE Transactions on Automatic Control*, 51(12), 1862–1877.