2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Macau, China, November 4-8, 2019

Specifying and Synthesizing Human-Robot Handovers

Alap Kshirsagar, Hadas Kress-Gazit, and Guy Hoffman

Abstract— **We present a controller for human-robot handovers that is automatically synthesized from high-level specifications in Signal Temporal Logic (STL). In contrast to existing controllers, this approach can provide formal guarantees on the timing of each of the handover phases. Using synthesis also allows end-users to specify and dynamically change the robot's behaviors using high-level requirements of goals and constraints rather than by tuning low-level controller parameters. We illustrate the proposed approach by replicating the behavior of existing handover strategies from the literature. We also identify specification parameters that are likely to lead to successful handovers using a public database of human-human handovers.**

Index Terms— **Human-Robot Handovers, Formal Methods, Signal Temporal Logic, Human-Robot Interaction**

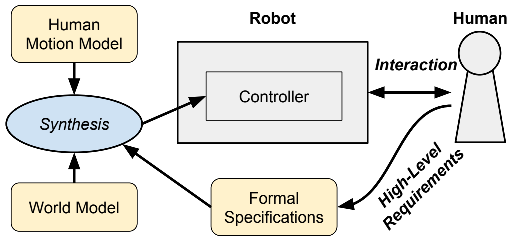


Fig. 1: Our approach to human-robot handovers uses the automatic synthesis of a robot controller from formal specifications written in Signal Temporal Logic. Users can change the robot's behavior with high-level requirements of goals and intuitive parameters such as the timing of different stages of handovers. The controller is then synthesized online based on a given human-motion prediction model and a world dynamics model.

I. Introduction

In this paper, we propose the automatic synthesis of robot controllers for human-robot handovers from formal specifications. Object handovers are a central aspect of human-robot collaboration in both industrial and domestic environments. Examples include a collaborative factory robot exchanging parts with a human co-worker, a surgical assistant robot transferring instruments to or from a surgeon, a warehouse robot helping a human shelve items, a domestic robot unloading a dishwasher, and a caregiver robot providing food or medicine to bedridden people. These tasks include both human-to-robot and robot-to-human handovers.

Most of the research on human-robot handovers uses hand-coded controllers [1]–[5], where the robot's behavior is predefined and programmed manually by engineers. A more flexible approach is learning-from-demonstration [6]–[10], where the robot learns the entire handover or some parameters associated with it via demonstrations. These demonstrations can consist of human-human handovers or of a human guiding the robot through the motion. Some researchers have also developed motion planners for offline generation of the robot's handover trajectory [11]–[13].

While these approaches can result in successful bi-directional human-robot handovers, they have several shortcomings. First, they do not provide guarantees on the timing of different stages of a handover. Such timing guarantees may be crucial in productivity oriented industrial tasks and fast-paced life-critical scenarios like surgery. Another drawback of existing approaches is that they do not allow users to specify and dynamically change behaviors using high-level abstractions of goals and constraints. Instead, they require users to tune controller parameters, which is often infeasible or non-intuitive. Finally, given the multitude of handover strategies that have been proposed in the literature for human-robot handovers, there is no unified framework to easily switch between those strategies. Our proposed approach, shown in Fig. 1, tries to address these limitations.

We exemplify the possibility of automatically synthesizing handovers by specifying the robot's handover behavior using Signal Temporal Logic (STL) [14] formulae. STL is well-suited to the handover domain as it enables reasoning about timings and distances. Using STL, the user can change the robot's behavior by specifying intuitive parameters like the timing of different phases of handovers and safety constraints. In this paper, we present specification templates both for human-to-robot and for robot-to-human handovers, and illustrate the flexibility of our approach by reproducing existing human-robot handover strategies found in the literature. We then use the same specifications in conjunction with a human-human handover data-set to identify parameters that can lead to successful human-robot handovers.

Sibley School of Mechanical and Aerospace Engineering, Cornell University, Ithaca, New York 14853
Email: {ak2458,hadaskg,hoffman}@cornell.edu

II. Related Work

A. Human-Robot Handovers

Existing approaches to human-robot handovers can be broadly categorized into two groups: offline and online. Most of the prior work on human-robot handovers has focused on offline approaches, in which the robot's motion is planned before the start of a handover. These approaches do not take into account the observed behavior of the human during a handover and, instead, the human has to adapt to the robot. Human-Robot Interaction (HRI) researchers have studied different aspects of handovers enabling offline planning, including the handover location and configuration [2], the effect of accompanied gaze behaviors [3], the effect of the user's level of experience [5]. All of these systems used hand-coded controllers. Going beyond hand-coding the full handover trajectory, a few researchers have proposed offline motion planners for robot-to-human handovers [11], [13]. Another direction of research has focused on learning robot motions from demonstrations of human-human handovers [6], [8], [10]. All of these approaches are offline and hence lack adaptability to the human's motion.

Some have proposed online controllers for human-robot handovers that do take into account the observed human motion [4], [7], [12], [15], [9]. Huang *et al.* [4], proposed three strategies for robot-to-human handovers, two of which enable the robot to adapt to the receiver's task demand. But they also used preprogrammed robot trajectories in these strategies. Yamane *et al.* [7] used a human-motion database to generate the robot's motion corresponding to the observed human motion, but this approach does not have mechanisms to respond to the human's motions that are not present in the database. Prada *et al.* [12], proposed a control system based on Dynamic Movement Primitives (DMP) formalism to drive the robot along human-like trajectories towards the human hand. Micelli *et al.* [15], proposed a proportional velocity controller to move the robot towards the human hand. Medina *et al.* [9], estimated the handover location online by modeling the human's motion as a linear dynamical system, and employed an impedance-based control scheme to drive the robot towards the predicted handover location. While these approaches enable the robot to adapt to the human's motion, they require tuning controller parameters (e.g., the weights of DMP terms or the velocity-tracking gain) to achieve a specific robot behaviors.

B. Automatic Synthesis of Robot Control for HRI

Automatic synthesis of robot controllers outside of HRI has been gaining interest in the past several years [16] and has been demonstrated on a variety of platforms including mobile robots, UAVs, autonomous cars, modular robots, mobile manipulators, swarm robots and humanoids. Still, very little work has been done regarding synthesis for HRI.

In the area of formal methods, Araiza-Illan *et al.* [17] and Webster *et al.* [18] applied probabilistic model-checking in a human-robot handover task, verifying a controller with respect to safety and liveness specifications. But these works consider a given controller and do not deal with the synthesis of one. Li *et al.* [19] presented a formalism for human-in-the-loop control synthesis and synthesized a semi-autonomous controller from temporal logic specifications, but the HRI aspects were limited to human intervention and correction of the robot's operation, and not to the actual handover interaction. To the best of our knowledge, there is no prior work on synthesis for human-robot handovers.

III. Signal Temporal Logic for Robot Control

Automatic synthesis of robot control [16] has been demonstrated using a number of formalisms, including discrete logic such as Linear Temporal Logic (LTL), probabilistic logic such as Probabilistic Computation Tree Logic (PCTL) and metric logic such as Signal Temporal Logic (STL). We choose STL for human-robot handovers, as it allows specifications that include distances, timing, and object states.

A. Signal Temporal Logic

STL formulae are composed of predicates $\pi^\gamma : \mathbb{R}^n \to \mathbb{B}$ over continuous signals. The truth value of a predicate π^γ is determined by the sign of a function $\gamma : \mathbb{R}^n \to \mathbb{R}$ of an underlying signal $\mathbf{s}$. STL formulae are defined recursively according to the following grammar:

$$\varphi ::= \pi^\gamma \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \Box_{[a,b]}\varphi \mid \varphi_1 \mathcal{U}_{[a,b]} \varphi_2 \mid \Diamond_{[a,b]}\varphi$$

where $a, b \in \mathbb{R}_{\geq 0}$. The operators $\Box$ ("Always"), $\Diamond$ ("Eventually") and $\mathcal{U}$ ("Until") can also be unbounded.

The satisfaction of an STL formula φ with respect to the signal $\mathbf{s}$ at time t is defined inductively as follows:

$$\begin{aligned}
(\mathbf{s},t) \models \pi^\gamma &\iff \gamma(\mathbf{s}(t)) > 0 \\
(\mathbf{s},t) \models \neg\pi^\gamma &\iff \neg((\mathbf{s},t) \models \pi^\gamma) \\
(\mathbf{s},t) \models \varphi_1 \wedge \varphi_2 &\iff (\mathbf{s},t) \models \varphi_1 \wedge (\mathbf{s},t) \models \varphi_2 \\
(\mathbf{s},t) \models \varphi_1 \vee \varphi_2 &\iff (\mathbf{s},t) \models \varphi_1 \vee (\mathbf{s},t) \models \varphi_2 \\
(\mathbf{s},t) \models \Box_{[a,b]}\varphi &\iff \forall t' \in [t+a, t+b], (\mathbf{s},t') \models \varphi \\
(\mathbf{s},t) \models \varphi_1 \mathcal{U}_{[a,b]} \varphi_2 &\iff \exists t' \in [t+a, t+b] \text{ s.t. } (\mathbf{s},t') \models \varphi_2 \\
&\qquad \wedge \forall t'' \in [t,t'], (\mathbf{s},t'') \models \varphi_1 \\
(\mathbf{s},t) \models \Diamond_{[a,b]}\varphi &\iff \exists t' \in [t+a, t+b] \text{ s.t. } (\mathbf{s},t') \models \varphi
\end{aligned}$$

A signal $\mathbf{s} = s_0 s_1 s_2 \ldots$ satisfies φ, denoted by $\mathbf{s} \models \varphi$, if $(\mathbf{s},0) \models \varphi$. Informally, $\mathbf{s} \models \Box_{[a,b]}\varphi$ if φ holds at each time between a and b. $\mathbf{s} \models \varphi_1 \mathcal{U}_{[a,b]} \varphi_2$ if φ_2 holds at some time c between a and b, and φ_1 holds at each time between a and c. $\mathbf{s} \models \Diamond_{[a,b]}\varphi$ if φ holds at some time between a and b.

B. Receding Horizon Control Synthesis from STL

For controller synthesis, STL specifications can be automatically transformed into mixed integer linear programs (MILP) and solved either globally for a sequence of control actions or iteratively in a receding horizon manner [20] [21]. In this paper, we use the receding horizon control (RHC) synthesis algorithm presented by Raman *et al.* [20]: Given a system of the form $x_{t+1} = f(x_t, u_t)$, initial state x_0, STL formula φ, cost function J and finite horizon H, the following optimization problem is solved at each time-step t:

$$\begin{aligned}
\operatorname{argmin}_{\mathbf{u}^{H,t}} \quad & J(\mathbf{x}(x_t, \mathbf{u}^{H,t}), \mathbf{u}^{H,t}) \\
\text{s.t.} \quad & \mathbf{x}(x_0, \mathbf{u}_0) \models \varphi
\end{aligned} \tag{1}$$

where $\mathbf{u}^{H,t} = u_0^t, u_1^t, u_2^t \ldots u_H^t$ is the horizon-H control input sequence computed at each time step and $\mathbf{x}(x_t, \mathbf{u}^{H,t}) = x_t, x_{t+1}, x_{t+2} \ldots x_{t+H}$ is the sequence of system states over the horizon H. Though the control input is computed over the horizon H, only the first control is applied at each time-step and the horizon is calculated again. Thus in Eq. 1, $\mathbf{u}_0 = u_0^0, u_0^1, u_0^2, \ldots$ is the applied control sequence consisting of only the first control inputs of $\mathbf{u}^{H,t}$ with $\mathbf{x}(x_0, \mathbf{u}_0) = x_0, x_1, x_2 \ldots$ being the sequence of system states as a result of the input $\mathbf{u_0}$.

IV. Formulation of Handovers in Signal Temporal Logic (STL)

To specify human-robot handovers in STL, we first define the following variables. Bold letters denote vectors.

Pose:

- $\mathbf{p_i} = [x_i, y_i, z_i]$ and $\mathbf{q_i} = [q_{x,i}, q_{y,i}, q_{z,i}, q_{w,i}]$ represent the position and orientation vectors in 3D space, jointly called 'pose'. Subscript $i \in \{r, h, o, l, d, \eta\}$ refers to: r = robot, h = human, o = object, l = handover location, d = object drop-off destination, η = robot home pose.
- $\mathbf{q}_\delta$ is the offset between the robot end-effector's orientation and the human hand's orientation, when they both hold the object.
- ε_p and ε_q are the permitted tolerances (errors) in the robot end-effector's position and orientation from any desired position and orientation.

Gripper and Hand:

- g_r and $g_h \in [0,1]$ are the robot gripper and human hand "openness" (0 for fully closed, 1 for fully open).
- $g_* \in [0,1]$ is the grip-width of the object.
- ε_g is the permitted tolerance (error) between a gripper/hand openness from a desired level of openness.

Force Sensing:

- $\alpha_r \in [0,1]$ is the proportion of the weight (or the "load-share") of the object supported by the robot.
- δ_1, δ_2 are thresholds on the load share specifying holding and releasing states.

Other:

- l_h is the radius of a "handover zone" centered around the robot's base.
- e is a boolean variable used to specify that the handover has started.

Assumptions: We represent all the poses in the frame attached to the base of the robot, as shown in Fig. 2. We assume that the human hand always remains in the dextrous workspace of the robot. We consider that the human is ready for the handover if the human hand is within a region of radius l_h centered at the robot's base, which we call the *handover zone*. For human-to-robot handovers, we assume that the human hand contains the object at the start of the handover, and also assume that the object's drop-off destination location is in the dextrous workspace of the robot. For robot-to-human handovers, we assume that the object is initially in the dextrous workspace of the robot. For simplicity, we consider that there is only one object in the workspace, but the formulation can be easily extended to multiple objects.

A. System Representation

We model the robot's end-effector's motion as a linear system with its pose and gripper openness making up the system state $\mathbf{x}$. The end-effector velocity and the gripper's opening velocity are the control input $\mathbf{u}$, constrained by $\mathbf{u_{min}} \leq \mathbf{u} \leq \mathbf{u_{max}}$.

$$\dot{\mathbf{x}} = \left[\dot{\mathbf{p}}_\mathbf{r}, \dot{\mathbf{q}}_\mathbf{r}, \dot{g}_r\right]^T = \mathbf{u} \tag{2}$$

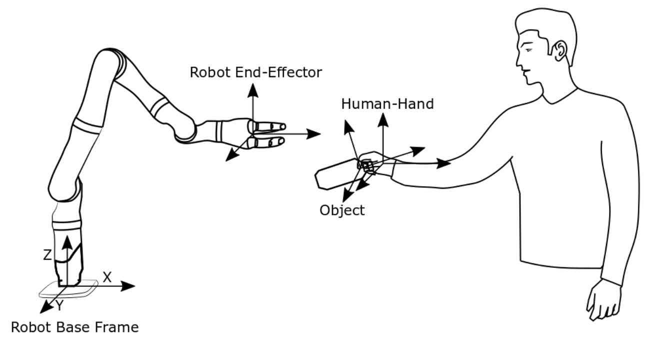


Fig. 2: Human-Robot object handover reference frames. All poses are expressed in the frame attached to the base of the robot.

We choose velocity control as it enables control over the timing of the pose trajectory. Also, velocity control is the most suitable choice for reactive trajectory modification and online motion planning [22]. We represent the world state $\mathbf{w}$ as the pose of the human's hand, the human's hand's openness, and the pose of the object.

$$\mathbf{w} = \left[\mathbf{p_h}, \mathbf{q_h}, g_h, \mathbf{p_o}, \mathbf{q_o}\right]^T \tag{3}$$

To simplify the specification formulae, we create sets of robot, human and object states and represent them using the discrete variable $o \in \{o_s, o_r, o_h, o_g\}$, defined in Table I. If the robot's gripper is equipped with a force sensor, the load-share estimate α_r is used to determine o, similar to Medina *et al.* [9]. Otherwise, we use the human's hand state g_h and the robot end-effector state g_r.

In the following sections we deal with human-to-robot handovers and robot-to-human handovers separately.

B. Specifications for Human-to-Robot Handovers

From a receiver's perspective, a handover consists of three phases: a reach phase in which the receiver moves to the handover location, a transfer phase in which the receiver grasps the object and the giver releases the object, and a retreat phase in which the receiver moves to the object's drop-off destination location. When the robot is the receiver, we specify its behavior in terms of timing constraints on these three phases (Table II top).

- Reach: The robot should reach the handover location $[\mathbf{p_l}, \mathbf{q_l}]$ within t_1 seconds after the handover signal e.
- Transfer: The robot should grasp the object within t_2 seconds after it reaches the object's location.
- Retreat: The robot should retreat to the object's drop-off location $[\mathbf{p_d}, \mathbf{q_d}]$ within t_3 seconds after it has the object and release the object in t_4 seconds after reaching the object's drop-off location.

C. Specifications for Robot-to-Human Handovers

From a giver's perspective, a handover consists of four phases: a pick-up phase in which the giver moves to the object's location and grasps the object, a reach phase in which the giver moves the object to the handover location, a transfer phase in which the giver releases the object and the

TABLE I: Discrete object states defined for robots without and with a gripper force sensor.

State	Formula (without Force Sensor)	Formula (with Force Sensor)
o_r (object is with the robot)	$(\lVert\mathbf{p_o}-\mathbf{p_r}\rVert \le \varepsilon_p \wedge \lvert\lvert g_r - g_*\rvert\rvert \le \varepsilon_g) \wedge \neg(\lVert\mathbf{p_o}-\mathbf{p_h}\rVert \le \varepsilon_p \wedge \lvert\lvert g_h - g_*\rvert\rvert \le \varepsilon_g)$	$\alpha_r > \delta_2$
o_h (object is with the human)	$\neg(\lVert\mathbf{p_o}-\mathbf{p_r}\rVert \le \varepsilon_p \wedge \lvert\lvert g_r - g_*\rvert\rvert \le \varepsilon_g) \wedge (\lVert\mathbf{p_o}-\mathbf{p_h}\rVert \le \varepsilon_p \wedge \lvert\lvert g_h - g_*\rvert\rvert \le \varepsilon_g)$	$\alpha_r \le \delta_1 \wedge \lVert\mathbf{p_o}-\mathbf{p_h}\rVert \le \varepsilon_p$
o_s (object is shared by both)	$(\lVert\mathbf{p_o}-\mathbf{p_r}\rVert \le \varepsilon_p \wedge \lvert\lvert g_r - g_*\rvert\rvert \le \varepsilon_g) \wedge (\lVert\mathbf{p_o}-\mathbf{p_h}\rVert \le \varepsilon_p \wedge \lvert\lvert g_h - g_*\rvert\rvert \le \varepsilon_g)$	$\delta_2 \ge \alpha_r > \delta_1$
o_g (object is on the ground)	$\neg(\lVert\mathbf{p_o}-\mathbf{p_r}\rVert \le \varepsilon_p \wedge \lvert\lvert g_r - g_*\rvert\rvert \le \varepsilon_g) \wedge \neg(\lVert\mathbf{p_o}-\mathbf{p_h}\rVert \le \varepsilon_p \wedge \lvert\lvert g_h - g_*\rvert\rvert \le \varepsilon_g)$	$\alpha_r \le \delta_1 \wedge \lVert\mathbf{p_o}-\mathbf{p_h}\rVert > \varepsilon_p$

receiver grasps the object, and a retreat phase in which the giver moves back to its original position. When the robot is the giver, we specify its behavior in terms of timing constraints on these four phases (Table II bottom).

- Pick-up: The robot should reach the object's location within t_5 seconds and grasp the object within t_6 seconds after reaching the object's location.
- Reach: The robot should take the object to the handover location $[\mathbf{p_l}, \mathbf{q_l}]$ within t_7 seconds after the handover signal e.
- Transfer: The robot should release the object within t_8 seconds after the object is shared by both.
- Retreat: The robot should retreat to a home position $[\mathbf{p}_\eta, \mathbf{q}_\eta]$ within t_9 seconds after the human has received the object.

V. Specifying Different Handover Strategies

To illustrate the flexibility of our approach, we list specifications for four different handover strategies, three of which are found in the literature. These differ only in the way the reach phase is specified. For each of these strategies, the specification in Table III replaces the reach phase specification in Table II.

- Proactive, Predetermined: The robot should reach a predefined or offline computed handover location $[\mathbf{p}_*, \mathbf{q}_*]$ without waiting for the human's hand to enter the handover zone. This behavior is similar to the robot's behavior presented by Moon *et al.* [3] and the "proactive" strategy presented by Huang *et al.* [4].
- Proactive, Towards Human: The robot should reach the human's hand without waiting for the human's hand to enter the handover zone.
- Reactive, Predetermined: The robot should reach a predefined or offline computed handover location $[\mathbf{p}_*, \mathbf{q}_*]$, only when the human hand is in the handover zone. This is similar to the "reactive" strategy by Huang *et al.* [4].
- Reactive, Towards Human: The robot should reach the human's hand, only when the human's hand is in the handover zone. This behavior is similar to the behaviors presented by Micelli *et al.* [15] and Medina *et al.* [9].

VI. Evaluation

In this section, we present simulated runs to demonstrate the different robot behaviors synthesized from the specifications described in Section IV. We also show how different choices of reach-time parameters affect the expected success rate of the handover, as evaluated on a public data-set of human-human handovers. We only present the simulations of the reach phase since the other phases have the same specifications in all the presented strategies. Also, we only consider the position trajectory of the human hand and synthesize the position trajectory of the robot, since the orientation trajectory of the robot depends on the offset $\mathbf{q}_\delta$ specific to the object being handed over.

A. Implementation

We use the MATLAB 'BluSTL' toolbox [23] to synthesize controllers from the system dynamics and specifications. This toolbox implements the RHC synthesis algorithm described in Section III-B. Since this toolbox accepts only linear and affine predicates, we use the l_1 norm in the specifications. For the objective function, J in Eq. 1, we use

$$J(\mathbf{x}(x_t, \mathbf{u}^{H,t}), \mathbf{u}^{H,t}) = \sum_{k=0}^{H} \lvert\lvert \mathbf{u}_{t+k} \rvert\rvert \tag{4}$$

to suggest that the robot move with minimum mean velocity.

The receding horizon control synthesis from the STL specifications depends on the predicted behavior of the environment w over the horizon H. Specifically, the prediction horizon H has to be greater than the maximum t_i value in the specifications. At each time-step, we predict the motion of the human by a Linear Dynamical System (LDS) $\dot{\mathbf{p_h}} = \mathbf{A}\mathbf{p_h}$. Similar to the approach used by Medina *et al.* [9], we use the pose data of the human for a pre-defined time interval before the current time-step and estimate the matrix $\mathbf{A}$ using least squares approximation. Then the predicted motion of the human is given by:

$$\mathbf{p_h}(t_0 + t) = \mathbf{p_h}(t_0) + (t\delta_t)\mathbf{A}\mathbf{p_h}(t_0) \ \forall\ t \in [0, H] \tag{5}$$

where H is the prediction horizon and δ_t is the sampling time. We update this estimate at each time-step using the position of the human hand and generate the control input for the next time-step. If no feasible control input is found, the robot stops.

B. Simulations

Fig. 3 shows the simulated robot end-effector's trajectories for each of the four reach-phase strategies for the same human hand motion. The values of the STL parameters are: t_1 (reach time) $= 3s$, $\varepsilon_p = 0.01m$, $|u_i| \le 1m/s$, δ_t (control input sampling interval) $= 0.2s$, $H = 15$ time-steps. For readability, we only show the x axis projection. Each strategy results in a different robot trajectory. In the "Predetermined" strategies (Fig. 3, top row), the robot goes to a fixed handover location regardless of the human's motion and thus may require adjustment on the human's part to reach the handover location. In the "Towards Human" strategies (Fig. 3, bottom row), the robot moves to the human hand's location. In the "Proactive" strategies (Fig. 3, left column), the robot starts to move towards the handover location even if the human hand

TABLE II: STL Specifications for Human-to-Robot Handovers

Robot's Role	Phase	Specification
Receiver	Reach	$\Box(e \Rightarrow \Diamond_{[0,t_1]}(\lVert\mathbf{p_l}-\mathbf{p_r}\rVert < \varepsilon_p \wedge \lVert\mathbf{q_l}-\mathbf{q_r}\rVert < \varepsilon_q))$
	Transfer	$\Box((\lVert\mathbf{p_o}-\mathbf{p_r}\rVert < \varepsilon_p \wedge (o == o_h \vee o == o_s)) \Rightarrow \Diamond_{[0,t_2]}(\lVert g_r - g_*\rVert < \varepsilon_g))$
	Retreat	$\Box((o == o_r) \Rightarrow \Diamond_{[0,t_3]}(\lVert\mathbf{p_r}-\mathbf{p_d}\rVert < \varepsilon_p \wedge \lVert\mathbf{q_r}-\mathbf{q_d}\rVert < \varepsilon_q))$
		$\Box((\lVert\mathbf{p_r}-\mathbf{p_d}\rVert < \varepsilon_p \wedge \lVert\mathbf{q_r}-\mathbf{q_d}\rVert < \varepsilon_q \wedge o == o_r) \Rightarrow \Diamond_{[0,t_4]}(\lVert g_r - 1\rVert < \varepsilon_g))$
Giver	Pick-up	$\Box((o == o_g) \Rightarrow \Diamond_{[0,t_5]}(\lVert\mathbf{p_o}-\mathbf{p_r}\rVert < \varepsilon_p \wedge \lVert\mathbf{q_o}-\mathbf{q_r}\rVert < \varepsilon_q))$
		$\Box((\lVert\mathbf{p_o}-\mathbf{p_r}\rVert < \varepsilon_p \wedge \lVert\mathbf{q_o}-\mathbf{q_r}\rVert < \varepsilon_q \wedge o == o_g) \Rightarrow \Diamond_{[0,t_6]}(\lVert g_r - g_*\rVert < \varepsilon_g))$
	Reach	$\Box(e \Rightarrow \Diamond_{[0,t_7]}(\lVert\mathbf{p_l}-\mathbf{p_r}\rVert < \varepsilon_p \wedge \lVert\mathbf{q_l}-\mathbf{q_r}\rVert < \varepsilon_q))$
	Transfer	$\Box((o == o_s) \Rightarrow \Diamond_{[0,t_8]}(\lVert g_r - 1\rVert < \varepsilon_g))$
	Retreat	$\Box((o == o_h) \Rightarrow \Diamond_{[0,t_9]}(\lVert\mathbf{p_r}-\mathbf{p_\eta}\rVert < \varepsilon_p \wedge \lVert\mathbf{q_r}-\mathbf{q_\eta}\rVert < \varepsilon_q))$

TABLE III: STL Specifications for Reach-Phase Strategies

Robot's Role	Strategy	Target	Specification
Receiver	Proactive	Predetermined	$\Box(\neg(o == o_r) \Rightarrow \Diamond_{[0,t_1]}(\lVert\mathbf{p_*}-\mathbf{p_r}\rVert < \varepsilon_p \wedge \lVert\mathbf{q_*}-\mathbf{q_r}\rVert < \varepsilon_q))$
		Towards Human	$\Box(\neg(o == o_r) \Rightarrow \Diamond_{[0,t_1]}(\lVert\mathbf{p_h}-\mathbf{p_r}\rVert < \varepsilon_p \wedge \lVert\mathbf{q_h}-\mathbf{q_r}-\mathbf{q_\delta}\rVert < \varepsilon_q))$
	Reactive	Predetermined	$\Box((\neg(o == o_r) \wedge \lVert\mathbf{p_h}\rVert \leq l_h) \Rightarrow \Diamond_{[0,t_1]}(\lVert\mathbf{p_*}-\mathbf{p_r}\rVert < \varepsilon_p \wedge \lVert\mathbf{q_*}-\mathbf{q_r}\rVert < \varepsilon_q))$
		Towards Human	$\Box((\neg(o == o_r) \wedge \lVert\mathbf{p_h}\rVert \leq l_h) \Rightarrow \Diamond_{[0,t_1]}(\lVert\mathbf{p_h}-\mathbf{p_r}\rVert < \varepsilon_p \wedge \lVert\mathbf{q_h}-\mathbf{q_r}-\mathbf{q_\delta}\rVert < \varepsilon_q))$
Giver	Proactive	Predetermined	$\Box((o == o_r) \Rightarrow \Diamond_{[0,t_7]}(\lVert\mathbf{p_*}-\mathbf{p_r}\rVert < \varepsilon_p \wedge \lVert\mathbf{q_*}-\mathbf{q_r}\rVert < \varepsilon_q))$
		Towards Human	$\Box((o == o_r) \Rightarrow \Diamond_{[0,t_7]}(\lVert\mathbf{p_h}-\mathbf{p_r}\rVert < \varepsilon_p \wedge \lVert\mathbf{q_h}-\mathbf{q_r}-\mathbf{q_\delta}\rVert < \varepsilon_q))$
	Reactive	Predetermined	$\Box((o == o_r \wedge \lVert\mathbf{p_h}\rVert \leq l_h) \Rightarrow \Diamond_{[0,t_7]}(\lVert\mathbf{p_*}-\mathbf{p_r}\rVert < \varepsilon_p \wedge \lVert\mathbf{q}*-\mathbf{q_r}\rVert < \varepsilon_q))$
		Towards Human	$\Box((o == o_r \wedge \lVert\mathbf{p_h}\rVert \leq l_h) \Rightarrow \Diamond_{[0,t_7]}(\lVert\mathbf{p_h}-\mathbf{p_r}\rVert < \varepsilon_p \wedge \lVert\mathbf{q_h}-\mathbf{q_r}-\mathbf{q_\delta}\rVert < \varepsilon_q))$

is not in the handover zone, while in "Reactive" strategies (Fig. 3, right column) the robot starts to move towards the handover location only when the human hand is in the handover zone. These strategies result in different values for human and robot idle times, which have been shown to be indicators of perceived human-robot fluency [24].

In addition to specifying overall strategies, the STL formulation allows users to specify high-level timing constraints for the handover, resulting in different controllers. To illustrate, Fig. 4 shows simulated robot trajectories for different values of t_1. The values of the other parameters are: $\varepsilon_p = 0.01m$, $|u_i| \leq 1m/s$, $\delta_t = 0.2s$, $H = 15$ time-steps. In this example, a specification of $t_1 = 1s$ (reaching the handover location within one second) results in the controller being unable to find a control input, causing the robot to stop. For $t_1 = 2s$ and $t_1 = 3s$ the robot reaches the handover location within the specified time limit, with different human idle times. The accompanying video shows the simulations.

C. Inferring Timing Parameters from a Handover Database

In addition to the possibility to synthesize a single controller for a given specification, our approach provides an additional benefit: the ability to estimate constraints on the high-level behavior of the robot under different circumstances. We illustrate this here by finding feasible bounds for t_1 based on a public database of human-human handovers [25]. This database of 1000 recordings was collected from 18 volunteers in 76 test configurations with different volunteer's starting positions, roles, objects to pass and motion strategies. In our evaluation, we only use the trajectories for "uncontrolled" (the experimenter is not one of the actors) and "no-approach" (the volunteers are already standing close to each other) configurations. This results in a dataset of 72 giving motions and 72 receiving motions.

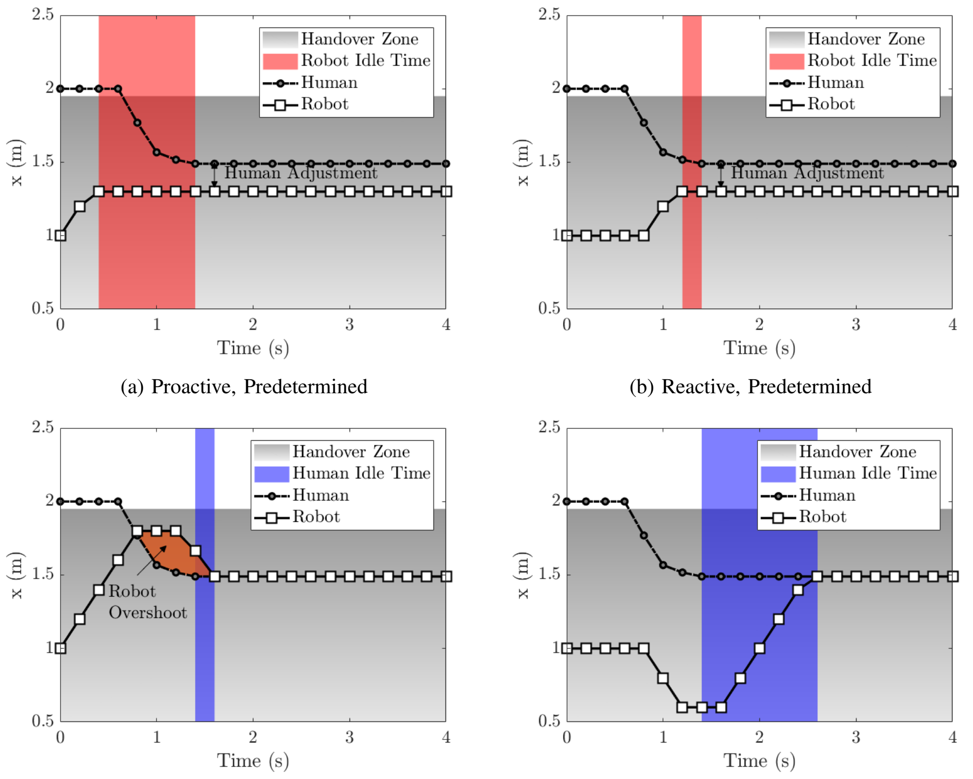
Table IV shows the percentage of the human's giving motions from the database for which the controller finds a feasible solution for a given reach-time value and a maximum velocity control input u_{max}. The values of the remaining STL parameters are: $\varepsilon_p = 0.01m$, $\delta_t = 0.2s$, $H = 20$ time-steps. An end-user could use results like these to estimate the timing, and thus the productivity, of a handover system given a safety cap on the robot's velocity.

TABLE IV: Handover success rate for different reach-times.

$u_{max}(m/s)$	t_1(s)	Successful Handovers
1	1s	14.3%
	2s	91.4%
	3s	100%
0.5	2s	2.9%
	3s	82.9%
	4s	97.1%

VII. Discussion

We present an approach to human-robot handovers that works at a high-level abstraction of goals and constraints. Our approach differs from existing handover controllers, which use hand-coded algorithms tuned via low-level parameters. In addition, the presented controller is synthesized online, enabling the robot to adapt to the human's observed behavior. To date, most of the literature on human-robot handovers uses offline motion planning, with very few online approaches. Also, almost all the work on handovers discusses a single robot behavior. Although it is conceivable to write a program with different robot behaviors and allow the user to switch between them, the approach in this paper proposes a more unified view of this central aspect of HRI. It also allows end-users to specify the handover in terms meaningful to them. Since it is difficult for naïve users to write STL specifications, we plan to develop an intuitive user interface using which they can modify the STL specifications presented in this paper. In the interface, users will be able to change the timing values of handover phases and switch between different strategies.



(a) Proactive, Predetermined (b) Reactive, Predetermined

(c) Proactive, Towards Human (d) Reactive, Towards Human

Fig. 3: Simulation runs of different handover strategies. "Proactive" strategies (left column) may result in shorter human idle time but longer robot idle time, while "Reactive" strategies (right column) may result in shorter robot idle time but longer human idle time. A predetermined handover location may require adjustment on the human's part (top row), while the human hand as the handover location (bottom row) does not require any adjustment on the human's part. However, this may result in robot trajectories with overshoot (bottom-left) or the robot initially moving away from the human (bottom-right), due to the prediction of the human-motion model.

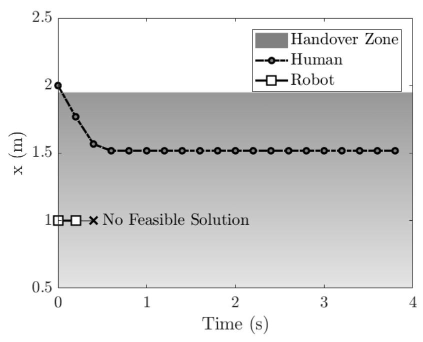
Using automatic synthesis based on formal models, our approach provides timing guarantees, which means that the robot will always obey timing constraints as long as the human behavior allows for it. It can also notify users of constraint violations. Furthermore, our approach enables estimates of constraint parameters based on human motion data. Another advantage of our approach is that alternative optimization criteria, for example, minimum jerk or minimum acceleration, can be used in Eq. 1 to change the robot's trajectory while obeying the timing constraints. In this paper, due to paucity of space we only presented simulation results by modeling the human's motion as a Linear Dynamical System. Alternative models such as linear/exponential extrapolation and neural networks could also be used to predict the human's motion.

In the formulation presented in this paper, we do not address obstacle or self-collision avoidance. This is a common limitation in existing online human-robot handover controllers [12], [15], [9], [7]. We can add obstacle location $\mathbf{p}_{\mathbf{obstacle}}$ as an additional environment variable in Eq. 3 and write a safety specification of the form $\square(||\mathbf{p_r} - \mathbf{p}_{\mathbf{obstacle}}|| > \varepsilon_p)$, so that the robot's end-effector does not collide with the obstacle. Furthermore, the collision avoidance of the robot's links can be encoded in STL by modeling the links, albeit at the expense of more complex computation.

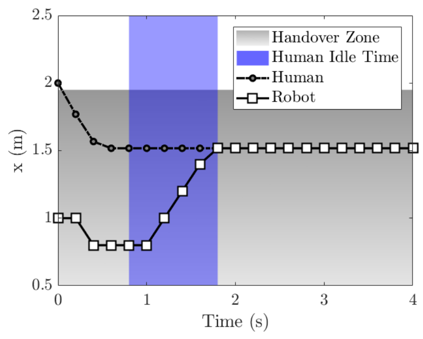
VIII. Conclusion and Future Work

We formulated the human-robot handover scenario using STL formulae and provided candidate specifications of eight different robot behaviors for bi-directional human-robot handovers. We demonstrated the feasibility of our approach via simulations, including using a public dataset of human-human handovers. To the best of our knowledge, this is the first work to use the automated synthesis of robot controllers from formal specifications for human-robot handovers. Our approach allows the user to specify the robot's behavior in terms of timing of different stages of the object handover. The controller is synthesized online in a receding horizon manner, with the human's motion being predicted at each time-step using a linear dynamical system model.

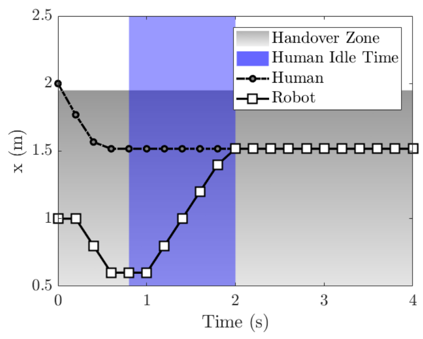
To demonstrate that our approach is practically feasible, we are currently implementing it on a collaborative robot arm. We will conduct human-participant experiments with naïve users to study the quantitative and qualitative outcomes of human-robot handovers with our approach and compare them with other state-of-the-art approaches from the literature. We are also extending our framework to include gaze and other gestures, which have been found to affect handover timing and quality.



(a) Reach-Time = 1s



(b) Reach-Time = 2s



(c) Reach-Time = 3s

Fig. 4: Representative simulations of different reach-time values (reactive, towards human strategy). A timing constraint of $t_1 = 1$s is infeasible given the safety cap on the robot's velocity. Increasing the constraint to $t_1 = 2$s allows the robot end-effector's trajectory to converge to the human hand's trajectory within the specified time limit. Increasing the reach time constraint from $t_1 = 2$s to $t_1 = 3$s increases the time taken by the robot end-effector's trajectory to converge to the human hand's trajectory and thus increases human idle time. This increase is less than 1s, because the timing parameter is only an upper bound.

REFERENCES

[1] M. Huber, M. Rickert, A. Knoll, T. Brandt, and S. Glasauer, "Human-robot interaction in handing-over tasks," in *17th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, 2008, pp. 107–112.

[2] K. Strabala, M. K. Lee, A. Dragan, J. Forlizzi, S. S. Srinivasa, M. Cakmak, and V. Micelli, "Toward seamless human-robot handovers," *Journal of Human-Robot Interaction*, vol. 2, no. 1, pp. 112–132, 2013.

[3] A. Moon, D. M. Troniak, B. Gleeson, M. K. Pan, M. Zheng, B. A. Blumer, K. MacLean, and E. A. Croft, "Meet me where I'm gazing: how shared attention gaze affects human-robot handover timing," in *9th ACM/IEEE International Conference on Human-Robot Interaction*, 2014, pp. 334–341.

[4] C.-M. Huang, M. Cakmak, and B. Mutlu, "Adaptive coordination strategies for human-robot handovers." in *Robotics: Science and Systems (RSS)*, 2015.

[5] S. M. zu Borgsen, J. Bernotat, and S. Wachsmuth, "Hand in hand with robots: Differences between experienced and naive users in human-robot handover scenarios," in *International Conference on Social Robotics*, 2017, pp. 587–596.

[6] M. Cakmak, S. S. Srinivasa, M. K. Lee, J. Forlizzi, and S. Kiesler, "Human preferences for robot-human hand-over configurations," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011, pp. 1986–1993.

[7] K. Yamane, M. Revfi, and T. Asfour, "Synthesizing object receiving motions of humanoid robots with human motion database," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2013, pp. 1629–1636.

[8] W. P. Chan, M. K. Pan, E. A. Croft, and M. Inaba, "Characterization of handover orientations used by humans for efficient robot to human handovers," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 1–6.

[9] J. R. Medina, F. Duvallet, M. Karnam, and A. Billard, "A human-inspired controller for fluid human-robot handovers," in *IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, 2016, pp. 324–331.

[10] D. Vogt, S. Stepputtis, B. Jung, and H. B. Amor, "One-shot learning of human–robot handovers with triadic interaction meshes," *Autonomous Robots*, vol. 42, no. 5, pp. 1053–1065, 2018.

[11] E. A. Sisbot and R. Alami, "A human-aware manipulation planner," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1045–1057, 2012.

[12] M. Prada, A. Remazeilles, A. Koene, and S. Endo, "Implementation and experimental validation of Dynamic Movement Primitives for object handover," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2014, pp. 2146–2153.

[13] J. Waldhart, M. Gharbi, and R. Alami, "Planning handovers involving humans and robots in constrained environment," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 6473–6478.

[14] O. Maler and D. Nickovic, "Monitoring temporal properties of continuous signals," in *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems*. Springer, 2004, pp. 152–166.

[15] V. Micelli, K. Strabala, and S. Srinivasa, "Perception and control challenges for effective human-robot handoffs," in *Robotics: Science and Systems (RSS) Workshop on RGB-D Cameras*, 2011.

[16] H. Kress-Gazit, M. Lahijanian, and V. Raman, "Synthesis for Robots: Guarantees and Feedback for robot behavior," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 211–236, 2018.

[17] D. Araiza-Illan, D. Western, A. Pipe, and K. Eder, "Coverage-Driven Verification," in *Haifa Verification Conference*, 2015, pp. 69–84.

[18] M. Webster, D. Western, D. Araiza-Illan, C. Dixon, K. Eder, M. Fisher, and A. G. Pipe, "An assurance-based approach to verification and validation of human–robot teams," *arXiv preprint arXiv:1608.07403*, 2016.

[19] W. Li, D. Sadigh, S. S. Sastry, and S. A. Seshia, "Synthesis for human-in-the-loop control systems," in *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, 2014, pp. 470–484.

[20] V. Raman, A. Donzé, M. Maasoumy, R. M. Murray, A. Sangiovanni-Vincentelli, and S. A. Seshia, "Model predictive control with signal temporal logic specifications," in *IEEE 53rd Annual Conference on Decision and Control (CDC)*, 2014, pp. 81–87.

[21] V. Raman, A. Donzé, D. Sadigh, R. M. Murray, and S. A. Seshia, "Reactive synthesis from signal temporal logic specifications," in *18th International Conference on Hybrid Systems: Computation and Control*, 2015, pp. 239–248.

[22] A. Zelenak, C. Peterson, J. Thompson, and M. Pryor, "The advantages of velocity control for reactive robot motion," in *ASME Dynamic Systems and Control Conference*, 2015, pp. V003T43A003–V003T43A003.

[23] V. Raman and A. Donzè, "BluSTL on github," https://github.com/BluSTL/BluSTL, accessed: 2018-11-10.

[24] G. Hoffman, "Evaluating fluency in human-robot collaboration," in *Robotics: Science and Systems (RSS) Workshop on Human Robot Collaboration*, 2013.

[25] A. Carf, F. Foglino, B. Bruno, and F. Mastrogiovanni, "A multi-sensor dataset of human-human handover," *Data in Brief*, vol. 22, pp. 109 – 117, 2019.