A Signal Temporal Logic Planner for Ergonomic Human–Robot Collaboration

Giuseppe Silano[1], Amr Afifi[2], Martin Saska[1], and Antonio Franchi[2,3,4]

***Abstract*— This paper proposes a method for designing human-robot collaboration tasks and generating corresponding trajectories. The method uses high-level specifications, expressed as a Signal Temporal Logic (STL) formula, to automatically synthesize task assignments and trajectories. To illustrate the approach, we focus on a specific task: a multi-rotor aerial vehicle performing object handovers in a power line setting. The motion planner considers limitations, such as payload capacity and recharging constraints, while ensuring that the trajectories are feasible. Additionally, the method enables users to specify robot behaviors that take into account human comfort (e.g., ergonomics, preferences) while using high-level goals and constraints. The approach is validated through numerical analyzes in MATLAB and realistic Gazebo simulations using a mock-up scenario.**

***Index Terms*— Aerial Systems: Applications, Multi-Rotor UAVs, Human-Aware Motion Planning.**

I. Introduction

In the field of robotics, Aerial Robots (ARs) and Multi-Rotor Aerial Vehicles (MRAVs) have gained significant attention in recent years due to their impressive agility, maneuverability, and ability to be equipped with a variety of onboard sensors [1]. Their modular design and versatility have made them suitable for a wide range of applications, including those which involve either contactless [2] or physical interaction with their surroundings [3].

There are many real-world examples where the use of aerial robots is advantageous, such as for work environments at heights, wind turbines, large construction sites, or power transmission lines [4], [5]. These types of settings often require specialized and trained personnel to use expensive equipment and special vehicles. The use of aerial robots as robotic co-workers [6], [7] in these scenarios can, for example, facilitate tasks by flying to the target location and carrying tools, reducing the physical and cognitive load on human operators, etc.. However, in order to realize these benefits, it is important to consider human ergonomics and safety [8] when designing aerial robots, particularly multi-rotors.

Despite the potential of MRAVs to work closely with human operators, their use in such scenarios is still limited. In contrast, there is a wealth of research on Human-Robot Interaction (HRI) involving ground robots and human partners [9]. For example, previous studies have looked at the use of manipulators to assist humans in handling heavy or bulky objects, or during assembly tasks [10]. The issue of object handover has also been extensively studied in the literature [11].

Therefore, to allow ARs to effectively collaborate with human workers and address critical ergonomic and safety concerns while additionally minimizing the physical and cognitive demands on human operators, advanced task and motion planning techniques are required. Temporal Logic (TL) can fulfill this role by providing a mathematical framework for expressing complex specifications that combine natural language commands with temporal and Boolean operators [12]. In particular, Signal Temporal Logic (STL) [13] is endowed with a metric called *robustness*, which not only allows one to determine whether a system's execution satisfies certain requirements, but also provides a measure of how well these requirements have been satisfied. This leads to an optimization problem aimed at maximizing the robustness score, thereby generating the optimal feasible trajectory for the system that meets desired specifications.

In this paper, we present an MRAV motion planner that leverages STL specifications to facilitate ergonomic human-robot collaboration. As a motivating example, we consider the task of an MRAV performing object handovers in a power line scenario, as depicted in Figure 1. The mission requirements are expressed as an STL formula. We maximize its robustness by formulating a nonlinear non-convex max-min optimization problem. To address the complexity of this nonlinear optimization, we employ a hierarchical approach that first solves an Integer Linear Programming (ILP) problem, and then passes the result to the final STL optimizer.

A. Related work

The process of handover typically encompasses multiple stages, including the *approach*, *reach*, and *transfer* phases, as noted in previous studies [6], [7]. Many researchers have typically studied each phase individually, but there are a few notable exceptions. In [14], a control architecture for fluid handovers that addresses all phases in a cohesive

[1]Faculty of Electrical Engineering, Department of Cybernetics, Czech Technical University in Prague, 12135 Prague, Czech Republic, (emails: {giuseppe.silano, martin.saska}fel.cvut.cz).

[2]Robotics and Mechatronics Department, Electrical Engineering, Mathematics, and Computer Science (EEMCS) Faculty, University of Twente, 7500 AE Enschede, The Netherlands, (emails: {a.n.m.g.afifi, a.franchi}utwente.nl).

[3]Department of Computer, Control and Management Engineering, Sapienza University of Rome, 00185 Rome, Italy, antonio.franchi@uniroma1.it

[4]LAAS-CNRS, Université de Toulouse, 31000 Toulouse, France, antonio.franchi@laas.fr

This work was partially funded by the European Union's Horizon 2020 research and innovation programme AERIAL-CORE under grant agreement no. 871479, by the CTU grant no. SGS23/177/OHK3/3T/13, by the Czech Science Foundation (GAČR) within research project no. 23-07517S, and by the OP VVV funded project CZ.02.1.01/0.0/0.0/16 019/0000765 "Research Center for Informatics".

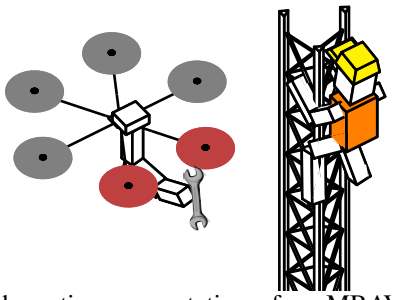

Fig. 1: A schematic representation of an MRAV delivering a tool to a human worker in a power line scenario.

manner is proposed. This approach takes into account the interactions that occur during the handover and specifically aims to minimize unwanted wrench components that are not essential for moving and holding the object. However, the proposed control scheme does not explicitly factor in safety and ergonomics, which are crucial considerations in a framework designed for collaboration between aerial robots and humans, particularly in high-risk environments.

The inclusion of human comfort and ergonomics into robot control and planning software has also been explored in the literature. One of the earliest works in this area is [15], which develops a manipulation planner accounting for various human factors, such as ergonomics and field of view. Subsequent studies have advanced this approach. For example, [16] proposes a method for computing human joint torques based on a whole-body dynamic model and then controlling a ground mobile manipulator to minimize the overloading of human joints. However, none of these works consider ARs as robotic co-workers interacting with a human operator.

In addition to the control and planning aspects, some studies concentrate on improving these elements by equipping ARs with the ability to perceive the human subject through sensors. This is crucial as any reduction in visibility of the human collaborator could negatively impact the task. Perception-constrained control is a key consideration in these studies. For example, [7] proposes an Nonlinear Model Predictive Control (NMPC) formulation that incorporates human ergonomics and comfort as objectives, while also enforcing perception and actuation limits. Other research focuses on using dynamic programming to ensure safety so that the robot can never cause injury to the interacting human. For instance, [6] presents a comprehensive framework formulated as a constrained quadratic programming problem for controlling an aerial manipulator during physical interactions with a human operator. While these approaches enable ARs to interact safely and ergonomically with a single human operator, they do not account for scenarios where multiple operators are involved and tools need to be carried multiple times throughout the mission.

Alternatively, the field of automatic synthesis of robot controllers for human-robot handovers from formal specifications has also been explored. For example, the authors in [17] present a controller for human-robot handovers automatically generated from high-level specifications in STL. This approach offers formal guarantees on the timing of each handover phase. In the realm of formal methods, [18] uses probabilistic model-checking in a human-robot handover task, validating a controller with respect to (w.r.t.) safety and liveness [19] specifications. Finally, the authors in [17] describe a formalism for human-in-the-loop control synthesis and set up a semi-autonomous controller from TL specifications. However, as far as the authors are aware, this is the first study that addresses the task assignment and trajectory generation problem for ARs, specifically for MRAVs, to enhance human-robot ergonomic collaboration.

B. Contributions

This paper presents a novel method in designing tasks and motion planning for human-robot collaboration prioritizing ergonomics. The approach is demonstrated using the specific task of an MRAV performing object handovers in a power line setting. The method employs STL to generate optimal trajectories that comply with the vehicle's dynamics and velocity and acceleration limits, while also fulfilling various mission specifications, such as collision avoidance and safety requirements. A hierarchical strategy is implemented to address the complexity of the problem, where an initial guess solution is obtained by solving an ILP problem to act as the starting point for the STL optimizer. This strategy builds upon prior work [20], [21] by allowing users to specify robot behaviors that take into account human comfort and preferences using high-level goals and constraints. This includes computing trajectories that consider payload capacity limitations and refilling stations for longer-duration operations. Additionally, a method for computing the initial solution for the optimization problem is proposed.

The proposed approach offers several key benefits: (i) by using STL formulae, the framework can take into account explicit time requirements, making it easy to adapt and customize for various applications; (ii) the concise and unambiguous STL formulation enables end-users to specify robot behavior in terms they can understand, such as the direction of approach and zones to avoid, reducing the need for hand-coded algorithms; (iii) using automatic synthesis based on formal models, our approach provides timing guarantees. This means that the vehicle will always obey timing constraints, as long as the human behavior allows for it. It also notifies users of any constraint violations.

The approach has been validated through numerical simulations in MATLAB. Additionally, Gazebo simulations were used to demonstrate the approach's effectiveness in a scenario that closely resembles real-world implementation.

II. Problem Description

The focus of this paper is to enhance the ergonomic collaboration between humans and robots. The scenario under examination is that of an MRAV equipped with a manipulation arm performing object handovers in a power line setting. This scenario is depicted in Figure 2. The objective is to design a trajectory for the drone that takes

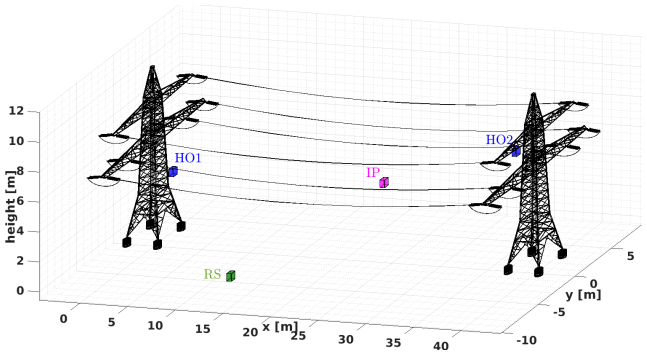

Fig. 2: Power line scenario for the ergonomic human-robot collaboration. Human operators (HO) are represented in blue, while the MRAV's initial position (IP) and the refilling station (RS) are depicted in magenta and green, respectively.

into account human ergonomic needs by approaching the operator from the front, either from the left or right, from above or below, and never from behind [8]. To simplify the problem, the location of the handover operation is represented as a 3D space for each operator. We assume that the MRAV begins the mission with a tool. For ease of understanding, we consider that only one object can be delivered at a time. However, the method can be easily extended to multiple objects. Once the MRAV reaches the operator, it is assumed that an onboard low-level controller handles the handover procedure, e.g. [6], [7]. The MRAV has limited velocity, acceleration, and payload capacity, meaning the number of tools it can carry is restricted. Furthermore, there is a refilling station on the ground along the power line where the drone can reload tools and resume operation. The goal is to plan a trajectory for the MRAV to complete the mission within a specified maximum time frame, while also satisfying dynamics and capacity constraints. Further, safety requirements must be met, such as avoiding obstacles and never approaching the operator from behind. It is assumed that a map of the environment, including a polyhedral representation of obstacles like power towers and cables, is known in advance.

III. Preliminaries

Let us consider a discrete-time dynamical system S represented in the form $x_{k+1} = f(x_k; u_k)$, where x_{k+1}, x_k 2 X $\mathbb{R}^n$ are the next and current states of the system S, respectively, and u_k 2 U $\mathbb{R}^m$ is the control input. Let us also assume that $f : X$ U $!$ X is differentiable in both arguments and locally Lipschitz. Therefore, given an initial state x_0 2 X_0 $\mathbb{R}^n$ and a time vector $\mathbf{t} = (t_0; :::; t_N)^>$ 2 $\mathbb{R}^{N+1}$, with N 2 $\mathbb{N}_{>0}$ being the number of samples that describe the evolution of the system S with sampling period T_s 2 $\mathbb{R}_{>0}$, we can define the finite control input sequence $\mathbf{u} = (u_0; :::; u_{N\ 1})^>$ 2 $\mathbb{R}^N$ as the input to provide to the system S to attain the unique sequence of states $\mathbf{x} = (x_0; :::; x_N)^>$ 2 $\mathbb{R}^{N+1}$.

Let us consider a Generically Tilted Multi-Rotor (GTMR) model [22] to describe the vehicle's dynamics. Also, let us denote with F_W and F_B the world frame and body frame reference systems, respectively. The body frame is attached to the GTMR such that the origin of the frame O_B coincides with the Center of Mass (CoM) of the vehicle. The position of the origin O_B of the body frame F_B w.r.t. the world frame F_W is denoted with $\mathbf{p} = (p^{(1)}; p^{(2)}; p^{(3)})^>$ 2 $\mathbb{R}^3$, while the velocity and acceleration of O_B in F_W are denoted with $\mathbf{v} = (v^{(1)}; v^{(2)}; v^{(3)})^>$ 2 $\mathbb{R}^3$ and $\mathbf{a} = (a^{(1)}; a^{(2)}; a^{(3)})^>$ 2 $\mathbb{R}^3$, respectively.

Hence, we can define the state sequence $\mathbf{x}$ and the control input sequence $\mathbf{u}$ of a GTMR model as $\mathbf{x} = (\mathbf{p}^{(1)}; \mathbf{v}^{(1)}; \mathbf{p}^{(2)}; \mathbf{v}^{(2)}; \mathbf{p}^{(3)}; \mathbf{v}^{(3)})^>$ and $\mathbf{u} = (\mathbf{a}^{(1)}; \mathbf{a}^{(2)}; \mathbf{a}^{(3)})^>$, where $\mathbf{p}^{(j)}$, $\mathbf{v}^{(j)}$, and $\mathbf{a}^{(j)}$, with $j = \{1; 2; 3\}$, represent the sequences of position, velocity, and acceleration of the vehicle along the j-axis of the world frame F_W, respectively.

Finally, let us denote with $p_k^{(j)}$, $v_k^{(j)}$, $a_k^{(j)}$, and t_k, with k 2 $\mathbb{N}_0$, the k-th element of the sequences $\mathbf{p}^{(j)}$, $\mathbf{v}^{(j)}$, $\mathbf{a}^{(j)}$, and vector $\mathbf{t}$, respectively.

A. Signal temporal logic

Definition 1 *(Signal Temporal Logic)*: STL is a concise and unambiguous specification language for describing the temporal behavior of real-valued signals [13]. One major benefit of using STL formulae for motion planning over traditional algorithms [23] is that all mission specifications can be encapsulated into a single formula $'$. For example, the statement "at least two vehicles must complete tasks A and B before task C is performed, one of them must execute task D within the time interval $[t_1; t_2]$, while all of them must avoid obstacles and remain within the designated workspace" can be expressed into a single STL formula $'$. The syntax and semantics of STL are detailed in [13], [24], however a brief overview is provided here. STL's grammar enables the representation of complex behavioral requirements using temporal operators, such as *until* (U), *always* ( ), and *eventually* ( ), as well as logical operators like *and* ($\wedge$), *or* ($\_$), and *negation* ($:$). These operators act on atomic propositions (also known as *predicates*), which are simple statements or assertions that are either *true* ($>$) or *false* ($?$). Examples of atomic propositions include belonging to a particular region or comparing real values (e.g., a distance exceeding a threshold). An STL formula $'$ is considered valid if it evaluates to a *true* ($>$) logic value, and invalid ($?$) otherwise. For instance, informally, the formula $'_1 U_I '_2$ means that $'_2$ must hold at some point within the time interval I and, until then, $'_1$ must hold without interruption.

B. Robust signal temporal logic

Definition 2 *(STL Robustness)*: Uncertainties, dynamic conditions, and unexpected events can all impact the satisfaction of an STL formula $'$ (Def. 1). To account for these factors and ensure a margin of satisfaction for an STL formula $'$, the concept of *robust semantics* for STL formulae has been developed [13], [24], [25]. This *robustness*, represented by , is a quantitative metric that helps guide the optimization process towards finding the best feasible

solution for meeting the statement (mission) requirements. It can be formally defined using the following recursive formulae:

$$
\begin{aligned}
p_i(\mathbf{x}; t_k) &= \mu_i(\mathbf{x}; t_k); \\
\neg\varphi(\mathbf{x}; t_k) &= -\varphi(\mathbf{x}; t_k); \\
\varphi_1 \wedge \varphi_2(\mathbf{x}; t_k) &= \min\left(\varphi_1(\mathbf{x}; t_k);\ \varphi_2(\mathbf{x}; t_k)\right); \\
\varphi_1 \vee \varphi_2(\mathbf{x}; t_k) &= \max\left(\varphi_1(\mathbf{x}; t_k);\ \varphi_2(\mathbf{x}; t_k)\right); \\
\Box_I \varphi(\mathbf{x}; t_k) &= \min_{t'_k \in [t_k + I]} \varphi(\mathbf{x}; t'_k); \\
\Diamond_I \varphi(\mathbf{x}; t_k) &= \max_{t'_k \in [t_k + I]} \varphi(\mathbf{x}; t'_k); \\
\varphi_1 U_I \varphi_2(\mathbf{x}; t_k) &= \max_{t'_k \in [t_k + I]} \min\left(\varphi_2(\mathbf{x}; t'_k)\right); \\
&\qquad \min_{t''_k \in [t_k; t'_k]} \left(\varphi_1(\mathbf{x}; t''_k)\right);
\end{aligned}
$$

where $t_k + I$ represents the Minkowski sum of the scalar t_k and the time interval I. The formulae above consist of a set of *predicates*, p_i, along with their corresponding real-valued function $\mu_i(\mathbf{x}; t_k)$, each of which is considered true if its robustness value is greater than or equal to zero, and false otherwise. The whole formula behaves like a logical formula, which is deemed false if any of the predicates are false. As an illustration, using the previous example of belonging to a particular region, we can first divide the mission into a set of predicates, i.e., bounds on distance relative to upper and lower margins. Next, we compute the robustness value associated with each predicate (how well or poorly the specification is being satisfied). Finally, all predicates are evaluated using the logical formulae presented earlier. The result is a numerical value that indicates whether the specification is being met and to what degree. Further information on this can be found in [13], [24], [25]. In this case, we say that $\mathbf{x}$ satisfies the STL formula φ at time t_k if $\varphi(\mathbf{x}; t_k) > 0$, and $\mathbf{x}$ violates φ if $\varphi(\mathbf{x}; t_k) \leq 0$.

Therefore, we can compute the control inputs $\mathbf{u}$ that maximize robustness over the set of finite state and input sequences $\mathbf{x}$ and $\mathbf{u}$, respectively. This optimal sequence $\mathbf{u}^\star$ is considered valid if $\varphi(\mathbf{x}^\star; t_k)$ is positive, with $\mathbf{x}^\star$ and $\mathbf{u}^\star$ satisfying the dynamics of the system. The larger the value of $\varphi(\mathbf{x}^\star; t_k)$, the more robust the system's behavior is considered to be.

Definition 3 *(Smooth Approximation)*: A limitation of the standard robustness measure is that it is non-smooth and non-convex for many useful specifications due to the inclusion of $\min$ and $\max$ operators. To overcome this issue, recent research has focused on smooth approximations of the robustness measure $\tilde{\varphi}(\mathbf{x}; t_k)$ [26], [27]. These approximations allow for the use of efficient gradient-based optimization methods and have been shown to perform well on a wide range of problems, providing significant speed and scalability improvements [28].

One of the approximations of the robustness measures is the Arithmetic-Geometric Mean (AGM) robustness. This approach retains many of the computational benefits of the most commonly used Log-Sum-Exponential (LSE) method [26]. Furthermore, AGM robustness is more conservative in the sense that trajectories derived using this approach tend to be more robust to external disturbances. However, the AGM robustness is almost smooth everywhere, i.e., its analytical gradient does not always exist. While this issue with singularities may pose challenges for the solver, it is still important to keep in mind that the optimization should not reach the boundary conditions. For example, in the case of belonging to the workspace area, there is no reason for the vehicle to be at the boundaries. In light of this, we choose the AGM as smooth approximation for the robustness measure $\tilde{\varphi}(\mathbf{x}; t_k)$. The full description of the AGM robustness syntax and semantics can be found in [27] and is not given here for the sake of brevity.

Definition 4 *(STL Motion Planner)*: By encoding the mission specifications detailed in Section II as an STL formula φ and replacing its robustness $\varphi(\mathbf{x}; t_k)$ with the smooth approximation $\tilde{\varphi}(\mathbf{x}; t_k)$ (as defined in Def. 3), the problem of generating trajectories for the GTMR model can be formulated as the optimization problem [20]:

$$
\begin{aligned}
\underset{\mathbf{p}^{(j)}; \mathbf{v}^{(j)}; \mathbf{a}^{(j)}}{\text{maximize}} \quad & \tilde{\varphi}(\mathbf{p}^{(j)}; \mathbf{v}^{(j)}) \\
\text{s.t.} \quad & |v_k^{(j)}| \leq v^{(j)}; |a_k^{(j)}| \leq a^{(j)}; \quad ; \\
& \mathbf{S}^{(j)}; \forall k = \{0; 1; \ldots; N - 1\}
\end{aligned}
\tag{1}
$$

where $v^{(j)}$ and $a^{(j)}$ are the maximum desired values of velocity and acceleration along the motion, respectively, and $\mathbf{S}^{(j)}(p_k^{(j)}; v_k^{(j)}; a_k^{(j)}) = (p_{k+1}^{(j)}; v_{k+1}^{(j)}; a_{k+1}^{(j)})^\top$ are the vehicle motion primitives along each j-axis of the world frame F_W encoding the splines presented in [20, eq.(2)].

IV. Problem Solution

In this section, we apply the STL framework presented in Section III to formulate the optimization problem in Section II, resulting in a nonlinear non-convex max-min problem formulated as a Nonlinear Programming (NLP) problem solved through dynamic programming (Section IV-A). To find a solution for this type of nonlinear problem in a reasonable amount of time, we generate an initial guess from a simplified ILP formulation (Section IV-B), that draws inspiration from prior research [21]. It is worth noting that the proposed approach distinguishes itself from previous efforts by incorporating ergonomics features while maintaining consistency with the previously identified objectives.

A. Motion planner

In this section, the requirements for the problem outlined in Section II are translated into the STL formula, φ. The mission to perform object handovers with an MRAV have two types of specifications. Firstly, safety must be maintained throughout the entire mission duration, t_N; the MRAV must remain within the designated area (φ_{ws}), avoid collisions with surrounding objects (φ_{obs}), and never approach to the operator from behind (φ_{beh}). Secondly, certain ergonomic-related objectives must be met at specific times in the mission duration; each human operator must be visited once by the MRAV and the vehicle must stay there for t_{han} (φ_{han}), with $t_{\text{han}} \leq t_N$, to perform the object handover. The MRAV must approach the operator from the front, either from the

left or right, from above or below, based on the operator's preferences ($'_{\text{pr}}$). Additionally, due to the limited carrying capacity of the MRAV, the vehicle must stop at a refilling station and remain there for a duration of t_{rs}, with $t_{\text{rs}} \quad t_N$, once its onboard supply of tools is depleted in order to replenish its supply ($'_{\text{cap}}$). Lastly, after completing handover operations, the MRAV must fly to its nearest refilling station ($'_{\text{rs}}$). All mission requirements can be expressed in the STL formula:

$$' = \quad _{t_N} '_{\text{ws}} \wedge '_{\text{obs}} \wedge '_{\text{beh}} \quad \bigvee \quad _{t_N} ('_{\text{han}} \wedge '_{\text{pr}} \wedge '_{\text{cap}}) \, U_{t_N} '_{\text{rs}}; \quad (2)$$

with

$$'_{\text{ws}} = \mathbf{p}^{(j)} \; 2 \; (\underline{p}^{(j)}_{\text{ws}}; p^{(j)}_{\text{ws}}); \quad (3a)$$

$$'_{\text{obs}} = \quad \mathbf{p}^{(j)} \; \overline{2} \; (^{n}\underline{p}^{(j)}_{\text{obs}}; {}^{n}p^{(j)}_{\text{obs}}); \quad (3b)$$

$$'_{\text{beh}} = \quad \mathbf{p}^{(j)} \; \overline{2} \; (^{n}\underline{p}^{(j)}_{\text{beh}}; {}^{n}p^{(j)}_{\text{beh}}); \quad (3c)$$

$$'_{\text{han}} = \quad _{t_{\text{han}}} \mathbf{p}^{(j)} \; 2 \; (^{n}\underline{p}^{(j)}_{\text{ho}}; {}^{n}p^{(j)}_{\text{ho}}); \quad (3d)$$

$$'_{\text{pr}} = \quad _{t_N} \mathbf{p}^{(j)} \; 2 \; (^{n;m}\underline{p}^{(j)}_{\text{pr}}; {}^{n;m}p^{(j)}_{\text{pr}}) \quad ; \quad (3e)$$

$$'_{\text{cap}} = \quad _{t_{\text{rs}}} (c == 0) \, \mathbf{p}^{(j)} \; 2 \; (\underline{p}^{(j)}_{\text{rs}}; p^{(j)}_{\text{rs}}); \quad (3f)$$

$$'_{\text{rs}} = \mathbf{p}^{(j)} \; 2 \; (\underline{p}^{(j)}_{\text{rs}}; p^{(j)}_{\text{rs}}); \quad (3g)$$

where equation (3a) ensures that the position of the MRAV, denoted by $\mathbf{p}^{(j)}$, remains within the designated workspace. The minimum and maximum values of the workspace along the j-axis of the world frame F_W are represented by $\underline{p}^{(j)}_{\text{ws}}$ and $p^{(j)}_{\text{ws}}$, respectively. Equations (3b), (3c), (3d), (3f), and (3g) establish guidelines for obstacle avoidance, operator safety, handover operations, payload capacity, and mission completion, respectively. The payload capacity of the MRAV is represented by $c \; 2 \; \mathbb{N}_{>0}$ as a positive integer. The vertices of the rectangular regions identifying obstacles, areas behind the operators, operators themselves, and refilling stations along the j-axis of the world frame F_W are represented by $\underline{p}^{(j)}_{\text{obs}}$, $\underline{p}^{(j)}_{\text{beh}}$, $\underline{p}^{(j)}_{\text{ho}}$, $\underline{p}^{(j)}_{\text{rs}}$, $p^{(j)}_{\text{obs}}$, $p^{(j)}_{\text{beh}}$, $p^{(j)}_{\text{ho}}$ and $p^{(j)}_{\text{rs}}$, respectively. Finally, equation (3e) accounts for the human operators' preferences for the drone's approach. These preferences include the operator's preferred approach direction, such as front, left, right, above, or below [8]. The preferences define specific areas represented by rectangular regions $(\underline{p}^{(j)}_{\text{pr}}; p^{(j)}_{\text{pr}})$ along the j-axis within the world frame F_W where the drone is permitted to approach the operator. These areas are established by taking the operator's orientation into account and reflecting the potential paths that the drone's trajectory can take when approaching the operator from different directions (i.e., front, left, right, above, or below). The use of the *eventually* operator $_{t_N}$ guarantees that the predicate (3e), i.e., the drone position $\mathbf{p}^{(j)}$ belonging to those rectangular regions, will hold somewhere within the time interval t_N.

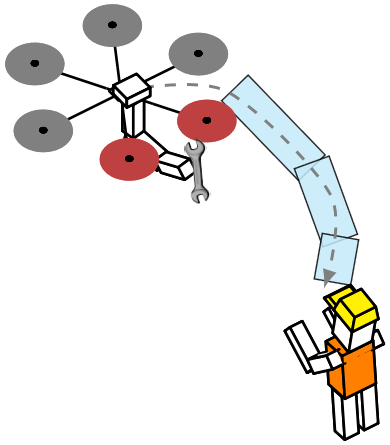

Fig. 3: An illustrative representation of an MRAV approaching a human operator. In blue are depicted the areas $(\underline{p}^{(j)}_{\text{pr}}; p^{(j)}_{\text{pr}})$ where the drone can approach the operator, while in gray is a sample output from the STL optimizer.

A schematic representation of the scenario is reported in Figure 3.

By using the specifications described in (2), we formulated the optimization problem described in Def. 4 to determine feasible trajectories that maximize the smooth robustness $\sim_{'}(\mathbf{x}; t_k)$ w.r.t. the given mission specifications $'$. In order to accomplish this, we must compute the robustness score for each individual predicate. The predicates within the STL formula (2) assess whether the MRAV's position falls within a specific region, as described in (3a)–(3g). When the MRAV is located within a region, it is assigned a positive robustness value. The higher the minimum Euclidean distance between the MRAV's trajectory and the region's boundaries, the greater the robustness. However, in (3b) and (3c), this relationship is inverted, with the MRAV's presence in an obstacle region or a space behind the human operator resulting in a negative robustness value. To provide a specific example, let us consider the $'_{\text{ws}}$ predicate (3a). This predicate can be mathematically expressed as follows:

$$'_{\text{ws}} = \min_k \quad \min(\; '_{(1)}; \; \underline{}'_{(1)}; \; '_{(2)}; \; \underline{}'_{(2)}; \; '_{(3)}; \; \underline{}'_{(3)}) \quad ; \quad (4)$$

with

$$'_{(j)} = p^{(j)}_{\text{ws}} \quad p^{(j)}_k; \qquad \underline{}'_{(j)} = p^{(j)}_k \quad \underline{p}^{(j)}_{\text{ws}}.$$

Similarly, the robustness metric for the non-belonging predicate $'_{\text{obs}}$ (3b) can be calculated by reversing the minimum distance values for each sample along the trajectory. Mathematically, this can be represented as:

$$'_{\text{obs}} = \min_k \quad \min(\; '_{(1)}; \; \underline{}'_{(1)}; \; '_{(2)}; \; \underline{}'_{(2)}; \; '_{(3)}; \; \underline{}'_{(3)}) \quad ; \quad (5)$$

where $'_{(j)}$ and $\underline{}'_{(j)}$, with $j = \{1; 2; 3\}$, can be computed by replacing $^{(j)}_{\text{ws}}$ and $\underline{}^{(j)}_{\text{ws}}$ in (4) with $^{(j)}_{\text{obs}}$ and $\underline{}^{(j)}_{\text{obs}}$, respectively.

It is worth mentioning that the NLP problem is tackled by means of dynamic programming, which has a tendency

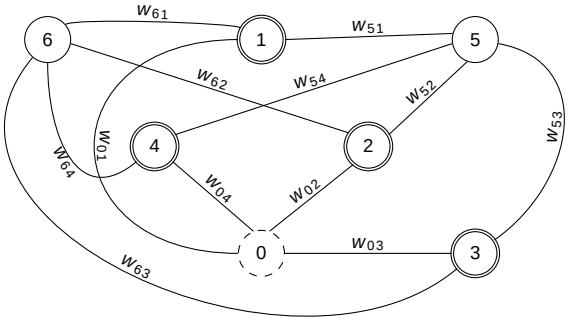

Fig. 4: Instance of the graph G assuming four human operators (double round nodes), two refilling stations (round solid nodes), and one vehicle. Arcs and weights w_{ij} denote the MRAVs's path. The depot is depicted by a round dahsed node.

to get trapped in local optima if the initial guess is not well-chosen [29], [30]. Therefore, selecting an appropriate initial guess is of paramount importance.

B. Initial guess

An appropriate initial guess is essential to arrive at optimal solutions for the STL motion planner within a reasonable time frame, and also to prevent the solver from getting stuck during the search for a feasible solution. The strategy for obtaining this initial guess involves simplifying the original problem by abstracting it in order to yield an optimization problem with fewer constraints. Specifically, the initial guess takes into account the mission requirements that pertain to visiting all human operators, while noting the MRAV payload capacity and refilling operations ($'_{\text{han}}$, $'_{\text{cap}}$, and $'_{\text{rs}}$). However, it disregards the obstacle avoidance and ergonomy requirements ($'_{\text{obs}}$, $'_{\text{ws}}$, $'_{\text{beh}}$, and $'_{\text{pr}}$), as well as mission time intervals (t_N, t_{han} and t_{rs}) as these constraints introduce the most complex nonlinearities and motion discontinuities in the problem.

A graph-based representation, including connections between the human operators and the refilling stations, is employed to formulate an ILP problem that models a type of capacitated Vehicle Routing Problem (VRP) [23]. The resulting solution assigns the human operators (those they deliver a tool to) to the vehicle and provides a navigation sequence for the MRAV.

The graph used to formulate the ILP problem is illustrated in Figure 4 and is characterized by the tuple $G = (V; E; W; C)$. The set of vertices denoted by V is comprised of the human operators (T), refilling stations (R), and the depot (O) where the MRAV is located at the initial time t_0. The number of elements in T, R, and O are represented by , r, and , respectively. The set of edges and their associated weights are represented by E and W, respectively. Furthermore, C represents the vehicle maximum payload capacity. In terms of connectivity, all vertices in T are fully connected to every vertex in $R \cup O$. Formally, let $e_{ij} \in E$ represent the edge connecting the vertices i and j, with $\{i; j\} \in V; i \neq j$ and $w_{ij} \in W$ the weight associated with e_{ij}. Given the dynamic constraints for the MRAV $v^{(j)}$ and $a^{(j)}$, we model the edge weights using Euclidean distances, implying that $w_{ij} = w_{ji}$. Therefore, minimizing the mission time can be equivalent to reducing the distance traveled by the vehicle.

To represent the number of times an edge is selected in the ILP solution, an integer variable $z_{ij} \in \mathbb{Z}_{\geq 0}$ is defined for each edge $e_{ij} \in E$, and holds that $z_{ij} = z_{ji}$. This variable is used to specify the number of times the corresponding edge is selected in the ILP solution. Thus, z_{ij} is limited to the set $\{0; 1\}$ if $\{i; j\} \in \{T; O\}$ and z_{ij} is limited to the set $\{0; 1; 2\}$ if $i \in R$ and $j \in T$. This limitation ensures that an edge between two human operators is never traversed twice and that the depot is only used as a starting point. This makes the proposed solution further suitable for wider scenarios where the MRAV can carry more than one tool at a time. Additionally, this allows for round trips between refilling stations and human operators in case there are no other tools to be delivered, i.e., $z_{ij} = 2$. The variable $z_{ij} = 0$ corresponds to non-traversed edges. By utilizing all of these defined variables, the ILP problem can be formulated as follows:

$$\underset{z_{ij}}{\text{minimize}} \quad \sum_{\{i;j\} \in V;\, i \neq j} w_{ij}\, z_{ij} \tag{6a}$$

$$\text{s.t.} \quad \sum_{i \in V;\, i \neq j} z_{ij} = 2; \ \forall j \in T; \tag{6b}$$

$$\sum_{i \in T} z_{0i} = 1; \tag{6c}$$

$$\sum_{i \in T;\, j \notin T} z_{ij} \geq 2h(T); \tag{6d}$$

where (6a) represents the objective function that encompasses the overall distance traversed by the MRAV. The constraints (6b) mandate that each human operator is visited only once. Equation (6c) ensures that each MRAV begins its mission at its depot and does not return. Constraints (6d) prevent the formation of tours that exceed the payload capacity of the MRAV or those which are not connected to a refilling station. This is achieved by using the function $h(T)$ [23] and dynamically adding these constraints as they are violated.

Once the optimal assignment for the ILP problem is obtained, motion primitives for the vehicle [20, eq.(2)] are used to obtain a dynamically feasible trajectory that correspond to its optimal assignment. The methodology to compute these trajectories is described in [20] and is not covered in this paper for conciseness. In summary, the motion primitives are calculated by fixing rest-to-rest motion between operators, resulting in zero velocity and acceleration in these places, and imposing the minimum feasible time that allows for the desired maximum values of velocity $v^{(j)}$ and acceleration $a^{(j)}$ along the MRAV's motion. The time intervals for handover t_{han} and refilling t_{rs} with the MRAV stopped in the corresponding regions are also taken into account for the trajectory.

Parameter	Symbol	Value
Payload capacity	c	$1\,[-]$
Max. velocity	$\bar{v}^{(j)}$	$1.1\,[\mathrm{m/s}]$
Max. acceleration	$\bar{a}^{(j)}$	$1.1\,[\mathrm{m/s^2}]$
Mission time	t_N	$23\,[\mathrm{s}]$
Handover time	t_{han}	$3\,[\mathrm{s}]$
Refilling time	t_{rs}	$3\,[\mathrm{s}]$
Sampling period	T_s	$0.05\,[\mathrm{s}]$
Number of samples	N	$460\,[-]$
Heading operator HO1	ψ_{ho1}	$\pi[\mathrm{rad}]$
Heading operator HO2	ψ_{ho2}	$0[\mathrm{rad}]$

TABLE I: Parameter values for the optimization problem.

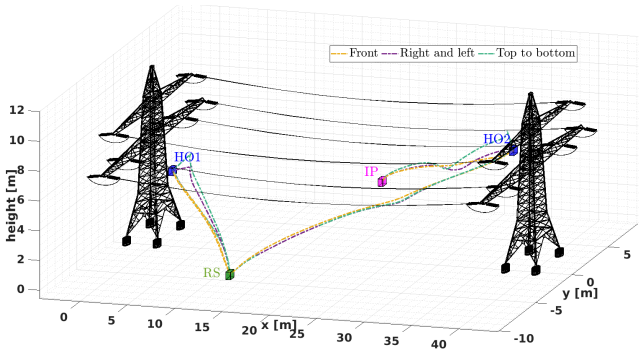

Fig. 5: Power line scenario. The trajectories considering different operators' preferred approach directions, such as front, left and right, and top to bottom, and are depicted in yellow, purple, and green, respectively.

V. Simulation Results

To validate the proposed planning approach, numerical simulations were carried out in MATLAB. At this stage, the vehicle dynamics and trajectory tracking controller were not included in the simulations. The trajectories were then verified for feasibility in realistic simulations performed in Gazebo while exploiting the benefits of software-in-the-loop simulations [4].

The optimization method was implemented in MATLAB R2019b, with the ILP problem formulated using the CVX framework and the STL motion planner using the CasADi library and IPOPT as the solver. All simulations were run on a computer with an i7-8565U processor (1:80 GHz) and 32GB of RAM running on Ubuntu 20.04. Figure 7 depicts a snapshot of the object handover task, while illustrative videos with the simulations are available at `http://mrs.felk.cvut.cz/stl-ergonomy`.

The proposed planning strategy was evaluated using the object handover scenario outlined in Section II. In particular, as shown in Figure 2, the simulation scenario consisted of a mock-up environment (50 m 20 m 15 m) with two human operators, one refilling station, and a single MRAV. The parameters and their corresponding values used to run the optimization problem are listed in Table I. It is worth noting that the handover time t_{han} and the refilling time t_{rs} were set to short symbolic times to avoid analyzing trajectories with excessive waiting times. The heading angle of the MRAV was adjusted by aligning the vehicle with the direction of movement when moving towards the human operator. Once the MRAV reaches the operator, it is assumed that an onboard low-level controller, e.g. [6], [7], handles the handover operation, thus adjusting the heading angle accordingly. The rectangular regions in which the drone was allowed to approach the operator were established taking into consideration the operators' heading, ho1 and ho2, as well as their preferred direction of approach (′ pr).

Figure 5 illustrates the planned trajectories along with the power towers and cables, human operators, and refilling station. The figure compares the trajectories computed by taking into account the preferred approach directions of the operators, including front, right and left, and top to bottom. The towers are 15 m tall and are positioned 40 m apart. The optimization problem was solved in 130 s and it took 1 s to find an initial guess solution. Note that, even though the initial guess solution retrieved by solving the ILP problem is the same, the STL optimizer rearranges the trajectory to maximize the robustness score. This basic scenario illustrates the decoupling of the final STL optimization from the ILP initial solution, which provides added versatility to the proposed human-robot collaboration planner. Additionally, it highlights the ease with which high-level ergonomic requirements can be integrated into the problem formulation. However, as the complexity increases, beginning with an initial solution far from the overall problem solution can lead to the STL optimizer becoming trapped in a local optimum.

Figure 6 demonstrates that the planned trajectories comply with the mission requirements. As can be observed from the graph, the vehicle velocity and acceleration stay within the permissible bounds ($[\underline{v}^{(j)}; v^{(j)}]$ and $[\underline{a}^{(j)}; a^{(j)}]$). For simplicity, the velocity and acceleration bounds are considered to be symmetric, i.e., $j\underline{v}^{(j)}j = jv^{(j)}j$ and $j\underline{a}^{(j)}j = ja^{(j)}j$. In addition, the time frames for both the handover and refilling operations are presented, highlighting how the user's preferences directly reflect on the MRAV motion.

VI. Conclusions

In this paper, a motion planning framework was introduced to enhance ergonomic human-robot collaboration for an MRAV with payload capacity limitations and dynamic constraints. The proposed method employs STL specifications to generate trajectories that comply with mission requirements, including safety, ergonomics, and mission time. This work extends a previous motion planner [20], [21] by introducing a method for tackling the nonlinear non-convex nature of the optimization problem through the use of an ILP approach. This approach utilizes a simplified version of mission specifications to provide a feasible initial solution for the STL framework to facilitate convergence of the algorithm. Numerical analyses in MATLAB and realistic simulations in Gazebo demonstrated the effectiveness of the proposed approach. Future work includes incorporating human operator fatigue into the problem formulation by using weights associated with Boolean and temporal operators to modulate the robustness. Additionally, research will be conducted on the use of conflicting temporal logic