Fast, Composable Rescue Mission Planning for UAVs using Metric Temporal Logic

Usman A. Fiaz   John S. Baras

Department of Electrical & Computer Engineering, Institute for Systems Research, University of Maryland, College Park, MD, USA (e-mail: fiaz@umd.edu, baras@umd.edu)

Abstract: We present a hybrid compositional approach for real-time mission planning for multi-rotor unmanned aerial vehicles (UAVs) in a time critical search and rescue scenario. Starting with a known environment, we specify the mission using Metric Temporal Logic (MTL) and use a hybrid dynamical model to capture the various modes of UAV operation. We then divide the mission into several sub-tasks by exploiting the invariant nature of safety and timing constraints along the way, and the different modes (i.e., dynamics) of the UAV. For each sub-task, we translate the MTL specifications into linear constraints and solve the associated optimal control problem for desired path, using a Mixed Integer Linear Program (MILP) solver. The complete path for the mission is constructed recursively by composing the individual optimal sub-paths. We show by simulations that the resulting suboptimal trajectories satisfy the mission specifications, and the proposed approach leads to significant reduction in computational complexity of the problem, making it possible to implement in real-time.

Keywords: Flying robots, mission planning and decision making, safety, trajectory and path planning, optimal control of hybrid systems

1. INTRODUCTION

Multi-rotor Unmanned Aerial Vehicles (UAVs) in general and quadrotors in particular, find enormous applications in several key areas of research in academia as well as industry. These include but are not limited to search and rescue and disaster relief, autonomous aerial grasping and transport, and aerial coverage and surveillance. The motivation for this expanding zeal towards multi-rotor UAVs is twofold. First, these are highly inexpensive robots which can be extensively used as testbeds for much of the ongoing research in aerial robotics. Secondly, these are extremely agile aircraft, capable of much higher maneuverability in comparison with the other UAV classes, namely fixed-wing and helicopter style UAVs. This salient feature also edges them as a feasible platform to operate in congested environments, such as crowded city skies and constrained indoor workspaces.

The enormous impact of quadrotors in autonomous search and rescue becomes evident especially during natural disasters, where it is impossible for humans and terrestrial robots to access highly cluttered spaces to rescue people from life threatening situations. However, the need to evacuate the targets within limited, finite time cannot be more emphasized.

In this paper, we propose a hybrid, compositional, optimization based method for real-time mission planning for quadrotors in a time critical search and rescue scenario. Starting with a known environment, we specify the mission using Metric Temporal Logic (MTL) and use a hybrid

dynamical model to capture the various modes of UAV operation. We then divide the mission into several sub-tasks, by exploiting the invariant nature of safety and timing constraints along the way, and the different modes (i.e., dynamics) of the UAV. For each sub-task, we translate the MTL specifications into linear constraints, and solve the associated optimal control problem for desired path, using a Mixed Integer Linear Program (MILP) solver. The complete path for the mission is constructed recursively by composing the individual optimal sub-paths. We show by simulations that the resulting suboptimal trajectories satisfy the mission specifications, and this approach leads to significant reduction in computational complexity of the problem, making it possible to implement in real-time.

Our proposed method ensures the safety of UAVs at all times and guarantees finite time mission completion. It is also shown that our approach scales up nicely for a large number of UAVs under some realistic assumptions on the environment of operation. Following are the main contributions of this paper:

- A hybrid optimization based framework for rescue mission planning for UAVs using MTL specifications under the assumption of known environment, and using a rich hybrid dynamical model for the UAVs.
- Decomposition of complex MTL specifications and their translation into linear constraints.
- Fast (i.e., real-time) and recursive computation of safe, composable, suboptimal trajectories for UAVs with finite time guarantees.
- Limitations and scalability results for the proposed approach for large number of UAVs in a constrained environment.

2. RELATED WORK

Given any high level task, it is a standard practice in classic motion planning literature (LaValle (2006)), to look for a set of trajectories, which the robot can follow while satisfying the desired task specifications. This gives rise to the notion of optimal path planning, which considers an optimal path in the sense of optimizing some suitable cost function and finding a control law, to go from one position to another while satisfying some constraints (Choset (2005)). Traditionally, methods such as potential functions (Xi et al. (2005)), have been used for multi-robot mission planning. However, they tend to fail in situations where the mission involves some finite time constraints or dynamic specifications. Aerial surveying of areas and search and rescue are examples of such tasks.

Temporal logic (Baier et al. (2008)) seems to address this problem, since it enables us to specify complex dynamic tasks in compact mathematical form. A bulk of modern motion planning literature is based on Linear Temporal Logic (LTL) (Goerzen et al. (2010)), which is useful for specifying tasks such as visiting certain objectives periodically, surveying areas, ensuring stability and safety etc. (Kantaros and Zavlanos (2018)). However, from a control theory perspective, LTL only accounts for timing in the infinte horizon sense i.e., it can only guarantee something will *eventually* happen and is not rich enough to describe finite time constraints.

On the other hand, Metric Temporal Logic (MTL) (Koymans (1990); Ouaknine and Worrell (2008)), can express finite time requirements between various events of the mission as well as on each event duration. This allows us to specify safety critical missions with dynamic task specifications and finite time constraints. An optimization based method for LTL tasks was proposed in (Karaman et al. (2008)), and (Wolff et al. (2014)), where they translate the LTL specifications to Mixed Integer Linear Programming (MILP) constraints, which are then used to solve an optimal control problem for a linear point-robot model. This work was extended in (Maity and Baras (2015)), where the authors used bounded time temporal constraints using extended LTL for motion planning with linear system models. However, all these works did not incorporate a rich dynamical model of the robot, and also illustrated significant computational complexity issues for the proposed methods in case of planning for multiple robots in 3D space.

In some recent papers, optimization based methods with MTL specifications for single (Zhou et al. (2015)) and multiple (Nikou et al. (2016)) robots, do guarantee safe and finite time mission completion. However, in both cases, the computation of the optimal trajectory is expensive (in the order of ∼500 sec computation time), and hence cannot be implemented in real-time. Moreover, these works put high constraints on the robot maneuverability, by limiting its dynamics to a simple linear (point-robot) model, which is in contradiction with the main reason for deploying quadrotors in constrained dynamic environments. Thus, in this paper, our intention is to use the rich dynamics of the robots in an intelligent way to divide and conquer the computationally complex problem of mission planning for multiple UAVs using finite time MTL constraints.

3. NOTATION AND PRELIMINARIES

3.1 System Dynamics

Any discrete time (possibly nonlinear) dynamical system can be represented in the form:
$$x(t+1) = f(t, x(t), u(t)) \tag{1}$$
where $x(t) \in \mathcal{X}$ is the state vector, $x(0) \in \mathcal{X}_0 \subseteq \mathcal{X}$ is the initial condition of the state, and $u(t) \in \mathcal{U} \subset \mathbb{R}^m$ is the set of control inputs for all $t = 0, 1, 2, \cdots$. Let us denote the trajectory for System (1), with initial condition x_0 at t_0, and input $u(t)$ as: $\mathbf{x}_{t_0, x_0, u(t)} = \{x(t) \mid t \geq t_0, x(t+1) = f(t, x(t), u(t)), \ x(t_0) = x_0\}$. For the sake of convenience, we use the shorthand notation i.e., $\mathbf{x}_{t_0}$ instead of $\mathbf{x}_{t_0, x_0, u(t)}$ to represent system trajectory whenever no explicit information about $u(t)$ and x_0 is required. As is in the usual control literature setting, a linearization of the discrete time dynamics for System (1) about a given trajectory (or operating point) looks like:
$$\hat{x}(t+1) = A(t)\hat{x}(t) + B(t)\hat{u}(t) \tag{2}$$
for all $t = 0, 1, 2, \cdots$. We use System (2) form dynamics in our problem formulation in Section 5.

3.2 Metric Temporal Logic (MTL)

The convention on MTL syntax and semantics followed in this paper is the same as presented in (Koymans (1990)).

*Definition 1. An atomic proposition is a statement with regard to the state variables x that is either **True** ($\top$) or **False** ($\perp$) for some given values of x.*

Let $\Pi = \{\pi_1, \pi_2, \cdots \pi_n\}$ be the set of atomic propositions which labels $\mathcal{X}$ as a collection of the areas of interest in some workspace, which can possibly be time varying. Then, we can define a map L which labels this workspace or environment as follows:
$$L : \mathcal{X} \times \mathcal{I} \to 2^{\Pi}$$
where $\mathcal{I} = \{[t_1, t_2] \mid t_2 > t_1 \geq 0\}$ and 2^{Π} denotes the power set of Π as usual. In general, $\mathcal{I}$ represents an interval of time but it may just also represent a time instance. For each trajectory of System (2) i.e., $\mathbf{x}_{t_0}$ as before, the corresponding sequence of atomic propositions, which $\mathbf{x}_{t_0}$ satisfies is given as: $\mathcal{L}(\mathbf{x}_0) = L(x(0), 0)L(x(1), 1)....$

We later specify the tasks formally using MTL formulae, which can incorporate finite timing constraints. These formulae are built on the stated atomic propositions (Definition 1) by following some grammar.

Definition 2. The syntax of MTL formulas are defined in accordance with the following rules of grammar:
$$\phi ::= \top \mid \pi \mid \neg\phi \mid \phi \vee \phi \mid \phi \mathbf{U}_I \phi$$

where $I \subseteq [0, \infty]$, $\pi \in \Pi$, $\top$ and $\neg\top(= \perp)$ are the Boolean constants for *true* and *false* respectively. $\vee$ represents the disjunction while $\neg$ represents the negation operator. $\mathbf{U}_I$ denotes the Until operator over the time interval I. Similarly, other operators (both Boolean and temporal) can be expressed using the grammar imposed in Definition 2. Some examples are conjunction ($\wedge$), always on I ($\Box_I$), eventually within I ($\Diamond_I$) etc. Further examples of temporal operators can be found in (Karaman et al. (2008)).

Definition 3. The semantics of any MTL formula ϕ is recursively defined over a trajectory x_t as:

$x_t \models \pi$ *iff* $\pi \in L(x(t), t)$
$x_t \models \neg\pi$ *iff* $\pi \notin L(x(t), t)$
$x_t \models \phi_1 \vee \phi_2$ *iff* $x_t \models \phi_1$ *or* $x_t \models \phi_2$
$x_t \models \phi_1 \wedge \phi_2$ *iff* $x_t \models \phi_1$ *and* $x_t \models \phi_2$
$x_t \models \bigcirc\phi$ *iff* $x_{t+1} \models \phi$
$x_t \models \phi_1 \mathbf{U}_I \phi_2$ *iff* $\exists t' \in I$ *s.t.* $x_{t+t'} \models \phi_2$ *and* $\forall\, t'' \leq t'$,
$x_{t+t''} \models \phi_1$.

Thus, for instance, the expression $\phi_1 \mathbf{U}_I \phi_2$ means the following: ϕ_2 is true within time interval I, and until ϕ_2 is true, ϕ_1 must be true. Similarly, the MTL operator $\bigcirc\phi$ means that ϕ is true at next time instance. $\square_I\phi$ means that ϕ is always true for the time duration or during the interval I, $\diamondsuit_I\phi$ implies that ϕ eventually becomes true within the interval I. More details on specifying tasks as MTL formulae can also be found in (Ouaknine and Worrell (2008)). Similar to the convention used in Definition 3, a system trajectory $\mathbf{x}_{t_0}$ satisfying an MTL specification ϕ is denoted as $\mathbf{x}_{t_0} \models \phi$.

3.3 The Workspace

Throughout this paper, we repeatedly refer to a time critical rescue mission defined on the constrained workspace shown in Fig. 1. It is a custom built CAD environment, designed with the intention to closely suit our problem. As shown, various areas of interest are marked on the workspace using different alphabets. The role of each of these areas will become obvious as we formulate the problem and specify the mission using MTL specifications in Section 5. For now, we briefly describe the mission in a two UAV setting. Starting from initial positions A and B, two quadrotor UAVs q_1 and q_2 need to rescue two objects located at F and G respectively from a constrained environment. The objects are accessible only through a window E, with dimensions such that it allows only one UAV to pass at a given time. Therefore, one of the UAVs has to wait at C for the other to pass first. It is assumed that there are no additional obstacles in the area other than the walls O and the UAVs themselves. The task for each UAV is to grasp its respective target object, and transport it to safety (marked H_1 and H_2 respectively for q_1 and q_2) in given finite time. While doing so, the UAVs need to avoid the obstacles O as well as each other, in particular at the window E. Later on, we also use prime region notation; e.g., A' to represent the same 2D region A. There A' represents an altitude (w.r.t. z-axis) variation of the quadrotor while it is in the same 2D region A.

4. QUADROTOR DYNAMICS

We adopt the generalized nonlinear model for the quadrotor presented in (Michael et al. (2010)). We build a hybrid model for the system with five linear modes, namely *Take off*, *Land*, *Hover*, *Steer*, and a task specific *Grasp* mode. The linearization for each mode is carried out separately about a different operating point. This enables our system to have rich dynamics with less maneuverability restrictions, while each mode still being linear. This is an important point and its significance becomes apparent once we formulate the problem, since it requires all constraints to be linear. (see Section 5)
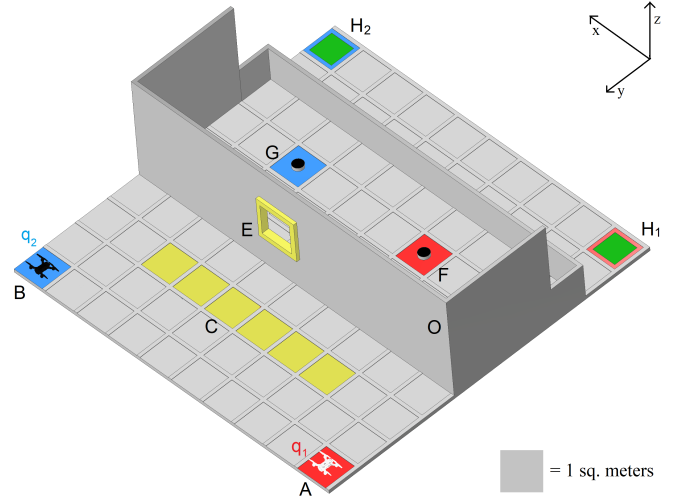

Fig. 1. CAD model for the workspace used. The environment is a 10x10x3 m^3 workspace which is divided into several 2D regions of interest that are labeled with alphabets and marked with different colors.

4.1 General Nonlinear Model

The dynamics of a quadrotor can be fully specified using two coordinate frames. One is a fixed earth (or world) frame, and the second is a moving body frame. Let the homogeneous transformation matrix from body frame to earth frame be $R(t)$, which is a function of time t. In state space representation, the quadrotor dynamics are represented as twelve states namely $[x, y, z, v_x, v_y, v_z, \phi, \theta, \psi, \omega_\phi, \omega_\theta, \omega_\psi]^T$, where $\xi = [x, y, z]^T$ and $v = [v_x, v_y, v_z]^T$ represent the position and velocity of the quadrotor respectively with respect to the body frame. $[\phi, \theta, \psi]^T$ are the angles along the three axes (i.e., roll, pitch, and yaw respectively), and $\Omega = [\omega_\phi, \omega_\theta, \omega_\psi]^T$ represents the vector containing their respective angular velocities. Under the rigid body assumptions on its airframe, the Newton-Euler formalism for quadrotor in earth frame is given by:

$$\dot{\xi} = v$$
$$\dot{v} = -g\mathbf{e_3} + \frac{F}{m}R\mathbf{e_3} \qquad (3)$$
$$\dot{R} = R\hat{\Omega}$$
$$\dot{\Omega} = J^{-1}(-\Omega \times J\Omega + \tau)$$

where J is the moment of inertia matrix for the quadrotor, g is the gravitational acceleration, $\mathbf{e_3} = [0, 0, 1]^T$, F is the total thrust produced by the four rotors, and $\tau = [\tau_x, \tau_y, \tau_z]^T$ is the torque vector, whose components are the torques applied about the three axes. So, F, τ_x, τ_y, and τ_z are the four control inputs to System (3).

4.2 Hybrid Model with Linear Modes

System (3) serves as a starting point for generating a hybrid model for the quadrotor with five modes, which are represented by a labeled transition system as shown in Fig. 2. As usual, the states (or modes) denote the action of the UAV, such as *Take off* and *Steer*, while the edges represent the change or switch to another action. The change is governed by some suitable guard condition. Note,
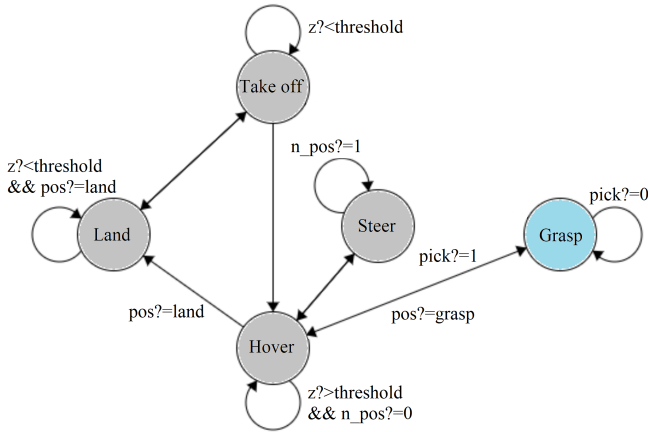
Fig. 2. The simplified hybrid dynamical model for the quadrotor. Some guard conditions are hidden for readability. We use linearized dynamics around different operating points for each mode. This makes the model rich in dynamics as well as linear at the same time.

that some edges do not exist; for example, the quadrotor cannot go from *Land* to *Hover* without taking the action *Take off*. Each state of the transition system follows certain dynamics, which result from a linearization of System (3) around a different operating point. For example, consider the *Hover* mode. One possible choice of operating points for the linearization of system dynamics in this mode is $\psi = 0$. This implies that the two states ψ and ω_ψ i.e., the yaw angle and its respective angular velocity are identically zero, and thus can be removed from the state space representation. Consequently, the state space dimension is reduced to ten, and the control set is reduced to three inputs as well; i.e., F, τ_x, and τ_y. The resulting linearized model can be written in standard (discrete time) form as: $\sigma(t+1) = A(t)\sigma(t) + B(t)\gamma(t)$, where $\sigma(t)$ is the state, and $\gamma(t)$ is the input (in vector notation), with the two system matrices given as:

$$A = \begin{bmatrix} \mathbf{0} & I & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \begin{bmatrix} 0 & g \\ -g & 0 \\ 0 & 0 \end{bmatrix} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & I \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} ; \quad B = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1/m \end{bmatrix} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ \mathbf{0} & I_{2\times3}J^{-1} \end{bmatrix}$$

where $I_{2,3} = [I_{2,2} \; \mathbf{0}_{2,1}]$, and all zero and identity matrices in $A(t)$ and $B(t)$ are of proper dimensions. We adopt similar procedure to linearize System (3) around other operating points for different modes, and obtain linearized dynamics for the hybrid model. Here, we omit the discussion about the selection of these operating points, but it can be found in (García et al. (2013)).

4.3 The Grasp Mode

Grasping in general is a very challenging problem, in particular when dexterity based manipulation is involved. Since *Grasp* is the only task specific dynamical mode of our system, it was advisable to simplify the grasping routine within the high level task. However, in case of aerial grasping, some passive mechanisms (Fiaz et al. (2019)) have been shown to be very reliable in grasping an object with

an instantaneous touchdown onto its surface (Fiaz et al. (2017, 2018)). Under this reliable passive aerial grasping assumption (i.e., instantaneous touchdown and grasp), we can express the *Grasp* mode as a switching combination of *Hover*, *Land*, and *Take off* dynamics with special guard conditions. This clearly simplifies the problem of having the need to introduce a complex gripper and its end-effector dynamics into the *Grasp* mode of the hybrid model. Figure 3 depicts this representation of the *Grasp* mode in terms of the *Hover*, *Land*, and *Take off* dynamics.
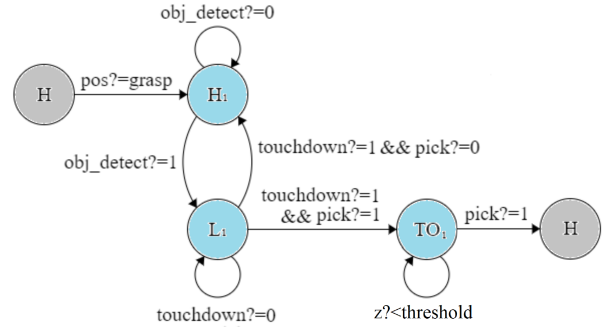

Fig. 3. The *Grasp* mode expressed as a combination of *Hover* (H_1), *Land* (L_1), and *Take off* (TO_1) modes (colored cyan), with special guard conditions.

5. METHOD: FORMULATION AND SOLUTION

Given the map of the environment, we can write down a mission specification for each quadrotor as an MTL formula ϕ_i. For the workspace described in Section 3, a possible MTL task specification for the ith quadrotor q_i can be written as:

$$\phi_i = \diamond_{[0,T_1]}(Object\ Location) \wedge \square_{[0,T_2]}(Object\ Location)'$$
$$\wedge \diamond_{[0,T_3]}\square(Safe\ Location) \wedge \square\neg(Obstacles) \wedge \square\neg q_j$$

where $i, j \in \{1, 2, ..., N\}$, $i \neq j$, and $q_j \in \mathcal{N}_i(t)$. $\mathcal{N}_i(t)$ represents the neighborhood set of quadrotor q_i i.e., all quadrotors q_j s.t. $\|q_i - q_j\| \leq \rho$, for some $\rho > 0$. T_3, T_2, T_1 are discrete time units, and the prime notation models the grasping action within the same area of the workspace with some altitude variation of the UAV along the z-axis. Thus, for instance, in the two UAV setting as shown in Fig. 1, a possible MTL specification for quadrotor q_1 is:

$$\phi_1 = \diamond_{[0,T_1]}(F) \wedge \square_{[0,T_2]}(F)' \wedge \diamond_{[0,T_3]}\square(H_1) \wedge \square\neg(O) \wedge \square\neg q_2$$

A similar mission specification ϕ_2 can be written for the quadrotor q_2 and so on. Now using the quadrotor dynamics and these MTL formulae ϕ_i, we can state the rescue mission planning as an optimal control problem.

5.1 Problem Statement and Formulation

We set up the described mission as a standard optimal control problem in discrete time. Given the system dynamics (2), the objective is to find a suitable control law that steers each quadrotor through some regions of interest in the workspace within desired time bounds, so that it evacuates the target safely to a desired location. This control also optimizes some cost function, while the associated task constraints are specified by an MTL expression. As

before, let ϕ_i denote the MTL formula for the mission specification, and $J(x_i(t, u_i(t)), u_i(t))$ be the cost function to be minimized. Then, the corresponding optimization problem for quadrotor q_i, $i \in \{1, 2, ..., N\}$ is given by:

Problem 1.

$$\min_{x_i, u_i} \quad J(x_i(t, u_i(t)), u_i(t))$$
$$\text{s.t.} \quad x_i(t+1) = A(t)x_i(t) + B(t)u_i(t)$$
$$\mathbf{x_{i}}_{t_0} \models \phi_i$$

Problem 1 is a discrete time optimal control problem with linear dynamics. However, it includes a complex MTL satisfiability constraint. In addition, notice that in our case, the hybrid model of the quadrotor has multiple linear modes. Therefore, in its current form, Problem 1 is not directly solvable.

We now describe two methods that transform Problem 1 into a set of readily solvable Mixed Integer Linear Programs (MILPs). First, we describe a method for translating the MTL satisfiability constaint into a set of linear constraints. This approach renders Problem 1 solvable as a MILP for a linear cost and a given dynamical mode of the system. Secondly, we decompose the complex MTL specifications ϕ_i into a set of simpler MTL formulae ϕ_i^k, $i \in \{1, 2, ..., N\}$ and $k \in \{1, 2, ..., M\}$. This suffices to breaking down the original problem into M sub-problems, each with a set of linear constraints and exactly one associated dynamical mode of the hybrid system. Combining both these methods, the resulting M MILPs are expected to have significantly reduced computational complexity than the parent problem.

5.2 MTL Formulae to Linear Constraints

This method is based on the approach presented in (Karaman et al. (2008)) where the authors translate LTL specifications into linear constraints. Along similar lines, we now present an approach to translate MTL specifications into mixed integer linear constraints. We start with a simple temporal specification and work through the procedure to convert it to mixed integer linear constraints. We then use this example as a foundation for translating the MTL operators into equivalent linear constraints.

Consider the constraint that a trajectory $x(t)$ lies within a convex polytope $\mathcal{K}$ at time t. Since $\mathcal{K}$ is convex, it can be represented as an intersection of a finite number of halfspaces. A halfspace can be represented as set of points, $\mathcal{H}_i = \{x : h_i^T x \leq a_i\}$. Thus, $x(t) \in \mathcal{K}$ is equivalent to $x(t) \in \cap_{i=1}^n \mathcal{H}_i = \cap_{i=1}^n \{x : h_i^T x \leq a_i\}$. So, the constraint $x(t) \in \mathcal{K} \ \forall \ t \in \{t_1, t_1 + 1, \cdots t_1 + n\}$ can be represented by the set of linear constraints $\{h_i^T x(t) \leq a_i\}$, $\forall \ i = \{1, 2, \cdots, n\}$ and $\forall t \in \{t_1, t_1 + 1, \cdots t_1 + n\}$.

In a polytopic environment, atomic propositions (see Definition 2), $p, q \in \Pi$, are related to state of the system via conjunction and disjunction of linear halfspaces (Karaman et al. (2008)). Let us consider the case of a convex polytope and let $b_i^t \in \{0, 1\}$ be some binary variables associated with the corresponding halfspaces $\{x(t) : h_i^T x(t) \leq a_i\}$ at time $t = 0, ..., N$. We can then force the constraint: $b_i^t = 1 \iff h_i^T x(t) \leq a_i$, by introducing the following linear constraints:

$$h_i^T x(t) \leq a_i + M(1 - b_i^t) \tag{4}$$

$$h_i^T x(t) \geq a_i - Mb_i^t + \epsilon$$

where M and ϵ are some large and small positive numbers respectively. If we denote $K_t^{\mathcal{K}} = \wedge_{i=1}^n b_i^t$, then $K_t^{\mathcal{K}} = 1 \iff x(t) \in \mathcal{K}$. This approach is extended to the general nonconvex case by convex decomposition of the polytope. Then, the decomposed convex polytopes are related using disjunction operators. Similar to conjunction, as is described later in this section, the disjunction operator can also be translated to mixed integer linear constraints.

Let $S_\phi(x, b, u, t)$ denote the set of all mixed integer linear constraints corresponding to a temporal expression ϕ. Using the described procedure, once we have obtained $S_p(x, b, u, t)$ for atomic propositions $p \in \Pi$, we can formulate $S_\phi(x, b, u, t)$ for any MTL formula ϕ. Now, for the Boolean MTL operators, such as $\neg$, $\wedge$, $\vee$, let $t \in \{0, 1, ..., N\}$, and as before $K_t^\phi \in [0, 1]$ be the continuous variables associated with the formula ϕ generated at time t with atomic propositions $p \in \Pi$. Then $\phi = \neg p$ is the negation of an atomic proposition, and it can be modeled as:

$$K_t^\phi = 1 - K_t^p \tag{5}$$

the conjunction operator, $\phi = \wedge_{i=1}^m p_i$, is modeled as:

$$K_t^\phi \leq K_t^{p_i}, \quad i = 1, ..., m \tag{6}$$

$$K_t^\phi \geq 1 - m + \sum_{i=1}^m K_t^{p_i}$$

and the disjunction operator, $\phi = \vee_{i=1}^m p_i$, is modeled as:

$$K_t^\phi \geq K_t^{p_i}, \quad i = 1, ..., m \tag{7}$$

$$K_t^\phi \leq \sum_{i=1}^m K_t^{p_i}$$

Similar to binary operators, temporal operators such as $\Diamond, \square$, and $\mathcal{U}$ can be modeled using linear constraints as well. Let $t \in \{0, 1, ..., N - t_2\}$, where $[t_1, t_2]$ is the time interval used in the MTL specification ϕ. Then, eventually operator: $\phi = \Diamond_{[t_1, t_2]} p$ is modeled as:

$$K_t^\phi \geq K_\tau^p, \quad \tau \in \{t + t_1, ..., t + t_2\} \tag{8}$$

$$K_t^\phi \leq \sum_{\tau = t + t_1}^{t + t_2} K_\tau^p$$

and always operator: $\phi = \square_{[t_1, t_2]} p$ is represented as:

$$K_t^\phi \leq K_\tau^p, \quad \tau \in \{t + t_1, ..., t + t_2\} \tag{9}$$

$$K_t^\phi \geq \sum_{\tau = t + t_1}^{t + t_2} K_\tau^p - (t_2 - t_1)$$

and until operator: $\phi = p \, \mathcal{U}_{[t_1, t_2]} \, q$ is equivalent to:

$$c_{tj} \leq K_q^j \quad j \in \{t + t_1, \cdots, t + t_2\}$$
$$c_{tj} \leq K_p^l \quad l \in \{t, \cdots, j - 1\}, j \in \{t + t_1, \cdots, t + t_2\}$$
$$c_{tj} \geq K_q^j + \sum_{l=t}^{j-1} K_p^l - (j - t) \quad j \in \{t + t_1, \cdots, t + t_2\}$$
$$c_{tt} = K_q^t \tag{10}$$
$$K_t^\phi \leq \sum_{j = t + t_1}^{t + t_2} c_{tj}$$
$$K_t^\phi \geq c_{tj} \quad j \in \{t + t_1, \cdots, t + t_2\}$$

The equivalent linear constraints for until operator (10) are constructed using a procedure similar to (Karaman et al. (2008)). The modification for MTL comes from the following result in (Koymans (1990)).

$$K_t^\phi = \bigvee_{j=t+t_1}^{t+t_2} \left((\wedge_{l=t}^{l=j-1} K_p^l) \wedge K_q^j \right).$$

All other combinations of MTL operators for example, eventually-always operator: $\phi = \Diamond_{[t_1,t_2]}\Box_{[t_3,t_4]}p$ and always-eventually operator: $\phi = \Box_{[t_1,t_2]}\Diamond_{[t_3,t_4]}p$ etc., can be translated to similar linear constraints using (5)-(10). In addition to the collective operator constraints, we need another constraint $K_0^\phi = 1$ as well, which suffices to the overall satisfaction of a task specification ϕ.

Using this approach, we can translate an MTL formula ϕ into a set of mixed integer linear constraints $S_\phi(x, b, u, t)$, which converts the associated optimal control problem (e.g. Problem 1) to a MILP for some linear cost function.

5.3 Decomposition of Complex MTL Formulae

Notice that the worst case complexity of the above MILP (Problem 1 with linear constraints and linear cost) is exponential i.e., $O(2^{mT})$, where m is the number of boolean variables or equivalently the number of halfspaces required to express the MTL formula, and T is the discrete time horizon. Therefore, it is logical to consider decomposing the task specification ϕ_i into several simpler sub-tasks ϕ_i^k, where $i \in \{1, 2, ..., N\}$ and $k \in \{1, 2, ..., M\}$.

In (Schillinger et al. (2018)), the authors proposed a timed-automata based approach to decompose a complex LTL specification into a finite number of simpler LTL specifications. In case of MTL, this method is not applicable in general, because MTL to Buchi automata conversion is not always possible. However, in the special case, where the finite time constraints are specified as intervals i.e., in case of Metric Interval Temporal Logic (MITL), we can break down a complex MTL specification (mission) into finite number of simpler MTL specifications (sub-tasks), if the sum of the finite timing intervals in decomposed MTL specifications does not violate the finite timing interval of the original MTL specification. We can state this proposition as the following theorem:

Theorem 1. Given an MITL specification ϕ_i, there exists some finite M-length decomposition ϕ_i^k, $k \in \{1, 2, ..., M\}$, s.t. $\wedge_{k=1}^{M}(\phi_i^k) \implies \phi_i$, if $\sum_{k=1}^{M} T_k \leq T_i$, where T_i is the finite timing interval for ϕ_i, and T_ks are the corresponding finite timing intervals for ϕ_i^k, $\forall k \in \{1, 2, ..., M\}$.

The proof follows directly from Theorem 2 in (Schillinger et al. (2018)). It can also be readily verified using a timed automata simulation in UPAAL (Behrmann et al. (2006)).

Notice that Theorem 1 does not guarantee an equivalence relationship between ϕ_i and ϕ_i^k. Moreover, this decomposition is not unique. However, if the mission is specified by an MITL specification and the system is represented by a hybrid model, which is the case in our problem, one convenient design choice is to pick such ϕ_i^k, for which there is only one associated dynamical mode of the system. We provide examples of such decomposition in Section 6.

5.4 Final Trajectory Generation

By decomposing the mission specification ϕ_i, and by using the MTL to linear constraints translation mechanism, we can replace Problem 1 with a collection of smaller optimization problems, each with a sub-task specification represented as an MTL formula ϕ_i^k, and an associated linear mode of the hybrid model.

Here, the linear cost function of choice is $\sum_{t=0}^{T} |u_i(t)|$, where T is the discrete time horizon for the optimal trajectory. Thus, our final formulation of the problem is given by:

Problem 2.

$$\min_{x_i, u_i} \quad \sum_{t=0}^{T} |u_i(t)|$$
$$\text{s.t.} \quad x_i(t+1) = A_l(t)x_i(t) + B_l(t)u_i(t)$$
$$\mathbf{x_{i}}_{t_0} \models \phi_i^k$$

where ϕ_i^k is the MTL specification for the kth sub-task for the ith UAV, $A_l(t)$, $B_l(t)$ are the linear system matrices for the lth mode, and $\mathbf{x_{i}}_{t_0}$ is the resulting optimal trajectory for the kth sub-task, with $i \in \{1, 2, 3, ..., N\}$, $k \in \{1, 2, 3, ..., M\}$, and $l \in \{1, 2, 3, ..., 5\}$.

Problem 2 represents a collection of MILPs, which can be solved recursively and efficiently using a MILP solver. The resultant trajectories are locally optimal for each individual sub-task, and their existence inherently guarantees safety and finite time completion of the respective sub-tasks. The final trajectory for the complete mission is generated over time by composing all the individual optimal sub-task trajectories. The final path is therefore not optimal but suboptimal with respect to the original mission specification ϕ_i for the ith UAV. However, despite this loss of global optimality, the advantages achieved in terms of reduction in computational complexity, and improved scalability with respect to the number of UAVs are far more important, as is shown in the following section.

6. SIMULATIONS AND RESULTS

We apply the proposed method for solving Problem 2 in the same workspace as shown in Fig. 1. The experiments are run through YALMIP-CPLEX solver using MATLAB interface on an Intel NuC. It is portable computer with an Intel core i7 @ 3.7 GHz CPU, an integrated Intel Iris GPU, and 16 GBs of memory. This setup is directly transferable to a quadrotor as a companion module for onboard computation.

We use a $2m$ neighborhood set threshold for the UAVs i.e., $\rho = 2m$. The discrete time horizon for simulation is $T = 30$, and the UAV altitude limit in *Hover* and *Steer* modes is set to $1.5m$. All dynamics are uniformly discretized at a rate of 5 Hz.

6.1 Case Study I: Validation (2 UAVs)

In the two UAV setting, for the mission ϕ_1, the sub-tasks for the quadrotor q_1 are specified as following:

$$\phi_1^1 = \Box(A) \wedge \Diamond_{[0,5]}(A)' \quad [mode : Take off]$$
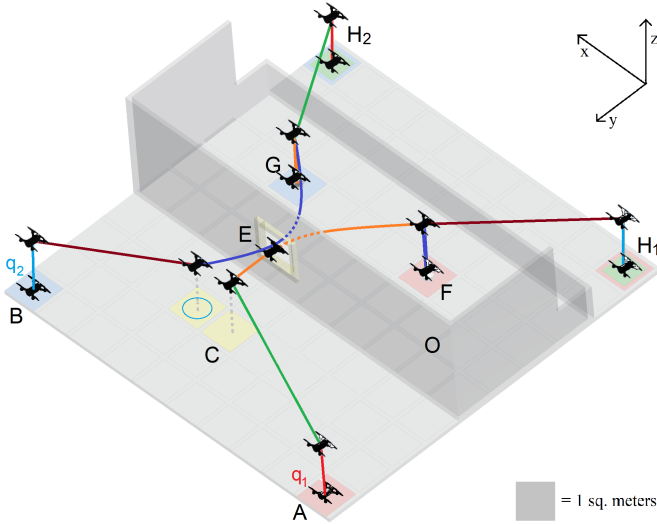$$\phi_1^2 = \Diamond_{[0,5]}(C) \wedge \Box\neg(O) \wedge \Box\neg(q_2) \quad [mode : Steer]$$

Fig. 4. The resultant composed trajectories for the subtasks for q_1 and q_2 operating simultaneously.

$$\phi_1^3 = \Diamond_{[0,10]}(F) \wedge \Box \neg(O) \wedge \Box \neg(q_2) \quad [mode:Steer]$$

$$\phi_1^4 = \Box(F) \wedge \Diamond_{[0,10]}(F)' \quad [mode:Grasp]$$

$$\phi_1^5 = \Diamond_{[0,10]}(H_1) \wedge \Box \neg(O) \wedge \Box \neg(q_2) \quad [mode:Steer]$$

$$\phi_1^6 = \Box(H_1) \quad [mode:Land]$$

Using the convention defined earlier, the specification ϕ_1^1 requires the quadrotor q_1 to attain desired threshold altitude (represented as A') while staying inside the 2D region marked A. ϕ_1^2 requires the quadrotor q_1 to reach C within 5 time units, and ϕ_1^3 requires it to reach F within 10 time units, while avoiding the obstacle O and the neighboring quadrotor q_2. ϕ_1^4 requires the UAV to grasp the object at F within 10 time units while staying in F, whereas ϕ_1^5 asks it to reach H_1 within 10 time units. Finally ϕ_1^6 forces q_1 to stay at H_1 indefinitely. The sub-tasks for quadrotor q_2 are specified in a similar fashion.

Notice, that the constraint $\Box \neg(q_j)$, where $j \in \mathcal{N}_i(t)$, $i \in \{1, 2\}$, enforces the quadrotors to avoid collision when within ρ proximity of its neighbors. In practice, the UAV that is the first to reach region C and is closest to the window E, gets to go through first. The other UAV has to wait in the default *Hover* mode at C.

Each UAV sequentially solves the set of MILPs for its respective sub-task specifications, and moves along the generated optimal trajectory for the corresponding sub-task. The final mission trajectory is generated by recursive composition of all these optimal sub-task trajectories. Figure 4 shows the resulting composed trajectories for both the quadrotors operating simultaneously. Both UAVs safely avoid the obstacles and evacuate their respective objects within given finite time limits. As expected, quadrotor q_2 waits at C. The number of circular rings at C (see Fig. 4) correspond to the waiting time for q_2 in terms of discrete time steps.

Table 1 provides some insight into the timing analysis of each sub-task for both quadrotors. A closer look at this data indicates that all timing constraints are indeed satisfied. Moreover, the computation time for each sub-task indicates that the proposed method can be implemented

Table 1. Timing analysis for the ϕ_i^k for $N = 2$

Task $\phi_i^{\mathbf{k}}$	Computation (sec)	Execution (steps)
ϕ_1^1 $(A - A')$	2.7	$2 \leq 5$
ϕ_1^2 $(A - C)$	6.3	$4 \leq 5$
ϕ_1^3 $(C - F)$	10.3	$7 \leq 10$
ϕ_1^4 $(F - F')$	3.0	$3 \leq 10$
ϕ_1^5 $(F - H_1)$	5.7	$6 \leq 10$
ϕ_1^6 $(H_1 - H_1')$	2.5	$2 \leq 5$
ϕ_2^1 $(B - B')$	2.7	$2 \leq 5$
ϕ_2^2 $(B - C)$	5.8	$4 \leq 5$
ϕ_2^3 $(C - G)$	11.1	$8 \leq 10$
ϕ_2^4 $(G - G')$	3.0	$3 \leq 10$
ϕ_2^5 $(G - H_2)$	5.7	$6 \leq 10$
ϕ_2^6 $(H_2 - H_2')$	2.5	$2 \leq 5$

in real-time. To the best of the authors knowledge, these are one of the fastest computation times reported in the existing MTL based planning literature. The secret to this reduction in computational complexity lies in our divide and conquer approach. From an implementation point of view, the performance can be further improved by using hardware which is optimized for computation (such as Nvidia Jetson TX2 etc.).

6.2 Case Study II: Scalability (N UAVs)

We now consider the case of N number of UAVs to investigate the scalability features of our method. For this case study, we increase the number of UAVs in the simulation setup, one at a time until one of the finite-time constraints in any of the sub-task specifications is violated for at least one of the UAVs. To keep the analysis consistent, we keep the finite timing constraints for all UAVs the same as before.

It turns out that for $N = 7$, the sub-task specification ϕ_7^3 fails the satisfaction criterion and hence no solution exists for this sub-task. That is, for $N > 6$, one of the UAVs fails to reach its respective target object within the finite time limit of 10 units. The reason is that beyond $N = 6$, one access point to the environment is not sufficient to meet the specified timing constraints for all the UAVs, since now the quadrotors have to queue up for longer duration at C in order to avoid collisions at E.

However, this problem can be solved simply either by increasing the finite time limits for this sub-task, or by executing the mission in an environment with multiple access points. Another way to look at this limitation is to identify that $N = 6$ is the maximum number of UAVs that can be deployed successfully under these safety and timing constraints in this particular workspace, and adding more UAVs does not provide any additional value in terms of the success of the mission. Therefore, it is more of a limitation of the workspace than our approach itself.

Figure 5 shows the resulting composed trajectories for 6 UAVs, in which case all safety and timing constraints are satisfied. The computation and execution times for the sub-tasks for $N = 6$ case, indicating the satisfaction of all timing constraints, can be found in the ArXiv version (Fiaz and Baras (2019)) of this paper.
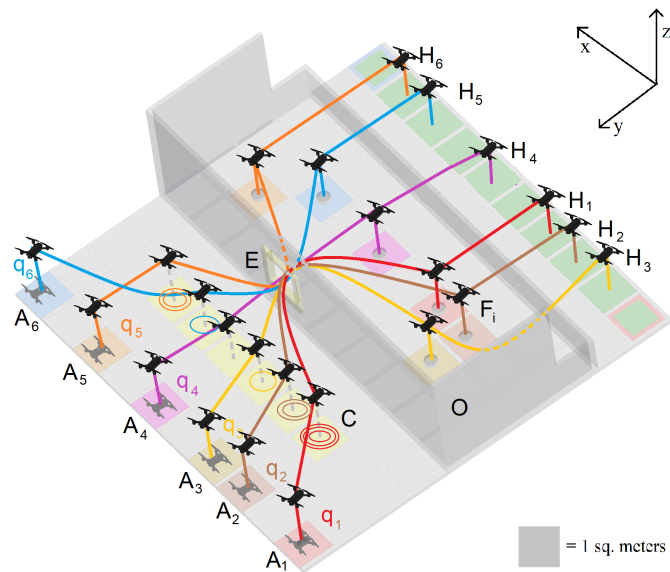
Fig. 5. The resultant composed trajectories for the subtasks for $N = 6$ UAVs operating simultaneously. As before, the number of circular rings at C correspond to the waiting time for the ith quadrotor in terms of discrete steps.

7. CONCLUSIONS AND PROSPECTS

We have proposed a hybrid compositional approach to rescue mission planning for quadrotors with MTL task specifications, and have presented an optimization based method which can be implemented in real-time. Using a simple yet realistic search and rescue test case, we have demonstrated the computational efficiency of our approach, and have shown that by breaking down the mission into several sub-tasks, and by using a hybrid model for the system, it is possible to solve the challenging problem of motion planning for multi-agent systems with rich dynamics and finite time constraints in real-time.

In addition to some promising results, this work also poses many new and interesting questions as well. For example, given a finite time constraint for the whole mission, what is the best or optimal way to divide the timing constraints among various sub-tasks. Of course it is a scheduling problem, and is dependent on many factors such as robot dynamics, its maximum attainable speed, and nature of the sub-tasks as well.

In this study, the individual sub-task timing constraints were constructed in a relaxed and uniform fashion. However, it is worth noticing that using a rich dynamical model for the UAV puts less constraints on its maneuverability, and hence can allow it to tackle more conservative finite time constraints as well.

We anticipate that the scalability features of this approach can be shown with even greater number of UAVs in an environment with multiple access points for instance. Detailed performance comparison of this hybrid approach with some decentralized collision avoidance methods will be beneficial as well. Extension of this work with different tasks, dynamic obstacles other than the UAVs themselves, conservative time constraints, and tolerances in both time

and space can also yield interesting results, and are all great directions for future work.

REFERENCES

Baier, C., Katoen, J.P., et al. (2008). *Principles of model checking*, volume 26202649. MIT press Cambridge.

Behrmann, G., David, A., Pettersson, P., Yi, W., and Hendriks, M. (2006). Uppaal 4.0.

Choset, H.M. (2005). *Principles of robot motion: theory, algorithms, and implementation*. MIT press.

Fiaz, U.A., Abdelkader, M., and Shamma, J.S. (2018). An intelligent gripper design for autonomous aerial transport with passive magnetic grasping and dual-impulsive release. In *AIM 2018*, 1027–1032. IEEE.

Fiaz, U.A. and Baras, J.S. (2019). Fast, composable rescue mission planning for uavs using metric temporal logic. *arXiv preprint arXiv:1912.07848*.

Fiaz, U.A., Toumi, N., and Shamma, J.S. (2017). Passive aerial grasping of ferrous objects. *IFAC-PapersOnLine*, 50(1), 10299–10304.

Fiaz, U.A., Shamma, J.S., and Abdelkader, M.A. (2019). Impulsive release mechanism and method. US Patent App. 16/266,800.

García, L.R., Dzul López, A.E., Lozano, R., and Pégard, C. (2013). Modeling the quad-rotor mini-rotorcraft. *Quad Rotorcraft Control*, 23–34.

Goerzen, C., Kong, Z., and Mettler, B. (2010). A survey of motion planning algorithms from the perspective of autonomous uav guidance. *JIRS*, 57(1-4), 65.

Kantaros, Y. and Zavlanos, M.M. (2018). Temporal logic optimal control for large-scale multi-robot systems: 10 400 states and beyond. In *CDC*, 2519–2524. IEEE.

Karaman, S., Sanfelice, R.G., and Frazzoli, E. (2008). Optimal control of mixed logical dynamical systems with ltl specifications. In *CDC*, 2117–2122. IEEE.

Koymans, R. (1990). Specifying real-time properties with metric temporal logic. *Real-time systems*, 2(4), 255–299.

LaValle, S.M. (2006). *Planning algorithms*. Cambridge university press.

Maity, D. and Baras, J.S. (2015). Motion planning in dynamic environments with bounded time temporal logic specifications. In *MED 2015*, 940–946. IEEE.

Michael, N., Mellinger, D., Lindsey, Q., and Kumar, V. (2010). The grasp multiple micro-uav testbed. *IEEE Robotics & Automation Magazine*, 17(3), 56–65.

Nikou, A., Tumova, J., and Dimarogonas, D.V. (2016). Cooperative task planning of multi-agent systems under timed temporal specifications. In *ACC*. IEEE.

Ouaknine, J. and Worrell, J. (2008). Some recent results in metric temporal logic. In *Formal Modeling and Analysis of Timed Systems*, 1–13. Springer.

Schillinger, P., Bürger, M., and Dimarogonas, D.V. (2018). Decomposition of finite ltl specifications for efficient multi-agent planning. In *DARS*, 253–267. Springer.

Wolff, E.M., Topcu, U., and Murray, R.M. (2014). Optimization-based trajectory generation with ltl specifications. In *ICRA 2014*.

Xi, W., Tan, X., and Baras, J.S. (2005). A hybrid scheme for distributed control of autonomous swarms. In *ACC 2005*, 3486–3491. IEEE.

Zhou, Y., Maity, D., and Baras, J.S. (2015). Optimal mission planner with timed temporal logic constraints. In *2015 European Control Conference (ECC)*. IEEE.