A Hybrid Compositional Approach to Optimal Mission Planning for Multi-rotor UAVs using Metric Temporal Logic

Usman A. Fiaz and John S. Baras

Abstract—This paper investigates a hybrid compositional approach to optimal mission planning for multi-rotor Unmanned Aerial Vehicles (UAVs). We consider a time critical search and rescue scenario with two quadrotors in a constrained environment. Metric Temporal Logic (MTL) is used to formally describe the task specifications. In order to capture the various modes of UAV operation, we utilize a hybrid model for the system with linearized dynamics around different operating points. We divide the mission into several sub-tasks by exploiting the invariant nature of various task specifications i.e., the mutual independence of safety and timing constraints along the way, and the different modes (i.e., dynamics) of the robot. For each sub-task, we translate the MTL formula into linear constraints, and solve the associated optimal control problem for desired path using a Mixed Integer Linear Program (MILP) solver. The complete path is constructed by the composition of individual optimal sub-paths. We show that the resulting trajectory satisfies the task specifications, and the proposed approach leads to significant reduction in computational complexity of the problem, making it possible to implement in real-time.

I. 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, disaster relief, surveillance, autonomous aerial transport, and entertainment. The motivation behind this expanding zeal towards multi-rotor UAVs is twofold. First, these are highly inexpensive robots as compared to their fixed-wing counterparts, and are therefore leading the way as a standard platform to operate in congested environments as well, such as crowded city skies and constrained indoor workspace.

The enormous impact of quadrotors in autonomous search and rescue missions becomes evident during natural disasters, where it is impossible for humans and terrestrial robots to access highly cluttered spaces to rescue people from life threatening situations. In such scenarios, these agile multi-rotor UAVs come to our rescue. They can either locate and grab a target themselves or can serve as a guide for other robots as well as humans to help evacuate a target. For this reason, an enormous amount of work has been done in designing new control strategies and planning algorithms for this specific class of aerial robots, with the aim to deploy them eventually in safety and time critical missions [1], [2].

Given any high level task, it is a standard practice in classical motion planning literature, to look for a trajectory or a set of trajectories, which the robot (i.e., the UAV) can follow (i.e., which its dynamics permit), 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 a control law, to go from one position to another in space while satisfying some constraints [3]. Traditionally, several interesting methods such as potential functions [4], and optimization through generation of probabilistic maps in high dimensional state-space [5], have been used for robot mission planning. However, all these methods tend to fail in situations where the task involves some finite time constraints or dynamic specifications. Aerial surveying of areas and time-critical search and rescue scenarios are two common examples of such tasks.

Temporal logic [6]–[8] 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) [9], [10], which is useful for specifying tasks such as visiting certain objects periodically, surveying areas, ensuring stability and safety etc. [11]–[13]. However, from a control theory perspective, LTL only accounts for timing in the infinite horizon sense i.e., it can only guarantee something will eventually happen and is not rich enough to describe finite time constraints. In addition, the traditional LTL formulation such as in [14], assumes a static environment, which does not admit incorporating dynamic task specifications.

On the other hand, Metric Temporal Logic (MTL) [15], [16], 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 was proposed in [17], and [18], where they translate the LTL task 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 [19] and [20], where the authors used bounded time temporal constraints using extended LTL, and a timed automata approach respectively, for motion planning with linear system models. However, both these approaches did not incorporate a well defined or rich dynamical model of the robot, and also illustrated significant computational complexity issues for the

The authors are with the Department of Electrical and Computer Engineering and the Institute for Systems Research at University of Maryland, College Park, 20742 MD, USA. {{fiaz,baras}@umd.edu}
proposed methods. Similarly, optimization based methods with MTL specifications were presented in [21], and [22] for single and multiple robots respectively. However, in both cases, the computation of the resulting trajectory is expensive (∼500 sec), and hence cannot be performed in real-time. Moreover, these works have put high constraints on the robot maneuverability, by limiting the dynamics to a simple linear (point-robot) model, which is contradictory to the primary reason for deploying quadrotors in constrained dynamic environments.

In this work, we propose a compositional optimization based approach for motion planning with two UAVs in a time critical search and rescue mission scenario. We use MTL specifications to formally describe the task, and unlike all of the described previous works, we utilize a complete hybrid dynamical model of the robot (i.e., the quadrotor), to capture its various modes of operation. We divide the mission into several sub-tasks based on the different dynamics of the robot, and the invariant nature of safety and timing constraints along the way. For each sub-task, we translate the MTL formulae into mixed integer linear constraints, and solve the associated optimization problem for desired optimal path using a MILP solver. The final path is generated by composing the individual optimal sub-paths. We show that by breaking down the motion planning problem into several sub-problems, we can compute the optimal sub-paths in real time while satisfying the safety and timing constraints. Thus, the main contribution of this paper is the demonstration of a real-time close-to-optimal mission planner for multiple UAVs with MTL specifications, while using a complete hybrid model for the robot with minimum and realistic assumptions.

The rest of the paper is organized as following. In Section II we define the problem and describe the task using a custom built workspace. Section III presents the notation used and some preliminaries on system dynamics, and MTL. In Section IV we define the hybrid dynamical model for the quadrotor and discuss the linearization strategy for its various modes. Section V describes the formulation of the optimization problem for the task, and its decomposition into several sub-tasks. Then, in Section VI we detail our approach to solving the problem. Section VII covers the results drawn from simulations. In Section VIII we summarize and analyze the outcomes of this work briefly, before concluding with some future prospects.

II. PROBLEM STATEMENT

We consider a simple time critical search and rescue mission defined on a constrained workspace as shown in Fig. I. The workspace is custom designed in Autodesk Inventor Professional environment, to closely suit the problem specifications. The mission is desired as follows: Starting from initial positions A and B, two quadrotor UAVs q1 and q2 need to rescue (i.e., evacuate) two objects (shown in black and white circular disks) from a constrained environment respectively. The objects are located at F and G respectively, and are accessible only through a window E, with dimensions such that it allows only one UAV to pass at a given time. Quadrotor q1 has priority over quadrotor q2 i.e., in the case that both quadrotors arrive at the window E at the same time, q2 has to wait for q1 to pass first. We also assume that there are no additional obstacles in the area other than the boundary walls O and the two UAVs. The specific task for each UAV is to grasp its respective target object, and transport it to safety (marked H1 and H2 respectively for q1 and q2 in the workspace) in given finite time. While doing so, the UAVs need to avoid the wall O as well as each other, in particular at the window E. In the later sections, we also use prime region notation; for example, A' to represent the same 2D region A. Here A' represents an altitude (w.r.t z-axis) variation of the quadrotor while it is in the same 2D region A.

III. NOTATION AND PRELIMINARIES

In this section, we describe some mathematical notation and preliminaries that are followed throughout, in rest of the paper. These are essential to set up the described problem statement as an optimal control problem with temporal constraints.

A. System Dynamics

Any general (possibly nonlinear) dynamical system can be represented in the form:

\[ \dot{x}(t) = f(t, x, u) \]

where for all time t continuous
- \( x(t) \in \mathcal{X} \subseteq \mathbb{R}^n \) is the state vector of the system
- \( x_0 \triangleq x(0) \in \mathcal{X}_0 \subseteq \mathcal{X} \) is the initial condition of the state vector and
- \( u(t) \in \mathcal{U} \subseteq \mathbb{R}^m \) is the set of control inputs which is constrained in the control set \( \mathcal{U} \).
A. General Nonlinear Model

Given a nonlinear model of the system, a linearization around an operating point \((x, u(t))\) is expressed as:
\[
\dot{x}(t) = A(t)\dot{x}(t) + B(t)\dot{u}(t)
\]
where for all time \(t\) continuous
- \(\dot{x}(t) = x(t) - x_s(t)\)
- \(\dot{u}(t) = u(t) - u_s(t)\)
- \(A(t) = \frac{\partial f}{\partial x}(t)\bigg|_{x=x_s,u=u_s}\)
- \(B(t) = \frac{\partial f}{\partial u}(t)\bigg|_{x=x_s,u=u_s}\)

If time \(t\) is discretized, then the system dynamics take the form:
\[
x(t+1) = f(t, x(t), u(t))
\]
where as before, \(x(t) \in X, x(0) \in X_0 \subseteq X\), and \(u(t) \in U\) for all \(t = 0, 1, 2, \ldots\). Let us denote the trajectory for System (1), with initial condition \(x_0\) at \(t_0\), and input \(u(t)\) as: \(x_{t_0,t_0+\tau}(t) = x(t) = f(t, x(t), u(t)), x(t_0) = x_0\). However, in this paper, for the sake of convenience, we use the shorthand notation i.e., \(x_{t_0}\) instead of \(x_{t_0,t_0+\tau}(t)\) to represent system trajectory whenever no explicit information about \(u(t)\) and \(x(t)\) is required. Similar to the the continuous time case, the corresponding linearized system in discrete time looks like:
\[
\dot{x}(t+1) = A(t)\dot{x}(t) + B(t)\dot{u}(t)
\]
for all \(t = 0, 1, 2, \cdots\). We use System (2) form dynamics in our final problem formulation in Section V

B. Metric Temporal Logic (MTL)

Now, we introduce some essential terminology and semantics for MTL. The convention on MTL syntax and semantics followed in this paper is the same as presented in [15]. More details on specifying tasks as MTL formulae can also be found in [16].

Definition 1: An atomic proposition is a statement with regard to the state variables \(x\) that is either True(\(T\)) or False(\(\bot\)) for some given values of \(x\). [10]

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

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 formulae are defined in accordance with the following rules of grammar:
\[
\phi ::= T | \pi | \neg\phi | \phi \land \phi | \phi U_I \phi
\]
where \(I \subseteq [0, \infty], \pi \in \Pi, T\) and \(\neg T(= \bot)\) are the Boolean constants for true and false respectively. \(\lor\) represents the disjunction while \(\neg\) represents the negation operator. \(U_I\) 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.

Definition 3: The semantics of any MTL formula \(\phi\) is recursively defined over a trajectory \(x_t\) as:
\[
x_t \models \pi \iff \pi \in L(x(t), t)
\]
\[
x_t \models \neg\phi \iff \phi \not\models x_t
\]
\[
x_t \models \phi_1 \lor \phi_2 \iff x_t \models \phi_1 \text{ or } x_t \models \phi_2
\]
\[
x_t \models \phi_1 \land \phi_2 \iff x_t \models \phi_1 \text{ and } x_t \models \phi_2
\]
\[
x_t \models \phi_0 \iff x_{t+1} \models \phi
\]
\[
x_t \models U_I \phi_0 \iff \exists t' \in I \text{ s.t. } x_{t+t'} \models \phi_0 \text{ and } \forall t'' \leq t', x_{t+t''} \models \phi_1.
\]

Thus, for instance, the expression \(\phi_1 U_I \phi_2\) means the following: \(\phi_2\) is true within time interval \(I\), and until \(\phi_2\) is true, \(\phi_1\) must be true. Similarly, the MTL operator \(\phi_0\) means that \(\phi\) is true at next time instance. \(\phi_0\) means that \(\phi\) is always true for the time duration or during the interval \(I\). \(\phi_1\) means that \(\phi\) eventually becomes true within the interval \(I\).

More complicated formulas can be specified using a variety of compositions of two or more MTL operators. For example, \(\phi_1 U_I \phi_2\) suffices to the following: within time interval \(I\), \(\phi\) will be eventually true and from that time instance, it will always be true for an interval or duration of \(I\). The remaining Boolean operators such as implication (\(\Rightarrow\)) and equivalence (\(\Leftrightarrow\)) can also be represented using the grammar rules and semantics given in Definition 2 and Definition 3.

Also, similar to the convention used in Definition 3, a system trajectory \(x_{t_0}\) satisfying an MTL specification \(\phi\) is denoted as \(x_{t_0} \models \phi\). This is the general temporal constraint representation, which we use later in the optimal control problem formulation.

IV. QUADROTOR DYNAMICS

We adopt the generalized nonlinear dynamics for the quadrotor model presented in [23]. 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 and present our solution approach, since it requires all constraints in the problem to be linear. (see Section VI).

A. 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 $[x, y, z, vx, vy, vz, \phi, \theta, \psi, \dot{\phi}, \dot{\theta}, \dot{\psi}]^T$, where $\xi = [x, y, z]^T$ and $v = [vx, vy, vz]^T$ represent the position and velocity of the quadrotor 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:

$$\begin{align*}
\dot{\xi} &= v \\
\dot{v} &= -g e_3 + F R e_3 \\
\dot{\Omega} &= J^{-1}(-\Omega \times J \Omega + \tau)
\end{align*}$$

where $J$ is the moment of inertia matrix for the quadrotor, $g$ is the gravitational acceleration, $e_3 = [0, 0, 1]^T$, $F$ 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, $F, \tau_x, \tau_y,$ and $\tau_z$ are the four control inputs to System [3].

### B. 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 robot, 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, that some edges donot exist; for example, the quadrotor cannot go from Land to Hover without taking the action Take off. Each state or mode of the transition system follows certain dynamics, which result from a linearization of System [3] around a different operating point. As an 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 $\psi$ and $\omega_\psi$ 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., $F, \tau_x,$ and $\tau_y$. 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:

$$A = \begin{bmatrix} 0 & I & 0 & 0 \\ 0 & 0 & -g & 0 \\ 0 & 0 & 0 & I \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1/m & 0 \\ 0 & I_{2 \times 3} J^{-1} \end{bmatrix}$$

where $I_{2 \times 3} = [I_{2 \times 2} \quad 0_{2 \times 1}],$ 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 [23] and [24].

### C. 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. Thankfully, in case of aerial grasping, some modern passive aerial grasping mechanisms [25] have been shown to be very reliable in grasping an object with an instantaneous touchdown onto its surface [26]. Under this instantaneous touchdown and grasp assumption, 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. Figure 3 depicts this representation of Grasp mode in terms of the Hover, Land, and Take off dynamics.
V. PROBLEM FORMULATION

The described search and rescue problem can be set up as a standard optimal control formulation in discrete time. Given some general system dynamics \([\mathbf{f}, \mathbf{b}]\), 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. This control also optimizes some cost function, while the associated task constraints are specified by an MTL expression. Let \(\phi\) denote the MTL formula for a task specification, and \(J(x(t), u(t))\) be the cost function to be minimized. For the stated search and rescue scenario, a possible MTL task specification for quadrotor \(q_1\) can be:

\[
\phi = \diamond [0, T_1] F \land \Box [0, T_2] F' \land \diamond [0, T_3] H_1 \land \Box \neg O \land \Box \neg q_2
\]

where \(T_3 > T_2 > T_1\) are discrete time units. A similar specification can be written for quadrotor \(q_2\). Then, the corresponding optimization problem is given as:

**Problem 1:**

\[
\begin{align*}
\min_{x,u} & \quad J(x(t), u(t)) \\
\text{subject to} & \quad x(t+1) = f(t, x(t), u(t)) \\
& \quad x_{t_0} = \phi
\end{align*}
\]

Problem 1 is a discrete time optimal control problem with non-linear system dynamics, which is difficult to solve even in cases with simple \(\phi\). In our particular case, it is only harder. Thus, a standard practice is to use linear dynamics at the expense of sacrificing maneuverability of the robot, in order to simplify the problem (as is done in [21] etc.).

**Problem 2:**

\[
\begin{align*}
\min_{x,u} & \quad J(x(t), u(t)) \\
\text{subject to} & \quad x(t+1) = A(t)x(t) + B(t)u(t) \\
& \quad x_{t_0} = \phi
\end{align*}
\]

However, since we are using a hybrid model for the system, and not a linearization about a single operating point only, we are not sacrificing the dynamics and hence the maneuverability of the robot as much. Indeed we have five different modes or sets of linear dynamics associated with the system. Therefore, now the question arises that given an MTL task specification \(\phi\), what linear dynamics or which mode is best suited to solve Problem 2. The answer to this question leads us to a very interesting conclusion. In general, the task specification \(\phi\) is complex and asks the robot to perform actions which require the use of multiple modes. This is true in our problem as well. Therefore, an efficient way to solve Problem 2 is to break down the original task into several sub-tasks by exploiting their invariant properties along the way, and based on the various modes associated with each sub-task.

Thus, having a hybrid dynamical model with linear modes (which are linearizations of System 2 at different operating points) enables us to break down the task specification \(\phi\) into several sub-tasks. In addition, a closer inspection of \(\phi\) reveals that all safety and timing constraints specified by \(\phi\) need not be satisfied by either of the robots during the whole mission. There are some critical safety constraints such as “always avoid \(O\)” that need to be satisfied most of the times, but majority of the constraints are purely local, based on the position of the robot in the workspace. For example, a timing constraint for quadrotor \(q_1\) to grasp the object at location \(F\) within 10 time units is independent from (or invariant of) a timing constraint for it to reach location \(C\) from \(A\) within 5 time units, and so on. The mutual invariance of such local constraints or sub-tasks can also be formally verified using UPAAL [27] for instance.

Thus, Problem 2 can be replaced by a collection of smaller optimization problems, each with a sub-task specification represented as an MTL formula \(\phi_{sub,i}\), with the associated \(i\)th-mode linear dynamics from the hybrid model. Further, we choose the cost function to be linear as well i.e., \(\sum_{t=0}^{N} |u_i(t)|\), where \(N\) is the horizon for the optimal trajectory. Thus, our final formulation of the problem is given as:

**Problem 3:**

\[
\begin{align*}
\min_{x_i, u_i} & \quad \sum_{t=0}^{N} |u_i(t)| \\
\text{subject to} & \quad x_i(t+1) = A_i(t)x_i(t) + B_i(t)u_i(t) \\
& \quad x_{t_0} = \phi_{sub,i}
\end{align*}
\]

where \(\phi_{sub,i}\) is the MTL specification for the \(i\)th sub-task, \(A_i(t), B_i(t)\) are the linear system matrices for the \(i\)th mode, and \(x_{t_0}\) is the resulting optimal trajectory for the sub-task \(\phi_{sub,i}\). For example, for quadrotor \(q_1\), one sub-task is to go from \(A\) to \(C\) in 5 time units. The MTL specification for this sub-task is given by \(\phi_{q1(AC)} = \diamond [0, 5] C \land \neg O\), and the associated dynamics are selected from the Steer mode.

VI. SOLUTION APPROACH

Having setup Problem 3 i.e., an optimization problem with linear cost function and linear dynamics, we now describe an approach to translate the MTL specification \(\phi_{sub,i}\) to linear constraints. Our method is based on the approach presented in [17] where the authors translate LTL specifications to linear constraints. We will start with a simple temporal specification and work through the procedure to convert it to mixed integer linear constraints. We will then use this example as a foundation for translating other MTL operators to equivalent linear constraints.

Consider the constraint that a trajectory \(x(t)\) lies within a convex polytope \(K\) at time \(t\). Since \(K\) is convex, it can be represented as an intersection of a finite number of halfspaces. A halfspace can be represented as set of points, \(H_i = \{x : h_i^T x \leq a_i\}\). Thus, \(x(t) \in K\) is equivalent to \(x(t) \in \cap_{i=1}^{n} H_i\) and \(x(t) \in H_i\). So, the constraint \(x(t) \in K \land \forall t \in \{t_1, t_1 + 1, \cdots, t_1 + n\}\) can be represented by the set of linear constraints \(\{h_i^T x(t) \leq a_i\}, i \in \{1, 2, \cdots, n\}\) 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 system state via conjunction and disjunction of linear halfspaces [17]. 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 \(t = 0, \cdots, N\). 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:
\[
h_i^T x(t) \leq a_i + M(1 - b_i^t)
\]
\[
h_i^T x(t) \geq a_i - M b_i^t + \epsilon
\]
where \( M \) and \( \epsilon \) are some large and small positive numbers respectively. If we denote \( K_i^t = \land_{i=1}^m b_i^t \), then \( K_i^t = 1 \iff x(t) \in K \). 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 \( \phi \). Using the described procedure, once we have obtained \( S_\phi(x, b, u, t) \) for atomic propositions \( p \in \Pi \), we can formulate \( S_\phi(x, b, u, t) \) for any MTL formula \( \phi \). Now, for the Boolean MTL operators, such as \( \neg, \land, \lor \), let \( t \in \{0, 1, \ldots, N\} \), and as before \( K_i^t \in [0,1] \) be the continuous variables associated with the formula \( \phi \) 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:
\[
K_i^t = 1 - K_i^p
\]
the conjunction operator, \( \phi = \land_{i=1}^m p_i \), is modeled as:
\[
K_i^\phi \leq K_{i}^{p_{i}}, \quad i = 1, \ldots, m
\]
\[
K_i^\phi \geq 1 - m + \sum_{i=1}^m K_i^{p_{i}}
\]
and the disjunction operator, \( \phi = \lor_{i=1}^m p_i \), is modeled as:
\[
K_i^\phi \geq K_{i}^{p_{i}}, \quad i = 1, \ldots, m
\]
\[
K_i^\phi \leq \sum_{i=1}^m K_i^{p_{i}}
\]
Similar to binary operators, temporal operators such as \( \diamond, \Box \), and \( \mathcal{U} \) can be modeled using linear constraints as well. Let \( t \in \{0, 1, \ldots, N-2\} \), where \( [t_1, t_2] \) is the time interval used in the MTL specification \( \phi \). Then, eventually operator: \( \phi = \Diamond_{[t_1, t_2]} p \) is modeled as:
\[
K_t^\phi \geq K_{t}^{p_{}}, \quad \tau \in \{t + t_1,\ldots, t + t_2\}
\]
\[
K_t^\phi \leq \sum_{\tau=t+t_1}^{t+t_2} K_{\tau}^p
\]
and always operator: \( \phi = \Box_{[t_1, t_2]} p \) is represented as:
\[
K_t^\phi \leq K_{t}^{p_{}}, \quad \tau \in \{t + t_1,\ldots, t + t_2\}
\]
\[
K_t^\phi \geq \sum_{\tau=t+t_1}^{t+t_2} K_{\tau}^p - (t_2 - t_1)
\]
and until operator: \( \phi = p \mathcal{U}_{[t_1, t_2]} q \) is equivalent to:
\[
c_{tj} \leq K_{q_{}}^{j}, \quad j \in \{t + t_1,\ldots, t + t_2\}
\]
\[
c_{tj} \leq K_{p_{}}^{j}, \quad l \in \{t,\ldots, j-1\}, j \in \{t + t_1,\ldots, t + t_2\}
\]
\[
c_{tj} \geq K_{q_{}}^{j} + \sum_{l=t}^{j-1} K_{p_{}}^{l} - (j-t), \quad j \in \{t + t_1,\ldots, t + t_2\}
\]
\[
c_{tt} = K_{q_{}}^{t}
\]
\[
K_t^\phi = \sum_{j=t+t_1}^{t+t_2} (\land_{j=t}^{j-1} K_{p_{}}^{j} \land K_{q_{}}^{j})
\]
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, at the end, we need the constraint \( K_0^\phi = 1 \), which suffices to the overall satisfaction of a task specification \( \phi \). Further details on this method can be found in [17].

Using this approach, we translate the sub-task MTL constraints i.e., \( \phi_{sub_i} \) to a set of mixed integer linear constraints \( S_\phi(x, b, u, t) \). This way Problem 3 converts into a Mixed Integer Linear Program (MILP), which can be solved using a solver, quite efficiently.

Notice, that the worst case complexity of this MILP is exponential i.e., \( \mathcal{O}(2^mN) \), where \( m \) is the number of boolean variables or equivalently the number of halfspaces required to express the MTL formula, and \( N \) is the discrete time horizon. This also backs our concept of decomposing the task specification \( \phi \) into several simpler sub-tasks \( \phi_{sub_i} \).

VII. SIMULATION AND RESULTS

We apply the described method for solving Problem 3 collection, in the same workspace as shown in Fig. 1. It is a custom built 10x10x3 cubic meter CAD environment which is readily importable across many simulation and computation software tools. In our case study, the simulation experiments are run through YALMIP-CPLEX 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.

For the given problem statement (i.e., the task \( \phi \)), the sub-task specifications for the quadrotor \( q_1 \) are specified as following along with their respective dynamics.
\[
\phi_{q1(\text{AA}')} = \Box A \land \Diamond_{[0,5]} A' \quad \text{[mode : Take off]}
\]
\[
\phi_{q1(\text{AC})} = \Diamond_{[0,5]} C \land \Box \neg O \quad \text{[mode : Steer]}
\]
\[
\phi_{q1(\text{CF})} = \Diamond_{[0,10]} F \land \Box \neg O \quad \text{[mode : Steer]}
\]
Furthermore, the same time. Fairness of this specification is easy to verify, and requires the quadrotor units. Finally grasp the object at F while avoiding the obstacle while staying inside the 2D region marked O. The Hover mode is implicitly present in all modes. In absence of q1, a similar trajectory for quadrotor q2 is obtained with its respective areas of interest.

\[ \phi_{q1}(FF') = □F \land \Diamond[0.10]F' \quad \text{[mode : Grasp]} \]
\[ \phi_{q1}(FH_1) = \Diamond[0.10]H_1 \land □¬O \quad \text{[mode : Steer]} \]
\[ \phi_{q1}(H_1H'_1) = □H_1 \quad \text{[mode : Land]} \]

where O represents the obstacle. Using the convention defined earlier, the specification \( \phi_{q1}(AA') \) requires the quadrotor q1 to attain desired threshold altitude (represented as A') while staying inside the 2D region marked A. \( \phi_{q1}(AC) \) requires the quadrotor q1 to reach C within 5 time units, and \( \phi_{q1}(CF) \) requires it to reach F within 10 time units, while avoiding the obstacle O. \( \phi_{q1}(FF') \) requires the robot to grasp the object at F within 10 time units while staying in F, whereas \( \phi_{q1}(FH_1) \) asks the robot to reach H1 within 10 time units. Finally \( \phi_{q1}(H_1H'_1) \) forces q1 to stay at H1 indefinitely. Similarly, the sub-task specifications for quadrotor q2 are given as following.

\[ \phi_{q2}(BB') = □B \land \Diamond[0.5]B' \quad \text{[mode : Take off]} \]
\[ \phi_{q2}(BD) = \Diamond[0.5]D \land □¬O \quad \text{[mode : Steer]} \]
\[ \phi_{q2}(D) = □D U(¬(pos(q1)? = C) \quad \text{[mode : Hover]} \]
\[ \phi_{q2}(DG) = \Diamond[0.10]G \land □¬O \quad \text{[mode : Steer]} \]
\[ \phi_{q2}(GG') = □G \land \Diamond[0.10]G' \quad \text{[mode : Grasp]} \]
\[ \phi_{q2}(GH_2) = \Diamond[0.10]H_2 \land □¬O \quad \text{[mode : Steer]} \]
\[ \phi_{q2}(H_2H'_2) = □H_2 \quad \text{[mode : Land]} \]

Notice, that quadrotor q2 has an additional constraint namely \( \phi_{q2}(D) \), which assigns priority to q1 by keeping q2 waiting at D, thus ensuring that both robots do not end up at E at the same time. Fairness of this specification is easy to verify, since it is impossible for q1 to be stuck at C indefinitely. Furthermore, the Hover mode is implicitly present in all sub-tasks and is involved in switching between different modes. However, there is no corresponding optimization required once the UAV is within the desired region. For the simulation, we use the cost function J as in Problem 3 with \( N = 30 \) i.e., the number of discrete time steps. The quadrotor dynamics for all modes are uniformly discretized at a rate of 5Hz.

The resulting trajectory for the complete mission is generated by concatenation of optimal sub-task trajectories. For quadrotor q1 alone, the final composed and continuous trajectory (interpolated between discrete points) is shown in Fig. 4. The quadrotors successfully avoid the obstacle and evacuate the object to safety within specified time. In absence of q1, a similar trajectory for quadrotor q2 is obtained with its respective areas of interest.

Figure 5 shows the resulting composed trajectories for both the quadrotors operating simultaneously. Again, both safely avoid the obstacle and evacuate their respective objects within given times. As expected, quadrotor q2 waits at D, for quadrotor q1, to avoid collision. The number of circular rings at D (see Fig. 5) correspond to the waiting time for q2, while it stays at D in Hover mode. Since, both these plots are displaying motion in 3D, it is not possible to visualize the satisfaction of timing constraints per each sub-task along a forth axis. However, we provide some insight to the timing analysis of each sub-task for both quadrotors in terms of computation time. In addition, a closer look at the simulation data indicates that all timing constraints are indeed satisfied.

Table II shows the computation times for each individual sub-task (\( \phi_{subi} \)) for both quadrotors. These numbers indicate that the proposed approach can be implemented in real-time. 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.).
TABLE I
COMPUTATION TIMES FOR SUB-TASKS ($\phi_{sub,i}$)

| Task ($q_1$) | Time (sec) | Task ($q_2$) | Time (sec) |
|--------------|------------|--------------|------------|
| $\phi_{q_1(A'1)}$ | 2.7 | $\phi_{q_2(B'H')}$ | 2.7 |
| $\phi_{q_1(C'1)}$ | 6.3 | $\phi_{q_2(B'D')}$ | 5.8 |
| $\phi_{q_1(C'F)}$ | 10.3 | $\phi_{q_2(D)}$ | 2.0 |
| $\phi_{q_1(F'F')}$ | 3.0 | $\phi_{q_2(DG)}$ | 11.1 |
| $\phi_{q_1(F'H_1)}$ | 5.7 | $\phi_{q_2(G'C')}$ | 3.0 |
| $\phi_{q_1(H_1'H_2)}$ | 2.5 | $\phi_{q_2(G'H_2)}$ | 5.7 |
| $\phi_{q_1(H_2'H_2)}$ | 2.5 |

VIII. DISCUSSION

We have proposed a hybrid compositional approach to 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 task into several sub-tasks, and using a hybrid model for system dynamics, it is possible to solve the challenging problem of motion planning with rich system 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 task, 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 our study, the individual sub-task timing constraint assignments were relaxed and uniform. However, it is worth noticing that using a complete 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. For example, the Steer mode in our model allows the quadrotor to achieve speeds as high as 0.5 m/s, which is not possible with the usual single mode Hover linearization only, which involves small angle (pitch, roll) assumptions.

The scalability features of this approach with regard to a large number of UAVs within a mission, need to be demonstrated. The invariant property of sub-tasks can also be proved rigorously using tools from formal methods. Extension of this work with more complicated tasks, dynamic obstacles, conservative constraints, and tolerances in both time and space can also yield interesting results, and are all great directions for future work.

IX. ACKNOWLEDGMENT

This work was partially supported by the ONR grant N00014-17-1-2622.

REFERENCES

[1] J. Latombe, Robot motion planning, ser. Kluwer international series in engineering and computer science. Boston: Kluwer Academic Publishers, 1991.
[2] S. M. LaValle, Planning algorithms. Cambridge university press, 2006.
[3] H. M. Choset, Principles of robot motion: theory, algorithms, and implementation. MIT press, 2005.
[4] W. Xi, X. Tan, and J. S. Baras, “A hybrid scheme for distributed control of autonomous swarms,” in Proceedings of the 2005 American Control Conference. IEEE, 2005, pp. 3486–3491.
[5] R. Sharma, “Locally efficient path planning in an uncertain, dynamic environment using a probabilistic model,” Robotics and Automation, IEEE Transactions on, vol. 8, no. 1, pp. 105–110, 1992.
[6] C. Baier, J.-P. Katoen et al., Principles of model checking. MIT press Cambridge, 2008, vol. 26202649.
[7] E. M. Clarke, O. Grumberg, and D. Peled, Model checking. MIT press, 1999.
[8] M. M. Quottrup, T. Bak, and R. Zamanabadi, “Multi-robot planning: A timed automata approach,” in Robotics and Automation, 2004. Proceedings. ICRA’04. 2004 IEEE International Conference on, vol. 5. IEEE, 2004, pp. 4417–4422.
[9] C. Goerzen, Z. Kong, and B. Mettler, “A survey of motion planning algorithms from the perspective of autonomous uav guidance,” Journal of Intelligent and Robotic Systems, vol. 57, no. 1-4, p. 65, 2010.
[10] R. Alur, Principles of cyber-physical systems. MIT Press, 2015.
[11] E. Frazzoli, M. A. Dahleh, and E. Feron, “Real-time motion planning for agile autonomous vehicles,” Journal of guidance, control, and dynamics, vol. 25, no. 1, pp. 116–129, 2002.
[12] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas, “Temporal-logic-based reactive mission and motion planning,” IEEE transactions on robotics, vol. 25, no. 6, pp. 1370–1381, 2009.
[13] E. Plaku and S. Karaman, “Motion planning with temporal-logic specifications: Progress and challenges,” AI communications, vol. 29, no. 1, pp. 151–162, 2016.
[14] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas, “Temporal-logic-based reactive mission and motion planning,” IEEE Transactions on Robotics, vol. 25, no. 6, pp. 1370–1381, 2009.
[15] R. Koymans, “Specifying real-time properties with metric temporal logic,” Real-time systems, vol. 2, no. 4, pp. 255–299, 1990.
[16] J. Ouaknine and J. Worrell, “Some recent results in metric temporal logic,” in Formal Modeling and Analysis of Timed Systems. Springer, 2008, pp. 1–13.
[17] S. Karaman, R. G. Sanfelice, and E. Frazzoli, “Optimal control of mixed logical dynamical systems with linear temporal logic specifications,” in 47th IEEE Conference on Decision and Control. IEEE, 2008, pp. 2117–2122.
[18] E. M. Wolff, U. Topcu, and R. M. Murray, “Optimization-based trajectory generation with linear temporal logic specifications,” in Int. Conf. on Robotics and Automation, 2014.
[19] D. Maity and J. S. Baras, “Motion planning in dynamic environments with bounded time temporal logic specifications,” in 2015 23rd Mediterranean Conference on Control and Automation (MED). IEEE, 2015, pp. 940–946.
[20] Y. Zhou, D. Maity, and J. S. Baras, “Timed automata approach for motion planning using metric interval temporal logic,” in 2016 European Control Conference (ECC). IEEE, 2016, pp. 690–695.
[21] ——, “Optimal mission planner with timed temporal logic constraints,” in 2015 European Control Conference (ECC). IEEE, 2015, pp. 759–764.
[22] A. Nikou, J. Tumova, and D. V. Dimarogonas, “Cooperative task planning of multi-agent systems under timed temporal specifications,” in 2016 American Control Conference (ACC). IEEE, 2016, pp. 7104–7109.
[23] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, “The grasp multiple micro-uav testbed,” IEE Robotics & Automation Magazine, vol. 17, no. 3, pp. 56–65, 2010.
[24] L. R. García Carrillo, A. E. Dzul López, R. Lozano, and C. Pégard, “Modeling the quad-rotor mini-rotorcraft,” Quad Rotorcraft Control, pp. 23–34, 2013.
[25] U. A. Fiaz, N. Toumi, and J. S. Shamma, “Passive aerial grasping of ferrous objects,” IFAC-PapersOnLine, vol. 50, no. 1, pp. 10 299–10 304, 2017.
[26] U. A. Fiaz, M. Abdelkader, and J. S. Shamma, “An intelligent gripper design for autonomous aerial transport with passive magnetic grasping and dual-impulsive release,” in 2018 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM). IEEE, 2018, pp. 1027–1032.
[27] G. Behrmann, A. David, K. G. Larsen, J. Håkansson, P. Pettersson, W. Yi, and M. Hendriks, “Uppaal 4.0,” 2006.