A Set-based Motion Planning Algorithm for Aerial Vehicles in the Presence of Obstacles Exhibiting Hybrid Dynamics

Adam Ames, Nan Wang, Ricardo G. Sanfelice

***Abstract*— This paper proposes a set-based motion planning algorithm to drive a quadrotor to a target location while avoiding multiple moving obstacles that have hybrid dynamics. By exploiting a library of motion primitives, the proposed planner is capable of rapidly generating trajectories that avoid obstacles with unknown angular velocities that instantaneously change at unknown impact times. The planner employs the reachable sets of the vehicle and of the obstacles to generate a set of safe trajectories from which a trajectory with lowest cost is selected. Simulations and experimental validation of the motion planner are presented.**

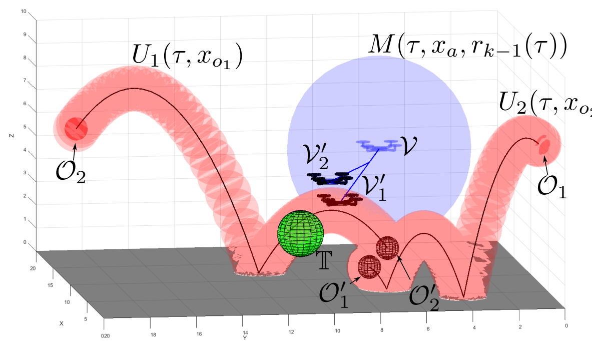

Fig. 1. The quadrotor $\mathcal{V}$ is moving towards the target $\mathbb{T}$ while avoiding the obstacles $\mathcal{O}_1$ and $\mathcal{O}_2$. The set $M(\tau, x_a, r_{k-1}(\tau))$ is the sphere where the quadrotor can move in τ time from initial state x_a with initial reference state $r_{k-1}(\tau)$. The initial states of obstacles $\mathcal{O}_1$ and $\mathcal{O}_2$ are given by x_{o_1} and x_{o_2}, with $\mathcal{O}_1'$ and $\mathcal{O}_2'$ being the obstacles at time τ. The sets $U_1(\tau, x_{o_1})$ and $U_2(\tau, x_{o_2})$ contain all the unsafe points around the path of $\mathcal{O}_1$ to $\mathcal{O}_1'$ and $\mathcal{O}_2$ to $\mathcal{O}_2'$ over the time period $[0, \tau]$. Moving directly towards the target causes a collision with $\mathcal{O}_2$ at $(10, 8.5, 3)$. The planner moves the quadrotor to $(10, 9, 4)$ avoiding the obstacles while moving to the target.

I. INTRODUCTION

Small-scale aerial vehicles keep gaining popularity due to their versatility and wide range of applications. Nowadays, these vehicles are used for search and rescue, fire monitoring, product delivery to just list a few. Their broad applicability has propelled research focusing on solving the motion planning problem with obstacle avoidance, leading to strategies ranging from velocity fields, which generates a vector field to drive the vehicle away from the obstacles and towards the goal [1], to graph search algorithms, which solve the motion planning problem using search algorithms [2]. A particularly popular approach is receding horizon control [3] [4], where the motion planning problem is recast into a constrained optimization problem. In this approach, the constrained optimization problem is recurrently solved over a receding horizon, using current measurements and predictions. Obstacle avoidance is achieved by imposing hard constraints within the optimization problem. In [5], a modular framework, called FaSTrack, is developed to enable motion planning that is fast, dynamically feasible, and safe in the presence of the static obstacles. In [6], a real-time motion planning framework with obstacle avoidance is proposed for quadrotors, but the dynamics of the obstacles are not considered. In [7], a set-based predictive control framework is proposed to optimize trajectories to safely guide a ground vehicle toward the target and predict dynamic obstacles that exhibit only continuous behavior. The motion of the dynamic obstacles is predicted using their discretized model. In [8], a nonlinear model predictive control approach is used to generate a series of safe control inputs for a quadrotor with obstacles that are classified to be static, linear, or projectile. The motion of obstacles is predicted using different continuous-time models depending on the type of obstacle.

While the collision avoidance problem has been widely studied, the case when obstacles exhibit hybrid dynamics, namely, continuous evolution and at times instantaneous changes, has been largely neglected. Obstacles with hybrid dynamics emerge in situations when the obstacles exhibit collisions with the environment, as shown in Figure 2, where a quadrotor is flying near the ground and must avoid the obstacle bouncing toward it. To the best of the authors' knowledge, the only article that allows the obstacle to bounce on the ground is [8]. However, the effect of spin at bounces is not modeled in [8], which negatively affects the fidelity of the predicted obstacle trajectories and may lead to collisions. Moreover, as we argue in this article, the planner may fail at finding a solution due to topological obstructions.

In this paper, we propose a motion planning algorithm that uses a receding horizon approach introducing reachable sets that are computed on the fly and that are used to define a safe motion plan. We define a set called *the mobility set*, that captures where the vehicle can reach over a finite time horizon and a set called *the unsafe set*, that characterizes where the obstacles can reach, based on current estimates. Using these two sets *the safe mobility set* is constructed. The cost for each trajectory in the safe mobility set is calculated and a trajectory with the lowest cost is selected. In comparison to [8], our approach differs as our solution

*This research has been partially supported by the National Science Foundation under Grant no. ECS-1710621, Grant no. CNS-1544396, Grant no. CNS-2039054, by Grant no. CNS-2111688, by the Air Force Office of Scientific Research under Grant no. FA9550-19-1-0053, Grant no. FA9550-19-1-0169, and Grant no. FA9550-20-1-0238, and by the Army Research Office under Grant no. W911NF-20-1-0253.

A. Ames, N. Wang, R. G. Sanfelice are with the Department of Electrical and Computer Engineering, University of California, Santa Cruz. Email: `adrames, nanwang, ricardo@ucsc.edu`

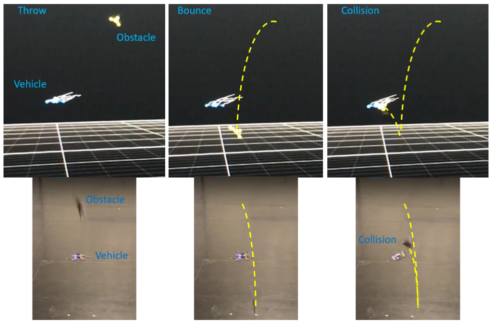

Fig. 2. Motion planning for obstacle avoidance without bounce prediction. The images at the top show the view from the motion capture cameras, with the blue frame showing the quadrotor and the yellow showing the obstacle. The images at the bottom are from the view of the person throwing the obstacle.

includes the spin effect in the obstacle model, resulting in higher fidelity trajectory predictions, and does not suffer from the pathologies typically induced by topological obstructions.

The paper is organized as follows. Preliminaries on hybrid systems are in Section II. The problem statement and proposed solution are outlined in Section III. Models of the vehicle and obstacles are presented in Section IV. The mathematical form of the problem statement is presented in Section V. The motion planning algorithm is detailed in Section VI. Simulation results are presented in Section VII and experimental results are presented in Section VIII.

Notation: The 2 norm of a vector x is denoted as $|x|$, with the infinity norm denoted as $|x|_\infty$. The set of real numbers is represented by $\mathbb{R}$. The set of real numbers greater than or equal to 0 is represented by $\mathbb{R}_{\geq 0}$. The subset Y of X is given as $Y \subset X$. The unit ball $\mathbb{B}$ in $\mathbb{R}^3$ is defined as $\mathbb{B} := \{x \in \mathbb{R}^3 : |x| \leq 1\}$. The set of natural numbers including zero is denoted as $\mathbb{N}$. The set of unit quaternions $\mathbb{S} \subset \mathbb{R}^4$ is defined as $\mathbb{S} := \{q \in \mathbb{R}^4 : |q| = 1\}$. The closure of a set S is $cl(S)$. The vector e_i contains all zeros except the i^{th} position which is 1. The distance from a vector p to a set Q is defined as $\text{dist}(p, Q) := \{\min(|p^\top q|_\infty) : q \in Q\}$. The set of $m \times n$ matrices is denoted $\mathbb{R}^{m \times n}$. The derivative of a differentiable matrix function with arguments $F : \mathbb{R}^{m\times n} \mapsto \mathbb{R}^{k \times l}$ is given by $\mathcal{D}_x(F(x)) = \frac{\partial vec(F(x))}{\partial vec(x)^\top}$ for each $x \in \mathbb{R}^{m \times n}$, with $vec(x)$ denoting the matrix x reshaped as a column vector. The determinant of a matrix R is denoted $det(R)$. For a vector $q \in \mathbb{R}^n$ and a set $P \subset \mathbb{R}^n$, $\arg\max_{p \in P} p^\top q$ is the largest value of $p^\top q$ for all $p \in P$. A sequence of variables x indexed by k is denoted as $\{x_k\}_{k \in \mathbb{N}}$. The Minkowski sum is the addition of two sets such that $A + B := \{a + b : a \in A, b \in B\}$. The range of the set valued map M is the set $\text{rge}\, M = \{y \in \mathbb{R}^n : \exists x \in \mathbb{R}^m$ such that $y \in M(x)\}$. The skew-symmetric matrix form of some vector x is $S(x)$.

II. Preliminaries on Hybrid Systems

A hybrid dynamical system is a system with both continuous and discrete dynamics. A hybrid system $\mathcal{H}$ with state $x \in \mathbb{R}^n$ can be modeled as, following [9],

$$\mathcal{H} := \begin{cases} \dot{x} \ \in F(x) & x \in C \\ x^+ \in G(x) & x \in D \end{cases} \tag{1}$$

The data (C, D, F, G) describes how the system state evolves over time. The flow map F describes the continuous evolution of x while the state is in the flow set C. The jump map G describes the discrete evolution while the state is in the jump set D.

A hybrid time domain $E \subset \mathbb{R}_{\geq 0} \times \mathbb{N}$ is defined as the union of intervals $E \cap ([0, T] \times \{0, 1, \ldots, J\}) = \bigcup_{j=0}^{J-1}([t_j, t_{j+1}], j)$ for each $(T, J) \in E$, with $t_0 = 0 \leq t_1 \leq t_2 \leq \ldots \leq t_J = T$. A hybrid arc ϕ is a function defined on a hybrid time domain, $\text{dom}\phi \subset \mathbb{R}_{\geq 0} \times \mathbb{N}$. A hybrid arc ϕ is a solution to the hybrid system $\mathcal{H}$ if the following conditions are met:

- $\phi(0, 0) \in cl(C) \cup D$
- For all $j \in \mathbb{N}$ such that $I^j := \{t : (t, j) \in \text{dom}\ \phi\}$ has nonempty interior, $\phi(t, j) \in C$ for all $t \in \text{int}\ I^j$ and $\phi(t, j) \in F(\phi(t, j))$ for almost all $t \in I^j$;
- For all $(t, j) \in \text{dom}\ \phi$ such that $(t, j + 1) \in \text{dom}\phi$, $(\phi(t, j)) \in D$ and $\phi(t, j + 1) \in G(\phi(t, j))$.

An input can be easily added to $\mathcal{H}$. For more details, see [9] and [10].

The finite-horizon reachability map for a hybrid system $\mathcal{H}$ collects states the system can reach from some initial state ξ over a given time interval $[0, T]$ with at most J jumps. This reachability map is defined in [11] as

$$\begin{aligned} \mathfrak{R}_{\mathcal{H}}(T, J, \xi) := \{\phi(t, j) : \phi \in \hat{\mathcal{S}}_{\mathcal{H}}(\xi), \\ (t, j) \in \text{dom}\ \phi \cap \mathfrak{T}(T, J)\} \end{aligned} \tag{2}$$

with $\hat{\mathcal{S}}_{\mathcal{H}}(\xi)$ denoting the set of all solutions to the hybrid system $\mathcal{H}$ from initial state ξ and $\mathfrak{T}(T, J)$ denoting the hybrid time horizon $[0, T] \times \{0, 1, 2, \ldots, J\}$.

III. Problem Statement and Proposed Solution

The goal of this paper is to solve the following problem:
Problem: Design an algorithm that generates a reference trajectory that steers a quadrotor to a desired hovering location while evading dynamic obstacles that exhibit hybrid dynamics.

To solve this problem, we propose a feedback motion-planning algorithm that generates a reference trajectory by computing and manipulating two sets. The first set is the mobility set, denoted $M(\tau, \xi_a, \xi_r)$, containing all trajectories the quadrotor can execute from initial state ξ_a and initial reference state ξ_r over the continuous time interval $[0, \tau]$, as shown by $M(\tau, x_a, r_{k-1}(\tau))$ in Figure 1. The second set is the unsafe set, denoted $U_i(\tau, \xi_o)$, containing all points the i^{th} obstacle can reach from initial state ξ_o over the time interval $[0, \tau]$. In Figure 1, the unsafe set for obstacles $\mathcal{O}_1$ and $\mathcal{O}_2$ are $U_1(\tau, x_{o_1})$ and $U_2(\tau, x_{o_2})$. With $M(\tau, \xi_a, \xi_r)$ and $U_i(\tau, \xi_o)$ for each i, the proposed algorithm generates a reference trajectory that drives the quadrotor to the target set using the reference tracking controller without colliding with any obstacle.

IV. Modeling the Obstacles and the Quadrotor

A. Modeling the Obstacles

Each obstacle has the hybrid dynamics of a bouncing ball moving in $\mathbb{R}^3$, with the state of the i-th obstacle given by

$$x_{o_i} := (p_{o_i}, v_{o_i}) \in \mathbb{R}^6 =: \mathcal{X}_{o_i} \tag{3}$$

where $p_{o_i} = (p_{x_{o_i}}, p_{y_{o_i}}, p_{z_{o_i}}) \in \mathbb{R}^3$ is position and $v_{o_i} = (v_{x_{o_i}}, v_{y_{o_i}}, v_{z_{o_i}}) \in \mathbb{R}^3$ is velocity. When moving in free air, the motion of a bouncing ball is modeled using the equations of projectile motion without air resistance, namely

$$(\dot{p}_{o_i}, \dot{v}_{o_i}) = (v_{o_i}, (0, 0, -\gamma)) =: F_{o_i}(x_{o_i}) \tag{4}$$

where $\gamma > 0$ is the gravity acceleration.

The impact between the obstacle and the ground is modeled as an instantaneous velocity change, leading to the definition of the jump map G_{o_i}. The restitution coefficient $\lambda \in (0, 1)$ models the energy loss at impacts. Accelerations in the x and y directions are generated by the spin of the ball. Since the angular velocities are not known, the exact trajectory cannot be predicted. Instead of a single velocity value after a jump, we define the set of possible velocities to capture the effect of all values within the expected angular velocity range. The set $\Sigma \subset \mathbb{R}$ contains all possible changes in linear velocity from the expected angular velocities of the ball. The impact is then modeled as

$$\begin{aligned} x_{o_i}^+ \in G_{o_i}(x_{o_i}) := \{p_{x_{o_i}}\} \times \{p_{y_{o_i}}\} \times \{0\} \times \{v_{x_{o_i}} \\ +\Sigma\} \times \{v_{y_{o_i}} + \Sigma\} \times \{-\lambda v_{z_{o_i}}\} \end{aligned} \tag{5}$$

where $+$ is the Minkowski sum. Each obstacle is assumed to be a point mass which impacts the ground at $p_{z_{o_i}} = 0$.

The combination of the continuous and discrete dynamics of the bouncing ball leads to the following hybrid model for each obstacle $\mathcal{O}_i$:

$$\mathcal{H}_{o_i} = (C_{o_i}, F_{o_i}, D_{o_i}, G_{o_i}), \tag{6}$$

where $C_{o_i} := \{x_{o_i} \in \mathbb{R}^6 : p_{z_{o_i}} \geq 0\}$, F_{o_i} is given in (4), $D_{o_i} := \{x_{o_i} \in \mathbb{R}^6 : p_{z_{o_i}} = 0, v_{z_{o_i}} \leq 0\}$, and G_{o_i} is given in (5).

B. Modeling the Quadrotor and Tracking Controller

The quadrotor $\mathcal{V}$ under the effect of a reference tracking feedback controller is modeled as in [12]. The quadrotor state is given as

$$x_b := (p_a, v_a, R_a, \omega_a) \in \mathbb{R}^3 \times \mathbb{R}^3 \times SO(3) \times \mathbb{R}^3 =: \mathcal{X}_b \tag{7}$$

where $p_a \in \mathbb{R}^3$ is position, $v_a \in \mathbb{R}^3$ is linear velocity, $R_a \in SO(3) := \{R \in \mathbb{R}^{3\times 3} : R^\top R = I_3, det(R) = 1\}$ is orientation, and $\omega_a \in \mathbb{R}^3$ is angular velocity, all of them with respect to the world inertial frame. The dynamics of the quadrotor are

$$\dot{p}_a = v_a, \qquad \dot{v}_a = -R_a e_3 \frac{f}{m} + g e_3 \tag{8}$$

$$\dot{R}_a = R_a S(\omega_a), \quad \dot{\omega}_a = -J^{-1} S(\omega_a) J \omega_a + J^{-1} \mathcal{M} \tag{9}$$

with quadrotor mass m, thrust input $f \in \mathbb{R}$, torque input $\mathcal{M} \in \mathbb{R}^3$, and inertia tensor $J \in \mathbb{R}^{3\times 3}$.

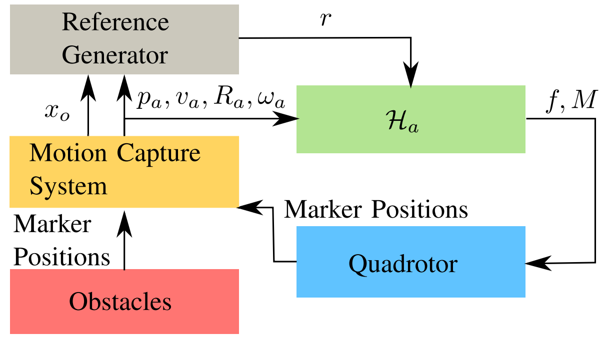

Fig. 3. Diagram of the feedback loop.

The hybrid tracking controller is comprised of two parts. The first part is a saturation controller which reduces the translational error between the state and the reference. The second part uses the output of the saturation controller with the current state and a memory variable to drive the quadrotor rotation to the reference. The controller is modeled by the hybrid system $\mathcal{H}_c = (C_c, F_c, D_c, G_c)$ with state $x_c \in \mathbb{R}^3 \times \{-1, 1\} \times \mathbb{S}^3 =: \mathcal{X}_c$ and input reference trajectory r. Definitions of C_c, F_c, D_c, and G_c can be found in [12].

The combined quadrotor and tracking controller is modeled by the hybrid system $\mathcal{H}_a = (C_a, F_a, D_a, G_a)$ with state

$$x_a := (x_b, x_c) \in \mathcal{X}_b \times \mathcal{X}_c =: \mathcal{X}_a \tag{10}$$

for a reference trajectory $r : \mathbb{R}_{\geq 0} \to \mathbb{R}^{12} \times SO(3) \times \mathbb{R}^3$ with

$$r(t) := (p_r(t), p_r^{(1)}(t), p_r^{(2)}(t), p_r^{(3)}(t), R_r(t), \omega_r(t)), \tag{11}$$

satisfying the following assumption.

Assumption 1: Given the vehicle's maximum snap $\mathcal{M}_p > 0$ and angular acceleration $\mathcal{M}_\omega > 0$, a reference trajectory is a solution $t \mapsto r(t)$ to

$$\dot{r} \in F_d(r) := (p_r^{(1)}, p_r^{(2)}, p_r^{(3)}, \mathcal{M}_p \mathbb{B}, R_r S(\omega_r), \mathcal{M}_\omega \mathbb{B}), \tag{12}$$

such that $\operatorname{rge} r \in \Omega_r$ for some compact set $\Omega_r \subset \mathbb{R}^{12} \times SO(3) \times \mathbb{R}^3$, satisfying $e_3^\top R_r(t) e_3 \geq 0$ for each $t \geq 0$.

The hybrid system $\mathcal{H}_a = (C_a, F_a, D_a, G_a)$ modeling the closed-loop controller and quadrotor dynamics with state x_a and input reference trajectory $t \mapsto r(t)$ is given by

$$C_a := C_c, \qquad D_a := D_c \tag{13}$$

$$\begin{aligned} F_a(x_a, r(t)) := \big(&v_a, -R_a e_3 \frac{f}{m} + g e_3, R_a S(\omega_a), \\ &- J^{-1} S(\omega_a) J \omega_a + J^{-1} \mathcal{M}, F_c(x_a, r(t))\big) \end{aligned} \tag{14}$$

$$G_a(x_a, r(t)) := \big(x_b, G_c(x_a, r(t))\big). \tag{15}$$

with the vehicle inputs thrust magnitude and torque $\mathcal{M}$ assigned by the hybrid controller $\mathcal{H}_c$. For more details on Assumption 1 and $\mathcal{H}_a$, the reader is referred to [12].

V. Mathematical Problem Statement

With the constructions in Section IV, the problem outlined in **Problem** in Section III is formulated mathematically as follows.

Problem 1: Given a quadrotor $\mathcal{V}$ with reference tracking controller whose dynamics are captured by $\mathcal{H}_a$, initial quadrotor state ξ_a, obstacles $\mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_N$ whose dynamics are captured by $\mathcal{H}_{o_i}, i \in \{1, 2, \ldots, \mathbb{N}\}$, initial obstacle states ξ_{o_i}, $i \in \{1, 2, \ldots, N\}$, minimum safe quadrotor obstacle distance $k_u > 0$, target closed set $\mathbb{T} \subset \mathbb{R}^3$, and duration $\tau > 0$, compute a reference trajectory $t \mapsto r(t)$ given by the sequence of signals $\{r_k\}_{k \in \mathbb{N}_{\geq 1}}$ with elements $t \mapsto r_k(t)$ for each k, such that

1) $r_1(0) = \xi_a$
2) For each $k \in \mathbb{N}_{\geq 1}$, with x_a and x_{o_i} being the current quadrotor and obstacle states at time $t_c = (k-1)\tau$
   a) r_{k-1} has been executed by the quadrotor for τ time.
   b) $r_k(t_c) = r_{k-1}(t_c)$.
   c) $r_k : [t_c, t_c + \tau] \to \Omega_r$ satisfies Assumption 1.
   d) The maximal solution ϕ_k to $\mathcal{H}_a$ from x_a for input r_k evolves for at least τ seconds of flow and satisfies $\mathrm{dist}\left(\phi_k(t,j), \mathfrak{R}_{\mathcal{H}_{o_i}}(\tau, \infty, x_{o_i})\right) \geq k_u$ for all $(t,j) \in \mathrm{dom}\phi_k \cap ([t_c, t_c+\tau] \times \mathbb{N})$ so the distance between any possible obstacle trajectory and the quadrotor is never below the minimum safe distance.
3) The maximal solution ϕ to $\mathcal{H}_a$ from ξ_a for input r, with r being the concatenation of all reference trajectories in $\{r_k\}_{k \in \mathbb{N}}$, satisfies
   a) ϕ is complete.
   b) There exists a finite time t_f such that the position component of the trajectory $p_\phi(t,j) \in \mathbb{T}$ for all $(t,j) \in \mathrm{dom}\ \phi$ such that $t \geq t_f$.

Condition 2b ensures that the reference does not jump between r_{k-1} and r_k. Requirement 2d on each trajectory r_k ensures obstacle avoidance by enforcing a minimum distance obstacle quadrotor distance. Convergence to the target set within some finite time is included by requirement 3b.

VI. The Motion-Planning Algorithm

To solve the Problem 1, we propose a set-based feedback motion planning algorithm. The algorithm operates over multiple iterations, with each iteration extending the executed reference trajectory by $\tau_e \in \mathbb{R}_{>0}$ seconds. To prevent the planner from selecting a trajectory from which there is no safe extension due to an obstacle outside of the τ_e horizon, each iteration plans for $\tau_p \in \mathbb{R}_{>0}$ seconds, where $\tau_p > \tau_e$. The proposed algorithm is as follows:

Step 1: Compute quadrotor mobility set M for the closed-loop quadrotor model in (14)-(15).
Step 2: Compute the unsafe set U_i for each obstacle using $\mathcal{H}_{o_i}$in (6).
Step 3: Build the safe mobility set M_S by removing trajectories from the mobility set that violate safety constraints.
Step 4: Solve an optimization problem selecting the lowest cost reference trajectory from the safe mobility set.
Step 5: Execute τ_e seconds of the reference trajectory.
Step 6: Go to Step 1 to replan from the current state.

A. Supporting Sets and Constraints

The quadrotor mobility set is the set of all possible reference trajectory, quadrotor trajectory pairs (r, ϕ). To generate this set, all possible reference trajectories must be generated. Since this is difficult to implement for most systems in practice, we approximate the set using a motion primitive library Ω_p. The library is generated using the quadrotor dynamics in (8)-(9) by using inputs f and $\mathcal{M}$ satisfying

$$-\dot{\omega} e_3 \frac{f}{m} - 2\left(RS(\omega)e_3 \frac{\dot{f}}{m}\right) - Re_3 \frac{\ddot{f}}{m} \in \mathcal{M}_p\mathbb{B},\ \mathcal{M} \in \mathcal{M}_\omega \mathbb{B}$$

The restrictions on f and $\mathcal{M}$ are to ensure all trajectories within Ω_r satisfy the conditions of Assumption 1. These reference trajectories are used as inputs to simulations of the closed-loop quadrotor model (10) from the current quadrotor state, with the resulting trajectories building the solution set $\hat{\mathcal{S}}_{\mathcal{H}_a}(\xi_a, r)$. The quadrotor mobility set $M(\tau, \xi_a, \xi_r)$ containing all reference trajectory quadrotor trajectory pairs (r, ϕ) for the given initial vehicle state ξ_a, motion primitive library Ω_p, and initial reference state ξ_r over the time interval $[0, \tau]$, is defined as

$$\begin{aligned} M(\tau, \xi_a, \xi_r) := \big\{(r, \phi) : \phi(t,j) = \phi_a(t,j)\ \forall (t,j) \in \mathrm{dom}\phi \\ \cap ([0,\tau] \times \mathbb{N}), \phi_a \in \hat{\mathcal{S}}_{\mathcal{H}_a}(\xi_a, r), \forall r \in \Omega_p, r(0) = \xi_r\big\} \end{aligned} \tag{16}$$

where the set $\hat{\mathcal{S}}_{\mathcal{H}_a}(\xi_a, r)$ contains all solutions to the hybrid system $\mathcal{H}_a$ from initial state ξ_a for reference trajectory r.

The set of all possible obstacles states is the unsafe set. For each obstacle, the unsafe set is denoted as $U_i(\tau, \xi_{o_i})$ for the given initial state ξ_{o_i} over the hybrid time horizon $[0, \tau] \times \mathbb{N}$ and is defined as

$$U_i(\tau, \xi_{o_i}) := \mathfrak{R}_{\mathcal{H}_{o_i}}(\tau, \infty, \xi_{o_i}). \tag{17}$$

Then, given initial states for all obstacles $\xi_o := \{\xi_{o_1}, \xi_{o_2}, \ldots, \xi_{o_n}\}$, the unsafe set $U(\tau, \xi_o)$ is defined as

$$U(\tau, \xi_o) := \bigcup_{i=1}^{N} U_i(\tau, \xi_{o_i}). \tag{18}$$

The safe mobility set, $M_S(\tau, \xi_a, \xi_r, \xi_o)$ is constructed by removing any trajectories in $M(\tau, \xi_a, \xi_r)$ that violate the minimum safe distance to the unsafe set $U(\tau, \xi_o)$. It is defined as follows:

$$\begin{aligned} M_S(\tau, \xi_a, \xi_r, \xi_o) = \big\{(r, \phi_a) \in M(\tau, \xi_a, \xi_r) : \\ \mathrm{dist}\big(p_{\phi_a}(t_a, j_a), p_{\phi_o}(t_o, j_o)\big) \geq k_u \text{ for all } \phi_o \in U(\tau, \xi_o), \\ \text{for all } (t_a, j_a) \in \mathrm{dom}\ \phi_a, (t_o, j_o) \in \mathrm{dom}\ \phi_o\big\} \end{aligned} \tag{19}$$

where $p_{\phi_a}(t,j)$ is the positional component of $\phi_a(t,j)$ and $p_{\phi_o}(t,j)$ is the positional component of $\phi_o(t,j)$. The constant k_u is the constraint on the minimum allowed distance between the quadrotor position and the unsafe set for a trajectory to be considered safe.

B. Problem Reformulation and Algorithm

Using the constructions above, Problem 1 can be reformulated with each r_k in the reference trajectory r solving the following.

Problem 2: Given $\mathbb{T} \subset \mathbb{R}^3$, planning window $\tau_p > 0$, execution window $\tau_e \in (0, \tau_p]$, previous reference trajectory

$r_{prev} : [0, \tau_p] \to \Omega_r$, previous reference cost κ_{prev}, vehicle state $x_a \in \mathcal{X}_a$, and obstacle state $x_o \in \mathcal{X}_o$, generate a reference trajectory $\hat{r} \in \Omega_p$ with domain $[0, \tau_p]$ that solves

$$\begin{aligned}
&\underset{\hat{r}}{\text{minimize}}\ \kappa(\hat{r}, \phi, r_{prev}, \kappa_{prev}) \\
&\quad \text{subject to} \\
&\qquad \text{C1)}\ \hat{r}(0) = r_{prev}(\tau_e) \\
&\qquad \text{C2)}\ (\hat{r}, \phi) \in M_S(\tau_p, x_a, r_{prev}, x_o)
\end{aligned}$$

where ϕ is the solution to $\mathcal{H}_a$ with state x_a and reference trajectory $\hat{r}$ over $[0, \tau_p] \times \mathbb{N}$.

The cost functional κ is defined as

$$\begin{aligned}
\kappa(\hat{r}, \phi, r_{prev}, \kappa_{prev}) := \operatorname{dist}(p_\phi(\tau_p, j_\tau), \mathbb{T}) \\
+\kappa_p(\hat{r}, r_{prev}, \kappa_{prev})
\end{aligned} \tag{20}$$

$$\kappa_p(\hat{r}, r_{prev}, \kappa_{prev}) := \begin{cases} 0 & \text{if } \hat{r}(t) = r_{prev}(t + \tau_e) \\ & \quad \text{for all } t \in [0, \tau_p - \tau_e] \\ k_h \kappa_{prev} & \text{otherwise} \end{cases}$$

with j_τ denoting the number of jumps at time τ_e and $p_\phi(t, j)$ denoting the position of the trajectory ϕ at (t, j). The hysteresis term κ_p is added to prevent chattering by penalizing references which do not extend the previous reference. The strength of the penalty is adjusted by the hysteresis tuning constant $k_h \geq 0$.

Solving Problem 2 for each reference while the previous reference is followed by the quadrotor, results in Algorithm 1.

Algorithm 1: Motion planning algorithm with input $(\mathbb{T}, \tau_p, \tau_e, \xi_a, \xi_o, \mathcal{H}_a, \mathcal{H}_{o_i})$

$\kappa_0 \leftarrow 0$, $r_0 \leftarrow 0$, $x_a \leftarrow \xi_a$, $x_o \leftarrow \xi_o$
for $k = 1, 2, \ldots$ **do**
  $M \leftarrow \emptyset$, $U \leftarrow \emptyset$, $M_S \leftarrow \emptyset$
  for all $\hat{r} \in \Omega_p, \hat{r}(0) = r_{k-1}(\tau_e)$ **do**
    The solution ϕ_a of $\mathcal{H}_a$ is simulated from x_a for reference $\hat{r}$ for $[0, \tau_p]$ seconds of flow
    $M \leftarrow M \cup \{(\hat{r}, \phi_a)\}$
  end for
  for $i \in \{1, 2, \ldots, N\}$ **do**
    The solution ϕ_{o_i} of $\mathcal{H}_{o_i}$ is simulated from x_{o_i} for $[0, \tau_p]$ seconds of flow
    $U \leftarrow U \cup \{\phi_{o_i}(t, j) : \forall (t, j) \in \operatorname{dom} \phi_{o_i} \cup ([0, \tau_p] \times \mathbb{N})\}$
  end for
  for all $(\hat{r}, \phi_a) \in M$ **do**
    for all $\phi_o \in U$ **do**
      if $\operatorname{dist}(p_{\phi_a}(t, j_a), p_{\phi_o}(t, j_o)) \geq k_u$, for all $(t, j_a) \in \operatorname{dom} \phi_a$ and all $(t, j_o) \in \operatorname{dom} \phi_o$ **then**
        $M_S \leftarrow M_S \cup \{(\hat{r}, \phi_a)\}$
      end if
    end for
  end for
  (r_k, ϕ) is the solution to Problem 2 given $(\mathbb{T}, \tau_p, \tau_e, r_{k-1}, \kappa_{k-1}, x_a, x_o)$
  $\kappa_k \leftarrow \kappa(r_k, \phi, r_{k-1}, \kappa_{k-1})$
  Execute r_k for τ_e seconds.
  Update x_a and x_o
end for

VII. Simulation Results

Simulations[1] have been performed using both a double integrator point mass model and the combined quadrotor controller model. The simulation in Figure 4 uses the combined hybrid controller and quadrotor dynamical model. The set of possible reference trajectories is approximated using 1728 different reference trajectories, generated from 12 increments of each pitch torque, roll torque, and thrust as inputs to (8)-(9). All reference trajectories have no yaw torque. The planning time is $\tau_p = 0.3$ seconds and execution time of $\tau_e = 0.05$ seconds. The mass, inertial tensor, and single motor maximum thrust are from the system identification of the Crazyflie quadrotor in [13]. The controller constants are the same as in the simulations of [12], except $\alpha = 0.5, \delta = 0.5$, and $\beta = k_v/4$.

VIII. Experimental Results

A. Experimental Setup

The experiments[2] use a Windows 10 computer with a dual core 3.20GHz processor with 8GB of memory. Quadrotor and obstacle position data is captured using eight motion capture cameras running at 120Hz. Velocities are calculated using the difference in position and capture time between two frames. The experimental quadrotor was a Crazyflie 2.0 controlled over 2.4GHz radio though the Crazyflie client. The obstacles are 0.08m diameter wiffle balls wrapped in retro-reflective tape to allow tracking by the motion capture system. To recover the restitution constant and Σ factor, one obstacle was tossed 15 times with different spins from a height of approximately 1.5m. The restitution constant is calculated as the average change in vertical velocity from before to after the impact. The set Σ is constructed using the largest change in horizontal velocity from before to after the impact. The ball has a restitution constant of $\lambda = 0.65$ and possible velocity change from spin of $\Sigma = [-0.02, 0.02]$m/s. The obstacles are thrown toward the quadrotor from a horizontal distance of 0.15m to 0.7m and a height between 0.7m and 0.85m with initial horizontal velocities between 0m/s and 0.01m/s and vertical velocities between -0.2m/s and 0.6m/s. The motion planner and controller are implemented in Matlab. The experimental quadrotor controller are comprised of four PIDs which drove the quadrotor to the reference. The PIDs output desired thrust, yaw, pitch, and roll values were sent to the Crazyflie client over ZeroMQ at a rate of 50Hz. The motion planner is using a planning window of $\tau_p = 0.3$s, execution window of $\tau_e = 0.28$s, and a minimum safe distance of $k_u = 0.2$m.

[1] Code available at `https://github.com/HybridSystemsLab/CrazyFlieAvoidanceSimulation`

[2] Code available at `https://github.com/HybridSystemsLab/CrazyFlieAvoidanceExperiment`

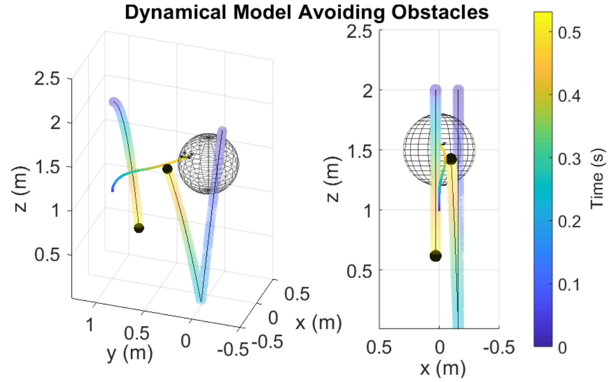

Fig. 4. Simulation avoiding two obstacles using a dynamical quadrotor model. The target set is the sphere with radius 0.3m centered at $(0, 0, 2)$. Initial quadrotor state is $(0, 1, 2, 0, -0.5, 0, I_{3\times3}, 0, 0, 0)$ with initial obstacle states $(-0.16, -0.2, 2, 0, 1, -8)$ and $(0.03, 1, 2, 0, -0.5, 0)$. Planning period was 0.3 seconds, execute period 0.05s, and obstacle radius of 0.05m.

B. Experimental Results

The quadrotor is able to reliably avoid the two obstacles and converge to the target set in each test. The motion planner has low computational delay with planner updates having a mean computation time of 83ms, median of 84ms, and a maximum time of 138ms. In Figure 6, the distance between the quadrotor and the obstacle of multiple experiments are shown, with the minimum being 0.16m. The violations of the minimum unsafe distance were due to the communication delay between the Crazyflie and the controller, which were not accounted for by the controller.

IX. CONCLUSION

In this paper, we presented a set-based feedback motion planner for a quadrotor avoiding bouncing ball-like dynamic obstacles with limited obstacle state information. Simulations using the hybrid controller and quadrotor model shows the ability of the algorithm to avoid multiple obstacles while converging to the target. Experimental results show feasibility of the proposed solution in real-world scenarios with limited computing power and real-time constraints.

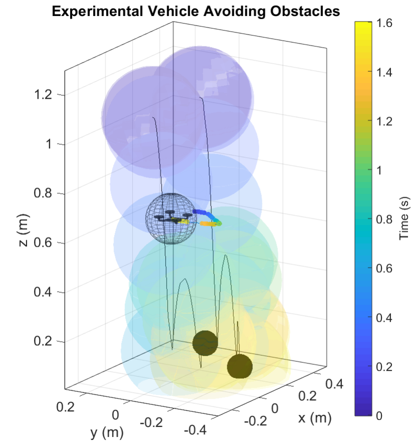

Fig. 5. Plot of the Crazyflie avoiding two obstacles. The Crazyflie has an initial position of $(0.29, -0.20, 0.68)$ and the obstacles had initial positions of $(0.44, -0.11, 1.08)$ and $(0.43, -0.10, 1.07)$. The light colored spheres depict the unsafe set, with the color of the sphere denoting the time as shown in the color bar. The unsafe set at time t is all spheres between t and $t + \tau_e$. The smaller colored spheres show the position of the Crazyflie over time. The wire frame sphere denotes the target set centered at $(0.18, -0.12, 0.67)$ with radius 0.1m

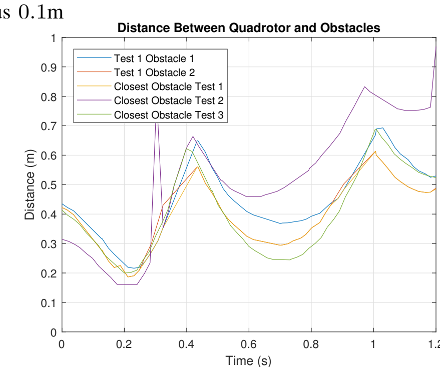

Fig. 6. Plot of distance between quadrotor and the closest of two obstacles over time for three experiment runs.

REFERENCES

[1] J. Rascón-Enríquez, L. A. García-Delgado, J. R. Noriega, A. García-Juárez, and E. S. Espinoza, "Tracking Control for Quad-Rotor Using Velocity Field and Obstacle Avoidance Based on Hydrodynamics," *Electronics*, vol. 9, no. 2, p. 233, Feb. 2020. [Online]. Available: https://www.mdpi.com/2079-9292/9/2/233

[2] C. Copot, A. Hernandez, T. Thoa Mac, and R. De Keyse, "Collision-free path planning in indoor environment using a quadrotor," in *2016 21st International Conference on Methods and Models in Automation and Robotics (MMAR)*, Aug. 2016, pp. 351–356.

[3] O. Andersson, O. Ljungqvist, M. Tiger, D. Axehill, and F. Heintz, "Receding-Horizon Lattice-Based Motion Planning with Dynamic Obstacle Avoidance," in *2018 IEEE Conference on Decision and Control (CDC)*, Dec. 2018, pp. 4467–4474, iSSN: 2576-2370.

[4] S. Liu, N. Atanasov, K. Mohta, and V. Kumar, "Search-based motion planning for quadrotors using linear quadratic minimum time control," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept. 2017, pp. 2872–2879, iSSN: 2153-0866.

[5] S. L. Herbert, M. Chen, S. Han, S. Bansal, J. F. Fisac, and C. J. Tomlin, "Fastrack: A modular framework for fast and guaranteed safe motion planning," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*. IEEE, 2017, pp. 1517–1522.

[6] R. E. Allen and M. Pavone, "A real-time framework for kinodynamic planning in dynamic environments with application to quadrotor obstacle avoidance," *Robotics and Autonomous Systems*, vol. 115, pp. 174–193, 2019.

[7] J. Crowley, Y. Zeleke, B. Altın, and R. G. Sanfelice, "Set-based predictive control for collision detection and evasion," in *2019 IEEE 15th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2019, pp. 541–546.

[8] B. Lindqvist, S. S. Mansouri, A.-a. Agha-mohammadi, and G. Nikolakopoulos, "Nonlinear MPC for Collision Avoidance and Control of UAVs With Dynamic Obstacles," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6001–6008, Oct. 2020.

[9] R. Goebel, R. G. Sanfelice, and A. R. Teel, *Hybrid Dynamical Systems: Modeling, Stability, and Robustness*. New Jersey: Princeton University Press, 2012.

[10] R. G. Sanfelice, *Hybrid Feedback Control*. New Jersey: Princeton University Press, 2021.

[11] M. Maghenem and R. G. Sanfelice, "Local lipschitzness of reachability maps for hybrid systems with applications to safety," in *Proceedings of the 23rd International Conference on Hybrid Systems: Computation and Control*, ser. HSCC '20. New York, NY, USA: Association for Computing Machinery, 2020. [Online]. Available: https://doi.org/10.1145/3365365.3382215

[12] P. Casau, R. G. Sanfelice, R. Cunha, D. Cabecinhas, and C. Silvestre, "Robust global trajectory tracking for a class of underactuated vehicles," *Automatica*, vol. 58, pp. 90–98, Aug. 2015. [Online]. Available: https://escholarship.org/uc/item/15z3c0cf

[13] J. Forster, "System Identification of the Crazyflie 2.0 Nano Quadrocopter," p. 147.