The Reachable Set of a Drone: Exploring the Position Isochrones for a Quadcopter

Mohammad M. Sultan, Daniel Biediger, Bernard Li, and Aaron T. Becker

***Abstract*— Quadcopters are increasingly popular for robotics applications. Being able to efficiently calculate the set of positions reachable by a quadcopter within a time budget enables collision avoidance and pursuit-evasion strategies.**

This paper examines the set of positions reachable by a quadcopter within a specified time limit using a simplified 2D model for quadcopter dynamics. This popular model is used to determine the set of candidate optimal control sequences to build the full 3D reachable set. We calculate the analytic equations that exactly bound the set of positions reachable in a given time horizon for all initial conditions. To further increase calculation speed, we use these equations to derive tight upper and lower spherical bounds on the reachable set.

I. Introduction

Quadcopters are popular due to their agility, ease of control, and low cost. All these are improving thanks to mass production and advances in sensing, batteries, and computation. The FAA estimates there are 1.25 million drones identified as model aircraft and predicts an annual growth of 13 percent [1]. Being able to derive the optimal control sequence to reach a given position (or alternately, to escape a current region) is useful for avoiding collisions with known obstacles [2]–[8], path planning [9]–[14], and evasion [15]–[18].

The goal of this paper is to calculate the *time-limited reachable set*, the set of all position states reachable by the quadcopter (or *drone*) at time t. This set is useful for many applications, including a drone seeking to escape a region in minimum time (Fig. 1), collision avoidance when planning trajectories near uncontrolled drones [19], and for drone counter-measures [20]. The time-limited reachable set is also useful for pursuit evasion tasks [21], [22]. Drones have been used for lighting effects at major events [23]. In future applications, drones could be used to simulate 3D animations and fireworks. In these applications, the maximum speed trajectory away from a point becomes significant.

Despite the usefulness of the reachable set, there remains a need for a closed-form solution. Current efforts have used numeric computation [24] or approximations [25]. One reason for this is that the dynamic model for a quadcopter requires 12 states, 4 inputs, and 8 parameters, as in [26] and [27]. To make the problem tractable, this paper uses the simplified 2D model of quadcopter dynamics proposed by Ritz, Hehn, and D'Andrea [4], [5]. We then integrate this model using the set of candidate optimal controllers and determine the extremal sets of inputs that construct the boundaries of this reachable set as a function of time.

This work was supported by the National Science Foundation under Grant No. [IIS-1553063] and [IIS-1849303].

Authors are with the Department of Electrical and Computer Engineering, University of Houston, Houston, TX 77204 USA {mmsultan,debiediger,bsli,atbecker}@uh.edu.

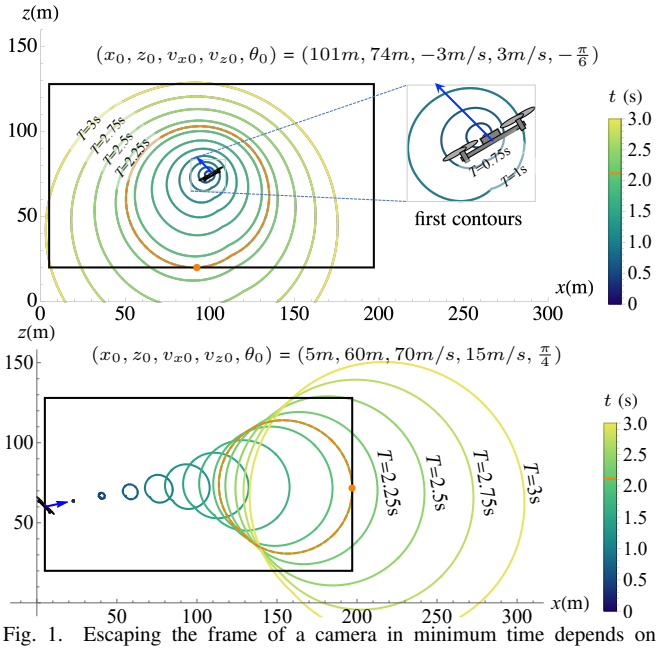


Fig. 1. Escaping the frame of a camera in minimum time depends on a drone's initial pose and velocity. The camera frame is drawn in black and contours of the drone's reachable set are drawn every 0.25 seconds. A slow-moving drone starting in the center of the camera frame can escape first by moving down (escaping in 2.12 s). A fast-moving drone entering the camera frame from the left can first escape at 2.11 s by moving at maximum velocity to the right.

A. 2D Simplified Model for a Quadcopter

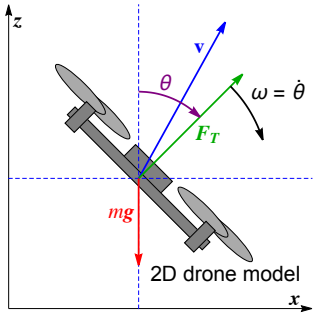
The system state is described by the quadcopter's horizontal position x, vertical position z, pitch angle θ and velocity $\mathbf{v}$ as shown schematically in Table I. The model assumes that the angular velocity can be controlled directly. This assumption is justified because modern quadcopters can reach high angular accelerations, but the on-board gyroscopic sensors limit the controllable angular velocity. The resulting system double integrates linear acceleration and single integrates angular velocity.

$$\begin{pmatrix} \ddot{x} \\ \ddot{z} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \frac{F_T}{m} \sin\theta \\ \frac{F_T}{m} \cos\theta - g \\ \omega \end{pmatrix}. \tag{1}$$

Here F_T is the magnitude of the net thrust from the thrusters, m the mass of the quadcopter, g is the gravitational acceleration, and ω the angular velocity. This model has been used

TABLE I. NOTATION

x	$\triangleq$	horizontal position
z	$\triangleq$	vertical position
θ	$\triangleq$	pitch angle
ω	$\triangleq$	angular velocity
t	$\triangleq$	time
g	$\triangleq$	gravity
m	$\triangleq$	mass of drone
u_T	$\triangleq$	thrust input
u_R	$\triangleq$	rotation input
$\mathbf{v}$	$\triangleq$	velocity, $\mathbf{v} = (v_x, v_z)$
F_T	$\triangleq$	thrust force
$\frac{F_T}{m}$	$\triangleq$	mass normalized thrust
t_T	$\triangleq$	thrust switching time
t_R	$\triangleq$	rotational switching time
$r_{\triangle}$	$\triangleq$	radius of inscribed circle
$r_{\bigcirc}$	$\triangleq$	radius of circumscribed circle
$\overline{*}$	$\triangleq$	maximum value of $*$
$\underline{*}$	$\triangleq$	minimum value of $*$
$\hat{\ }$	$\triangleq$	non-dimensionalized variable
$\dot{\ }$	$\triangleq$	first derivative (velocity)
$\ddot{\ }$	$\triangleq$	Second derivative (acceleration)



extensively, for instance in planning aggressive maneuvers through obstacles [11], designing robust controllers [28], learning optimal quadcopter controllers [6], and for grasping a moving target with a quadcopter [29].

This model is more complicated than the venerable Dubins car model [30] because it has an initial velocity, the effects of gravity, two control inputs, and doubly integrates the actuation. Details and a visual comparison between the Dubins car model and 2D quadrotor model are given in the appendix [31].

The system can then be non-dimensionalized using the following substitutions from [4]:

$$\hat{t} = \overline{\omega} t, \qquad \hat{x} = \frac{\overline{\omega}^2}{g} x, \qquad \hat{z} = \frac{\overline{\omega}^2}{g} z. \tag{2}$$

Here $\overline{\omega}$ is the maximum rotational rate. These substitutions enable the following dynamic model:

$$\begin{pmatrix} \dot{\hat{x}} \\ \ddot{\hat{x}} \\ \dot{\hat{z}} \\ \ddot{\hat{z}} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \dot{\hat{x}} \\ u_T \sin\theta \\ \dot{\hat{z}} \\ u_T \cos\theta - 1 \\ u_R \end{pmatrix}. \tag{3}$$

In this model, u_T is the thrust control input, u_R is the rotational control input, and u_T and u_R are the transformed control vector:

$$u_T = \frac{F_T}{mg}, \qquad u_R = \frac{\omega}{\bar{\omega}}. \tag{4}$$

The non-dimensionalized system describes the quadcopter dynamics with only two model parameters, ($\underline{u_T}$,$\overline{u_T}$), the maximum and minimum net thrust of the motors. u_T and u_R have the following ranges:

$$\underline{u_T} \leq u_T \leq \overline{u_T}, \tag{5}$$

$$-1 \leq u_R \leq 1. \tag{6}$$

The analysis in this paper is valid for any values of $(\bar{\omega}, \underline{u_T}, \overline{u_T})$. To generate plots requires picking values for these parameters. As in [4], for the plots in this paper we assume the rotation input is saturated to $\bar{\omega} = 10$ rad/s. We examine several sets of thrust parameters.

B. Candidate optimal controllers to bound the reachable set

The controls that bring the system to the boundary of the reachable set must satisfy Pontryagin's maximum principle (PMP). The candidate PMP controls to reach a desired orientation and position state were identified in [4]. They proved that the optimal thrust is a bang-bang control input where u_T is always either minimized or maximized: $u_T = \{\underline{u_T}, \overline{u_T}\}$. The rotational input was identified to always be either $-1 or +1$, unless it is in a singular arc.

In this work we are only interested in the reachable positions, not orientations. Our candidate controllers are therefore a subset of those from [4]. We use a process similar to that used to calculate the sets reachable by a Dubins car in [32]–[35] and a Reed-Shepp car in [36]. The optimal paths are specified by letters, with C standing for a path segment at the maximum turning rate and S for a straight path segment. Although moving a Dubins car from one (x, y, θ) configuration to another may require at most three path segments, reaching the position isochrones never requires more than two path segments. This is because any position that can be accessed by a CSC path can be accessed in less time by a CS path that first turns the robot to point to the goal and then moves straight to the goal. Similarly, to move a drone outwards in minimum time, the drone should be turned (at the maximum rotation rate) such that its velocity vector points toward the target, and then fly with no rotation towards the target at the maximum velocity. For small time budgets, the reachable set is also bounded by first rotating in one direction and then the other while moving at minimum velocity.

Like the Dubins car, the drone can follow a CS path by first spinning and then going straight. This is accomplished by switching u_R from 1 to 0 at some time $\hat{t}_R$. Unlike the Dubins car, we can also control thrust. Because all acceleration inputs are double integrated, it is sometimes optimal to move with minimum thrust until some time $\hat{t}_T$, when the drone is pointing more in the direction it needs to move, and then switch to maximum thrust. These will be clarified in the following section.

II. GENERATING THE REACHABLE SET

The reachable set is generated by a subset of the candidate optimal controllers, and requires only two switching times. This section defines these switching times, and explains how to compute the reachable set with arbitrary initial conditions, extends the analysis to 3D, and provides equations for circles that upper and lower bound the reachable set (spheres in 3D).

A. Two switching times form the reachable set

Trajectories that span the range of reachable positions can be parameterized using just two switching times: the thrust

switching time $\hat{t}_T$ and the rotational switching time $\hat{t}_R$. For the thrust switching time, if $\hat{t} < \hat{t}_T$ then the thrust will be $\underline{u_T}$ and otherwise the drone will fly with thrust $\overline{u_T}$. For the rotational switching time there are two cases: curve straight (CS) and curve curve (CC). If the trajectory is in CS then if $\hat{t} < \hat{t}_R$ the drone will rotate clockwise and otherwise the drone will move straight. If the trajectory is in CC then if $\hat{t} < \hat{t}_R$ the drone will rotate clockwise and otherwise will rotate counter clockwise.

$$\theta_{CS}(\hat{t}, \hat{t}_R) = \min(\hat{t}, \hat{t}_R) \tag{7}$$

$$\theta_{CC}(\hat{t}, \hat{t}_R) = \min(\hat{t}, \hat{t}_R - \hat{t}) \tag{8}$$

$$u_T(\hat{t}, \hat{t}_T) = \begin{cases} \underline{u_T} & \hat{t} \le \hat{t}_T \\ \overline{u_T} & \hat{t} > \hat{t}_T \end{cases} \tag{9}$$

$$\ddot{\hat{x}} = u_T(\hat{t}, \hat{t}_T) \sin\left(\theta(\hat{t}, \hat{t}_R)\right) \tag{10}$$

$$\ddot{\hat{z}} = u_T(\hat{t}, \hat{t}_T) \cos\left(\theta(\hat{t}, \hat{t}_R)\right) - 1 \tag{11}$$

The equations for $\hat{x}$ and $\hat{z}$ are obtained by double integrating $\ddot{\hat{x}}$ and $\ddot{\hat{z}}$. Because these useful equations are long, they are listed in the appendix [31] and Fig. 11. Animations and interactive demonstrations aide understanding both the configuration space and the reachable set. See `https://youtu.be/FidiVw0_yfs` and [37] for these visualizations.

B. Reachable set from zero initial conditions

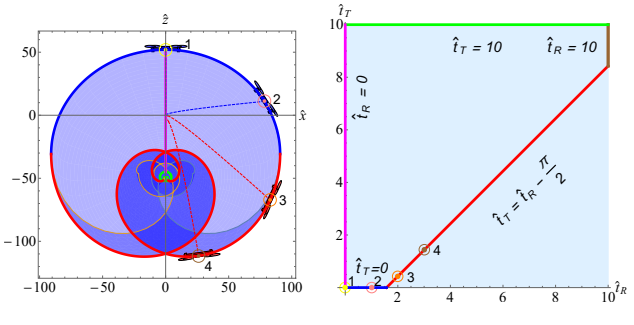


Fig. 2. The reachable set (left) and the configuration space (right) at $\hat{t} = 10$. The two switching times $(\hat{t}_R, \hat{t}_T)$ form a configuration space. While any configuration space pair is feasible and map to $(\hat{x}, \hat{z})$ coordinates, the colored lines in the configuration space map to the maximal bounds of the reachable set. Circled dots in the configuration space correspond with the four drones and their paths in the workspace. Knowing that $(\underline{u_T}, \overline{u_T}) = (0.102, 2.04)$

The plots in this subsection assume that the quadcopter starts at zero state in an obstacle-free workspace. Converting these plots from the non-dimensionalized coordinate space is a simple scaling derived by inverting (2).

For a drone that starts with initial conditions set to zero, Fig. 2 shows two plots. The left plot of Fig. 2 is the reachable set of $(\hat{x}, \hat{z})$ positions. The colored boundary lines match those in the configuration space plot at the right of Fig. 2. The configuration space plots $\hat{t}_R$ vs. $\hat{t}_T$, with $\hat{t}_R$ on the x-axis and $\hat{t}_T$ on the y-axis. For CS the blue line represents candidate control inputs that use full thrust all the time, which occurs when $\hat{t}_T = 0$. The magenta line represents when the drone moves straight always $(\hat{t}_R = 0)$. The green line represents

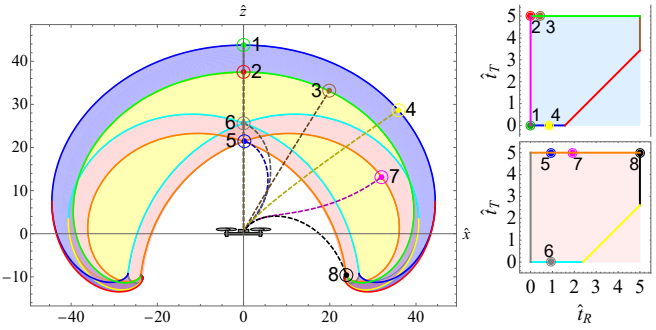


Fig. 3. The CC region must be computed if the $\underline{u_T} > 0$. This plot shows the reachable set when $\hat{t} = 5$, $\underline{u_T} = 4$ and $\overline{u_T} = 4.5$. The blue regions are accessed by CS trajectories, the pink regions by CC trajectories, and the yellow by controllers that are not optimal candidates.

control sequences that always move with minimum thrust $(\hat{t}_T = \hat{t})$. The brown line are inputs that always rotate $(\hat{t}_R = \hat{t})$, and red when $\hat{t}_T = \hat{t}_R - \pi/2$.

If $\underline{u_T} = 0$, then CS covers the entire reachable set. If $\underline{u_T} > 0$, then CS upper bounds the reachable set and CC under bounds the set. This is shown in Fig. 3, which also shows the CC and CS configuration space.

In the CC configuration space plots shown in Fig. 3, the cyan, black, gray and orange lines correspond to the blue, brown, magenta and green lines in the CS region. However, the yellow line occurs when $\hat{t}_T \approx \frac{\pi}{36}(23 + 1.5(\hat{t} - 2))$.

Isochrones at four different times are shown in Fig. 4.

C. Reachable set from non-zero initial conditions

The reachable set for a drone has the same shape for all initial velocities, positions, and orientations. However, initial velocities and positions translate the set and initial orientations rotate the set, as shown in Fig. 1.

If the drone starts at $(\hat{x}_0, \hat{z}_0)$ with initial velocity $(v_{\hat{x}_0}, v_{\hat{z}_0})$, the final reachable set at time $\hat{t}$ is translated by

$$(\hat{x}_0 + \hat{t} v_{\hat{x}_0}, \hat{z}_0 + \hat{t} v_{\hat{z}_0}), \tag{12}$$

which is computed by integrating the initial conditions. For non-zero initial orientations, rotate the reachable set

$$\text{about the point } (0, -\hat{t}^2/2) \text{ by } \theta_0 \tag{13}$$

before applying the translation (12). This point is the final position of an unpowered drone accelerated by gravity in the non-dimensionalized system.

D. 2D to 3D

To generate the 3D reachable workspace, we rotate the 2D reachable set about the line where $\hat{t}_R = 0$. This line is drawn in magenta in Fig. 5. This simplification assumes that the drone has the same maximum rotation rate in any combination of roll and pitch. This assumption is reasonable because the maximum rotation rate is limited by the sensor data rate [4], not the torque generated by the motors. So even though an $\times$-shaped configuration of thrusters can roll and pitch more quickly than a $+$-shaped configuration, these do not change the reachable set.

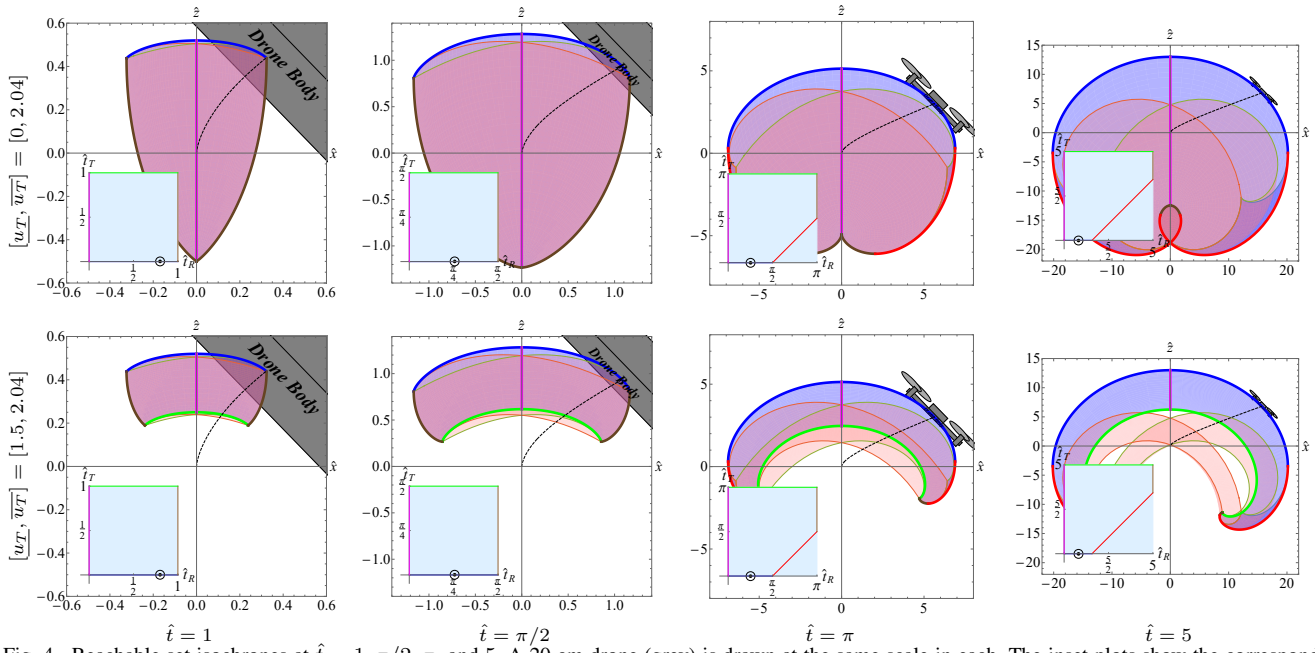


Fig. 4. Reachable set isochrones at $\hat{t} = 1, \pi/2, \pi$, and 5. A 20 cm drone (grey) is drawn at the same scale in each. The inset plots show the corresponding $(\hat{t}_R, \hat{t}_T)$ configuration-space diagrams. The reachable set for CS inputs is drawn in blue, and for CC inputs in pink. The top row uses $\underline{u_T} = 0$ and the bottom row uses $\underline{u_T} = 1.5$. Both use $\overline{u_T} = 2.04$. Dashed lines are the CS path from the origin to the location at $(\hat{t}_R, \hat{t}_T)$

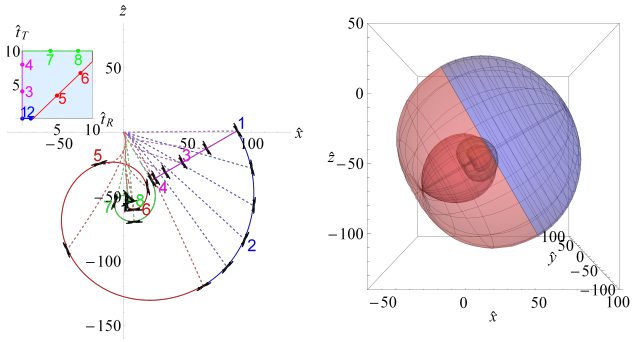


Fig. 5. The 3D reachable set is formed by rotating the 2D plot about the axis where $\hat{t}_R = 0$ (the magenta line). The blue manifold represents the limit for a drone with maximum thrust at all times, and the red manifold shows the set reachable for a drone starting with minimum thrust until $\hat{t}_T$ and maximum rotation until $\hat{t}_R$, with maximum thrust and straight flying otherwise. Shown with $(\underline{u_T}, \overline{u_T}) = (0, 2.04)$, $\hat{t} = 10$ and with $\theta_0 = \frac{\pi}{3}$.

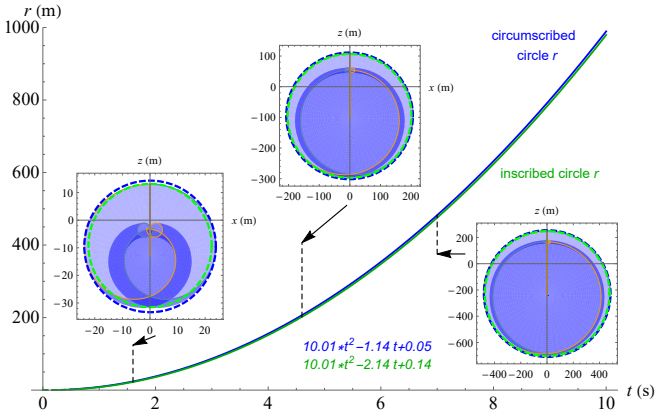


Fig. 6. Dimensionalized maximum inscribed and minimum circumscribed radii $r_{\triangle}(t)$ and $r_{\bigcirc}(t)$ as a function of time for curve-straight CS inputs. The fit is generated analytically using (14), (15), (16), and (17).

E. Reachable set inscribed and circumscribed circles

To simplify analysis, we can upper and lower bound the reachable set by circles in 2D (or spheres in 3D). We call these the inscribed and circumscribed circles with symbols $\triangle$ and $\bigcirc$. Dimensionalised and non-dimensionalized inscribed and circumscribed circle radii as a function of time are plotted in Fig. 6 and Fig. 7, along with the bounding circles in three times, where $(\underline{u_T}, \overline{u_T}) = (0, 2.04)$ for both figures. These figures and analysis are given for a drone with an zero initial configuration, but nonzero initial configurations rotate these circles by (13) and then translate them by (12).

The maximum inscribed circle for time $\hat{t} \geq \frac{2}{3}\pi$ is defined by two points: the maximum and the minimum distance travelled along the initial orientation, θ_0. The maximum distance is $\hat{z}(\hat{t}, 0, 0)$, with maximum thrust and no turning. The minimum distance is achieved by the red boundary control inputs from Fig. 2 that rotate to invert the drone and switch to maximum thrust to intercept the z axis. We define the rotation switching time as $\hat{t}_{R\triangle}$ and the thrust switching time as $\hat{t}_{T\triangle}$. With these, the inscribed circle center is at $\big(0, \hat{z}_{\triangle}(\hat{t})\big)$ and has radius $\hat{r}_{\triangle}(\hat{t})$, as given by (14) and (15):

$$\hat{z}_{\triangle}(\hat{t}) = \frac{\hat{z}(\hat{t}, 0, 0) - \hat{z}(\hat{t}, \hat{t}_{R\triangle}, \hat{t}_{T\triangle})}{2}, \tag{14}$$

$$\hat{r}_{\triangle}(\hat{t}) = \frac{\hat{z}(\hat{t}, 0, 0) + \hat{z}(\hat{t}, \hat{t}_{R\triangle}, \hat{t}_{T\triangle})}{2}. \tag{15}$$

For time $\hat{t} < \frac{2}{3}\pi$, the maximum circle is defined by multiple points, and is solved numerically.

The minimum circumscribed circle has a simple analytic solution for $\hat{t} \geq \frac{3}{2}\pi$ with center at $\big(0, \hat{z}_{\bigcirc}(\hat{t})\big)$ and has radius

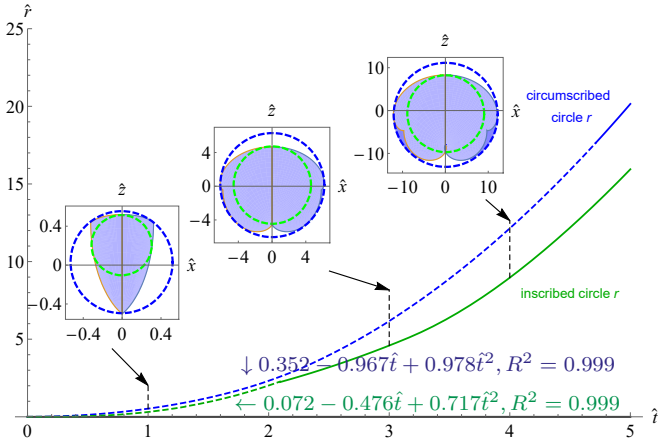


Fig. 7. Non-dimensionalized maximum inscribed and minimum circumscribed radii $\hat{r}_{\triangle}(\hat{t})$ and $\hat{r}_{\bigcirc}(\hat{t})$ as a function of $\hat{t}$ for curve-straight CS inputs. Dashed lines are numerical solutions while the solid lines are analytical solutions using (15) and (17)

$\hat{r}_{\bigcirc}(\hat{t})$:

$$\begin{aligned}\hat{z}_{\bigcirc}(\hat{t}) &= \hat{z}(\hat{t}, \frac{\pi}{2}, 0) \\ &= -\frac{\hat{t}^2}{2} + \overline{u_T}(1 - \frac{\pi}{2} + \hat{t}), \qquad (16) \\ \hat{r}_{\bigcirc}(\hat{t}) &= \hat{x}(\hat{t}, \frac{\pi}{2}, 0) \\ &= \frac{\overline{u_T}}{8}((\pi - 2\hat{t})^2 + 8(\hat{t} - 1)). \qquad (17)\end{aligned}$$

For time $\hat{t} < \frac{3}{2}\pi$, the minimum circle is defined by either two or three points, and is solved numerically. These circles can be converted to spheres that bound the 3D set following the procedure in Section II-D.

III. Applications

This section presents three representative applications for using a drone's reachable set.

A. Escaping a bounded region

If a drone is attempting to remain unobserved and is caught in the frame of a camera, it must determine which direction to move in order to escape the camera frame in the minimum time. The solution is to compute the reachable set as a function of time, and then calculate the first time that the reachable set intersects the edge of the camera frame. This procedure is illustrated in Fig. 1. Escaping the frame of a camera in minimum time depends on the drone's initial pose and velocity, as calculated in Section II-C. For each example, the camera frame corresponds to an HD camera that observes a region 192 m×108 m. This camera frame is drawn in black and contours of the drone's reachable sets are drawn every 0.25 seconds. A slow-moving drone starting in the center of the camera frame can escape first by moving down (escaping in 2.12 s). A fast-moving drone entering the camera frame from the left can first escape at 2.11 s by moving at maximum velocity to the right. The escape times are computed using a binary search over the reachable set equations in the Appendix [31] using Mathematica. With a

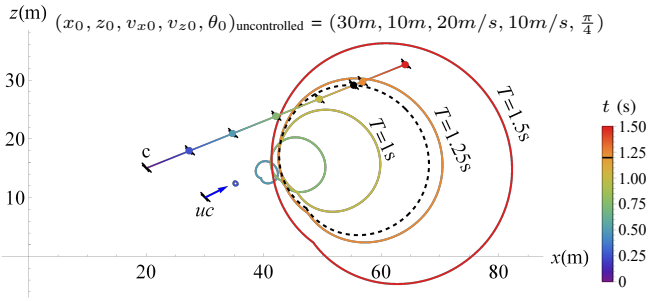


Fig. 8. Contours of the reachable set enable worst-case analysis for collision avoidance. In this image, the controlled drone has planned a linear flight trajectory (shown in blue). The uncontrolled drone's reachable set is plotted every 0.25 seconds. The controlled drone's position is disjoint from the reachable set until $t = 1.198$ s, which is the first possible collision, and is illustrated by a black point.

3.1 GHz Intel Core i7 laptop and a resolution of 0.001 s this calculation required 0.42 s and 0.44 s for the top and bottom drones respectively, but only 0.048 s and 0.047 s using the circumscribed circle fit (17).

B. Collision Avoidance

The reachable set has applications for many pursuit-evasion problems. A common problem arises in air traffic control: given a drone with a known trajectory (the *controlled* drone), and other drones with known initial position and velocity, but unknown future trajectories (the *uncontrolled* drones), when is the first possible collision? Using the full reachable set enables anticipating all scenarios. Figure 8 shows a representative example. The controlled drone coordinates are $(20, 15)$, and it is following a linear trajectory $(x, z) = (2.94t + 20, 1.18t + 15)$. The uncontrolled drone starts at $(30, 10)$ with velocity $(v_x, v_z) = (20, 10)$. The controlled drone's flight path is not guaranteed to be clear. The controlled drone's position is disjoint from the reachable set until $t = 1.198$ s, which is the first possible collision, generated by the uncontrolled drone using $\hat{t}_T = 0$, $\hat{t}_R = 0.74$. This type of worst case planning is becoming more relevant as learning-based automatic controllers have enabled flight acrobatics that exceed the capabilities of human pilots [38].

C. Drone countermeasures

The circumscribed set can be used for drone countermeasures if the effect of the counter measure has a circular cross section. Examples include the anti-drone net-deploying system in [20], or the use of *flak*, explosive anti-aircraft shells [39], [40]. To compute if a net-deploying system can guarantee capture with a net of radius r_{net}, and a delivery time t_{net}, calculate the circumscribed circle for the drone. If $r_{\bigcirc}(t_{\text{net}}) < r_{\text{net}}$, the drone can be captured and the net should be launched at the (x, z) center of the reachable set, as specified in Section II-E.

IV. Experiment

Our experiment replicated, with a physical drone, the optimal control sequences that generate the reachable set isochrones shown in Fig. 2. We selected an Emax TinyHawk

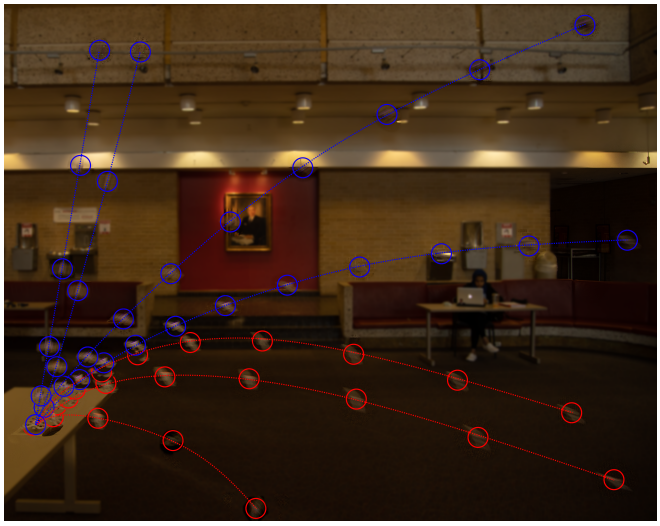


Fig. 9. Composite image with TinyHawk drone in *acro* mode (red) and *horizon* mode (blue). The composite image shows that the drone in both modes matches the predicted isochrone produced by Fig. 2.

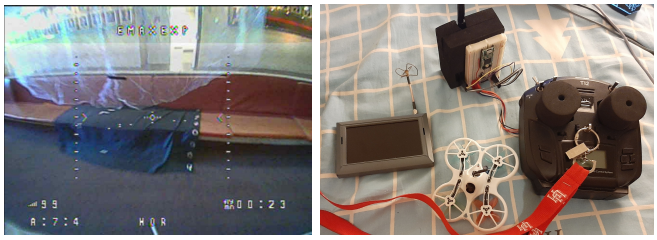


Fig. 10. (Left) Video from the pilot's FPV goggles while performing path 2 in Fig. 2. (Right) Photo of the experiment setup, including the Jumper T12 flight controller, the modified TinyHawk drone, the modified transmitter setup, and a FPV video screen.

1.1 drone, a model commonly used in drone racing. We connected a Jumper T-12 radio controller to a Arduino-compatible Teensy 3.2 micro-controller. The Teensy then mirrors the pulse-position modulation signal created by the pilot to a commercial RC transmitter module. The transmitter controls the TinyHawk which runs *BetaFlight*, a popular operating system for quadcopters. Figure 10 shows the hardware setup and a frame from the pilots FPV feed. The drone was placed in a zero-initial condition on a table to provide a safety offset from the ground. A stationary camera mounted to a tripod was setup to photograph the drone flights for approximately $t = 1$ second. The camera captured still frames of the flight at eight frames per second.

A trained pilot then replicated optimal trajectories similar to the four paths generated in Fig. 2 by applying an ω change to meet the desired angle. Betaflight provides two flight modes used for the test [41], *acro* which provides no stabilization, and *horizon* attempts to keep the drone at level position, trying to keep θ close to 0. Using both flight modes, the pilot was able to replicate the movement of optimal trajectories. These trajectories are drawn in Fig. 9, showing that this reachable set is a good indicator of possible movements a drone can perform in the real world. We recorded IMU data through a attached data recorder, as well as other information including flight footage, which we plan to use in future experiments and calculations.

V. Conclusion

This paper presented analytical equations for calculating the set of positions reachable by a quadcopter as a function of time, using a common non-dimensionalized quadcopter dynamics model. Solutions were provided for all initial conditions. Equations for the curve that bounds this set were given. Also provided were equations for the inscribed and the circumscribed circle of this set.

Future work will include the addition of terminal velocity in the model, and also the examination of the 3D (x, y, θ) reachable sets and the bounding curves for this expanded set. There are also opportunities to extend the experimental results with physical quadcopters, including exciting experiments of inverted flight under maximum thrust.

Appendix: 2D Drone reachable set equations: $\hat{x}(\hat{t}, \hat{t}_R, \hat{t}_T)$, $\hat{z}(\hat{t}, \hat{t}_R, \hat{t}_T)$.

For zero initial conditions, curve straight (CS) case when $\hat{t}_R \leq \hat{t}_T$:

$$\hat{x} = \begin{cases} \frac{u_T}{2}(\hat{t} - \sin(\hat{t})) & \hat{t} \leq \hat{t}_R \\ \frac{t u_T}{2} + (\hat{t}_R - \hat{t})\underline{u_T}\cos(\hat{t}_R) + \frac{1}{2}((\hat{t} - \hat{t}_R)^2 - 2)\underline{u_T}\sin(\hat{t}_R) & \hat{t}_R < \hat{t} \leq \hat{t}_T \\ \frac{t u_T}{2} + (\hat{t}_R - \hat{t})\underline{u_T}\cos(\hat{t}_R) + \frac{1}{2}(\underline{u_T}(\hat{t}_R^2 + 2\hat{t}\hat{t}_T - \hat{t}_T^2 - 2 - 2\hat{t}\hat{t}_R) + \overline{u_T}(\hat{t} - \hat{t}_T)^2)\sin(\hat{t}_R) & \hat{t}_T < \hat{t} \end{cases} \quad (2)$$

$$\hat{z} = \begin{cases} \frac{-\hat{t}^2}{2} + \underline{u_T} - \underline{u_T}\cos(\hat{t}) & \hat{t} \leq \hat{t}_R \\ \frac{-\hat{t}^2}{2} + \underline{u_T} + \frac{1}{2}(-2 + (\hat{t} - \hat{t}_R)^2)\underline{u_T}\cos(\hat{t}_R) + (\hat{t} - \hat{t}_R)\underline{u_T}\sin(\hat{t}_R) & \hat{t}_R < \hat{t} \leq \hat{t}_T \\ \frac{-\hat{t}^2}{2} + \underline{u_T} + \frac{1}{2}(\underline{u_T}(-2 - 2\hat{t}\hat{t}_R + \hat{t}_R^2 + 2\hat{t}\hat{t}_T - \hat{t}_T^2) + \overline{u_T}(\hat{t} - \hat{t}_T)^2)\cos(\hat{t}_R) + (\hat{t} - \hat{t}_R)\underline{u_T}\sin(\hat{t}_R) & \hat{t} \geq \hat{t}_T \end{cases} \quad (3)$$

For zero initial conditions, curve straight (CS) case when $\hat{t}_R > \hat{t}_T$:

$$\hat{x} = \begin{cases} \frac{u_T}{2}(\hat{t} - \sin(\hat{t})) & \hat{t} \leq \hat{t}_T \\ \frac{t u_T}{2} - (\hat{t} - \hat{t}_T)(\underline{u_T} - \overline{u_T})\cos(\hat{t}_T) - \overline{u_T}\sin(\hat{t}) + (\overline{u_T} - \underline{u_T})\sin(\hat{t}_T) & \hat{t}_T < \hat{t} \leq \hat{t}_R \\ \frac{t u_T}{2} + (-\hat{t} + \hat{t}_R)\overline{u_T}\cos(\hat{t}_R) + \frac{1}{2}(\overline{u_T}(-2 + (\hat{t} - \hat{t}_R)^2)\sin(\hat{t}_R) - (\underline{u_T} - \overline{u_T})((\hat{t} - \hat{t}_T)\cos(\hat{t}_T) + \sin(\hat{t}_T)) & \hat{t} > \hat{t}_R \end{cases} \quad (4)$$

$$\hat{z} = \begin{cases} \frac{-\hat{t}^2}{2} + \underline{u_T} - \underline{u_T}\cos(\hat{t}) & \hat{t} \leq \hat{t}_T \\ \frac{-\hat{t}^2}{2} + \underline{u_T} - \overline{u_T}\cos(\hat{t}) - (\underline{u_T} - \overline{u_T})(\cos(\hat{t}_T) + (-\hat{t} + \hat{t}_T)\sin(\hat{t}_T)) & \hat{t}_T < \hat{t} \leq \hat{t}_R \\ \frac{-\hat{t}^2}{2} + \underline{u_T} + \frac{1}{2}(-2 + (\hat{t} - \hat{t}_R)^2)\overline{u_T}\cos(\hat{t}_R) + (\hat{t} - \hat{t}_R)\overline{u_T}\sin(\hat{t}_R) - (\underline{u_T} - \overline{u_T})\left(\cos(\hat{t}_T) + (\hat{t}_T - \hat{t})\sin(\hat{t}_T)\right) & \hat{t} \geq \hat{t}_R \end{cases} \quad (5)$$

For non-zero initial conditions, curve straight (CS) case when $\hat{t}_R \leq \hat{t}_T$:

$$\hat{x} = \begin{cases} \frac{u_T}{2}(\hat{t}\cos(\theta_0) + \sin(\theta_0) - \sin(\hat{t} + \theta_0)) + v_{z_0}\hat{t} & \hat{t} \leq \hat{t}_R \\ \frac{u_T}{2}(\hat{t}\cos(\theta_0) + (-\hat{t} + \hat{t}_R)\cos(\hat{t}_R + \theta_0) + \sin(\theta_0)) + \frac{1}{2}(-2 + (\hat{t} - \hat{t}_R)^2)\underline{u_T}\sin(\hat{t}_R + \theta_0)) + v_{x_0}\hat{t} & \hat{t}_R < \hat{t} \leq \hat{t}_T \\ \frac{u_T}{2}(\hat{t}\cos(\theta_0) + (-\hat{t} + \hat{t}_R)\cos(\hat{t}_R + \theta_0) + \sin(\theta_0)) + \frac{1}{2}(\underline{u_T}(-2 - 2\hat{t}\hat{t}_R + \hat{t}_R^2 + 2\hat{t}\hat{t}_T - \hat{t}_T^2) + \overline{u_T}(\hat{t} - \hat{t}_T)^2)\sin(\hat{t}_R + \theta_0) + v_{z_0}\hat{t} & \hat{t} > \hat{t}_T \end{cases} \quad (6)$$

$$\hat{z} = \begin{cases} \underline{u_T}\cos(\theta_0) - \frac{\hat{t}}{2}(\hat{t} + 2\underline{u_T}\sin(\theta_0)) - \underline{u_T}\cos(\hat{t} + \theta_0) + v_{z_0}\hat{t} & \hat{t} \leq \hat{t}_R \\ \underline{u_T}\cos(\theta_0) - \frac{\hat{t}}{2}(\hat{t} + 2\underline{u_T}\sin(\theta_0)) + \frac{1}{2}(-2 + (\hat{t} - \hat{t}_R)^2)\underline{u_T}\cos(\theta_0 + \hat{t}_R) + (\hat{t} - \hat{t}_R)\underline{u_T}\sin(\theta_0 + \hat{t}_R) + v_{z_0}\hat{t} & \hat{t}_R < \hat{t} \leq \hat{t}_T \\ \underline{u_T}\cos(\theta_0) + \frac{1}{2}(\underline{u_T}(-2 - 2\hat{t}\hat{t}_R + \hat{t}_R^2 + 2\hat{t}\hat{t}_T - \hat{t}_T^2) + \overline{u_T}(\hat{t} - \hat{t}_T)^2)\cos(\theta_0 + \hat{t}_R) + \\ (\hat{t} - \hat{t}_R)\underline{u_T}\sin(\theta_0 + \hat{t}_R) - \frac{\hat{t}}{2}(\hat{t} + 2(\underline{u_T}\sin(\theta_0 + \hat{t}_R)) + v_{z_0}\hat{t} & \hat{t} \geq \hat{t}_T \end{cases} \quad (7)$$

For non-zero initial conditions, curve straight (CS) case when $\hat{t}_R > \hat{t}_T$:

$$\hat{x} = \begin{cases} \frac{u_T}{2}(\hat{t}\cos(\theta_0) + \sin(\theta_0) - \sin(\hat{t} + \theta_0)) + v_{z_0}\hat{t} & \hat{t} \leq \hat{t}_T \\ \frac{u_T}{2}\hat{t}\cos(\theta_0) - (\underline{u_T} - \overline{u_T})\left((\hat{t} - \hat{t}_T)\cos(\theta_0 + \hat{t}_T) + \sin(\hat{t}_T + \theta_0)\right) + \underline{u_T}\sin(\theta_0) - \overline{u_T}\sin(\hat{t} + \theta_0) + v_{z_0}\hat{t} & \hat{t}_T < \hat{t} \leq \hat{t}_R \\ \frac{u_T}{2}\hat{t}\cos(\theta_0) + \underline{u_T}\sin(\theta_0) + (-\hat{t} + \hat{t}_R)\overline{u_T}\cos(\hat{t}_R + \theta_0) + \frac{1}{2}(\overline{u_T}(-2 + (\hat{t} - \hat{t}_R)^2)\sin(\hat{t}_R + \theta_0) \\ -(\underline{u_T} - \overline{u_T})((\hat{t} - \hat{t}_T)\cos(\hat{t}_T + \theta_0) + \sin(\hat{t}_T + \theta_0)) + v_{z_0}\hat{t} & \hat{t} > \hat{t}_R \end{cases} \quad (8)$$

$$\hat{z} = \begin{cases} \underline{u_T}\cos(\theta_0) - \frac{\hat{t}}{2}(\hat{t} + 2\underline{u_T}\sin(\theta_0)) - \underline{u_T}\cos(\hat{t} + \theta_0) + v_{z_0}\hat{t} & \hat{t} \leq \hat{t}_T \\ \underline{u_T}\cos(\theta_0) - \frac{\hat{t}^2}{2} - \overline{u_T}\cos(\hat{t} + \theta_0) - (\underline{u_T} - \overline{u_T})(\cos(\hat{t}_T + \theta_0) + (\hat{t} - \hat{t}_T)\sin(\hat{t}_T + \theta_0)) \\ -\hat{t}\underline{u_T}\sin(\theta_0) + v_{z_0}\hat{t} & \hat{t}_T < \hat{t} \leq \hat{t}_R \\ \underline{u_T}\cos(\theta_0) - \frac{\hat{t}^2}{2} + \frac{1}{2}(-2 + (\hat{t} - \hat{t}_R)^2)\overline{u_T}\cos(\hat{t}_R + \theta_0) + (\hat{t} - \hat{t}_R)\overline{u_T}\sin(\hat{t}_R + \theta_0) \\ -(\underline{u_T} - \overline{u_T})(\cos(\hat{t}_T + \theta_0) + (\hat{t} - \hat{t}_T)\sin(\hat{t}_T + \theta_0)) - \hat{t}\underline{u_T}\sin(\theta_0) + v_{z_0}\hat{t} & \hat{t} \geq \hat{t}_R \end{cases} \quad (9)$$

For zero initial conditions, curve curve (CC) case when $\hat{t}_R \leq \hat{t}_T$:

$$\hat{x} = \begin{cases} \frac{u_T}{2}(\hat{t} - \sin(\hat{t})) & \hat{t} \leq \hat{t}_R \\ \frac{u_T}{2}(\hat{t} + 2(-\hat{t} + \hat{t}_R)\cos(\hat{t}_R) - \sin(\hat{t}) + \sin(\hat{t} - 2\hat{t}_R) + \sin(\hat{t}_R)) & \hat{t}_R < \hat{t} \leq \hat{t}_T \\ \frac{u_T}{2}\hat{t} + 2(-\hat{t} + \hat{t}_R)\underline{u_T}\cos(\hat{t}_R) + (\hat{t} - \hat{t}_T)(\underline{u_T} - \overline{u_T})\cos(2\hat{t}_R - \hat{t}_T) - \underline{u_T}\sin(\hat{t}) & \\ +\overline{u_T}\sin(\hat{t} - 2\hat{t}_R) + \underline{u_T}\sin(\hat{t}_R) - \underline{u_T}\sin(2\hat{t}_R - \hat{t}_T) + \overline{u_T}\sin(2\hat{t}_R - \hat{t}_T) & \hat{t} > \hat{t}_T \end{cases} \quad (10)$$

$$\hat{z} = \begin{cases} \frac{-\hat{t}^2}{2} + \underline{u_T}(1 - \cos(\hat{t})) & \hat{t} \leq \hat{t}_R \\ \frac{-\hat{t}^2}{2} + \underline{u_T}\left(1 - \cos(\hat{t} - 2\hat{t}_R) + 2(\hat{t} - \hat{t}_R)\sin(\hat{t}_R)\right) & \hat{t}_R < \hat{t} \leq \hat{t}_T \\ \frac{-\hat{t}^2}{2} + \underline{u_T}\left(1 - \frac{\overline{u_T}}{\underline{u_T}}\cos(\hat{t} - 2\hat{t}_R) + 2(\hat{t} - \hat{t}_R)\sin(\hat{t}_R)\right) - (\underline{u_T} - \overline{u_T})\left(\cos(2\hat{t}_R - \hat{t}_T)\right. & \\ \left.+(\hat{t} - \hat{t}_T)\sin(2\hat{t}_R - \hat{t}_t)\right) & \hat{t} \geq \hat{t}_T \end{cases} \quad (11)$$

For zero initial conditions, curve curve (CC) case when $\hat{t}_R > \hat{t}_T$:

$$\hat{x} = \begin{cases} \frac{u_T}{2}(\hat{t} - \sin(\hat{t})) & \hat{t} \leq \hat{t}_T \\ \frac{u_T}{2}\hat{t} - (\hat{t} - \hat{t}_T)(\underline{u_T} - \overline{u_T})\cos(\hat{t}_T) - (\underline{u_T} + \overline{u_T})\sin(\hat{t}) + \overline{u_T}\sin(\hat{t}_T) & \hat{t}_T < \hat{t} \leq \hat{t}_R \\ \frac{u_T}{2}\hat{t} - (\hat{t} - \hat{t}_T)(\underline{u_T} - \overline{u_T})\cos(\hat{t}_T) + \overline{u_T}(\sin(\hat{t} - 2\hat{t}_R) + \sin(\hat{t}_T)) + 2(-\hat{t} + \hat{t}_R)\overline{u_T}\cos(\hat{t}_R) - \underline{u_T}\sin(\hat{t}) & \hat{t} > \hat{t}_R \end{cases} \quad (12)$$

$$\hat{z} = \begin{cases} \frac{-\hat{t}^2}{2} + \underline{u_T} - \underline{u_T}\cos(\hat{t}) & \hat{t} \leq \hat{t}_T \\ \frac{-\hat{t}^2}{2} + \underline{u_T} - \overline{u_T}\cos(\hat{t}) \quad - (\underline{u_T} - \overline{u_T})(\cos(\hat{t}_T) + (\hat{t}_T - \hat{t})\sin(\hat{t}_T)) & \hat{t}_T < \hat{t} \leq \hat{t}_R \\ \frac{-\hat{t}^2}{2} + \underline{u_T} - \overline{u_T}\cos(\hat{t} - 2\hat{t}_R) - (\underline{u_T} - \overline{u_T})(\cos(\hat{t}_T) + (\hat{t}_T - \hat{t})\sin(\hat{t}_T)) + 2(\hat{t} - \hat{t}_R)\overline{u_T}\sin(\hat{t}_R) & \hat{t} \geq \hat{t}_R \end{cases} \quad (13)$$

Fig. 11. The equations for the reachable set of a quadcopter. See Appendix [31] for a full-size version.

References

[1] F. A. Forecast, "Fiscal years 2019-2039," *Federal Aviation Administration*, 2016. [Online]. Available: https://www.faa.gov/data_research/aviation/aerospace_forecasts/media/FY2019-39_FAA_Aerospace_Forecast.pdf

[2] V. Rubies-Royo, D. Fridovich-Keil, S. Herbert, and C. J. Tomlin, "A classification-based approach for approximate reachability," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 7697–7704.

[3] G. Yang and V. Kapila, "Optimal path planning for unmanned air vehicles with kinematic and tactical constraints," in *Proceedings of the 41st IEEE Conference on Decision and Control, 2002.*, vol. 2. IEEE, 2002, pp. 1301–1306.

[4] R. Ritz, M. Hehn, S. Lupashin, and R. D'Andrea, "Quadrocopter performance benchmarking using optimal control," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2011, pp. 5179–5186.

[5] M. Hehn, R. Ritz, and R. D'Andrea, "Performance benchmarking of quadrotor systems using time-optimal control," *Autonomous Robots*, vol. 33, no. 1-2, pp. 69–88, 2012.

[6] T. Tomić, M. Maier, and S. Haddadin, "Learning quadrotor maneuvers from optimal control and generalizing in real-time," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 1747–1754.

[7] B. Krogh and C. Thorpe, "Integrated path planning and dynamic steering control for autonomous vehicles," in *Proceedings. 1986 IEEE International Conference on Robotics and Automation*, vol. 3. IEEE, 1986, pp. 1664–1669.

[8] L. V. Santana, A. S. Brandão, and M. Sarcinelli-Filho, "Navigation and cooperative control using the ar. drone quadrotor," *Journal of Intelligent & Robotic Systems*, vol. 84, no. 1-4, pp. 327–350, 2016.

[9] S. LaValle, *Planning algorithms*. Cambridge University Press, 2006.

[10] L. Yang, J. Qi, J. Xiao, and X. Yong, "A literature review of UAV 3D path planning," in *Proceeding of the 11th World Congress on Intelligent Control and Automation*. IEEE, 2014, pp. 2376–2381.

[11] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments," in *Robotics Research*. Springer, 2016, pp. 649–666.

[12] W. G. Aguilar, S. Morales, H. Ruiz, and V. Abad, "RRT* GL based optimal path planning for real-time navigation of UAVs," in *International Work-Conference on Artificial Neural Networks*. Springer, 2017, pp. 585–595.

[13] H. Oleynikova, M. Burri, Z. Taylor, J. Nieto, R. Siegwart, and E. Galceran, "Continuous-time trajectory optimization for online UAV replanning," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 5332–5339.

[14] M. Beul and S. Behnke, "Analytical time-optimal trajectory generation and control for multirotors," in *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, June 2016, pp. 87–96.

[15] R. Vidal, S. Rashid, C. Sharp, O. Shakernia, J. Kim, and S. Sastry, "Pursuit-evasion games with unmanned ground and aerial vehicles," in *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*, vol. 3. IEEE, 2001, pp. 2948–2955.

[16] R. Vidal, O. Shakernia, H. J. Kim, D. H. Shim, and S. Sastry, "Probabilistic pursuit-evasion games: theory, implementation, and experimental evaluation," *IEEE transactions on robotics and automation*, vol. 18, no. 5, pp. 662–669, 2002.

[17] E. Camci and E. Kayacan, "Game of drones: UAV pursuit-evasion game with type-2 fuzzy logic controllers tuned by reinforcement learning," in *2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, 2016, pp. 618–625.

[18] D. Falanga, K. Kleber, and D. Scaramuzza, "Dynamic obstacle avoidance for quadrotors with event cameras," *Science Robotics*, vol. 5, no. 40, 2020.

[19] C. Goerzen and M. Whalley, "Sensor requirements for autonomous flight," in *International Conference on Unmanned Aircraft Systems*, 2012, pp. 12–15.

[20] M. Vrba, D. Het, and M. Saska, "Onboard marker-less detection and localization of non-cooperating drones for their safe interception by an autonomous aerial system," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3402–3409, Oct 2019.

[21] E. Boidot, A. Marzuoli, and E. Feron, "Optimal navigation policy for an autonomous agent operating in adversarial environments," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 3154–3160.

[22] L. E. Will and C. W. Schuety, "The American way of swarm: A machine learning strategy for training autonomous systems," Ph.D. dissertation, Monterey, CA; Naval Postgraduate School, 2018.

[23] "Behind the winter olympics' drone show," Jan 2019. [Online]. Available: https://airandspace.si.edu/stories/editorial/behind-winter-olympics-drone-show

[24] J. Jamieson, "Trajectory generation and tracking for drone racing," Ph.D. dissertation, University of Strathclyde, 2018.

[25] M. Chen, S. Herbert, and C. J. Tomlin, "Fast reachable set approximations via state decoupling disturbances," in *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, 2016, pp. 191–196.

[26] Z. Benić, P. Piljek, and D. Kotarski, "Mathematical modelling of unmanned aerial vehicles with four rotors," *Interdisciplinary Description of Complex Systems: INDECS*, vol. 14, no. 1, pp. 88–100, 2016.

[27] S. Kurak and M. Hodzic, "Control and estimation of a quadcopter dynamical model," *Periodicals of Engineering and Natural Sciences (PEN)*, vol. 6, no. 1, pp. 63–75, 2018.

[28] H. Liu, D. Li, J. Xi, and Y. Zhong, "Robust attitude controller design for miniature quadrotors," *International Journal of Robust and Nonlinear Control*, vol. 26, no. 4, pp. 681–696, 2016.

[29] R. Spica, A. Franchi, G. Oriolo, H. H. Bülthoff, and P. R. Giordano, "Aerial grasping of a moving target with a quadrotor UAV," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 4985–4992.

[30] L. E. Dubins, "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents," *American Journal of mathematics*, vol. 79, no. 3, pp. 497–516, 1957.

[31] M. M. Sultan, D. Biediger, B. Li, and A. T. Becker, "Appendices for the reachable set of a drone:exploring the position isochrones for a quadcopter," Oct 2020. [Online]. Available: https://github.com/aabecker/RoboticSwarmControlLab/blob/master/papers/2021_Optimal_Drone_Escape_Appendices.pdf

[32] V. S. Patsko and V. L. Turova, "Level sets of the value function in differential games with the homicidal chauffeur dynamics," *International Game Theory Review*, vol. 3, no. 01, pp. 67–112, 2001.

[33] V. Patsko, S. Pyatko, and A. Fedotov, "Three-dimensional reachability set for a nonlinear control system," *Journal of Computer and Systems Sciences International C/c of Tekhnicheskaia Kibernetika*, vol. 42, no. 3, pp. 320–328, 2003.

[34] V. Patsko, A. Fedotov, S. Kumkov, and S. Pyatko, "Informational sets in model problems of aircraft tracking," *IFAC Proceedings Volumes*, vol. 38, no. 1, pp. 112–117, 2005.

[35] J.-D. Boissonnat and X.-N. Bui, *Accessibility region for a car that only moves forwards along optimal paths*. INRIA France, 1994.

[36] V. S. Patsko and V. L. Turova, "From Dubins car to Reeds and Shepps mobile robot," *Computing and visualization in science*, vol. 12, no. 7, pp. 345–364, 2009.

[37] M. Sultan and A. T. Becker, "Reachable Set for a Drone, Wolfram Demonstrations Project," Dec. 2019. [Online]. Available: https://demonstrations.wolfram.com/ReachableSetForADrone/

[38] E. Kaufmann, A. Loquercio, R. Ranftl, M. Müller, V. Koltun, and D. Scaramuzza, "Deep drone acrobatics," *arXiv preprint arXiv:2006.05768*, 2020.

[39] J. Judson, "Industry pitches munitions designed to defeat drones," Sep 2017. [Online]. Available: https://www.defensenews.com/digital-show-dailies/dsei/2017/09/13/how-to-shoot-down-a-drone-industry-pitches-munitions-designed-to-take-them-out/

[40] G. Harkins, "Navy LCS gun could get potent airburst rounds to take out drones," Jan 2020. [Online]. Available: https://www.military.com/daily-news/2020/01/16/navy-lcs-gun-could-get-potent-airburst-rounds-take-out-drones.html

[41] The Betaflight Open Source Flight Controller Firmware Project, "Betaflight," https://github.com/betaflight/betaflight, 2015.