2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
September 24–28, 2017, Vancouver, BC, Canada

Aggressive Collision Avoidance with Limited Field-of-View Sensing

Brett T. Lopez and Jonathan P. How

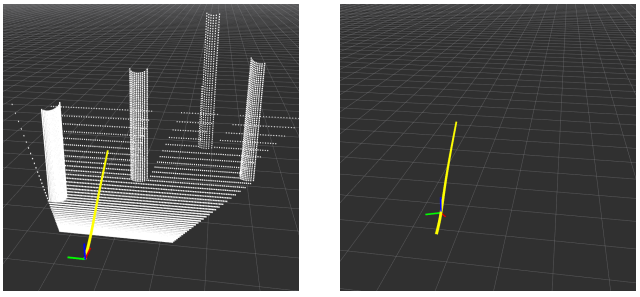
Abstract— **Quadrotors that navigate through unknown, cluttered environments have only recently begun to emerge following the development of small form-factor sensing and computing hardware and computationally efficient collision avoidance algorithms. Computation time of planning algorithms has significantly decreased in part to using local information as opposed to using a global map for collision avoidance. Safe planning with local information, however, restricts the direction of travel to remain within the perception system's field-of-view (FOV). The vehicle's motion becomes more constrained with body-mounted narrow FOV sensors, reducing vehicle maneuverability and speed. This work presents a relaxed-constraint Model Predictive Control framework that allows motions outside the perception FOV with guaranteed safety. The key aspect of this approach is the ability to safely choose a motion primitive generated in the past. Simulation and hardware results shows the new framework improves time to goal and flight path efficiency in environments with varying levels of clutter.**

I. INTRODUCTION

Autonomous navigation in unknown environments is essential for many robot applications and requires an integrated perception, path planning, and state estimation approach. The tight coupling between perception and obstacle avoidance places strong emphasis on gaining useful information about the environment. For body-mounted sensors with limited vertical field-of-view (FOV), the sensor (and vehicle) must be kept near level to ensure oncoming obstacles are observed and avoided. Hence, only low acceleration maneuvers are allowed, subsequently reducing vehicle maneuverability and speed. This paper presents the **R**elaxed-constraint **T**riple **I**ntegrator **P**lanner (**R-TIP**) that overcomes the constraints imposed by body-mounted perception systems without the need for complicated hardware.

Advancements in lightweight sensing and computing have led to quadrotor platforms capable of navigating in unknown environments. These platforms can be broadly categorized as either using local [1]–[6] or global [7], [8] representations of the environment for collision avoidance. The use of local information has received increased attention because of the reduced computation time, which enables real-time planning on the timescales necessary for fast flight. However, this planning paradigm is constrained to operate within the sensor's FOV since the planning decision is made with the most recent perception data. Assumptions about the environment [1], maneuver duration [2], or severely low acceleration maneuvers [3]–[6] are needed to accommodate the FOV constraint – all of which reduce the vehicle's ability to dodge obstacles at high-speed.

Aerospace Controls Laboratory, Massachusetts Institute of Technology, 77 Massachusetts Ave., Cambridge, MA, USA btlopez, jhow@mit.edu



(a) Point cloud when vehicle is near level.

(b) Point cloud when vehicle is at large attitude.

Fig. 1. A quadrotor navigating in an unknown environment. (a): The quadrotor (RGB coordinate system), nearly level, observes a series of obstacles, shown in white, and generates a motion primitive, in yellow, to avoid them. (b): As the vehicle accelerates the obstacles are no longer in the field-of-view (FOV). However, the planner presented in this work can still avoid the obstacles by selecting a safe primitive generated in the past.

The main contribution of this work is a framework that allows safe execution of high-acceleration motion primitives while still planning with local information. Since high-acceleration maneuvers violate the FOV constraint, a relaxed-constraint Model Predictive Control (MPC) [9], [10] framework is used to derive the necessary conditions for which the primitive is guaranteed safe (Section IV). The key component of the **R-TIP** algorithm is its ability to choose a safe motion primitive generated in the past (when the perception system was still producing useful information). Remembering past decisions enables the planner to make smarter decisions given the current information provided by the perception system. More informed decisions and large acceleration maneuvers enhances collision avoidance to the level needed for high-speed flight. In addition, more efficient flight paths are achieved since the best primitive over a *sequence* of replans is executed, as opposed to the instantaneous best primitive. **R-TIP** selects a primitive within 5ms of receiving a point cloud by using the approach set-forth by [1] that entails intelligently sampling a minimum-time motion primitive for collisions using a k-d tree representation of the local environment. **R-TIP**'s ability to safely, aggressively, and efficiently navigate though unknown environments is a direct consequence of the relaxed-constraint framework and low computation time.

II. RELATED WORK

Collision avoidance algorithms can be classified by how they represent the environment. A global representation includes the entire observation history and typically takes the form of an occupancy grid [11], [12]. Building a global

representation of the environment, or a global map, is necessary for many navigation tasks making its use in local collision avoidance very practical. However, the vehicle's top speed is capped by the long computation time associated with map maintenance and collision checking with a large data structure. For instance, [7] presented a quadrotor system that generated 3-D trajectories in a globally consistent map at 1Hz. The minimum-snap trajectory optimization algorithm, originally presented in [13] and refined in [14], was modified to handle state constraints, improve numerical stability, and reduce computation time. [15] expanded [7] by reducing the computation time to 50ms and outperformed other RRT-based algorithms with smoothing in terms of computation and average path length at the expense of successfully finding a trajectory. [16] presented a novel optimization formulation that generated minimum-snap trajectories that satisfied higher-order state constraints. This approach relied on identifying free-space flight corridors within an occupancy grid to find collision-free trajectories. Run-times greater 35ms were reported. The low speed and tame flights demonstrated in the aforementioned works is a direct consequence of the long computation time.

Conversely, a local representation contains the most recent observation of the environment. Planning with local information entails using instantaneous perception data for collision avoidance and is appealing for computationally constrained platforms requiring high-rate collision avoidance. However, it is particularly vulnerable to perception FOV limits and dead-ends. [5], and later [4], used a local map to evaluate motion primitives generated with the maximum dispersion algorithm [17] to guide a quadrotor through a forest. The approach included a local map with fading memory to aid in narrow FOV and backtracking maneuvers. [3] built a local occupancy grid with the most recent perception data and generated a minimum-jerk trajectory in 150ms. [6] constructed a *k*-d tree with instantaneous perception data and solved a quadratically constrained quadratic program to generate a trajectory within 100ms. [2] presented simulation results of a 2-D probabilistic motion primitive framework that was robust to position noise and had a total computation time of 5ms. [1] demonstrated in hardware aggressive 3-D collision avoidance using a minimum-time formulation, adapted from [18], with a computation time between 2-5ms. With the exception of [1], [2], the aforementioned works severely limit acceleration to accommodate FOV limits thereby reducing the ability to avoid obstacles. Further, [2] assumed the maneuver duration was short compared to the speed of oncoming obstacles (and thus ignored the vertical FOV constraint) while [1] assumed no obstacles were close to the vehicle during the start-up acceleration phase. While substantial improvement in computation time has been achieved, body-mounted, narrow FOV sensors still pose a challenge for these planners.

III. Background

A. Motivation

Planning with local information entails making a decision with the most recent perception data so the sensor must be kept nearly level for oncoming obstacles to be observed and avoided. This constrains the vehicle to only execute low-acceleration maneuvers. The impact low acceleration has on performance in terms of obstacle avoidance is quantified in Section VII. However, it is instructive to quantitatively understand why limited FOV sensors limit peak acceleration.

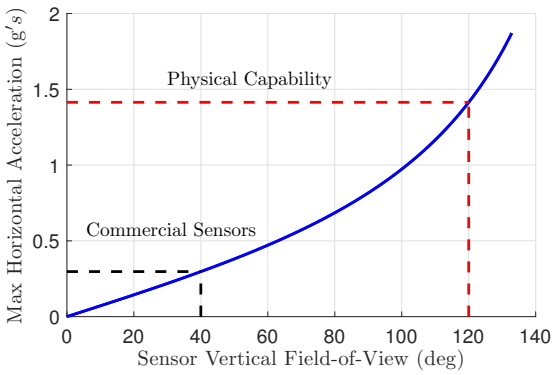


Fig. 2. Maximum horizontal acceleration as a function of sensor vertical field-of-view (FOV). Commercial sensors typical have a 40deg vertical FOV, thus limiting the maximum pitch to 20deg and horizontal acceleration to 0.3g; assuming the sensor is aligned with the vehicle's body axis. However most quadrotors have a 2:1 thrust-to-weight ratio so they can produce a maximum horizontal acceleration of 1.41g. This would require a sensor with a 120deg vertical FOV; such a large FOV is not practical.

The primary implication of the FOV constraint is that the vehicle's pitch, caused by forward acceleration, cannot exceed half of the sensor's vertical FOV (assuming the sensor is aligned with the vehicle's body axis) to ensure observability of oncoming obstacles. In practice the forward and lateral acceleration are coupled so limiting one limits the other. The total acceleration the vehicle can generate is thus constrained, thereby reducing the vehicle's ability to avoid obstacles. The relation between the sensor's vertical FOV and maximum horizontal acceleration is shown in Fig. 2. Commercially available sensors typically have a 40deg vertical FOV, limiting the maximum pitch to 20deg and horizontal acceleration to 0.3g. However, most quadrotors have a 2:1 thrust-to-weight ratio so they can produce a maximum horizontal acceleration of 1.41g. Fig. 2 shows that to achieve 1.41g horizontal acceleration a 120deg vertical FOV sensor is needed to ensure oncoming obstacles are observable; such a large FOV is not practical. As a result, only 21.2% of the total available acceleration can be utilized for collision avoidance with typical commercial sensors.

It is important to acknowledge that a gimbal or additional sensors can artificially increase the perception system's vertical FOV and alleviate the issues presented above. However, many systems cannot accommodate the additional hardware for a gimbal or the acquisition and processing of data from multiple sensors. Thus, efficient algorithmic solutions, like the one proposed in this work, are needed to address the FOV limitations imposed by common perception sensors.

B. Problem Statement

Quadrotors are differentially flat [13] so their entire state can be represented by the flat variables $\boldsymbol{x} = [x\ y\ z\ \psi]^T$,

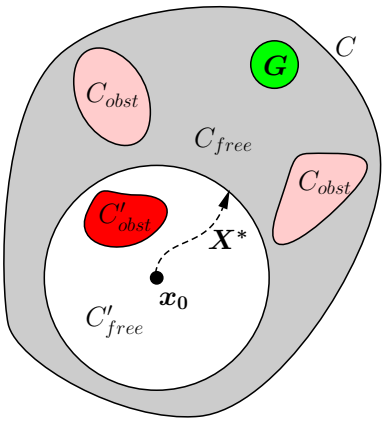


Fig. 3. Illustration of receding horizon planning. The goal is for the vehicle to navigate from its initial state $\boldsymbol{x_0}$ to a goal location $\boldsymbol{G} \in \mathbb{R}^3$ such that $\boldsymbol{\mathcal{X}} \cap C_{obst} = \{\emptyset\}$ where $\boldsymbol{\mathcal{X}}$ is the set of future configurations of the vehicle. However, only $C'_{obst} \subseteq C_{obst}$ is known by the vehicle because of limiting sensing so paths are generated in a receding horizon fashion.

where ψ is the vehicle's yaw, and their derivatives. The quadrotor is modeled as a state and input constrained triple integrator system. This model is sufficiently accurate even though it does not capture the full nonlinear dynamics [1], [18], [19].

Define the quadrotor state $\mathbf{x}$, assuming yaw is constant, as:

$$\mathbf{x} = \begin{bmatrix} \boldsymbol{x}^T & \dot{\boldsymbol{x}}^T & \ddot{\boldsymbol{x}}^T \end{bmatrix}^T = \begin{bmatrix} \boldsymbol{x}^T & \boldsymbol{v}^T & \boldsymbol{a}^T \end{bmatrix},^T \tag{1}$$

and control input $\mathbf{u} = \dddot{\boldsymbol{x}} = \boldsymbol{j}$, where $\boldsymbol{v}$, $\boldsymbol{a}$, and $\boldsymbol{j}$ are the vehicle's velocity, acceleration, and jerk, respectively.

Let C be the configuration of the vehicle and C_{free} and C_{obst} be the free space and obstacle configuration where $C = C_{free} \cup C_{obst}$, as illustrated in Fig. 3. Only the sets $C'_{free} \subseteq C_{free}$ and $C'_{obst} \subseteq C_{obst}$ are known by the vehicle because of limited sensing. Further, let $\boldsymbol{X}^* = \{\boldsymbol{x}^*(t)\} \in \mathcal{Z}$ be a time-indexed path, chosen in a receding horizon fashion, such that $\boldsymbol{X}^* \cap C'_{obst} = \{\emptyset\}$, is thrice differentiable $\boldsymbol{X}^* \in \mathcal{C}^3$, and satisfies the constraints:

$$0 \leq \ ||\boldsymbol{v}^*(t)||_2 \ \leq v_{\max} \tag{2}$$

$$a_{\min} \leq \ ||\boldsymbol{a}^*(t)||_2 \ \leq a_{\max} \tag{3}$$

$$||\boldsymbol{j}^*(t)||_2 \ \leq j_{\max}, \qquad \forall t > t_0, \tag{4}$$

where $a_{\min} > 0$ is required to keep the motors spinning. The goal is for the vehicle to navigate from its initial position $\boldsymbol{x_0}$ to a goal location $\boldsymbol{G} \in \mathbb{R}^3$ using an action set $\mathcal{Z}$ such that $\boldsymbol{\mathcal{X}} \cap C_{obst} = \{\emptyset\}$ where $\boldsymbol{\mathcal{X}} = \{\boldsymbol{X}^*_0, \boldsymbol{X}^*_1, \cdots\}$ is the entire set of future configurations of the vehicle and $\boldsymbol{X}^*_i$ is the path selected at planning epoch i.

IV. Relaxed-Constraint Framework

Receding horizon planning requires the perception sensor be kept near level so the vehicle can sense and avoid oncoming obstacles. This is especially important when planning with local information since the planning decision is made with the most recent observation. Hence, it is common practice to limit the peak acceleration along a path so that:

$$||\boldsymbol{a}^*(t)||_2 \ \leq a_{\max,\mathrm{FOV}}, \qquad \forall t > t_0, \tag{5}$$

where $a_{\max,\mathrm{FOV}}$ depends on the vertical FOV (Section III-A) and is less than the vehicle's actual maximum acceleration $a_{\max,\mathrm{act}}$. However, under certain conditions the constraint in (5) can be relaxed while still guaranteeing safety provided that:

$$||\boldsymbol{a}^*(t)||_2 \ \leq a_{\max,\mathrm{act}}, \qquad \forall t > t_0, \tag{6}$$

the world is accurately modeled, and the violation is sufficiently short. To see this, consider a path $\bar{\boldsymbol{X}}^*$ such that $\bar{\boldsymbol{a}}^*(t)$ violates (5) but satisfies (6). Let d_{perc} and d_{stop} be the vehicle's perception range and maximum stopping distance. Further assume that (5) is violated for a *finite* period of time (i.e. $\exists\ t'$ s.t. $||\bar{\boldsymbol{a}}^*(t)||_2 \ \leq a_{\max,\mathrm{FOV}}\ \forall t > t' \geq t_0$). This ensures the distance traveled during the constraint violation, d_{viol}, is finite since $\bar{\boldsymbol{v}}^*(t)$ is also bounded. Recall that $\bar{\boldsymbol{X}}^* \cap C'_{obst} = \{\emptyset\}$ by construction, then provided that:

$$d_{\mathrm{viol}} < d_{\mathrm{perc}} - d_{\mathrm{stop}}, \tag{7}$$

the path $\bar{\boldsymbol{X}}^*$ is guaranteed safe. Intuitively, (7) is the distance such that the vehicle is allowed to travel without gaining any perception information while still maintaining the ability to stop if an obstacle were just outside its maximum sensing range. Hence, (7) provides a precise definition for how sufficiently short the constraint violation can be.

A consequence of (7) is that the perception range must be longer than the vehicle's stopping distance. Although this is usually true in practice, the proposed formulation works best when $d_{\mathrm{perc}} \approx 2d_{\mathrm{stop}}$. It may appear that this approach alleviates the challenge of narrow FOV sensors by placing more emphasis on long-range sensing. However, as will be shown in Section VII, large accelerations, which this formulation allows, shortens the necessary perception range for collision avoidance at a given speed. Thus, long-ranged sensing is not required.

V. Algorithm

A. Overview

The **R**elaxed-constraint **T**riple **I**ntegrator **P**lanner (**R-TIP**) improves the **TIP** algorithm presented in [1] by using the relaxed-constraint receding horizon framework to enable safe high-acceleration maneuvers. **R-TIP** intelligently samples a minimum-time motion primitive for collisions using a k-d tree representation of the local environment and has the ability to choose a motion primitive generated in the past with guaranteed safety. The system architecture is shown in Fig. 4. A point cloud is provided to the local planner (**R-TIP**), which performs high-rate collision avoidance at the same rate as the perception system. **R-TIP** sends time indexed position and velocity reference commands to a position controller [19] that also has access to a state estimate. Acceleration and jerk are used for feedforward. It is assumed a global planner is providing a goal for the local planner. **R-TIP** and **TIP** are related, so the following focuses on the main components.

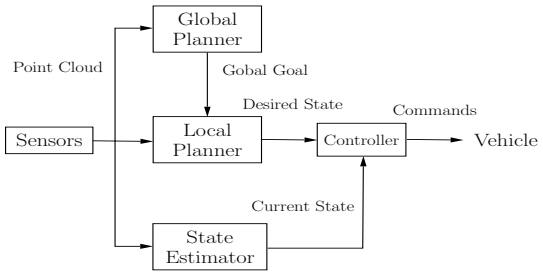
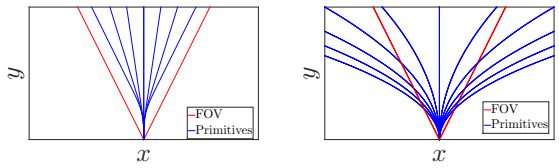


Fig. 4. System architecture used in this work. A point cloud is used for collision avoidance by the local planner. Position and velocity reference commands are sent from the local planner to a position controller. Acceleration and jerk are used for feedforward. It is assumed that a global planner is sending a global goal to the local planner.



(a) Dynamically feasible, state-based primitives. (b) Control-based primitives.

Fig. 5. 2-D primitives comparison. (a): The primitives used in this work are dynamically feasible and easily satisfy state constraints, such as remaining within the sensor's field-of-view. (b): Control-based primitives, generated by sampling the allowable control inputs, are always dynamically feasible but are not guaranteed to satisfy state constraints.

B. Motion Primitives

A computationally efficient technique to ensure all possible paths remain in the sensor's horizontal and vertical FOV is using state-based motion primitives. State-based primitives are generated online and easily incorporate state constraints. Fig. 5 shows a 2-D example of state- and control-based primitives. All of the state-based primitives in Fig. 5a satisfy the horizontal FOV constraint as compared to only a few of the control-based primitives in Fig. 5b. In 3-D, state-based primitives satisfy both the horizontal and vertical FOV constraint.

The desired final state of a primitive is specified as a desired speed and direction. In this work, speed is kept constant while possible directions are uniformly sampled from within the FOV. Each possible direction is assigned a cost that ensures the vehicle makes progress to the goal. The cost of direction i consists of a stage and terminal cost:

$$J_i = \underbrace{\phi^2_{\text{last},i}}_{\text{Stage Cost}} + \underbrace{\theta^2_{\text{goal},i}}_{\text{Terminal Cost}}, \tag{8}$$

where the stage cost is the angle difference $\phi_{\text{last},i}$ between direction i and the last selected direction. This heuristic prioritizes directions similar to the one last selected and dictates the planner's reluctance to large direction changes. The terminal cost is the angle difference between direction i and the heading to the global goal $\theta_{\text{goal},i}$ and ensures the vehicle makes progress to the goal. Note (8) is independent of the world model by design.

With an initial and desired final state, a minimum-time, state and input constrained 3-D motion primitive is generated to point the velocity vector in the desired direction. The control input for a triple integrator system switches at most twice when the final position constraint is relaxed [19]. The problem reduces to finding the times at which the control input switches. The switching times for the inactive constraint case entails solving a quadratic equations whereas the active constraint case involves solving a linear system of equations [19].

C. Collision Checking

Collision checking is generally the most computationally intensive component of path planning. This work intelligently samples a primitive for potential collisions and is more computationally efficient than the standard naive sampling technique [1], [21] since far fewer collision checks are performed. An illustration of the method is shown in Fig. 6. When a new plan is generated, the distance to the closest obstacle is calculated, as in Fig. 6a. Observe that the entire path within the sphere of radius $d_{obst,\min}$ is guaranteed to be collision free. The next time-instance that a collision is possible is when $||\boldsymbol{r}_d^I(t_1^*)|| = d_{obst,\min}$. Since a desired top-speed $v_{\max}$ is known, t_1^* can be approximated as $t_1^* \approx \frac{d_{obst,\min}}{v_{\max}}$. Even if the vehicle is not traveling at top speed, it is guaranteed to be within the collision-free sphere for $t \leq t_1^*$. The process of finding the closest obstacle and evaluating the path at the approximate boundary of the collision-free sphere is repeated, Fig. 6b, until either the closest point is an intermediate goal $\boldsymbol{I}$ at the edge of the sensing horizon or an obstacle is closer than a predefined safe distance d_{buffer}. If a collision is predicted, the next best primitive is evaluated.

If none of the primitives extend out to the sensing horizon, then the primitive cost is modified to:

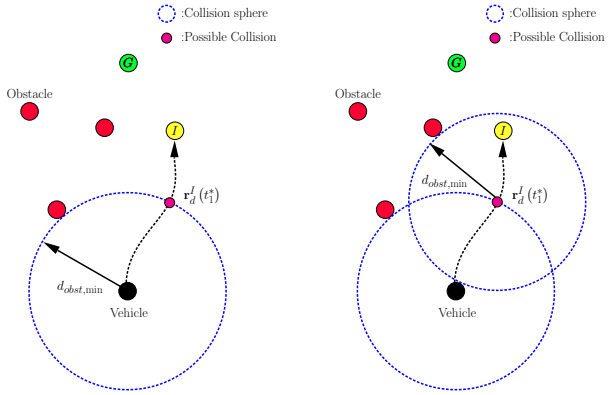
$$J_i = (d_{\text{perc}} - d_i)^2, \tag{9}$$

where d_{perc} is the maximum sensing range and d_i is the primitive's length that must be greater than the stopping distance. This prioritizes primitives with the longest length, so that the vehicle can maintain its top speed.

D. Primitive History

Each time a primitive is selected for execution, it, its cost, and the vehicle's position are saved for evaluation at the next planning event. If, at the next replan, a lower cost primitive exists and is collision free, it is selected for execution and saved. Conversely, if only higher cost primitives exist, which often occurs during aggressive maneuvers, then so long as (7) is satisfied and no new collision is predicted, the last saved primitive is executed. For additional safety, the saved primitive is rechecked for collisions each time new perception data is available. Once (7) is violated or a new collision is predicted, the planner selects a new primitive or executes a stop maneuver. This procedure incurs no significant additional computation time given the collision checking speed up presented in Section V-C.

Note that the above discussion made no mention of the vehicle's (and hence sensor's) attitude – selecting the saved primitive is solely based on the primitive's cost. Hence, the



(a) First collision check. (b) Second collision check.

Fig. 6. Collision checking through intelligent sampling of the path. (a): The closest obstacle is calculated when a new path is generated. (b): The path is then re-evaluated at the next possible collision time, which occurs at the edge of the collision-free sphere. This process is repeated until either the closest point is an intermediate goal $\boldsymbol{I}$ at the edge of the sensing horizon or a collision is detected. The next best primitive is evaluated if a collision is detected.

perception system could be generating useful information but the planner is still executing the saved primitive. Thus, the optimal primitive over a *sequence* of replans is selected, as opposed to the instantaneous best primitive. This is especially useful given that a slight change in perspective can create a significantly different outcome when planning with local information. As will be seen in Section VII, more efficient flight paths are a direct consequence of the chosen implementation.

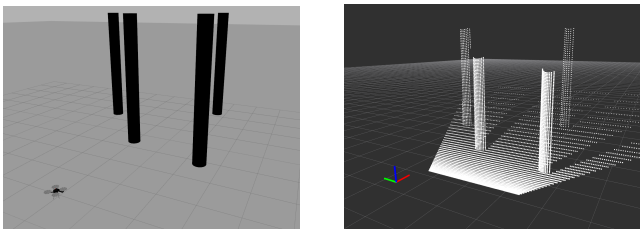
The constraint violation d_{viol} is a tuneable parameter that in practice works best with $d_{\text{viol}} \approx d_{\text{stop}}$ so $d_{\text{perc}} \approx 2d_{\text{stop}}$. Long constraint violation requires a longer perception range, which might not be possible. In addition, a higher cost primitive may actually be the correct choice given how the world is configured. Unfortunately this is nearly impossible to know a priori. Therefore, it is this author's contention that setting the constraint violation duration to be approximately the stopping distance is a good balance between short-term gain and long-term risk while maintaining safety.

It is also important to acknowledge that in real physical systems the maximum sensing range and vehicle's position are not precisely known. Typically these quantities can be modeled as a random variable with some distribution. Thus, a chance constrained version of (7) can be used by system's with large perception and state uncertainty.

VI. Test Environment

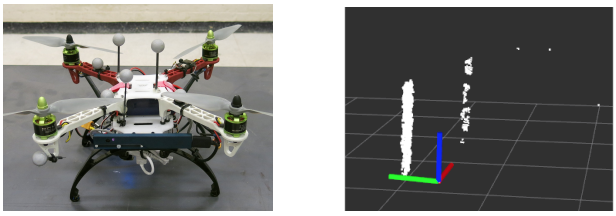
A. Simulation

The simulation environment is a combination of custom and open source code. The dynamics engine is custom and written in C++. Gazebo [22] and the hector_quadrotor package [23] were used to simulate perception data (in the form of a point cloud) as the quadrotor navigated custom worlds. The quadrotor model was changed to match the one used in this work (see Section VI-B). 3-D depth perception was simulated using the Asus model with a 40deg FOV, 12m maximum range, and 60Hz frame rate. A sample world and the quadrotor model are shown in Fig. 7a. The corresponding perception data is shown Fig. 7b.



(a) Quadrotor model and sample world. (b) Corresponding simulated point cloud.

Fig. 7. Simulation environment. (a): Quadrotor model and custom pole world created in Gazebo robot simulator. (b): Simulated point cloud using the Asus model with a 40deg FOV, 12m maximum range, and 60Hz frame rate. The coordinate system is representative of the quadrotor's position. Each grid cell is 1mx1m.



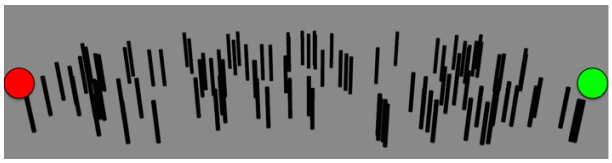
(a) Flight vehicle. (b) R200 point cloud.

Fig. 8. Hardware used in this work. (a): Flight vehicle equipped with a Jetson TX1 for onboard perception, planning, and control. The Intel® RealSense™ R200 RGB-D camera, front, is used for onboard perception. (b): Point cloud using the R200's onboard IR emitter, a maximum range of 4m is achieved indoors. Pillars are placed every 2m. The coordinate system is representative of the camera's position. Each grid cell is 1mx1m.

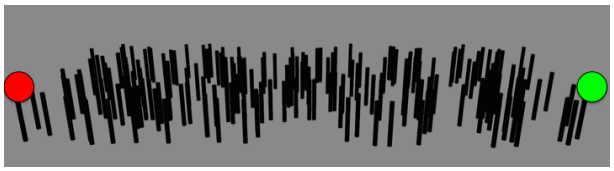
B. Hardware

The quadrotor used in this work is shown in Fig. 8a. The frame is composed of arms from a DJI F330 quadrotor and a custom Delrin base plate. The vehicle is equipped with a Jetson TX1 to enable onboard perception, planning, and control. A Vicon motion capture system is used, but only for vehicle state estimation – all obstacle knowledge is inferred from the onboard processing of the vehicle-mounted sensors. The vehicle has a 42% hover throttle with a 4S battery, leaving enough control authority for aggressive flight experiments. The total vehicle weight is 1.2kg. The maximum acceleration and jerk were experimentally found to be $20^{\text{m}}/_{\text{s}^2}$ and $60^{\text{m}}/_{\text{s}^3}$, respectively.

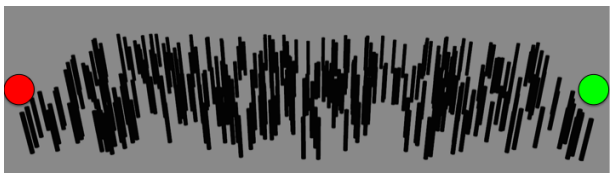
The 34g Intel® RealSense™ R200 RGB-D camera was used for perception. This lightweight sensor uses an IR stereo camera pair to provide 3-D depth information at a maximum rate of 60fps with a 480×360 resolution. Fig. 8b shows the point cloud using the sensor's onboard IR emitter with pillars spaced every 2m. The onboard emitter projects a static infrared pattern to increase scene texture [24]. Only two pillar are observed so the maximum indoor range is 4m. The maximum range of the sensor was limited to 3m given the small flight volume the experiments were conducted in.



(a) Low Difficulty, $\rho = 0.1$.



(b) Moderate Difficulty, $\rho = 0.2$.



(c) High Difficulty, $\rho = 0.3$.

Fig. 9. Perspective view of sample simulation environments of low, moderate, and high difficulty. The vehicle starts at the origin (●) and navigates to the goal location (●) while avoiding obstacles. Obstacle locations are not known beforehand.

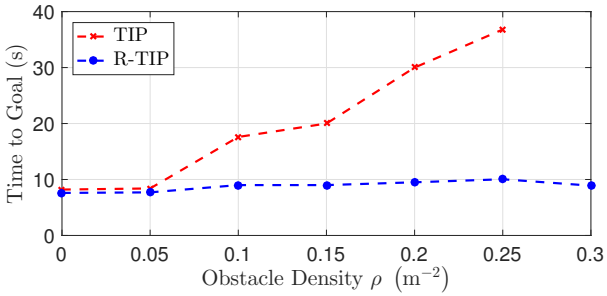


Fig. 10. Time to goal for **R-TIP** and **TIP** as a function of obstacle density ρ. Time to goal for **TIP** grows monotonically with obstacle density while **R-TIP** shows relatively little variation implying **R-TIP** is superior for fast flight in cluttered environments. **TIP** was not able to complete the $\rho = 0.3$ case.

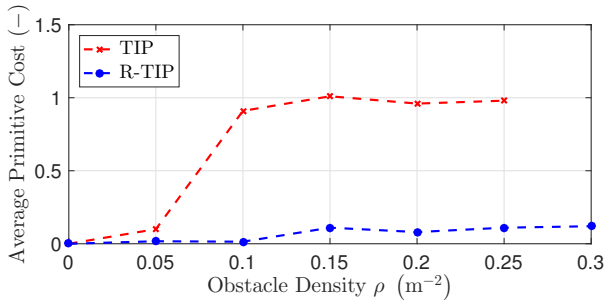


Fig. 11. Average primitive cost for **R-TIP** and **TIP** as a function of obstacle density ρ. **R-TIP** has a lower average primitive cost than **TIP** $\rho > 0$ implying more efficient flight paths. The average primitive cost saturates because of the saturation in the cost function formulation.**TIP** was not able to complete the $\rho = 0.3$ case.

VII. Analysis

A. Simulation

R-TIP and **TIP** were tested extensively in simulation to quantify the performance of the proposed relaxed-constraint receding horizon framework. Sample test environments are shown in Fig. 9. Obstacle locations were sampled from a uniform distribution to populate a 50m x 20m grid. Test difficulty was varied from low (Fig. 9a) to high (Fig. 9c) by increasing obstacle density ρ. The vehicle starts at the origin (●) and attempts to navigate to the goal (●) at the top speed of $8^m/_s$. The peak horizontal acceleration for **R-TIP** was 1.41g (the vehicle's maximum) and 0.3g for **TIP** (maximum allowed for a 40deg FOV sensor). Time to goal and average primitive cost were the two metrics chosen to evaluate performance. Time to goal captures the vehicle's ability to navigate quickly and average primitive cost quantifies the efficiency of the flight path through the environment.

Fig. 10 shows the time to goal with **R-TIP** and **TIP** for different obstacle densities. Time to goal for **TIP** grows monotonically with obstacle density where **R-TIP** shows relatively little variation. **TIP** is expected to have a longer time to goal given that low peak acceleration results in a longer time to reach top speed (confirmed by the longer time at $\rho = 0$). However, this difference is small: approximately an additional 1.6s. Hence, the results show that the relaxed-constraint framework improves maneuverability and enables fast flight for varying degrees of obstacle density. Note that the times reported are the best times for a set of 10 simulations at each obstacle density and for each planner. **TIP** consistently got stuck and was unable to navigate to the goal for $\rho = 0.3$ since only low acceleration maneuvers were permitted.

The average primitive cost for the best time to goal performance of **R-TIP** and **TIP** for different obstacle densities is shown in Fig. 11. Primitive cost is a metric for plan efficiency. **R-TIP** is shown to have a lower average primitive cost than **TIP** for $\rho > 0$ indicating more efficient paths. The improved performance of **R-TIP** is a direct consequence of the relaxed-constraint framework since the last optimal primitive is available given the relaxation is still valid. Hence, the planner has access to a lower cost primitive at each replan. Note that the average primitive cost saturates because of the saturation in the modified cost function given by (9). **TIP** consistently got stuck and was unable to navigate to the goal for $\rho = 0.3$ since only low acceleration maneuvers were permitted.

The primary role of saving the last primitive in **R-TIP** is to enable large acceleration maneuvers. The memory indicator, 1 if the planner is using the saved primitive and 0 otherwise, is a metric that quantifies how often the planner uses the past primitive for collision avoidance. Fig. 12 shows the memory indicator for the $\rho = 0.15$ test case. The saved primitive is used 56.5% of the total flight indicating large acceleration maneuvers were often executed. The high frequency confirms the intuition that large accelerations are a crucial component of collision avoidance in highly-cluttered environments at high-speed. Similar percentages were observed for the other test environments as well.

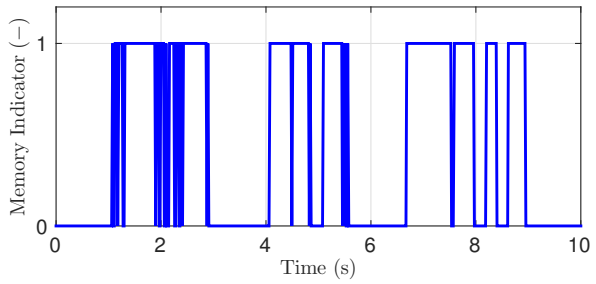


Fig. 12. Memory indicator for $\rho = 0.15$ test case. 1 indicates the planner is using the saved primitive and 0 indicates using a new primitive. The saved primitive is used 56.5% of the total flight indicating large acceleration maneuvers were often used. Similar percentages were observed for the other test environments.

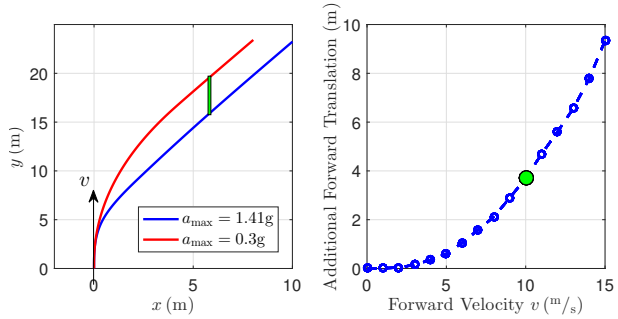


Fig. 13. Impact of maximum allowable acceleration on collision avoidance and perception range. (a): Two paths with different maximum allowable acceleration. The lower acceleration path requires more distance to complete the maneuver. (b): Additional forward translation for the lower acceleration path as a function of initial forward velocity. The curve implies that low-acceleration maneuvers requires a longer sensing range to achieve the same performance as high-acceleration maneuvers. Note the green line in (a) corresponds to the green dot in (b).

Fig. 13 shows the benefit of larger accelerations by comparing two paths with different maximum accelerations. The 0.3g path in Fig. 13a requires longer forward distance (denoted by the green line) to complete the maneuver than the 1.41g path, given the maneuver is started at the same time. It is clear that the 0.3g path must be started earlier and how much earlier is given by Fig. 13b for different forward velocities v. Earlier execution requires the perception system have a longer sensing range to achieve the same performance. Thus, **R-TIP** is capable of aggressive, high-speed navigation without the need for wide FOV and long-range perception systems.

B. Hardware

R-TIP was tested in the obstacle course shown in Fig. 14. Only one obstacle is observed at takeoff and the second row is just outside the sensor's range (3m). This configuration displays the important role (7) plays in **R-TIP** since following a primitive for too long will result in a crash. The vehicle traverses a 7m distance with $4^{\text{m}}/_{\text{s}}$ speed. The stopping distance for $4^{\text{m}}/_{\text{s}}$ is approximately 1.2m so $d_{\text{perc}} \approx 2d_{\text{stop}}$, as desired.

Fig. 15 shows a series of data visualization stills for the flight test. The perception data is shown in white, primitive in yellow, and goal in green. The RGB coordinate system

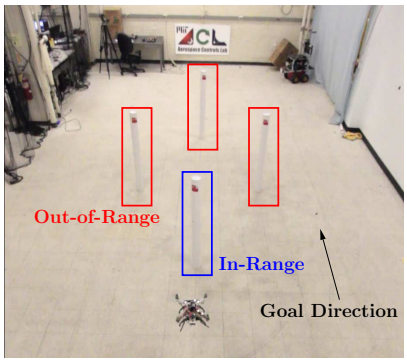
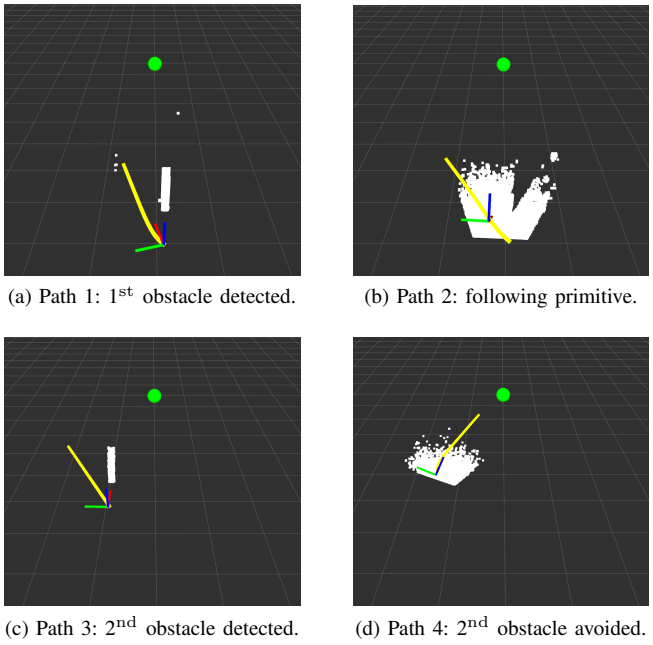


Fig. 14. Test course for **R-TIP** where the vehicle is required to aggressively accelerate up to $4^{\text{m}}/_{\text{s}}$. Only one of the four obstacles is known at takeoff. The second row is placed just outside the maximum sensing range. This configuration displays the important role (7) plays in **R-TIP** since following a primitive for too long will result in a crash.



(a) Path 1: 1st obstacle detected.

(b) Path 2: following primitive.

(c) Path 3: 2nd obstacle detected.

(d) Path 4: 2nd obstacle avoided.

Fig. 15. Perception (white), motion primitive (yellow), and goal (green) visualization for hardware experiments. (a): **R-TIP** selects a primitive that avoids the first obstacle that is seen at takeoff. (b): The first obstacle exits the FOV during large acceleration but **R-TIP** intelligently selects a past primitive and subsequently avoids the first obstacle. (c): The vehicle reaches top speed and the second obstacle is observed. (d): The second obstacle is avoided and the vehicle heads towards to goal.

is representative of the vehicle's position and orientation. The first obstacle is observed at takeoff so **R-TIP** plans a path that avoids it (Fig. 15a). As the vehicle pitches forward to accelerate up to speed, the first obstacle begins to exit the FOV (Fig. 15b). The planner intelligently selects the primitive generated when the first obstacle was still observable thereby ensuring successful avoidance. Once near top speed the vehicle levels out and observes the second obstacle (Fig. 15b). The vehicle subsequently avoids the second obstacle and heads towards the goal (Fig. 15c)

VIII. Conclusion and Future Work

Body-mounted, narrow FOV sensing poses significant challenges to planners that operate with instantaneous perception data. With this planning paradigm, a decision is made with the most recent observation so the sensor (and vehicle) must be kept near level to observe obstacles – subsequently limiting vehicle speed. This work presented the **R**elaxed-constraint **T**ripple **I**ntegrator **P**lanner that overcomes this challenge by safely choosing a motion primitive generated in the past. The necessary condition for safety was derived using a relaxed-constraint MPC framework. Simulation results showed the new framework improves time to goal and flight path efficiency for environments with varying degrees of clutter. **R-TIP** was also tested in hardware where the environment required the planner to switch between past and new primitives. Future work includes expanding the planner to handle dynamic obstacles and state and perception uncertainty.

IX. Acknowledgments

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. 1122374 and by the DARPA Fast Lightweight Autonomy (FLA) program. The author would also like to acknowledge Boeing Research & Technology for support of the indoor flight facility and hardware.

References

[1] B. T. Lopez and J. P. How, "Aggressive 3-d collision avoidance for high-speed navigation," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017.

[2] P. Florence, J. Carter, and R. Tedrake, "Integrated perception and control at high speed: Evaluating collision avoidance maneuvers without maps," in *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2016.

[3] S. Liu, M. Watterson, S. Tang, and V. Kumar, "High speed navigation for quadrotors with limited onboard sensing," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1484–1491.

[4] S. Daftry, S. Zeng, A. Khan, D. Dey, N. Melik-Barkhudarov, J. A. Bagnell, and M. Hebert, "Robust monocular flight in cluttered outdoor environments," *arXiv preprint arXiv:1604.04779*, 2016.

[5] D. Dey, K. S. Shankar, S. Zeng, R. Mehta, M. T. Agcayazi, C. Eriksen, S. Daftry, M. Hebert, and J. A. Bagnell, "Vision and learning for deliberative monocular cluttered flight," in *Field and Service Robotics*. Springer, 2016, pp. 391–409.

[6] F. Gao and S. Shen, "Online quadrotor trajectory generation and autonomous navigation on point clouds," in *Safety, Security, and Rescue Robotics (SSRR), 2016 IEEE International Symposium on*. IEEE, 2016, pp. 139–146.

[7] M. Burri, H. Oleynikova, M. W. Achtelik, and R. Siegwart, "Real-time visual-inertial mapping, re-localization and planning onboard mavs in unknown environments," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, 2015, pp. 1872–1878.

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

[9] J. B. Rawlings and K. R. Muske, "The stability of constrained receding horizon control," *IEEE transactions on automatic control*, vol. 38, no. 10, pp. 1512–1516, 1993.

[10] R. J. M. Afonso and R. K. H. Galvão, *Infeasibility Handling in Constrained MPC*. INTECH Open Access Publisher, 2012.

[11] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.

[12] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous Robots*, vol. 34, no. 3, pp. 189–206, 2013.

[13] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2520–2525.

[14] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for quadrotor flight," in *International Conference on Robotics and Automation*, 2013.

[15] H. Oleynikova, D. Honegger, and M. Pollefeys, "Reactive avoidance using embedded stereo vision for mav flight," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 50–56.

[16] J. Chen, T. Liu, and S. Shen, "Online generation of collision-free trajectories for quadrotor flight in unknown cluttered environments," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1476–1483.

[17] C. J. Green and A. Kelly, "Toward optimal sampling in the space of paths," in *In Proceedings of the International Symposium of Robotics Research*. Citeseer, 2007.

[18] M. Hehn and R. DAndrea, "Quadrocopter trajectory generation and control," in *World Congress*, vol. 18, no. 1, 2011, pp. 1485–1491.

[19] B. T. Lopez, "Low-latency trajectory planning for high-speed navigation in unknown environments," Master's thesis, Massachusetts Institute of Technology, 2016.

[20] D. E. Kirk, *Optimal control theory: an introduction*. Courier Corporation, 2012.

[21] J. Bialkowski, M. Otte, S. Karaman, and E. Frazzoli, "Efficient collision checking in sampling-based motion planning via safety certificates," *The International Journal of Robotics Research*, p. 0278364915625345, 2016.

[22] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, vol. 3. IEEE, 2004, pp. 2149–2154.

[23] J. Meyer, A. Sendobry, S. Kohlbrecher, U. Klingauf, and O. von Stryk, "Comprehensive simulation of quadrotor uavs using ros and gazebo," in *3rd Int. Conf. on Simulation, Modeling and Programming for Autonomous Robots (SIMPAR)*, 2012, p. to appear.

[24] "Intel® RealSense™ Camera R200," https://software.intel.com/sites/default/files/managed/d7/a9/realsense-camera-r200-product-datasheet.pdf, 2016, online; accessed 27 August 2016.