EasyChair Preprint

№ 1454

Fast Exploration using Multirotors: Analysis, Planning, and Experimentation

Kshitij Goel, Micah Corah, Curtis Boirum and Nathan Michael

September 4, 2019

Fast Exploration using Multirotors: Analysis, Planning, and Experimentation

Kshitij Goel, Micah Corah, Curtis Boirum, and Nathan Michael

Abstract This work presents a system and approach for rapid exploration of unknown environments using aerial robots. High-speed flight with multirotor air vehicles is challenging due to limited sensing range, use of on-board computation, and constrained dynamics. For robots operating in unknown environments, the control system must guarantee collision-free operation, and for exploration tasks, the system should also select sensing actions to maximize information gain with respect to environment. To this end, we present a motion primitive-based, receding-horizon planning approach that maximizes information gain, accounts for platform dynamics, and ensures safe operation. Analysis of motions parallel and perpendicular to frontiers given constraints on sensing and dynamics leads to bounds on safe velocities for exploration. This analysis and the bounds obtained, inform the design of the motion primitive approach. Simulation experiments in a complex 3D environment demonstrate the utility of the motion primitive actions for rapid exploration and provide a comparison to a reduced motion primitive library that is appropriate for online planning. Experimental results on a hexarotor robot with the reduced library demonstrate rapid exploration at speeds above $2.25\,\mathrm{m/s}$ under a varying clutter in an outdoor environment which is comparable to and exceeding the existing state-of-the-art results.

1 Introduction

Fast and safe exploration of unknown environments is an important aspect of robotics applications such as urban search-and-rescue and disaster response. In such scenarios, it is essential for a robot or a team of robots to find survivors or objects in an unstructured and unknown environment quickly while maintaining collision-free

Kshitij Goel · Micah Corah · Curtis Boirum · Nathan Michael
Carnegie Mellon University, Pittsburgh, PA, e-mail: {kshitij, micahcorah, cboirum, nmichael}@cmu.edu
This work was supported by DOE (DE-EM0004067) and industry.

operation. Aerial robots are particularly suitable for this task and have been used for mapping hazardous environments [16] and in prior work on exploration scenarios [1, 3, 6, 7]. Despite this interest, relatively little is known about the relationship between the dynamics of aerial robots and performance in exploration such as what limits performance in these systems or what can be done to improve performance in aerial exploration. This work investigates these limitations and to design planners suitable for exploration that satisfy these constraints.

Most works on robotic exploration study ground robots with largely trivial dynamics [11, 19, 20] and have been unconcerned with the role of speed in exploration. However, given the increasing application of aerial robots as sensing platforms, recent works have begun to consider the effects of system dynamics on high-speed exploration and navigation in unknown environments. Cieslewski et al. [6] propose a strategy based on maintaining rapid forward motion by driving the system toward frontier cells (cells on boundary of free and unknown space [20]) within the camera field-of-view. While the authors note that reaction time for obstacles avoidance can limit speeds in exploration, they provide little discussion of why this happens or how it can be avoided. This work considers a broader variety of sensing actions that can avoid these limitations and also incorporates sensing and planning time into action design. Also, Liu et al. [12] provide similar discussion as we do regarding how time delay in processing sensor data together with constrained dynamics can limit velocities when navigating unknown environments.

This work applies an information-based planning approach which reasons explicitly about information gain from future sensor observations [3, 4, 11]. As in previous works on robotic exploration [7, 8, 11], we use a randomized planning strategy based on Monte-Carlo tree search [2, 5]. Here, we focus on design of a library of motion primitives suitable for exploration at high speeds given the relationship between sensing constraints and maximum safe velocities. We evaluate this approach in simulation experiments demonstrating exploration of a large warehouse environment and via outdoor field tests with a hexarotor robot.

2 Steady-State Velocity Analysis

This section presents analysis of exploration performance for an aerial robot operating for steady-state conditions such as continuous motion toward a frontier. This analysis produces bounds on velocity and rates of entropy reduction, given constraints on dynamics and sensing. We leverage these insights in Sect. 3 to design motion primitive actions for rapid exploration.

2.1 System Model and Safety Constraints

This work applies a simplified double-integrator quadrotor model with acceleration and velocity constraints for analysis of limits on exploration performance, which can be thought of as a relaxation of dynamics models that are commonly used for

position and attitude control of multirotor vehicles [13, 15]. Let $\mathbf{r} = [x, y, z]^{\mathrm{T}}$ be the position of the vehicle in an inertial world frame $\mathcal{W} = \{\mathbf{x}_{\mathcal{W}}, \mathbf{y}_{\mathcal{W}}, \mathbf{z}_{\mathcal{W}}\}$, and let the body frame be $\mathcal{B} = \{\mathbf{x}_{\mathcal{B}}, \mathbf{y}_{\mathcal{B}}, \mathbf{z}_{\mathcal{B}}\}$. Assuming small roll and pitch angles, and given the yaw angle ψ, the system state is $\boldsymbol{\xi} = [\mathbf{r}^{\mathrm{T}}, \psi, \dot{\mathbf{r}}^{\mathrm{T}}, \dot{\psi}]^{\mathrm{T}}$. The derivatives of position and yaw satisfy bounds on velocity and acceleration

$$||\dot{\mathbf{r}}||_2 \leq V_{\max} \qquad ||\ddot{\mathbf{r}}||_2 \leq A_{\max} \qquad |\dot{\psi}| \leq \Omega_{\max} \tag{1}$$

where $||\cdot||_2$ is the L2-norm.

Further, the robot is equipped with a forward-facing depth sensor with range of r_{depth} for use in mapping. However, while navigating the environment, the robot must also be able to avoid collisions with obstacles.

The requirement for collision-free operation restricts the set of actions that a multirotor can safely execute while navigating in an unknown environment. A planning policy can ensure collision-free operation by guaranteeing that the robot is able to stop entirely within unoccupied space $\mathcal{X}_{\text{free}}$, given an appropriate collision radius r_{coll}, such as in the work of Janson et al. [9]. In the worst case, any voxel in the unknown space $\mathcal{X}_{\text{unk}}$ may be revealed to be occupied and so possibly force the robot to stop within $\mathcal{X}_{\text{free}}$.

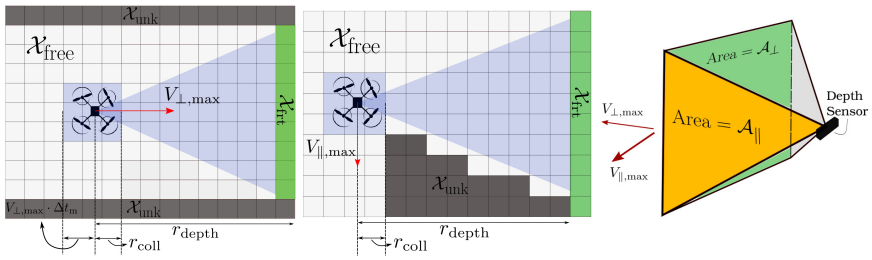
The robot plans once every Δt_{p} seconds, and there is also latency Δt_{m} for acquiring depth information and integrating it into the occupancy map. The sensor data is Δt_{m} seconds old at the beginning of planning, and once the planner is done, the robot executes the selected action for another Δt_{p} so that the total effective latency is no more than $\Delta t_{\text{l}} = \Delta t_{\text{m}} + 2\Delta t_{\text{p}}$. Note that, although latency may be unpredictable in practice, the robot will not depend on consistent latency to maintain safe operation.

2.2 Steady-State Exploration Scenarios

Figure 1 illustrates two possible scenarios for steady-state motion with respect to a frontier. For the perpendicular case (Fig. 1a) the robot moves continuously toward a frontier and may have to avoid obstacles at the edge of the sensor range. As discussed in Sect. 2.1, the robot must be able to avoid collisions with obstacles in the unknown environment even if there are not any there. This means that the the robot must always be prepared to stop before reaching the frontier. For the parallel case (Fig. 1b) the robot moves along the frontier through space that has already been mapped. When known space is free of obstacles, the robot may continue to do so at the maximum allowable velocity. This scenario can also be thought of as the limit for a spiral motion—which will arise later in the experimental results—as the curvature becomes very small.

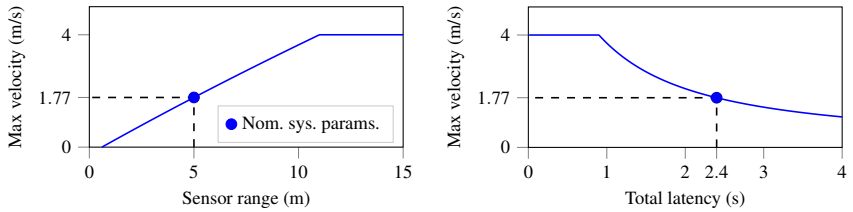
2.3 Bounds on Velocity

Given the system model and constraints for exploration scenarios, we now proceed with calculation of the velocity bounds for motion perpendicular and parallel to the



(a) Perpendicular scenario **(b)** Parallel scenario **(c)** Sensor cone

Fig. 1: Steady-state exploration scenarios. Analysis in Sect. 2 presents upper bounds on velocities are for a double integrator system **(a)** for motion perpendicular to a frontier ($V_{\perp,\max}$) and **(b)** for motion parallel to it ($V_{\parallel,\max}$). To ensure safety in the perpendicular case, the robot has to stop within a user-specified collision radius from the frontier ($\mathcal{X}_{\text{frt}}$), i.e. within $r_{\text{depth}} - r_{\text{coll}}$ from the current state. For the parallel case, no such restrictions exist since the robot is moving in the explored space which, ideally, is free ($\mathcal{X}_{\text{free}}$). **(c)** Combining the area of the projection of the sensor cone in the direction of motion with the bounds on velocity leads to upper bounds on rates of novel voxels observed during exploration.



(a) Velocity ($V_{\perp,\max}$) vs. sensor range (r_{depth}) **(b)** Velocity ($V_{\perp,\max}$) vs. total latency (Δt_{l})

Fig. 2: Maximum achievable velocity moving toward a frontier ($V_{\perp,\max}$) according to (2) based on parameters used in Tab. 1 and varying **(a)** sensor range and **(b)** total latency (which consists of latency for the mapping system and time for planning). The circle marks the maximum velocity at which the robot can move toward unknown space for the parameters used in this paper (see Tab. 1) which is less than half the overall velocity bound. Approaching this velocity limit requires either sensor range exceeding 10 m, both decreased planning time and mapping latency, or some combination of the two.

frontier, $V_{\perp,\max}$ and $V_{\parallel,\max}$ respectively. Maximum velocity toward the frontier is computed based on motion at a constant velocity followed by stopping at maximum deceleration to satisfy the safety constraint. The expression for $V_{\perp,\max}$ is a function of acceleration ($A_{\max}$), maximum sensing range (r_{depth}), the collision radius (r_{coll}), and the latency in planning Δt_{l} (see Fig. 1a) and is given by

$$
\begin{aligned}
V_{\perp,\max} &= \min(V_{\max}, V'_{\perp,\max}) \\
V'_{\perp,\max} &= A_{\max} \cdot \left(\sqrt{\Delta t_{\text{l}}^2 + 2\frac{r_{\text{depth}} - r_{\text{coll}}}{A_{\max}}} - \Delta t_{\text{l}} \right).
\end{aligned}
\tag{2}
$$

Table 1: Steady-state upper bounds on velocity and rate of entropy reduction for the scenarios described in Sect. 2.2. All values are computed for a planning time of 1 Hz, mapping latency 0.4 s, sensor point cloud of size 9.93 m × 5.68 m based on the RealSense D435 depth sensor with image size 424px × 240px and a maximum depth r_{depth} of 5 m. Occupancy grid resolution is 20 cm with an overall bound on top speed $V_{\max}$ at 4 m/s, and collision radius r_{coll} set at 0.6 m.

Value/Cases	Area (m^2)	Velocity (m/s)	Volume rate (m^3/s)	Entropy rate (bits/s)
Perpendicular ($\perp$)	56.4	1.77	99.83	1.25×10^4
Parallel ($\parallel$)	57.19	4.00	228.8	2.86×10^4
$\perp$**, rapid yaw**	56.8	1.77	100.5	1.26×10^4
$\parallel$**, rapid yaw**	78.05	4.00	312.2	3.90×10^4

Figure 2 shows the variation of this bound with r_{depth} and Δt_{l} for the parameters used in this work. For motion parallel to a frontier (see Sect. 2.2 and Fig. 1b), there are no obstacles in the direction of motion. Therefore, the steady-state upper bound on the velocity moving parallel to the frontier is identical to the maximum achievable by the system, i.e. $V_{\parallel,\max} = V_{\max}$.

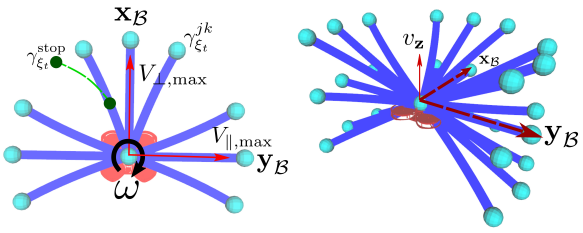
The entropy reduction then can also be bounded for each scenario terms of the sensor geometry (see Fig. 1c) and steady-state velocities by projecting the sensing volume in the direction of motion. Here, we also introduce the possibility of rapid yaw motion during either motion. Results are shown in Tab. 1. Note that moving parallel to the frontier can provide significantly improved performance.

3 Action Representation

This section details the design of available actions for the proposed motion planning framework. We define a trajectory generation scheme, related parameters and conventions, and action design specifics leveraging insights gained in Sect. 2.

3.1 Motion Primitive Library Generation

Safe and accurate high-speed flight requires large and smooth linear acceleration and angular velocity references. Smoothness for such references depends on higher derivatives of position, jerk and snap [14]. For this work, the actions that are avialable to the robot are snap-continuous, forward arc [21] motion primitives, which have previously been applied to high-speed teleoperation of multirotors [18]. Given the differentially-flat state of the multirotor at time t, $\boldsymbol{\xi}_t = [x, y, z, \psi]^{\mathrm{T}}$, denote an available action parameterization as $\mathbf{a} = [v_{\mathbf{x}}, v_{\mathbf{z}}, \omega]$ where $v_{\mathbf{x}}$ and $v_{\mathbf{z}}$ are velocities in the body frame $\mathbf{x}_{\mathcal{B}}$ and $\mathbf{z}_{\mathcal{B}}$ directions, and ω is the body yaw rate. Actions are discretized using user-specified maximum velocity bounds in $\mathbf{x}_{\mathcal{B}} - \mathbf{y}_{\mathcal{B}}$ plane (ω variation, N_ω primitives) and in $\mathbf{z}_{\mathcal{B}}$ direction ($v_{\mathbf{z}}$ variation, $N_{\mathbf{z}}$ primitives) to obtain a motion primitive library (MPL) $\Gamma_{\boldsymbol{\xi}_t}$ given by (Fig. 3):



(a) Variation in ω **(b)** Variation in $v_\mathbf{z}$

Fig. 3: Actions in $\mathbf{x}_\mathcal{B} - \mathbf{y}_\mathcal{B}$ plane at the multirotor state $\boldsymbol{\xi}_t$, $\gamma_{\boldsymbol{\xi}_t}^{jk}$ (blue), are generated using discretized velocity bounds obtained from the analysis in Sect. 2. The set of such primitives at each state is termed a motion primitive library (MPL) $\Gamma_{\boldsymbol{\xi}_t}$. MPLs are sampled in directions perpendicular ($\mathbf{x}_\mathcal{B}$) and parallel ($\mathbf{y}_\mathcal{B}, -\mathbf{y}_\mathcal{B}$) to the sensor scans with speeds bounded by $V_{\perp,\max}$ and $V_{\parallel,\max}$ respectively, see **(a)**. Variation in $\mathbf{z}_\mathcal{B}$ direction using a user-specified bound on vertical velocity, $V_\mathbf{z}$, yields the final action space shown in **(b)**. Dynamically feasible stopping trajectories $\gamma_{\boldsymbol{\xi}_t}^{\text{stop}}$ are available for each primitive (green) for safety (only one shown for brevity).

$$\Gamma_{\boldsymbol{\xi}_t} = \{\gamma_{\boldsymbol{\xi}_t}^{jk} \mid j \in [1, N_\omega], k \in [1, N_\mathbf{z}], \|[v_\mathbf{x}, v_\mathbf{y}]\| \leq V_{\max}, \|v_\mathbf{z}\| \leq V_\mathbf{z}, \|\omega\| \leq \Omega_{\max}\} \quad (3)$$

where, $\|\cdot\|$ denotes L2-norm of a vector, $V_{\max}$ and $V_\mathbf{z}$ are the bounds on speed in $\mathbf{x}_\mathcal{B} - \mathbf{y}_\mathcal{B}$ plane and $\mathbf{z}$ direction respectively, and $\Omega_{\max}$ is the bound on body yaw rate. For a given action discretization, the motion primitive $\gamma_{\boldsymbol{\xi}_t}^{jk}$ is an 8^{th} order polynomial in time with fixed start and end point velocities and unconstrained position at the end. The velocity at the end point, at time τ, follows by forward propagating unicycle kinematics using the current state and the action parameterization while higher order derivatives up to snap, endpoints are kept zero:

$$\dot{\boldsymbol{\xi}}_\tau = [v_\mathbf{x} \cos\psi, v_\mathbf{x} \sin\psi, v_\mathbf{z}, \omega]^\mathrm{T}, \ \psi = \omega\tau, \ \boldsymbol{\xi}_\tau^{(j)} = \mathbf{0}, \text{ for } j \in \{2,3,4\} \quad (4)$$

where $\boldsymbol{\xi}^{(j)}$ denotes the j^{th} time derivative of $\boldsymbol{\xi}$. The stopping trajectories at any $\boldsymbol{\xi}_t$ ($\gamma_{\boldsymbol{\xi}_t}^{\text{stop}}$, Fig. 3) can be sampled by keeping $\dot{\boldsymbol{\xi}}_t = \mathbf{0}$.

In contrast to the fixed duration (τ) primitives presented in [18], the duration for the primitive is kept minimum via a line search from the minimum possible duration (Δt_p) to the user-specified maximum duration ($\tau_{\max}$). The search terminates when the first dynamically feasible motion primitive is obtained. This dynamic feasibility check is based on pre-specified empirically observed bounds on linear acceleration and linear jerk $L2$-norms. This search achieves having the trajectory in the desired end point velocity $\dot{\boldsymbol{\xi}}_t$ in the minimum time possible from the current state.

3.2 Action Space for Fast Exploration

The action space for the proposed planner is a collection of MPLs, defined by (3), generated using linear velocities based on bounds obtained in Sect. 2, $V_{\perp,\max}$ and

$V_{\|,\max}$. The planner uses 6 MPLs to represent the action space, $\mathcal{A}_{\text{act}} = \{\Gamma^i_{\boldsymbol{\xi}_t} \mid i \in [1,6]\}$, and sets of upper bounds on linear velocities (Tab. 2) define each of these different MPLs. These MPLs include both high-speed actions for navigating the environment and actions that mimic steady-state conditions described Sect. 2.2. Later, in Sect. 5, we highlight effects on exploration performance due to these components, especially the high speed parallel and low speed perpendicular MPLs.

4 Action Selection

We formulate the action selection problem as a finite-horizon optimization seeking to maximize cumulative information gain [4], and build upon previous work [7, 8, 11] on robotic exploration using Monte Carlo tree search (MCTS).

Most MCTS-based planners follow four steps: selection, expansion, simulation playout, and backpropagation of statistics [2, 5]. Such planners usually construct a search tree iteratively by random rollout from a previously unexpanded node selected based on upper-confidence bounds for trees (UCT) [2]. Prior works [7, 8] have applied MCTS in planning for exploration using multirotors using a UCT-based selection policy, information gain rewards, and random simulation playout over a finite horizon. We extend this approach by adding considerations for model constraints into the node expansion phase of MCTS.

4.1 Information-Theoretic Exploration Objectives

Following a similar approach as our prior work [8], the planner optimizes an objective with two components: a local information reward based on Cauchy-Schwarz quadratic mutual information (CSQMI) [4], and a global reward for decrease the shortest path distance to points in the state space that are expected to provide significant information gain [8]. For any candidate action, $\gamma_{\boldsymbol{\xi}_t}$, we compute the local information gain $\mathcal{I}_\gamma$ over user-specified time intervals and treat the joint information gain as a reward for the MCTS planner. The distance reward serves to direct the robot toward unexplored and possibly distant regions of the environment once the local environment is exhausted of information causing the local information reward to decrease. Alternatively, designers might substitute competing routing and scheduling approaches [10] for the distance cost subject to with tradeoffs in computational cost and system design.

4.2 Safety at High Speeds

Given the action representation described in Sect. 3, we require the planner to ensure safety while expanding nodes. Specifically, the trajectory tracked by the controller should both respect constraints on the dynamics and remain in known free space for all time. To satisfy this condition, before sending any trajectory to the robot, we

require knowledge of a trajectory that will bring the robot to a stop—and potentially keep it there—afterward. As such, the robot will avoid collision, even if the planner fails to provide any output. If ever planning fails, the known stopping trajectory is sent to the robot, and the robot will continue to replan from a future reference point.

5 Results

This section describes hardware and simulation results for the proposed exploration approach. The simulation results evaluate performance in a warehouse-like environment which serves as a representative example of a large-scale exploration task. The hardware results demonstrate exploration at high speeds using a hexarotor platform under various degrees of clutter. Unless otherwise noted, the configuration of the robot for simulation matches the hardware.

5.1 Aerial Platform

Platform used for experiments is a hexarotor aerial robot (Fig. 7a) with a forward-facing depth camera for mapping (Realsense D435) with a 89.57° by 59.24° field of view and a range of $r_{\text{depth}} = 5.0\,\text{m}$. The robot itself weighs 55.37 N, has a thrust to weight ratio of 3.5, and has a diameter of 0.89 m from motor to motor. Limits on acceleration and jerk are set to $A_{\max} = 10\,\text{m/s}^2$ and $J_{\max} = 35\,\text{m/s}^3$ respectively, based on empirical data available for the platform. Unless otherwise stated, the planning horizon is kept at 4 seconds for all experiments. For both simulation and hardware experiments, mapping and planning run on a computationally constrained quad-core embedded CPU Gigabyte Brix 6500U. The robot obtains odometry estimates via a downward-facing camera and IMU using a monocular Visual-Inertial Navigation System (VINS-Mono [17]), previously used for high-speed teleoperation of a multirotor [18]. This state estimation component, only present for hardware experiments, is executed on a quad-core NVIDIA Tegra TX2 on-board the vehicle. Contrary to perfect state estimation in simulation experiments, for the hardware experiments the robot only has access to odometry for navigation and is susceptible to drift. For the purpose of this work, we will continue to emphasize the role of planning and speed in the exploration experiments and will comment briefly on ramifications of drift on outcomes. Future iterations of this platform will seek to combine a local mapping strategy [8] with a complete SLAM system.

5.2 Simulated Exploration of a Warehouse Environment

The simulations demonstrate exploration of a large warehouse environment (pictured in Fig. 4). These trials are repeated for three system configurations which vary the motion primitive library and the computational setting:

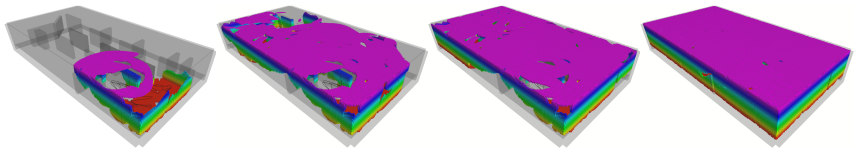
Table 2: Motion primitive libraries used to construct action space using Eq. 3 from Sect. 3 and the bounds obtained after applying analysis presented in Sect. 2. The vertical velocity bound ($V_{\mathbf{z}}$) and the speed bound in $\mathbf{x}_{\mathcal{B}} - \mathbf{y}_{\mathcal{B}}$ plane ($V_{\max}$) are kept at $0.3\,\mathrm{m/s}$ and $4.0\,\mathrm{m/s}$ respectively. The total number of primitives for a MPL is $N_{\text{prim}} = N_{\omega} \cdot N_{\mathbf{z}}$.

(a) Large Library

ID	Type	Max. Speed	Dir.	N_{ω}	$N_{\mathbf{z}}$	N_{prim}
1	Yaw	0	ψ	1	1	1
2	$\perp$	$V_{\perp,\max}$	$\mathbf{x}_{\mathcal{B}}$	9	5	45
3	$\perp$	$V_{\max}$	$\mathbf{x}_{\mathcal{B}}$	9	5	45
4	$\parallel$	$V_{\max}$	$\mathbf{y}_{\mathcal{B}}$	9	5	45
5	$\parallel$	$V_{\max}$	$-\mathbf{y}_{\mathcal{B}}$	9	5	45
6	Z	$V_{\mathbf{z}}$	$\mathbf{z}$	1	5	5

(b) Minimal Library

ID	Type	Max. Speed	Dir.	N_{ω}	$N_{\mathbf{z}}$	N_{prim}
1	Yaw	0	ψ	1	1	1
2	$\perp$	$V_{\perp,\max}$	$\mathbf{x}_{\mathcal{B}}$	3	3	9
3	$\perp$	$V_{\max}$	$\mathbf{x}_{\mathcal{B}}$	3	3	9
4	$\parallel$	$V_{\max}$	$\mathbf{y}_{\mathcal{B}}$	3	3	9
5	$\parallel$	$V_{\max}$	$-\mathbf{y}_{\mathcal{B}}$	3	3	9
6	Z	$V_{\mathbf{z}}$	$\mathbf{z}$	1	3	3

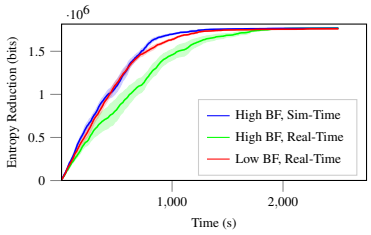


(a) $t = 100\,\mathrm{s}$ **(b)** $t = 500\,\mathrm{s}$ **(c)** $t = 1000\,\mathrm{s}$ **(d)** $t = 1500\,\mathrm{s}$

Fig. 4: Occupied map at different stages of exploration of a simulated three-dimensional $60\,\mathrm{m} \times 30\,\mathrm{m} \times 11\,\mathrm{m}$ warehouse environment used for experiments. The map is colored based on Z height.

- **High branching factor (BF), Sim-Time**: The planner uses the large motion primitive library (Tab. 2a) for exploration. The simulation and clock pause at the end of each planning round until the MCTS planner completes a user-defined number (3000) of iterations. The simulation time then does not include this additional time spent in planning.
- **High BF, Real-Time**: The planner uses the large motion primitive library for exploration, but the simulation of the multirotor runs in real time. The planner runs in an anytime fashion on a computer comparable to the on-board computer used for flight experiments presented in Sect. 5.3 while simulators for the camera and dynamics run on a separate computer.
- **Low BF, Real-Time**: The planner uses the minimal motion library (Tab. 2b) for exploration and the computational configuration is the same as the above.

These experiments first establish baseline performance (High BF, Sim-Time) given access to a variety of motion primitives and a relatively large amount of planning time. The latter two configurations demonstrate online planning in a configuration that closely matches the hexarotor platform used in this paper. These experiments seek to demonstrate the role of computational constraints in design of the motion primitive library. For each configuration, we provide several trials, one for each of 5 given starting locations.



Time to reach (s):	25%	50%	75%	90%
High BF, Sim-Time	**157.7**	**365.7**	**602.5**	**788.3**
High BF, Real-Time	267.4	501.6	866.5	1183.2
Low BF, Real-Time	193.2	382.5	604.8	901.1

Fig. 5: (left) Entropy reduction vs. time for each of the simulation trials. Transparent patches show standard-error. (right) Average time to reach fractions of the maximum entropy reduction over all trials (1.766×10^6 bits). While the High BF, Sim-Time case dominates in terms of entropy reduction, the Low BF, Real-Time case is able to provide similar performance. Note that the different configuration are described at the beginning of Sect. 5.2 and that BF denotes branching factor.

Each trial lasts 2500 seconds which provides ample time to explore the entire environment. For all trials, the perpendicular velocity $V_{\perp,\max}$ is further limited to $1.25\,\mathrm{m/s}$, below the value of $1.77\,\mathrm{m/s}$ obtained in Sect. 2 which we find admits forward motion at a constant speed given the trajectory generation approach used for motion primitive design. The maximum speed is set to more than three times greater at $V_{\max} = 4.0\,\mathrm{m/s}$.

Figure 5 summarizes exploration performance for each experiment. The high branching factory Sim-Time case which has access to extra planning time performs at least as well or better than the other configurations in terms of entropy reduction. However, the configuration with same motion primitive library and real time is significantly impaired and requires between approximately 1.3 to 1.8 times as long to reach reported levels of entropy reduction. The lower branching factor case matches the first configuration much more closely. As such, this latter configuration is appropriate for deployment on the compute-constrained hexarotor platform.

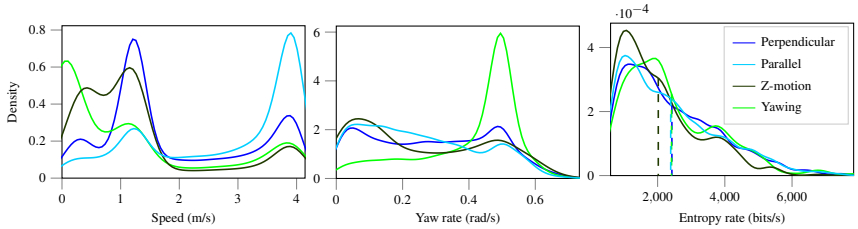


Fig. 6: Exploration performance by action. Plots provide estimates of probability densities (via kernel density estimation) for speed, yaw rate, and entropy rate conditional on the type of action (Tab. 2) being executed by the robot. All densities are also conditional on a significant entropy reduction rate (greater than 600 bits/s) in order to emphasize performance characteristics for actions that directly contribute to the map rather than traversal of known space or time periods after exploration is effectively complete. Note that, even though the option can has access to high-speed motions perpendicular to frontiers, entropy reduction for perpendicular actions occurs primarily at lower speeds (1.25m/s) in accordance with the analysis in Sect. 2.

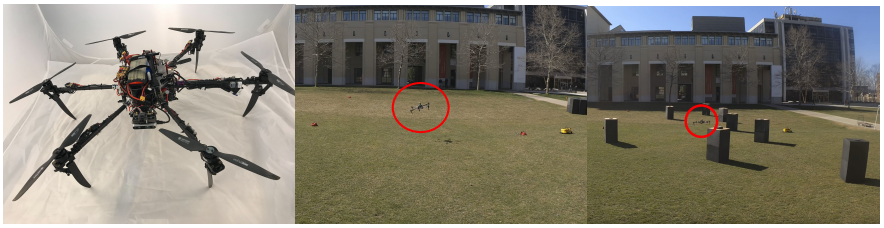
Table 3: Performance statistics for the high branching factor, Sim-Time simulation study. Unless otherwise stated, rates refer to average performance over time-periods when tracking a given action type. All statistics include any time the robot is tracking a given action, except for the entropy reduction rate in the last row, which is conditional on a significant entropy reduction rate (greater than 600 bits/s). Best (or nearly equivalent) values are bolded.

Actions:	$\perp$	$\parallel$	**Z**	**Yaw**	**Stop**
Selection frequency	0.343	**0.479**	0.089	0.088	0.001
Total entropy red, norm.	**0.40**	**0.41**	0.06	0.12	0.01
Average speed (m/s)	2.163	**2.778**	0.959	1.114	1.611
Average yaw rate (rad/s)	0.286	0.234	0.170	**0.381**	0.080
Entropy red. rate (bits/s):	**2425**	**2389**	2020	**2414**	1283

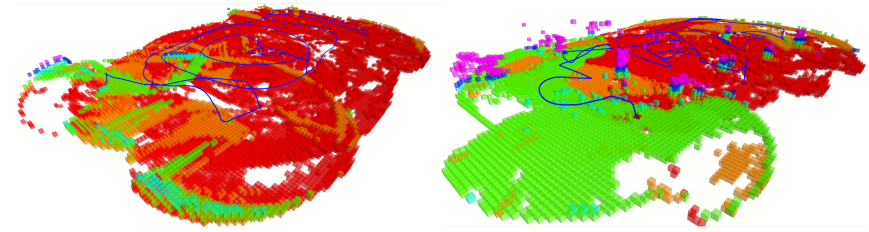
In addition to being able to explore an environment rapidly and completely, we characterize the roles of the motion primitive actions in the exploration task. Figure 6 shows density estimates plots for speeds, yaw rates, and entropy rate labelled by the type of action selected by the MCTS planner for execution for periods when the entropy reduction rate is significant (greater than $600\,\text{bits}/\text{s}$) so as to separate exploration actions from traversal of the environment and time periods after exploration is effectively complete. This threshold corresponds to a knee point in the overall distribution of entropy reduction rates: 94.4% of all entropy reduction occurs above this threshold but during only 27.9% of time during the trials. Interestingly, the time rate of entropy reduction is largely consistent across action types. However, as expected, motions perpendicular to the frontier primarily contribute to entropy reduction at reduced speed despite both low-speed and high-speed primitives being available. Table 3 shows provides further statistics for the different kinds of actions. Even though entropy reduction rates are similar across actions when the entropy reduction rate is significant, the planner selects motions parallel and perpendicular to frontiers most frequently, and those actions account for more than 80% of all entropy reduction.

5.3 Hardware Experiments under Varied Conditions

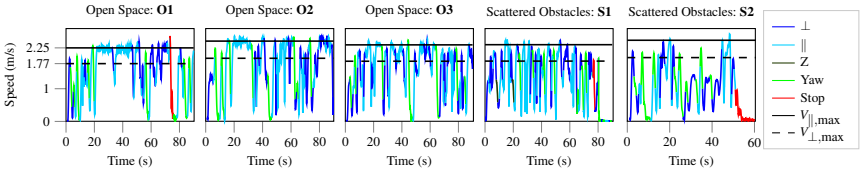
Real-world autonomous exploration experiments are conducted using the aerial platform described in Sect. 5.1 (Fig. 7a) in two outdoor scenarios: (1) Open space (Fig. 7b), and (2) Scattered obstacles (Fig. 7c). Total exploration duration is limited to 90 seconds to minimize the effects of state estimation drift on the resulting map. During each scenario, the robots explores while confined to $12\,\text{m} \times 24\,\text{m} \times 2\,\text{m}$ bounding box. The robot starts at the same position in the bounding box for each trial in both scenarios. The bounds on the speed for perpendicular ($V_{\perp,\max}$) and parallel ($V_{\parallel,\max}$) motions, are set at $1.25\,\text{m}/\text{s}$ and $2.25\,\text{m}/\text{s}$ respectively. The explored maps and robot trajectory for two experiments, one from each scenario, are shown in Figs. 7d,e. Speeds achieved by the vehicle during the experiments are shown in



(a) Multirotor with depth sensor **(b)** Open space scenario **(c)** Obstacles scenario



(d) Open space scenario exploration after 90 s **(e)** Obstacles scenario exploration after 90 s



(f) Speeds attained during outdoor exploration experiments

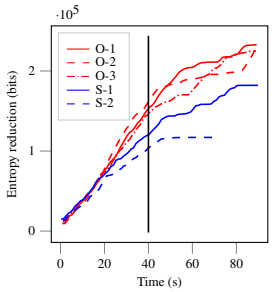
Fig. 7: **(a)** Multirotor used for hardware experiments in two real world scenarios: **(b)** open space and **(c)** space with scattered obstacles. **(d)**, **(e)**, and **(f)** show the explored maps (color gradient based on Z height), overall paths, and speeds attained by the robot after 90 s of 3D exploration using the proposed exploration approach.[1]

[1] Video: https://youtu.be/YXt4yiTpOAc

Fig. 7f Figure 8 provides plots of reduction of map entropy as well as summary statistics. Even though the trials were relatively short, the odometry often drifted significantly by the end This drift likely contributed to the robot getting stuck behind an obstacle during the trial **S2**. For this reason, we only use the first for the first 40 seconds of each trial when computing summary statistics unless otherwise noted.

As shown in Fig. 7f, the odometry system reports that the robot reaches and slightly exceeds the maximum desired reference speed[2] in each trial, primarily while executing motions parallel to the frontier. Figure 7d shows a particularly notable example of this behavior where the robot executed an outward spiraling motion soon after the start of the trial.

[2] The robot may exceed reference speeds due to error in tracking the position reference because of environment disturbances and inaccuracies in the system model.



Trials:	O1	O2	O3	S1	S2
Entropy red. at 40s (bits$\times 10^5$)	1.53	1.61	1.46	1.2	1.04
Entropy red. final (bits$\times 10^5$)	2.32	2.27	2.25	1.82	1.16
Average speed (m/s)	1.50	1.56	1.56	1.42	0.98
Average yaw rate (rad/s)	0.39	0.39	0.33	0.33	0.31
Max. speed (m/s)	2.39	2.38	2.38	2.40	2.29
Max. yaw rate (rad/s)	0.66	0.59	0.55	0.60	0.63

Fig. 8: Entropy reduction for hardware trials and summary statistics. Except for the final entropy reduction, all statistics are computed over the first 40 second of each trial (shown by the black bar in the entropy reduction plot).

6 Conclusion and Future Work

We have investigated how the dynamics of aerial platforms and the geometry of common sensors impact selection of control actions in robotic exploration. We have obtained bounds on the velocity for different kinds of motions and applied this analysis to the design of a motion primitive library and information-based planner suitable for rapid exploration with aerial robots. We have demonstrated this approach both in simulated exploration of a large warehouse environment and in outdoor experiments with only on-board computation. This system produces interesting and intuitive motions in practice such as outward spirals for rapid coverage of open space. Further, the experimental results demonstrate speeds exceeding $2.25\,\mathrm{m/s}$ for both open and cluttered environments, which matches and slightly exceeds prior state-of-the-art results [6].

The analysis illuminates competing directions for improvements to speed and entropy reduction performance. Decreasing planning time and latency, such as by improved efficiency or reactive planning [6] or simply increasing sensor range, can improve speeds moving perpendicular to frontiers which may be especially important in highly cluttered environments where motion parallel to frontiers is not viable. At the same time, the ability to safely and rapidly navigate known environments is also tightly coupled to exploration performance both for motions parallel to frontiers and when traversing known space to new unexplored regions. Thus, improvements to state-estimation, mapping, and planning under uncertainty are also critical to increasing speed and entropy reduction rates in exploration.

Acknowledgements The authors would like to thank Xuning Yang for help in discussion and implementation of forward arc motion primitives used in this work.

References

1. Bircher, A., Kamel, M., Alexis, K., Oleynikova, H., Siegwart, R.: Receding horizon path planning for 3D exploration and surface inspection. Auton. Robots **42**(2), 291–306 (2018)
2. Browne, C., Powley, E., Whitehouse, D., Lucas, S., Cowling, P.I., Rohlfshagen, P., Tavener, S., Perez, D., Samothrakis, S., Colton, S.: A survey of Monte Carlo tree search methods. IEEE Trans. on Comput. Intell. and AI in Games **4**(1), 1–43 (2012)
3. Charrow, B., Kahn, G., Patil, S., Liu, S., Goldberg, K., Abbeel, P., Michael, N., Kumar, V.: Information-theoretic planning with trajectory optimization for dense 3D mapping. In: Proc. of Robot.: Sci. and Syst. Rome, Italy (2015)
4. Charrow, B., Liu, S., Kumar, V., Michael, N.: Information-theoretic mapping using Cauchy-Schwarz quadratic mutual information. In: Proc. of the IEEE Intl. Conf. on Robot. and Autom. Seattle, WA (2015)
5. Chaslot, G.: Monte-Carlo tree search. Ph.D. thesis, Universiteit Maastricht (2010)
6. Cieslewski, T., Kaufmann, E., Scaramuzza, D.: Rapid exploration with multi-rotors: A frontier selection method for high speed flight. In: Proc. of the IEEE/RSJ Intl. Conf. on Intell. Robots and Syst. Vancouver, Canada (2017)
7. Corah, M., Michael, N.: Distributed matroid-constrained submodular maximization for multi-robot exploration: theory and practice. Auton. Robots **43**(2), 485–501 (2019)
8. Corah, M., O'Meadhra, C., Goel, K., Michael, N.: Communication-efficient planning and mapping for multi-robot exploration in large environments. IEEE Robot. Autom. Letters **4**(2), 1715—1721 (2019)
9. Janson, L., Hu, T., Pavone, M.: Safe motion planning in unknown environments: Optimality benchmarks and tractable policies. In: Proc. of Robot.: Sci. and Syst. Pittsburgh, PA (2018)
10. Kulich, M., Faigl, J., Přeučil, L.: On distance utility in the exploration task. In: Proc. of the IEEE Intl. Conf. on Robot. and Autom. Shanghai, China (2011)
11. Lauri, M., Ritala, R.: Planning for robotic exploration based on forward simulation. Robot. Auton. Syst. **83**, 15–31 (2016)
12. Liu, S., Watterson, M., Tang, S., Kumar, V.: High speed navigation for quadrotors with limited onboard sensing. In: Proc. of the IEEE Intl. Conf. on Robot. and Autom. Stockholm, Sweden (2016)
13. Mahony, R., Kumar, V., Corke, P.: Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor. IEEE Robot. Autom. Mag. **19**(3), 20–32 (2012)
14. Mellinger, D., Kumar, V.: Minimum snap trajectory generation and control for quadrotors. In: Proc. of the IEEE Intl. Conf. on Robot. and Autom. Shanghai, China (2011)
15. Michael, N., Mellinger, D., Lindsey, Q., Kumar, V.: The grasp multiple micro-uav testbed. IEEE Robot. Autom. Mag. **17**(3), 56–65 (2010)
16. Michael, N., Shen, S., Mohta, K., Mulgaonkar, Y., Kumar, V., Nagatani, K., Okada, Y., Kiribayashi, S., Otake, K., Yoshida, K., Ohno, K., Takeuchi, E., Tadokoro, S.: Collaborative mapping of an earthquake-damaged building via ground and aerial robots. J. Field Robot. **29**(5), 832–841 (2012)
17. Qin, T., Li, P., Shen, S.: Vins-mono: A robust and versatile monocular visual-inertial state estimator. IEEE Trans. Robotics **34**(4), 1004–1020 (2018). DOI 10.1109/TRO.2018.2853729
18. Spitzer, A., Yang, X., Yao, J., Dhawale, A., Goel, K., Dabhi, M., Collins, M., Boirum, C., Michael, N.: Fast and agile vision-based flight with teleoperation and collision avoidance on a multirotor. In: Proc. of the Intl. Sym. on Exp. Robot. Springer, Buenos Aires, Argentina (2018). To be published
19. Stachniss, C., Grisetti, G., Burgard, W.: Information gain-based exploration using rao-blackwellized particle filters. In: Proc. of Robot.: Sci. and Syst. (2005)
20. Yamauchi, B.: A frontier-based approach for autonomous exploration. In: Proc. of the Intl. Sym. on Comput. Intell. in Robot. and Autom. Monterey, CA (1997)
21. Yang, X., Agrawal, A., Sreenath, K., Michael, N.: Online adaptive teleoperation via motion primitives for mobile robots. Auton. Robots (2018)