Autonomous Cave Surveying with an Aerial Robot

Wennie Tabib, Kshitij Goel, John Yao, Curtis Boirum, and Nathan Michael

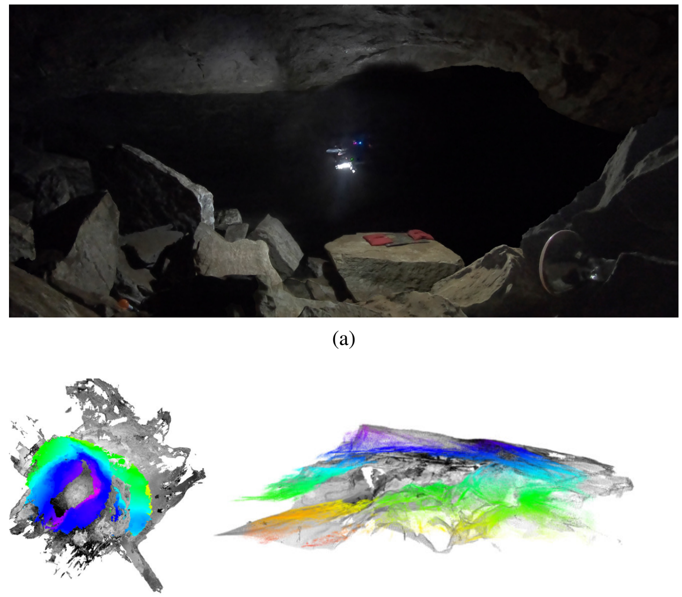
***Abstract*—This paper presents a method for cave surveying in complete darkness with an autonomous aerial vehicle equipped with a depth camera for mapping, downward-facing camera for state estimation, and forward and downward lights. Traditional methods of cave surveying are labor-intensive and dangerous due to the risk of hypothermia when collecting data over extended periods of time in cold and damp environments, the risk of injury when operating in darkness in rocky or muddy environments, and the potential structural instability of the subterranean environment. Robots could be leveraged to reduce risk to human surveyors, but undeveloped caves are challenging environments in which to operate due to low-bandwidth or nonexistent communications infrastructure. The potential for communications dropouts motivates autonomy in this context. Because the topography of the environment may not be known *a priori*, it is advantageous for human operators to receive real-time feedback of high-resolution map data that encodes both large and small passageways. Given this capability, directed exploration, where human operators transmit guidance to the autonomous robot to prioritize certain leads over others, lies within the realm of the possible.**

Few state-of-the-art, high-resolution perceptual modeling techniques quantify the time to transfer the model across low bandwidth radio communications channels which have high reliability and range but low bandwidth. To bridge this gap in the state of the art, this work compactly represents sensor observations as Gaussian mixture models and maintains a local occupancy grid map for a motion planner that greedily maximizes an information-theoretic objective function. The methodology is extensively evaluated in long duration simulations on an embedded PC and deployed to an aerial system in Laurel Caverns, a commercially owned and operated cave in Southwestern Pennsylvania, USA. A video of the simulation and hardware results is available at https://youtu.be/iwi3p7IENjE.

***Index Terms*—aerial system, perceptual modeling, exploration, autonomy**

I. Introduction

THE process of cave surveying, which consists of marking stations and measuring the distances between them, has changed relatively little since the 19th century [1, p. 1532]. Gunn [1] predicts that advancements in technology may fundamentally change this method in the 21st century. While substantial advancements in sensing, 3D reconstruction, and autonomy have been made within the last decade, these advancements have not propogated to cave surveying. This paper addresses this gap in the state of the art through the development and rigorous testing of an autonomous aerial system that explores and maps caves. Fig. 1a illustrates the aerial system flying through the Dining Room of Laurel Caverns. A GMM map is created in real-time and onboard the vehicle during a hardware trial and resampled to produce the pointcloud, colored from red to purple according to z-height, shown in Figs. 1b and 1c. A FARO[1] scan of the environment is used as ground truth and shown in grey.



(a) (b) (c)

Fig. 1: (a) An autonomous aerial system explores the Dining Room of Laurel Caverns. (b) A top-down view of the map built during an exploration trial colored from red to purple to denote z-height. Survey-grade FARO LiDAR data (intensity data shown in grey) provides ground truth and has dimensions approximately $30\,\mathrm{m} \times 29\,\mathrm{m} \times 12\,\mathrm{m}$. (c) illustrates a cross-section of the map produced during exploration. The terrain is highly unstructured and consists of large boulders and rocks caused by breakdown.

Cave surveying is challenging because surveyors remain still for extended periods of time and are exposed to water, cool air, and rock which can lead to hypothermia [2]. More worrisome yet is the potential for getting lost or trapped in a cave [3] because specialized training is required to perform challenging extractions where a caver may be physically incapacitated. The Barbara Schomer Cave Preserve in Clarion County, PA, has a cave that is particularly challenging to survey for two reasons: (1) the small size of the passages (typical natural passages are $0.75\,\mathrm{m}$ high and $0.75\,\mathrm{m}$ wide), and (2) the mazelike nature of the passages that are estimated to be $60\,\mathrm{km}$ in length [4, p. 10]. 50 cave surveyors have been involved in 30 trips to survey a total of $2522\,\mathrm{m}$ of passage (B. Ashbrook, personal communication, March 1, 2020). Each trip is 4-5 hours in duration and there are currently over 90 unexplored leads in the cave. Fig. 2 illustrates an excerpt of the current working map of the cave and Fig. 3 illustrates a caver sketching for the cave survey.

The authors are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 15213 USA e-mail: {wtabib, kgoel1, johnyao, cboirum, nmichael}@andrew.cmu.edu.

[1] A FARO is a survey grade 3D laser that emits pointclouds with color. https://www.faro.com/

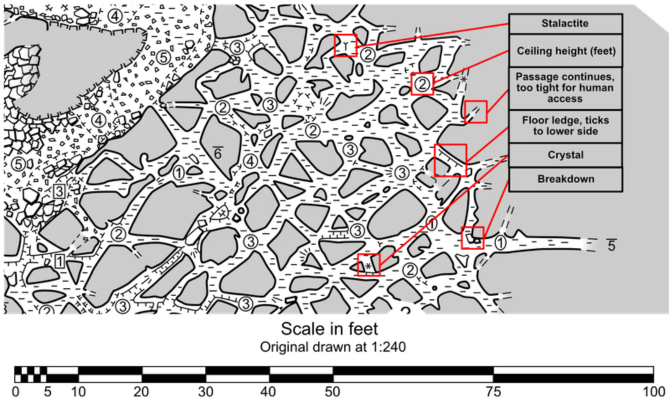


Fig. 2: An excerpt of the current working map for the cave on the Barbara Schomer Cave Preserve in Clarion County, PA. The map is encoded with terrain features. Note the passages marked too tight for human access in the top-right of the image. Aerial robots could be deployed to these areas to collect survey data. Image courtesy of B. Ashbrook.

Fig. 3: A caver sketches a passageway to produce content for the map shown in Fig. 2. Image courtesy of H. Wodzenski. J. Jahn pictured.

Robotic operations in undeveloped subterranean environments are challenging due to limited or nonexistent communications infrastructure that increases the risk of failure from data dropouts. Dang et al. [5] find that autonomy is necessary when operating underground because reliable high-bandwidth wifi connections are impossible to maintain after making turns. Low-frequency radio is commonly used in subterranean environments because it penetrates rock better than high-frequency radios (like walkie-talkies) [6], but the disadvantage is that radio has low-bandwidth and data dropouts increase as range increases. Given these constraints, cave surveying robots must be autonomous to robustly operate in the presence of communication dropouts and represent the environment in a way that is both high-resolution and compact so that data can be transmitted over low-bandwidth connections to operators. The ability to transmit high-resolution maps is desirable because human operators can direct exploration towards a particular lead of interest if the robot enters a passageway with multiple outgoing leads.

Gaussian mixture models (GMMs) are ideal for perceptual modeling in the cave survey context because they represent depth information compactly. This paper seeks to address the problem of constrained communications bandwidth robotic exploration by proposing an autonomous system that leverages these compact perceptual models to transmit high-resolution information about the robot's surroundings over low-bandwidth connections. Depth sensor observations are encoded as GMMs and used to maintain a consistent local occupancy grid map. A motion planner selects smooth and continuous trajectories that greedily maximize an information-theoretic objective function.

This work presents an extended version of prior work [7] introducing all contributions:

1) a method for real-time occupancy reconstruction from GMMs with LiDAR sensor observations,
2) an information-theoretic exploration system that leverages the occupancy modeling technique, and
3) evaluation of the exploration system in simulation and real-world experiments.

This manuscript presents the following additional contributions:

1) an extension of the real-time occupancy reconstruction from GMMs that can accomodate limited field of view depth cameras,
2) a motion planning framework that performs well for both LiDAR and limited field-of-view depth cameras, and
3) extensive evaluation of the exploration system in simulation and on hardware in a cave.

The paper is organized as follows: Section II surveys related work, Sections III, IV and V describe the proposed methodology, Section VI presents the experimental design and results, and Section VII concludes with a discussion of the limitations and future work.

II. Related Work

Exploration systems to date have largely left questions about the cost of transmitting data to operators unaddressed. However, it is becoming increasingly important for robots to transmit sensor and perceptual modeling data compactly to enable collaborative field operations with humans in undeveloped environments that lack a robust communications infrastructure. Planetary exploration is an application for which data rates are severely limited. For example, the Viking 1 Lander had a maximum data rate of 16 kilobits per second [8] which coupled with the large distance between Earth and Mars meant that operators had to wait 19 minutes before receiving the first images of the Martian surface [9].

Beyond space exploration, Murphy et al. [10] find that semiautonomy in the form of autonomous navigation coupled with human-assisted perception is preferred over total autonomy for robots operating in subterranean domains given that teleoperation was required to recover the autonomous 700 kg ATV-type Groundhog vehicle when it became stuck in a mine. However, human-assisted perception is predicated on the ability to transfer perceptual information to human operators. From a cave survey perspective, a significant challenge is the ability to access leads and passageways that may be unreachable to humans as shown in Fig. 2. If the lead contains a passageway with several outgoing leads, it is useful for a human operator to have the ability to direct the exploration in one direction over others (e.g., one passageway is larger than another or contains interesting features).

However, few works consider methods to map caves with aerial or ground robotic systems. Kaul et al. [11] develop the bentwing robot to produce maps of cave environments. The approach, which requires a pilot to remotely operate the vehicle, may be challenging due to the cognitive load required to stabilize attitude and position simultaneously. The bentwing uses a rotated 2D laser scanner to collect data that are post-processed into a globally consistent map. Similarly, Tabib and Michael [12] develop a simultaneous localization and mapping strategy that represents sensor observations using GMMs and produce maps via post-processing from data collected from an aerial system equipped with a 3D LiDAR. In contrast, this work produces a map in real-time and onboard the autonomous robot suitable for information-theoretic planning that is sufficiently compact to be transmitted to operators over low-bandwidth connections and used for human-assisted perception.

The DEep Phreatic THermal eXplorer (DEPTHX) vehicle was developed to explore and characterize the biology of the Sistema Zacatón cenotes, or underwater sinkholes, as an analog mission for the search for life underneath Europa's ice [13]. The probe combines LiDAR to map above the water table with sonar to map phreatic zones. The authors employ a Deferred Reference Octree data structure to represent the environment and mitigate the memory required to represent the underlying evidence grid. The cost of transmitting data to enable information sharing with human operators is not considered.

While many exploration approaches have leveraged voxel-based occupancy modeling strategies for information-theoretic planning [14], the large memory demands of using occupancy grid maps remains. The Normal Distribution Transform Occupancy Map (NDT-OM) mitigates this limitation somewhat [15] by encoding a Gaussian density into occupied voxels with the reasononing that larger voxels may be used as compared to a traditional occupancy grid map. However, for exploration in large environments, this technique suffers from the same drawbacks as the occupancy grid map.

To overcome this limitation, this work builds upon prior work by OMeadhra et al. [16] that compresses sensor observations as GMMs for the purpose of occupancy reconstruction, by developing real-time local occupancy mapping for information-theoretic planning using both 360° and limited FoV sensor models. Because sensor observations are stored as GMMs, local occupancy maps are constructed on-the-fly and only GMMs need to be transmitted between the robot and operator which results in a much smaller memory footprint. Another advantage is that occupancy grid maps of variable resolution may be used to plan paths at various resolutions (though this is not demonstrated in this work).

While GMMs have been used for compact perceptual modeling [17], occupancy modeling [16], and multi-robot exploration [18], these works do not demonstrate real-time operation. Our prior work [7] addresses this gap in the state of the art by proposing an exploration system that leverages

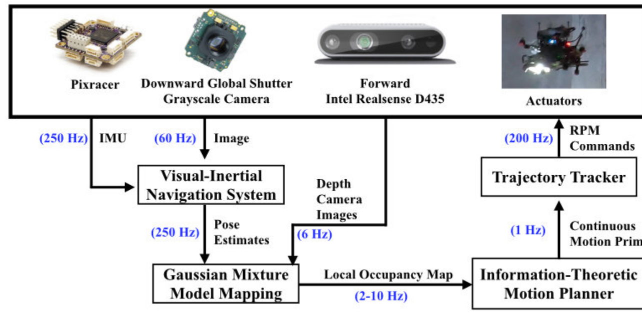


Fig. 4: Overview of the autonomous exploration system presented in this work. Using pose estimates from a visual-inertial navigation system (Section VI-C1) and depth camera observations, the proposed mapping method (Section IV-A and Section IV-B) builds a memory-efficient continuous approximate belief representation of the environment while creating local occupancy grid maps in real-time. A motion primitives-based information-theoretic planner (Section V) uses this local occupancy map to generate snap-continuous forward-arc motion primitive trajectories that maximize the information gain over time.

GMMs for real-time 3D information theoretic planning and perceptual modeling using a 360° field of view (FoV) sensor on computationally constrained platforms. The choice of 360° sensor field of view (FoV) has implications for mapping, planning, and hardware design so this paper extends the prior work by considering both 360° and limited FoV depth sensors.

III. Overview

The proposed exploration system consists of mapping, information-theoretic planning, and a monocular visual-inertial navigation system (Fig. 4). A brief review of GMMs is detailed in Section IV-A. Section IV-B develops the GMM-based local occupancy grid mapping strategy used by the planning approach to generate continuous trajectories that maximize an information-theoretic objective (Section V).

IV. Mapping

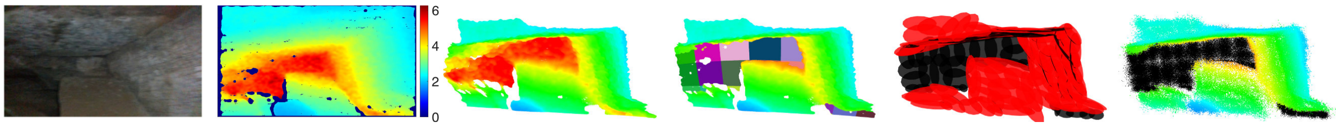
A. Gaussian Mixture Models for Perception

The approach leverages GMMs to compactly encode sensor observations for transmission over low-bandwidth communications channels. The GMM provides a generative model of the sensor observations from which occupancy may be reconstructed by resampling from the distribution and raytracing through a local occupancy grid map. Formally, the GMM is a weighted sum of M Gaussian probability density functions (PDFs). The probability density of the GMM is expressed as

$$p(\boldsymbol{x}|\boldsymbol{\Theta}) = \sum_{m=1}^{M} \pi_m \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_m, \boldsymbol{\Lambda}_m)$$

where $p(\boldsymbol{x}|\boldsymbol{\Theta})$ is the probability density for the D-dimensional random variable $\boldsymbol{x}$ and is parameterized by $\boldsymbol{\Theta} = \{\pi_m, \boldsymbol{\mu}_m, \boldsymbol{\Lambda}_m\}_{m=1}^{M}$. $\pi_m \in \mathbb{R}$ is a weight such that $\sum_{m=1}^{M} \pi_m = 1$ and $0 \leq \pi_m \leq 1$, $\boldsymbol{\mu}_m$ is a mean, and $\boldsymbol{\Lambda}_m$ is a covariance matrix for the m^{th} D-dimensional Gaussian probability density function of the distribution. The multivariate probability density for $\boldsymbol{x}$ is written as

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Lambda}_i) = \frac{|\boldsymbol{\Lambda}_i|^{-1/2}}{(2\pi)^{D/2}} \exp\Big(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^T \boldsymbol{\Lambda}_i^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_i)\Big).$$



(a) Color Image (b) Depth Image (c) PointCloud (d) Free Space Windows (e) GMM (f) Resampled Data

Fig. 5: Overview of the approach to transform a sensor observation into free and occupied GMMs. (a) A color image taken onboard the robot exploring Laurel Caverns. (b) A depth image corresponding to the same view as the color image with distance shown as a heatmap on the right hand side (in meters). (c) illustrates the point cloud representation of the depth image. (d) In the mapping approach, points at a distance smaller than a user-specified max range r_d (in this case $r_d = 5\,\mathrm{m}$) are considered to be occupied, and a GMM is learned using the approach detailed in Section IV-A1. Points at a distance further than r_d are considered free, normalized to a unit vector, and projected to r_d. The free space points are projected to image space and windowed using the technique detailed in Section IV-A2 to decrease computation time. Each window is shown in a different color. (e) The GMM representing the occupied-space points is shown in red and the GMM representing the free space points is shown in black. Sampling 2×10^5 points from the distribution yields the result shown in (f). The number of points to resample is selected for illustration purposes and to highlight that the resampling process yields a map reconstruction with an arbitrary number of points.

In this work, a depth observation taken at time t and consisting of N points, $\mathcal{Z}_t = \{\boldsymbol{z}_t^1, \ldots, \boldsymbol{z}_t^n, \ldots, \boldsymbol{z}_t^N\}$, is used to learn a GMM. Estimating optimal GMM parameters $\boldsymbol{\Theta}$ remains an open area of research [19]. This work employs the Expectation Maximization (EM) algorithm to solve the maximum-likelihood parameter estimation problem, which is guaranteed to find a local maximum of the log likelihood function [20]. To make the optimization tractable, EM introduces latent variables $\mathbf{C} = \{c_{nm}\}$ for each point $\boldsymbol{z}_t^n$ and cluster m and iteratively performs two steps: expectation (E) and maximization (M) [20, 21, 22].

The E step calculates the expected value of the complete-data log-likelihood $\ln p(\mathcal{Z}_t, \mathbf{C}|\boldsymbol{\Theta})$ with respect to the unknown variables $\mathbf{C}$ given the observed data $\mathcal{Z}_t$ and current parameter estimates $\boldsymbol{\Theta}^i$, which is written as $E[\ln p(\mathcal{Z}_t, \mathbf{C}|\boldsymbol{\Theta})|\mathcal{Z}_t, \boldsymbol{\Theta}^i]$ [21]. This amounts to evaluating the posterior probability, β_{nm}, using the current parameter values $\boldsymbol{\Theta}^i$ (shown in Eq. (1)) [20]

$$\beta_{nm} = \frac{\pi_m \mathcal{N}(\boldsymbol{z}_t^n | \boldsymbol{\mu}_m^i, \boldsymbol{\Lambda}_m^i)}{\sum_{j=1}^{M} \pi_j \mathcal{N}(\boldsymbol{z}_t^n | \boldsymbol{\mu}_j^i, \boldsymbol{\Lambda}_j^i)}, \tag{1}$$

where β_{nm} denotes the responsibility that component m takes for point $\boldsymbol{z}_t^n$. The M step maximizes the expected log-likelihood using the current responsibilities, β_{nm}, to obtain updated parameters, $\boldsymbol{\Theta}^{i+1}$ via the following:

$$\boldsymbol{\mu}_m^{i+1} = \sum_{n=1}^{N} \frac{\beta_{nm} \boldsymbol{z}_t^n}{\sum_{n=1}^{N} \beta_{nm}} \tag{2}$$

$$\boldsymbol{\Lambda}_m^{i+1} = \sum_{n=1}^{N} \frac{\beta_{nm} (\boldsymbol{z}_t^n - \boldsymbol{\mu}_m^{i+1})(\boldsymbol{z}_t^n - \boldsymbol{\mu}_m^{i+1})^T}{\sum_{n=1}^{N} \beta_{nm}} \tag{3}$$

$$\pi_m^{i+1} = \frac{\sum_{n=1}^{N} \beta_{nm} \boldsymbol{x}_n}{\sum_{n=1}^{N} \beta_{nm}}. \tag{4}$$

Every iteration of EM is guaranteed to increase the log likelihood and iterations are performed until a local maximum of the log likelihood is achieved [20].

The E step is computationally expensive because a responsibility β_{nm} is calculated for each cluster m and point $\boldsymbol{z}_t^n$, which amounts to NM responsibility calculations. In the M step, every parameter must be updated by iterating over all N samples in the dataset. In practice, a responsibility matrix $\mathbf{B} \in \mathbb{R}^{N \times M}$ is maintained whose entries consist of the β_{nm} to estimate the parameters $\boldsymbol{\Theta}$.

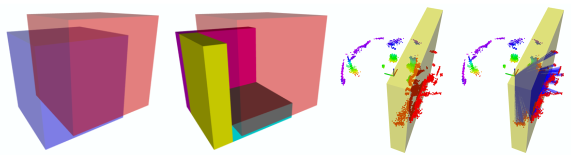
Following the work of OMeadhra et al. [16], distinct occupied $\mathcal{G}(\boldsymbol{x})$ (detailed in Section IV-A1) and free $\mathcal{F}(\boldsymbol{x})$ (detailed in Section IV-A2) GMMs are learned to compactly represent the density of points observed in the environment (Fig. 5). The process by which $\mathcal{F}(\boldsymbol{x})$ and $\mathcal{G}(\boldsymbol{x})$ are created is illustrated in Figs. 5c and 5d. Because the GMM is a generative model, one may sample from the distribution (Fig. 5f) to generate points associated with the surface model and reconstruct occupancy (detailed in Section IV-B).

1) Occupied Space: For points with norms less than a user-specified maximum range r_d, the EM approach is adapted from [12] to ignore points that lie outside a Mahalanobis distance greater than λ. Because Gaussians fall off quickly, points far away from a given density will have a small effect on the updated parameters for that density. By reducing the number of points, this decreases the computational cost of the EM calculation. Only points that satisfy the following Mahalanobis-bound are considered:

$$\lambda < \sqrt{(\boldsymbol{x}_n - \boldsymbol{\mu}_m^1)^T (\boldsymbol{\Lambda}_{\boldsymbol{m}}^{\mathbf{1}})^{-1} (\boldsymbol{x}_n - \boldsymbol{\mu}_m^1)} \tag{5}$$

where the superscript 1 denotes the initialized values for the mean, covariance, and weight. This approach differs from our prior work Tabib et al. [7]; we utilize the approach in [12] as it yields greater frame-to-frame registration accuracy in practice. Frame-to-frame registration is not used in this work and is left as future work.

2) Free Space: To learn a free space distribution, points with norms that exceed the maximum range r_d are projected to r_d. The EM approach from Section IV-A1 is used to decrease the computational cost of learning the distribution. To further decrease the cost, the free space points are split into windows in image space and GMMs consisting of n_f components are learned for each window. The windowing strategy is employed for learning distributions over free space points because it yields faster results and the distributions cannot be used for frame-to-frame registration. The number of windows and components per window is selected empirically. Fig. 5d illustrates the effect of the windowing using colored patches and Fig. 5e illustrates the result of this windowing technique with black densities. Once the free space distributions are learned for each



(a) (b) (c) (d)

Fig. 6: Overview of the method by which occupancy is reconstructed (a) The blue bounding box b_{t+1} is centered around $\mathbf{X}_{t+1}$ and red bounding box b_t is centered at $\mathbf{X}_t$. (b) illustrates the novel bounding boxes in solid magenta, teal, and yellow colors that represent the set difference $b_{t+1} \setminus b_t$. (c) Given a sensor origin shown as a triad, resampled pointcloud shown in Fig. 5f, and novel bounding box shown in yellow, each ray from an endpoint to the sensor origin is tested to determine if an intersection with the bounding box occurs. The endpoints of rays that intersect the bounding box are shown in red. (d) illustrates how the bounding box occupancy values are updated. Endpoints inside the yellow volume update cells with an occupied value. All other cells along the ray (shown in blue) are updated to be free.

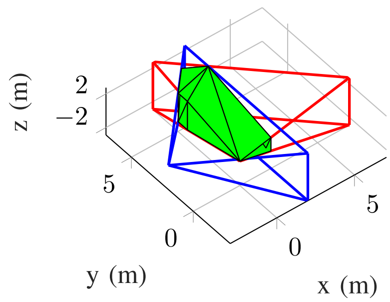


Fig. 7: For limited FoV sensors, the FoV is approximated by the illustrated blue and red rectangular pyramids. These FoVs may also be represented as tetrahedra. To determine if a sensor position should be stored, the overlapping volume between the two approximated sensor FoVs is found.

window the windowed distributions are merged into a single distribution.

Let $\mathcal{G}_i(\boldsymbol{x})$ be a GMM trained from N_i points in window i and let $\mathcal{G}_j(\boldsymbol{x})$ be a GMM trained from N_j points in window j, where $\sum_{w=1}^{W} N_w = N$ for sensor observation $\mathcal{Z}_t$ and W windows. $\mathcal{G}_j(\boldsymbol{x}) = \sum_{k=1}^{K} \tau_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\nu}_k, \boldsymbol{\Omega}_k)$ may be merged into $\mathcal{G}_i(\boldsymbol{x}) = \sum_{m=1}^{M} \pi_m \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_m, \boldsymbol{\Lambda}_m)$ by concatenating the means, covariances, and weights. However, care must be taken when merging the weights as they must be renormalized to sum to 1 [23]. The weights are renormalized via Eqs. (6) and (7):

$$N^* = N_i + N_j \tag{6}$$

$$\boldsymbol{\pi}^* = \begin{bmatrix} \frac{N_i \pi_1}{N^*} & \dots & \frac{N_i \pi_m}{N^*} & \frac{N_j \tau_1}{N^*} & \dots & \frac{N_j \tau_k}{N^*} \end{bmatrix}^T \tag{7}$$

where $m \in [1, \dots, M]$ and $k \in [1, \dots, K]$ denote the mixture component in GMMs $\mathcal{G}_i(\boldsymbol{x})$ and $\mathcal{G}_j(\boldsymbol{x})$, respectively. $N^* \in \mathbb{R}$ is the sum of the support sizes of $\mathcal{G}_i(\boldsymbol{x})$ and $\mathcal{G}_j(\boldsymbol{x})$. $\boldsymbol{\pi}^* \in \mathbb{R}^{M+K}$ are the renormalized weights. The means and covariances are merged by concatenation.

B. Local Occupancy Grid Map

The occupancy grid map [24] is a probabilistic representation that discretizes 3D space into finitely many grid cells $\mathbf{m} = \{m_1, ..., m_{|\mathbf{m}|}\}$. Each cell is assumed to be independent and the probability of occupancy for an individual cell is denoted as $p(m_i|\mathbf{X}_{1:t}, \mathcal{Z}_{1:t})$, where $\mathbf{X}_{1:t}$ represents all vehicle states up to and including time t and $\mathcal{Z}_{1:t}$ represents the corresponding observations. Unobserved grid cells are assigned the uniform prior of 0.5 and the occupancy value of the grid cell m_i at time t is expressed using log odds notation for numerical stability.

$$l_{t,i} \triangleq \log\left(\frac{p(m_i|\mathcal{Z}_{1:t}, \mathbf{X}_{1:t})}{1 - p(m_i|\mathcal{Z}_{1:t}, \mathbf{X}_{1:t})}\right) - l_0$$

When a new measurement $\mathcal{Z}_t$ is obtained, the occupancy value of cell m_i is updated as

$$l_{t,i} \triangleq l_{t-1,i} + L(m_i|\mathcal{Z}_t)$$

where $L(m_i|\mathcal{Z}_t)$ denotes the inverse sensor model of the robot and l_0 is the prior of occupancy [24].

Instead of storing the occupancy grid map $\mathbf{m}$ that represents occupancy for the entire environment viewed since the start of exploration onboard the vehicle, a local occupancy grid map $\bar{\mathbf{m}}_t$ is maintained centered around the robot's pose $\mathbf{X}_t$. The local occupancy grid map moves with the robot, so when regions of the environment are revisited, occupancy must be reconstructed from the surface models $\mathcal{G}(\boldsymbol{x})$ and $\mathcal{F}(\boldsymbol{x})$. To reconstruct occupancy at time $t+1$ given $\bar{\mathbf{m}}_t$, the set difference of the bounding boxes b_t and b_{t+1} for $\bar{\mathbf{m}}_t$ and $\mathbf{m}_{t+1}$, respectively, are used to compute at most three non-overlapping bounding boxes (see Figs. 6a and 6b for example). The intersection of the bounding boxes remains up-to-date, but the occupancy of the novel bounding boxes must be reconstructed using the surface models $\mathcal{G}(\boldsymbol{x})$ and $\mathcal{F}(\boldsymbol{x})$. Raytracing is an expensive operation [25], so time is saved by removing voxels at the intersection of b_t and b_{t+1} from consideration.

The local occupancy grid map at time $t+1$, $\bar{\mathbf{m}}_{t+1}$, is initialized by copying the voxels in local grid $\bar{\mathbf{m}}_t$ at the intersection of b_{t+1} and b_t. In practice, the time to copy the local occupancy grid map is very low (on the order of a few tens of milliseconds) as compared to the cost of raytracing through the grid. Not all Gaussian densities will affect the occupancy reconstruction so to identify the GMM components that intersect the bounding boxes a KDTree [26] stores the means of the densities. A radius equal to twice the sensor's max range is used to identify the components that could affect the occupancy value of the cells in the bounding box. A ray-bounding box intersection algorithm [27] checks for intersections between the bounding box and the ray from the sensor origin to the density mean. Densities that intersect the bounding box are extracted into local submaps $\bar{\mathcal{G}}(\boldsymbol{x})$ and $\bar{\mathcal{F}}(\boldsymbol{x})$. Points are sampled from each distribution and raytraced to their corresponding sensor origin to update the local grid map (example shown in Figs. 6c and 6d).

As the number of mixture components in the distribution increases over time in one region, updating the occupancy becomes increasingly expensive as the number of points needed to resample and raytrace increases. The next sections detail the differences in limiting the potentially unbounded number of points depending on whether the sensor model has a 360° or limited FoV.

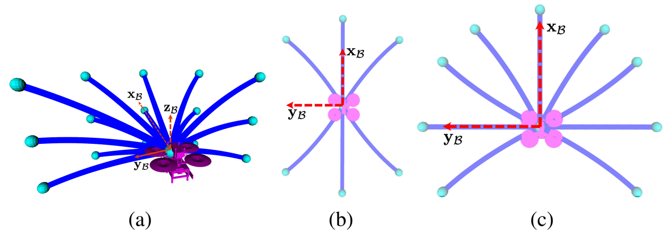


Fig. 8: Action space design for the proposed information-theoretic planner. (a) shows a single motion primitive library generated using bounds on the linear velocity along $\{\mathbf{x}_{\mathcal{B}}, \mathbf{z}_{\mathcal{B}}\}$ and the angular velocity along $\{\mathbf{z}_{\mathcal{B}}\}$. (b) and (c) show top-down views of the motion primitive library collections used when the sensor model is a LiDAR [7] and a depth camera [29] respectively (off-plane primitives are not shown). The proposed planner can be used with either of these sensors using the appropriate action space designs explained in Section V-A.

C. 360° *FoV Sensor Model*

A small, fixed-size bounding box around the current pose with half-lengths h_x, h_y, and h_z is used to determine if a prior observation was made within the confines of the bounding box. This bounding box approach works for sensors that have a 360° field of view such as the 3D LiDAR used in this work, but does not readily extend to depth sensors with smaller fields of view (discussed in the next section). If a prior observation was made within the bounding box the current observation, $\mathcal{Z}_t$ is not stored as a GMM. This has the effect of limiting the number of components that are stored over time.

D. *Limited FoV Sensor Model*

The limited FoV sensor model is directional, so it is approximated by two non-intersecting tetrahedra such that their union forms a rectangular pyramid (shown in Fig. 7). For two sensor FoVs the intersection between the four pairs of tetrahedra is calculated and the intersection points found. The convex hull of the intersection points is converted to a polyhedron mesh with triangular facets. The volume of the convex hull is found by summing individual volumes of the tetrahedrons that make up the polyhedron [28]. The overlap is estimated as a percentage of overlapping volume between two sensor FoVs and a sensor observation is only stored if its overlap exceeds a user-defined threshold. In this way, the number of components that represent a given location can be reduced while ensuring that the environment is covered.

V. Planning for Exploration

A motion planner designed for exploration of *a priori* unknown and unstructured spaces with an aerial robot must satisfy three requirements: (1) reduce entropy of the unknown map, (2) maintain collision-free operation, and (3) return motion plans in real-time. Several previous works provide information-theoretic frameworks towards meeting these objectives [30, 14, 31]. Julian et al. [30] use the Shannon mutual information between the map and potential sensor observations as a reward function to generate motion plans for exploration; however, computational requirements limit the number of potential trajectories over which the reward can be calculated. In contrast, the Cauchy-Schwarz Quadratic mutual information (CSQMI) has been demonstrated for real-time exploration with aerial robots [14, 7]. This work utilizes an information-theoretic planning strategy using CSQMI as the primary reward function, extending our prior work [7] to support limited FoV sensors in addition to 360° FoV sensors. The proposed framework can be divided into two stages: (1) action space generation and (2) action selection. At the start of any planning iteration, the planner uses the action generation strategy (detailed in Section V-A) to generate a set of candidate actions up to a user-specified planning horizon using motion primitives. The action selector evaluates the collision-free and dynamically feasible subset of the action space using CSQMI as a reward function, returning the most informative plan to execute during the next planning iteration (see Sections V-B and V-C).

A. *Action Space Generation*

This section describes: (1) background on the trajectory representation using motion primitive generation [32], and (2) the design of the action space.

1) Forward-Arc Motion Primitive: Accurate position control of multirotors presumes continuity in the supplied tracking references up to high-order derivatives of position [33]. To represent a candidate trajectory, this paper utilizes sequential forward-arc motion primitives [32]. We use an extension to this work that ensures differentiability up to jerk and continuity up to snap [34]. Given the multirotor state at a time t, $\boldsymbol{\xi}_t = [x, y, z, \theta]^\top$, linear velocities in the body frame $[v_{\mathbf{x}_{\mathcal{B}}}, v_{\mathbf{z}_{\mathcal{B}}}]$, and the angular velocity about $\mathbf{z}_{\mathcal{B}}$ axis, $\omega_{\mathbf{z}_{\mathcal{B}}}$, the forward-arc motion primitive is computed as a polynomial function of time generated using the following high-order constraints:

$$\begin{aligned} \dot{\boldsymbol{\xi}}_\tau &= [v_{\mathbf{x}_{\mathcal{B}}} \cos\theta, v_{\mathbf{x}_{\mathcal{B}}} \sin\theta, v_{\mathbf{z}_{\mathcal{B}}}, \omega_{\mathbf{z}_{\mathcal{B}}}] \\ \boldsymbol{\xi}_\tau^{(n)} &= \mathbf{0} \text{ for } n = 2, 3, 4 \end{aligned} \tag{8}$$

where $\{.\}^{(n)}$ denotes the n^{th} time derivative, and τ is the specified duration of the motion primitive. Motion primitives in the $\mathbf{y}_{\mathcal{B}}$ direction can also be obtained by replacing $v_{\mathbf{x}_{\mathcal{B}}}$ by $v_{\mathbf{y}_{\mathcal{B}}}$ in the above constraints. Later, we will use a combination of these directions to define the action space for the exploration planner that can operate with either a LiDAR or depth camera.

2) Motion Primitive Library (MPL): A motion primitive library (MPL) is a collection of forward-arc motion primitives generated using a user-specified discretization of the robot's linear and angular velocities [32]. Let $\mathbf{a} = \{v_{\mathbf{x}_{\mathcal{B}}}, v_{\mathbf{y}_{\mathcal{B}}}, v_{\mathbf{z}_{\mathcal{B}}}, \omega_{\mathbf{z}_{\mathcal{B}}}\}$ be an action set that is generated with user-specified maximum velocity bounds in the $\mathbf{x}_{\mathcal{B}} - \mathbf{y}_{\mathcal{B}}$ plane and the $\mathbf{z}_{\mathcal{B}}$ direction. The motion primitive library is then given by the set (Fig. 8a):

$$\Gamma_{\boldsymbol{\xi}_t} = \{\gamma_{\boldsymbol{\xi}_t}(\mathbf{a}_{jk}, \tau) \mid \|[v_{\mathbf{x}}, v_{\mathbf{y}}]\| \leq V_{\max}, \|v_{\mathbf{z}}\| \leq V_{\mathbf{z}}, \|\omega\| \leq \Omega\} \tag{9}$$

where $j \in [1, N_\omega]$ and $k \in [1, N_{\mathbf{z}}]$ define the action discretization for one particular primitive.

For each MPL, an additional MPL containing stopping trajectories at any state $\boldsymbol{\xi}_t$ can be generated by fixing the desired end point velocity to zero, $\dot{\boldsymbol{\xi}}_\tau = 0$. These stopping trajectories are scheduled one planning round away from the starting time of the planning round. These trajectories help ensure safety in case the planner fails to compute an optimal action.

3) Designing the Action Space: The final action space, $\mathcal{X}_{\text{act}}$, is a collection of MPLs selected according to three criteria: (1) rate of information gain, (2) safety, and (3) limitations in compute. Prior work [7] provides such a design for a 360° FoV sensor (LiDAR). Goel et al. [29] present an analysis on how these three factors influence $\mathcal{X}_{\text{act}}$ for a limited FoV depth sensor. This work extends [7] using the analysis in [29], yielding a motion planner amenable for exploration with either a LiDAR or a depth sensor and that ensures similar exploration performance in either case (see Section VI).

a) Action Space for 360° FoV Sensors: 360° FoV sensors are advantageous in an exploration scenario because of three factors: (1) 360° depth data from the sensor allows for visibility in all azimuth directions, (2) a larger volume is explored per unit range when compared to a limited FoV sensor, and (3) yaw in-place motion does not help gain information. The first factor enables backward and sideways motion into the action space $\mathcal{X}_{\text{act}}$ without sacrificing safety (Fig. 8b). The second factor influences the entropy reduction: for the same trajectory, a sensor with a larger FoV will explore more voxels compared to the limited FoV case. The third factor reduces the number of motion primitive libraries in the action space to yield increased planning frequency. An example of an action space designed while considering these factors is presented in [7] and the corresponding parameters are shown in Table Ia.

These factors indicate that the same action space $\mathcal{X}_{\text{act}}$ cannot be used for limited FoV cameras if comparable exploration performance is to be maintained. This motivates the need for an alternate and informed action design for the limited FoV cameras.

b) Action Space for Limited FoV Sensors: Goel et al. [29] show that for an exploration planner using a limited FoV sensor, the design of the action space $\mathcal{X}_{\text{act}}$ can be informed by the sensor model. The authors consider a depth sensor to design $\mathcal{X}_{\text{act}}$ by incorporating the sensor range and FoV, among other factors. This work follows a similar approach yielding an action space that contains MPLs in both the $\mathbf{x}_{\mathcal{B}}$ and $\mathbf{y}_{\mathcal{B}}$ directions (Fig. 8c). The parameters to construct the MPL collection comprising $\mathcal{X}_{\text{act}}$ are shown in Table Ib. Note that there is an additional MPL corresponding to a yaw-in-place motion, unlike the 360° FoV case, to compensate for the limited FoV of the depth camera. For further detail on how to obtain these parameters, please refer to [29, 35].

B. Information-Theoretic Objective

The action selection policy uses CSQMI as the information-theoretic objective to maximize the information gain over time. CSQMI is computed at k points along the primitive $\gamma_{\boldsymbol{\xi}_t}$, and the sum is used as a metric to measure the expected local information gain for a candidate action $\mathcal{I}_\gamma$. However, this design may result in myopic decision-making. Therefore, frontiers are also incorporated to model the global spatial distribution of information [36]. This global reward, denoted by $\mathcal{V}_\gamma$, is calculated based on the change in distance towards a frontier along a candidate action. Using the node state $\boldsymbol{\xi}_0$, end point state $\boldsymbol{\xi}_\tau$, and a distance field constructed based on the position of the frontiers, this reward can be calculated as $\mathcal{V}_\gamma = d(\boldsymbol{\xi}_0) - d(\boldsymbol{\xi}_\tau)$, where $d(\boldsymbol{\xi}_t)$ denotes the distance to the nearest voxel in the distance field from state $\boldsymbol{\xi}_t$ [18].

MPL ID	Vel., Time	N_ω, $N_{\mathbf{z}}$	N_{prim}
1	$v_{\mathbf{x}_\mathcal{B}}, \tau$	3, 5	15
2	$v_{\mathbf{x}_\mathcal{B}}, 2\tau$	3, 5	15
3	$-v_{\mathbf{x}_\mathcal{B}}, \tau$	3, 5	15
4	$-v_{\mathbf{x}_\mathcal{B}}, 2\tau$	3, 5	15

(a) **LiDAR**

MPL ID	Vel., Time	N_ω, $N_{\mathbf{z}}$	N_{prim}
1	$v_{\mathbf{x}_\mathcal{B}}, \tau$	3, 5	15
2	$v_{\mathbf{x}_\mathcal{B}}, 2\tau$	3, 5	15
3	$v_{\mathbf{y}_\mathcal{B}}, \tau$	3, 5	15
4	$v_{\mathbf{y}_\mathcal{B}}, 2\tau$	3, 5	15
5	$-v_{\mathbf{y}_\mathcal{B}}, \tau$	3, 5	15
6	$-v_{\mathbf{y}_\mathcal{B}}, 2\tau$	3, 5	15
7	$\omega_{\mathbf{z}_\mathcal{B}}, \tau$	1, 5	5

(b) **Depth Camera**

TABLE I: Discretization used to construct the action space $\mathcal{X}_{\text{act}}$ for the simulation experiments for (a) LiDAR and (b) depth camera cases. Total number of primitives for a MPL are denoted by $N_{\text{prim}} = N_\omega \cdot N_{\mathbf{z}}$. The base duration τ was kept at 3 s for all experiments.

Algorithm 1 Overview of Action Selection for Exploration

```
1:  input: X_act, X_free
2:  output: γ*_ξt                                    ▷ best action
3:  for Γ_ξt ∈ X_act do
4:      for γ_ξt ∈ Γ_ξt do
5:          feasible ← SAFETYCHECK(γ_ξt, γ^stop_ξt, X_free)
6:          if feasible then
7:              I_γ ← INFORMATIONREWARD(γ_ξt)
8:              V_γ ← FRONTIERDISTANCEREWARD(γ_ξt)
9:          else
10:             I_γ ← 0.0, V_γ ← 0.0
11: return γ*_ξt ← argmax_{γ_ξt ∈ X_act} [I_γ + V_γ]
```

C. Action Selection

Using the rewards described in the preceding section, the objective for the motion planner is defined as follows [29, 18]:

$$\begin{aligned} &\underset{\gamma_{\boldsymbol{\xi}_t}}{\arg\max}\ \mathcal{I}_\gamma + \alpha\,\mathcal{V}_\gamma \\ &\text{s.t. } \gamma_{\boldsymbol{\xi}_t} \in \mathcal{X}_{\text{act}} \end{aligned} \tag{10}$$

where α is a weight that adjusts the contribution of the frontier distance reward. Recall, the goal is to maximize this reward function in real-time on a compute-constrained aerial platform. Previous information-theoretic approaches that construct a tree and use a finite-horizon planner either do not use a global heuristic [37] or are not known to be amenable for operation on compute-constrained platforms [18]. In this work, a single-step planner is used with the action space $\mathcal{X}_{\text{act}}$ consisting of motion primitives of varying duration for real-time performance (see Table I). Due to this choice, the planner computes rewards over candidate actions that extend further into the explored map from the current position. In this manner, longer duration candidate actions provide a longer lookahead than the case when all candidate actions are of the same duration even in single-step planning formulations (see Table I).

The action selection procedure is detailed in Algorithm 1. For every candidate action $\gamma_{\boldsymbol{\xi}_t}$ in the action space $\mathcal{X}_{\text{act}}$, a safety

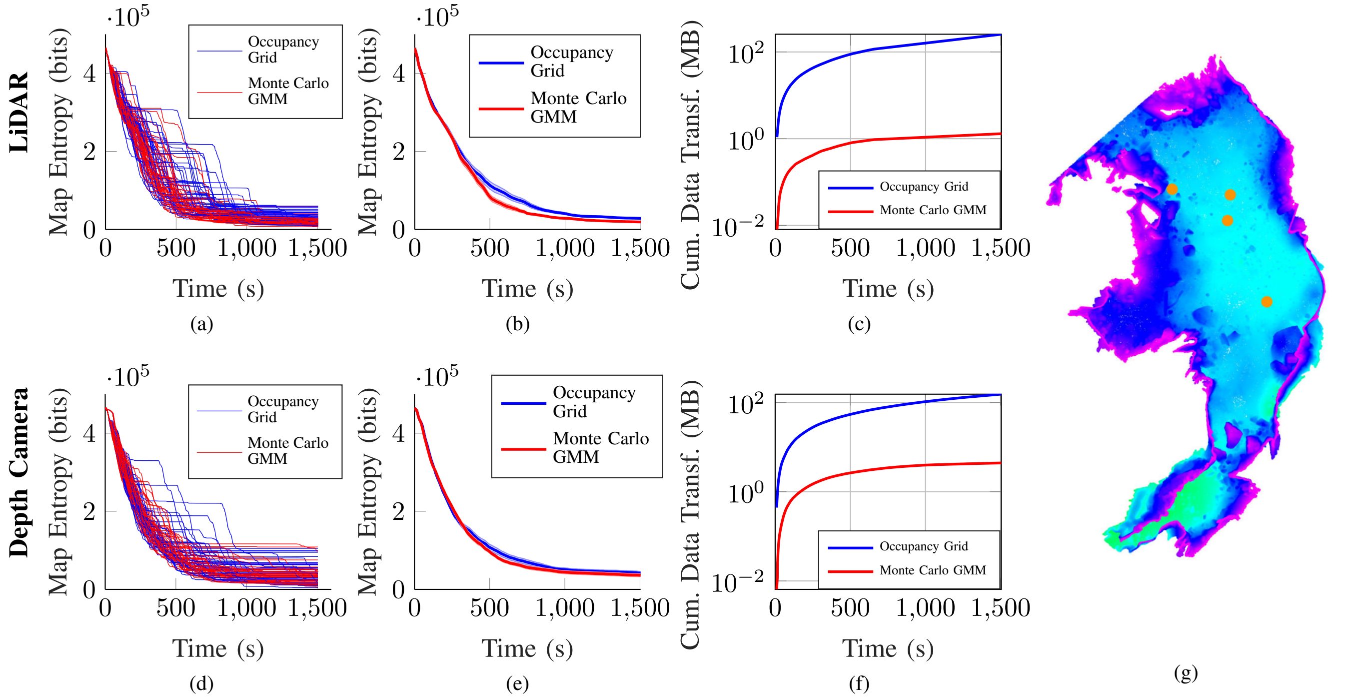


Fig. 9: Exploration statistics for simulation experiments. The first row of results pertains to the LiDAR sensor model and the second row to the depth camera sensor model. (a) and (d) illustrate the map entropy over time for 160 trials (80 trials per sensor model and 40 trials per mapping method), (b) and (e) illustrate the average map entropy over time for each method. Although both methods achieve similar entropy reduction, MCG uses significantly less memory according to the average cumulative data transferred shown in (c) and (f). When the LiDAR sensor model is used, the average cumulative data transferred at the end of $1500\,\mathrm{s}$ is $1.3\,\mathrm{MB}$ for the MCG approach and $256\,\mathrm{MB}$ for the OG approach. When the depth camera sensor model is used, the average cumulative data transferred at the end of $1500\,\mathrm{s}$ is $4.4\,\mathrm{MB}$ for the MCG approach and $153\,\mathrm{MB}$ for the OG approach. The MCG method represents a decrease of approximately one to two orders of magnitude as compared to the OG method for the LiDAR and depth camera sensor models, respectively. The experiments are conducted in the simulated cave environment shown in Fig. 9g. The four starting positions are shown as orange dots.

check procedure is performed to ensure that this candidate and the associated stopping action ($\gamma^{\text{stop}}_{\boldsymbol{\xi}_t}$) are dynamically feasible and lie within free space $\mathcal{X}_{\text{free}}$ (Line 5). The free space check is performed using a Euclidean distance field created from locations of occupied and unknown spaces in the robot's local map given a fixed collision radius [38]. Checking that the stopping action is also feasible ensures that the planner never visits an inevitable collision state, which is essential for safe operation [39]. If the action is feasible, the local information reward ($\mathcal{I}_\gamma$, Line 7) and frontier distance reward ($\mathcal{V}_\gamma$, Line 8) are determined as described in Section V-B. The planner then returns the action with the best overall reward (Line 11).

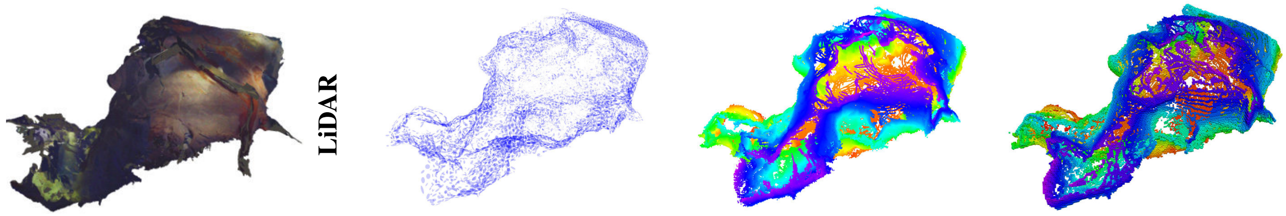
VI. Experimental Design and Results

This section details the experimental design to validate the approach. Results are reported for both real-time simulation trials and field tests in a cave. The following shorthand is introduced for this section only: MCG will refer to the Monte Carlo GMM mapping approach and OG mapping will refer to the Occupancy Grid mapping approach. The mapping and planning software is run on an embedded Gigabyte Brix 8550U with eight cores and $32\,\mathrm{GB}$ RAM, for both hardware and simulation experiments. Simulation results are presented for both LiDAR and depth camera sensor models, but hardware results are reported only for the depth camera case[2]. Unless otherwise noted, the parameters for simulation and hardware experiments are equal.

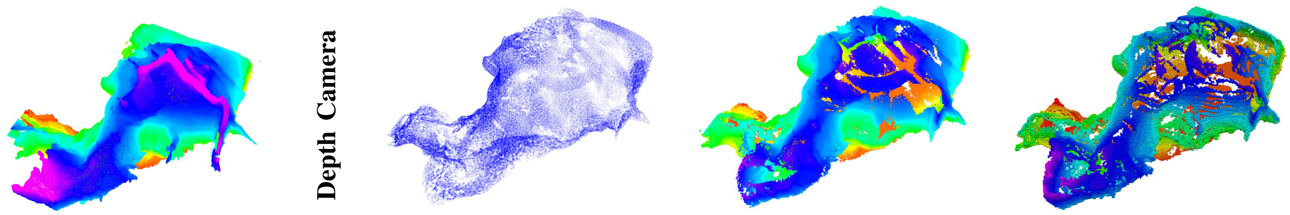
A. Comparison Metrics

To calculate the memory requirements for the OG mapping approach, the incremental OG map is transmitted as a changeset pointcloud where each point consists of 4 floating point numbers: $\{x, y, z, \text{logodds}\}$. The changeset is computed after insertion of every pointcloud. A floating point number is assumed to be 4 bytes, or 32 bits. For the MCG approach, the cumulative data transferred is computed by summing the cost of transmitted GMMs. Each mixture component is transmitted as 10 floating point numbers: 6 numbers for the covariance matrix (because the covariance matrix is symmetric), three numbers for the mean, and one number for the mixture component weight. One additional number is stored per GMM that represents the number of points from which the GMM was learned. The sensor origin is also stored for each GMM using 6 numbers (three to represent translation and three to represent rotation as Euler angles). To ensure a fair comparison of the exploration performance between the two approaches, a global

[2]The prior work upon which this manuscript is developed leveraged a $6.7\,\mathrm{kg}$ aerial system with LiDAR. To support improved experimental convenience, an alternative platform was developed that results in lower size, weight, and power consumption as compared to the previous platform. For the LiDAR system hardware results, please see [7].



(a) Simulated Cave Environment (b) GMM map (1σ covariances) (c) Resampled pointcloud (d) Dense voxel map



(e) Points from Rapps PLY (f) GMM map (1σ covariances) (g) Resampled pointcloud (h) Dense voxel map

Fig. 10: The colorized mesh used in simulation experiments is shown in (a) and produced from FARO scans of a cave in West Virginia. After $1500\,\mathrm{s}$ of exploration with a LiDAR sensor model, the resulting (b) MCG map is shown with 1σ covariances and densely resampled with 1×10^6 points to obtain the reconstruction shown in (c). (d) illustrates the dense voxel map produced after a $1500\,\mathrm{s}$ trial with $20\,\mathrm{cm}$ voxels. (e) illustrates the pointcloud from the mesh shown in (a). (f) illustrates the MCG map with 1σ covariances, which is densely resampled with 1×10^6 points, to obtain the reconstruction shown in (g). (h) illustrates the dense voxel map with $20\,\mathrm{cm}$ voxels after $1500\,\mathrm{s}$ of exploration with the depth camera sensor model. The reconstruction accuracy for (c), (d), (g), and (h) are shown in Table II. All pointclouds shown are colored from red to purple according to z-height.

occupancy grid serves as a referee and is maintained in the background with a voxel resolution of $0.2\,\mathrm{m}$. This occupancy grid is used to compute the map entropy over time [40], thus measuring the exploration progress during a simulation or a hardware experiment.

Approach	**LiDAR**		**Depth Camera**	
	Mean (m)	**Std** (m)	**Mean** (m)	**Std** (m)
MCG	1.8×10^{-2}	2.5×10^{-2}	1.3×10^{-2}	1.9×10^{-2}
OG	6.2×10^{-2}	3.9×10^{-2}	6.3×10^{-2}	3.9×10^{-2}

TABLE II: Reconstruction error for Fig. 10

B. Simulation Experiments

The exploration strategy is evaluated with 160 real-time simulation trials over approximately 67 hours in a $30\,\mathrm{m} \times 40\,\mathrm{m} \times 6\,\mathrm{m}$ environment constructed from colorized FARO pointclouds of a cave in West Virginia (see Fig. 10a). In each simulation, the multirotor robot begins exploration from one of four pre-determined starting positions and explores for $1500\,\mathrm{s}$. For each starting position, 10 exploration trials are run which means a total of 40 trials for each mapping approach and sensor configuration for a total of 160 trials. The end time of $1500\,\mathrm{s}$ is set empirically and based on the total time required to fully explore the cave environment. Note that ground truth state estimates are used for these simulation experiments, while the hardware experiments in Section VI-C rely on visual-inertial odometry (see Section VI-C1).

1) LiDAR Simulations: The LiDAR has a max range of $5.0\,\mathrm{m}$ and operates at $10\,\mathrm{Hz}$ for all simulation experiments. The motion planning parameters used in the action space design are shown in Table Ia. For all simulation trials, the maximum speed in the $\mathbf{x}_\mathcal{B} - \mathbf{y}_\mathcal{B}$ plane is $\|V_{\max}\| = 0.75\,\mathrm{m/s}$, the maximum speed along the $\mathbf{z}_\mathcal{B}$ axis is $V_\mathbf{z} = 0.5\,\mathrm{m/s}$, and the maximum yaw rate is $\Omega = 0.25\,\mathrm{rad/s}$. CSQMI is computed at the end point of the candidate action ($k = 1$). $\lambda = 5$ and $n_f = 2$ for all simulations and hardware trials.

The simulation trials demonstrate that MCG achieves similar exploration performance as OG, which indicates that the approximations made by the former enables real-time performance without compromising exploration or map reconstruction quality (Figs. 9b and 10c). Figure 9c depicts the cumulative data that must be transferred to reproduce the OG and MCG maps remotely. After $1500\,\mathrm{s}$, transferring the MCG map requires $1.3\,\mathrm{MB}$ as compared to $256\,\mathrm{MB}$ to incrementally transfer the OG map. The MCG approach significantly outperforms the OG approach in terms of cumulative data transfer requirements. A representative example of the reconstructed GMM map for one trial from Fig. 9a is shown in Fig. 10b. Resampling 1×10^6 points from the distribution yields the map shown in Fig. 10c. The MCG approach has lower average reconstruction error as compared to the OG approach (Fig. 10d) as shown in Table II.

2) Depth Camera Simulations: The depth camera sensor model also has a max range of $5.0\,\mathrm{m}$ and operates at $10\,\mathrm{Hz}$ for all simulation experiments. A collection of motion primitive libraries used for the simulation experiments is shown in Table Ib. The velocity bounds for the simulation experiments are the same as in the LiDAR case.

Similar to the LiDAR results, MCG outperforms OG in terms of memory efficiency while maintaining similar exploration performance. Figure 9f depicts the cumulative amount of data comparison in this case. After $1500\,\mathrm{s}$, transferring the MCG map requires $4.4\,\mathrm{MB}$ as compared to $153\,\mathrm{MB}$ to incrementally transfer the OG map. Because the LiDAR has a larger field of view and lower resolution (i.e., fewer points for

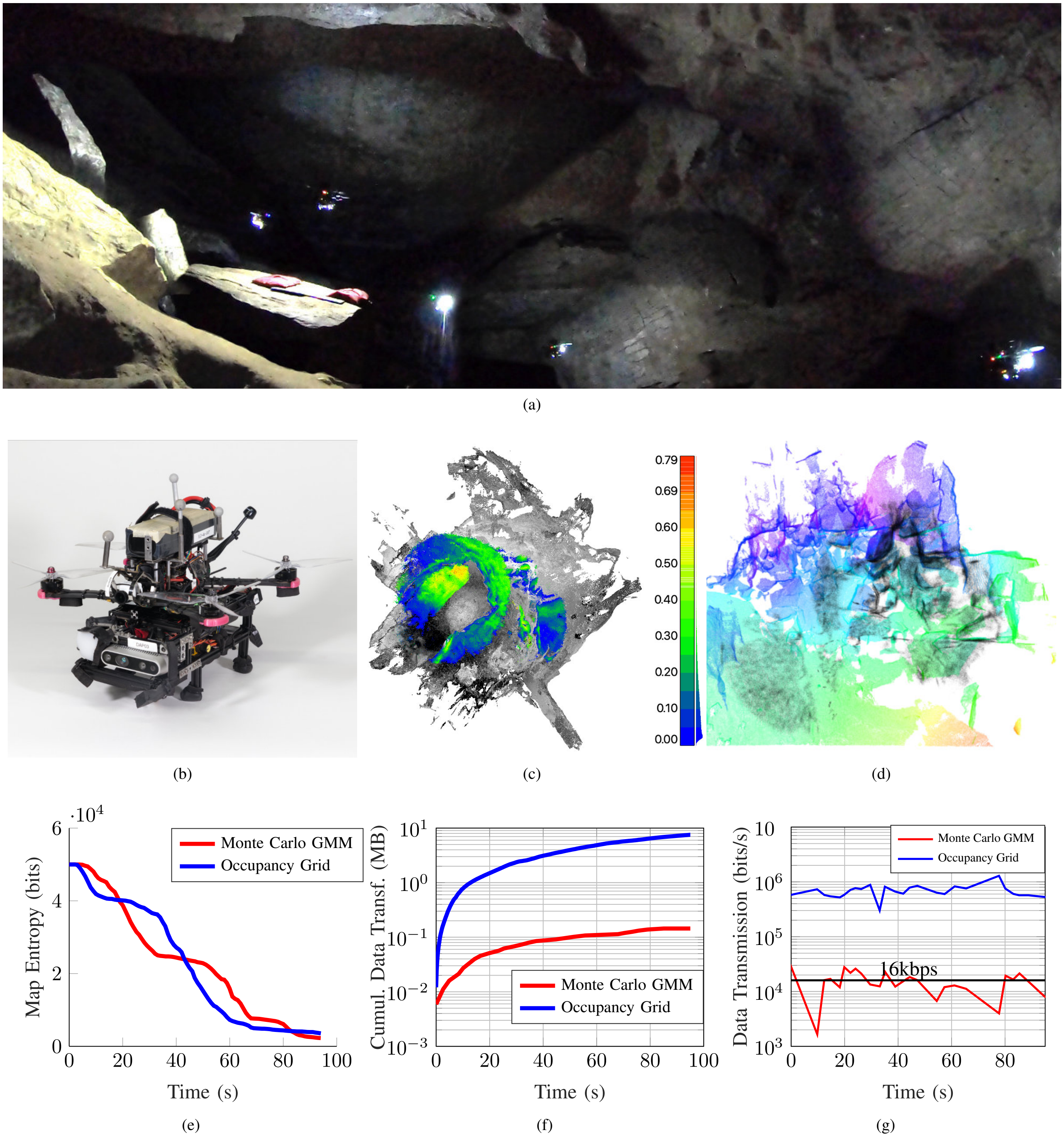


Fig. 11: (a) A single aerial system explores the Dining Room of Laurel Caverns in Southwestern Pennsylvania. Still images of the robot exploring the environment are super-imposed to produce this figure. (b) The aerial system with dimensions $0.25\,\mathrm{m} \times 0.41\,\mathrm{m} \times 0.37\,\mathrm{m}$ including propellers carries a forward-facing Intel Realsense D435 for mapping and downward-facing global shutter MV Bluefox2 camera (not shown). The pearl reflective markers are used for testing in a motion capture arena but are not used during field operations to obtain hardware results. Instead, a tightly-coupled visual-inertial odometry framework is used to estimate state during testing at Laurel Caverns. (c) illustrates the reconstruction error of the resampled GMM map as compared to the FARO map by calculating point-to-point distances. The distribution of distances is shown on the right-hand side. The mean error is $0.14\,\mathrm{m}$ with a standard deviation of $0.11\,\mathrm{m}$. In particular, there is misalignment in the roof due to pose estimation drift. (d) A subset of the resampled GMM map (shown in black) is overlaid onto the FARO map (shown in colors ranging from red to purple) that displays the breakdown in the middle of the Dining Room. (e) The entropy reduction and (f) cumulative data transferred for one trial for each of the Monte Carlo Gmm mapping and OG mapping approaches are shown. While the map entropy reduction for each approach is approximately similar, the GMM mapping approach transmits significantly less memory than the OG mapping approach ($0.1\,\mathrm{MB}$ as compared to $7.5\,\mathrm{MB}$). (g) illustrates the bit rate for each approach in a semi-logarithmic plot where the vertical axis is logarithmic. The black line illustrates how the approaches compare to 16kbps. For comparison, 16kbps is sufficient to transmit a low resolution (176 × 144 at 5 fps compressed to 3200 bit/frame) *talking heads* video [41, 42].

the same area of coverage) as compared to the depth camera, this sensor model covers more voxels given a fixed voxel resolution. The voxel grid relies on the Markovian assumption of the Bayes filter, which causes conflicting observations of the voxel (free vs. occupied) to rapidly change the probability of occupancy. This results in more memory transmitted to reflect these changes and as the field of view increases, the number of affected voxels increases. The probability of occupancy of a voxel for all experiments is not clamped even if it exceeds a threshold. Similar to the LiDAR reconstruction, the MCG has lower average reconstruction error as compared to the OG approach (see Fig. 10h) as shown in Table II.

C. Hardware Experiments

1) Visual-Inertial Navigation and Control: State estimates are computed from IMU and downward facing camera observations via VINS-Mono [43], a tightly-coupled visual-inertial odometry framework that jointly optimizes vehicle motion, feature locations, and IMU biases over a sliding window of monocular images and pre-integrated IMU measurements. The loop closure functionality of VINS-Mono is disabled to avoid having relocalization-induced discontinuities in the trajectory estimate, which would have significant implications for occupancy mapping and is left as future work.

For accurate trajectory tracking, a cascaded Proportional-Derivative (PD) controller is used with a nonlinear Luenberger observer to compensate for external acceleration and torque disturbances acting on the system [44]. To improve trajectory tracking, the controller uses angular feedforward velocity and acceleration terms computed from jerk and snap references computed from the reference trajectory's 8^{th} order polynomial (Fig. 4).

Additionally, a state machine enables the user to trigger transitions between the following modes of flight operation: (1) takeoff, (2) hover, (3) tele-operation, (4) autonomous exploration, and (5) landing. The results presented in the next section all pertain to the autonomous exploration mode.

2) Implementation Details: The exploration framework is deployed to the aerial system shown in Fig. 11b, a $2.5\,\mathrm{kg}$ platform equipped with a forward-facing Intel Realsense D435, downward-facing MV Bluefox2 camera, and downward and forward facing lights from Cree Xlamp XM-L2 High Power LEDs (Cool White 6500K). The MV Bluefox2 and D435 cameras operate at $60\,\mathrm{Hz}$ and output images of size 376×240 and 848×480, respectively. The MV Bluefox2 images are used in state estimation and the D435 depth images are throttled to $6\,\mathrm{Hz}$ for the the mapping system. The D435 camera estimates depth by stereo matching features in left- and right-camera images augmented through a dot pattern projected by an IR projector. The laser power of the IR projector on the D435 is increased from a default value of $150\,\mathrm{mW}$ to $300\,\mathrm{mW}$ in order to improve observation quality in darkness.

The robot is equipped with an Auvidea J120 carrier board with NVIDIA TX2 and Gigabyte Brix 8550U that communicate over ethernet. The TX2 performs state estimation and control functions while the Brix performs mapping and planning. The flight controller used for all experiments is the Pixracer, but the platform is also equipped with a Betaflight controller as a secondary flight controller. Switching between the two controllers can be done via a switch on the RC transmitter. The drone frame is an Armattan Chameleon Ti LR 7" on which a Lumenier BLHeli_32 32bit 50A 4-in-1 electronic speed controller is mounted. The aerial system is a quadrotor that uses T-Motor F80 Pro 1900KV motors and DAL Cyclone 7056C propellers.

For all hardware experiments, $\|V_{\max}\| = 0.5\,\mathrm{m/s}$, $V_{\mathbf{z}} = 0.25\,\mathrm{m/s}$, and the motion primitives with duration 2τ in Table Ib are disabled. These choices for are made to operate safely in Laurel Caverns (Section VI-C3).

3) Laurel Caverns: The approach is tested in total darkness at Laurel Caverns[3], a commercially operated cave system in Southwestern Pennsylvania consisting of over four miles of passages[4].

Figure 11a illustrates a composite image from several still images of the robot exploring the Laurel Caverns Dining Room. Two experiments were conducted, one for each of the MCG and OG approaches for a $95\,\mathrm{s}$ duration. The map entropy reduction over time is shown in Fig. 11e and is similar for both approaches, while the cumulative data transferred (Fig. 11f) to represent the maps is greater than an order of magnitude lower for the MCG approach as compared to the OG approach. The rate of data transferred in Fig. 11g is calculated using Euler differentiation but note that the accuracy is affected by the limited number of samples. During hardware trials, a bounding box was used to constrain the exploration volume.

VII. Conclusions and Future Work

The results presented in this paper comprise the beginning of an exciting line of research for autonomous cave surveying and mapping by aerial systems. By leveraging GMMs to compactly represent the environment, a high-fidelity perceptual model is achieved that is amenable to transmission across low-bandwidth communications channels. The method is demonstrated with $360°$ and limited field of view sensors and a planning framework amenable to both sensor models is also described. Several avenues are left as future work. Cave maps are typically annotated with important terrain features such as stalactites, stalagmites, and breakdown (see Fig. 2 for example), but this work does not consider the problem of terrain feature classification and encoding. Additionally, waterproof, rugged, and easy-to-use 2D maps are critical for cave rescuers or explorers to avoid getting lost in caves. Methods to project the 3D map information from the robot to 2D are needed to fill this gap in the state of the art. Finally, the deployment of multiple robots to increase the speed of exploration is of interest for large passages or maze caves. Introducing re-localization strategies to curb drift over

[3] http://laurelcaverns.com/

[4] The authors acknowledge that caves are fragile environments formed over the course of tens of thousands to millions of years. Laurel Caverns was chosen as a test site because it has relatively few speleothems due to its sandstone overburden and the high silica content of the Loyalhanna limestone [45]. The authors worked with cave management to select a test site that contained low speleothem growth to minimize risk of damage to the cave. Cave management monitored all flights. No flights were executed near delicate formations.

long duration flights and yield more consistent maps would also be beneficial for multi-robot operations. Beyond cave applications, this work has relevance for search and rescue, planetary exploration, and tactical operations where humans and robots must share information in real-time.

VIII. Acknowledgments

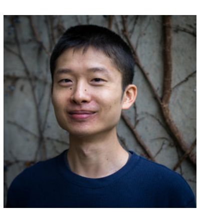
The authors thank Hope Brooks and Ryan Maurer for facilitating experiments at Laurel Caverns and thank David Cale for granting permission to test at Laurel Caverns. The authors also thank Dennis Melko for support and guidance regarding test sites, lending equipment and teaching the authors about caving. The authors thank Bert Ashbrook for his insights and information regarding the cave on the Barbara Schomer Cave Preserve in Clarion County, PA. The authors thank Hunter Wodzenski and Jeff Jahn for providing images used in this work. Finally, the authors thank Xuning Yang for fruitful discussions about motion primitives-based planning and Aditya Dhawale, Arjav Desai, Tim Lee, Matt Collins, and Micah Corah for feedback on this manuscript.

Wennie Tabib received the B.S. degree in computer science in 2012, the M.S. degree in robotics in 2014, and the Ph.D. degree in computer science in 2019 from Carnegie Mellon University, Pittsburgh, PA, USA.

She is a Postdoctoral Fellow in the Robotics Institute at Carnegie Mellon University. She researches perception, planning, and learning algorithms to enable safe autonomy in significantly three-dimensional, complex environments. Her current research develops methods to enable aerial systems to explore subterranean environments. Wennie is also a member of the National Speleological Society (NSS 69985), Mid-Atlantic Karst Conservancy, Pittsburgh Grotto and Loyalhanna Grotto.

Kshitij Goel received the B.Tech. degree in aerospace engineering in 2017 from the Indian Institute of Technology (IIT) Kharagpur, Kharagpur, WB, India.

Kshitij is a M.S. student in Robotics at Carnegie Mellon University, researching fast motion planning for multirotors operating in unknown environments. His current work focuses on robustly deploying teams of multirotors to rapidly explore challenging real world scenarios in real time.



John Yao received the B.A.Sc. degree in aerospace engineering from the University of Toronto, Toronto, Canada, in 2013 and the M.S. degree in robotics from Carnegie Mellon University (CMU), Pittsburgh, PA, USA in 2016.

John is a Ph.D. Candidate in the Robotics Institute at CMU. His research interests include visual-inertial state estimation and resource-constrained sensor fusion for autonomous robots.

Curtis Boirum received the B.S. degree in physical science from Eureka College, Eureka, IL, USA, in 2008. He received the B.S. and M.S. degrees in mechanical engineering from Bradley University, Peoria, IL, USA, in 2009 and 2011, respectively. He received the M.S. degree in robotics from Carnegie Mellon University, Pittsburgh, PA, USA, in 2015.

Curtis is a systems engineer for the Resilient Intelligent Systems Lab who designs, builds, and operates drones and ground robots ranging in size from 100g to 7kg.

Nathan Michael received the Ph.D. degree in mechanical engineering from the University of Pennsylvania, Philadelphia, PA, USA, in 2008.

He is an Associate Research Professor in the Robotics Institute of Carnegie Mellon University; Director of the Resilient Intelligent Systems Lab; author of over 160 publications on control, perception, and cognition for resilient intelligent single and multi-robot systems; nominee or recipient of nine best paper awards; recipient of the Popular Mechanics Breakthrough Award and Robotics Society of Japan Best Paper Award (of 2014); PI of past and ongoing research programs supported by ARL, AFRL, DARPA, DOE, DTRA, NASA, NSF, ONR, and industry; and Chief Technical Officer of Shield AI. Nathan develops resilient intelligent autonomous systems capable of individual and collective intelligence through introspection, adaptation, and evolvement in challenging domains.

References

[1] J. Gunn, *Encyclopedia of Caves and Karst Science*. New York, NY: Taylor & Francis, 2004.

[2] B. Aulenbach *et al.*, *Caving Basics: A Comprehensive Guide for Beginning Cavers*, 4th ed. Huntsville, AL: National Speleological Society, 2016.

[3] R. L. Breisch, *Lost in a Cave: Applying Graph Theory to Cave Exploration*. Huntsville, AL: National Speleological Society, 2011.

[4] E. F. Frank, *Nittany Grotto News*, vol. 32, no. 5, p. 10, 1985.

[5] T. Dang *et al.*, "Field-hardened robotic autonomy for subterranean exploration," in *Conf. on Field and Service Robot.*, Tokyo, Japan, Aug. 2019.

[6] D. Gibson, "Channel characterisation and system design for sub-surface communications," Ph.D. dissertation, University of Leeds, 2003.

[7] W. Tabib, K. Goel, J. Yao, M. Dabhi, C. Boirum, and N. Michael, "Real-time information-theoretic exploration with gaussian mixture model maps," in *Proc. of Robot.: Sci. and Syst.*, FreiburgimBreisgau, Germany, June 2019.

[8] P. J. Westwick, *Into the black: JPL and the American space program, 1976-2004*. New Haven: Yale University Press, 2007.

[9] (2018) August 1975 - viking 1 launched. [Online]. Available: https://www.nasa.gov/directorates/heo/scan/images/history/August1975.html

[10] R. R. Murphy, J. Kravitz, S. L. Stover, and R. Shoureshi, "Mobile robots in mine rescue and recovery," *IEEE Robot. Autom. Mag.*, vol. 16, no. 2, pp. 91–103, 2009.

[11] L. Kaul, R. Zlot, and M. Bosse, "Continuous-time three-dimensional mapping for micro aerial vehicles with a passively actuated rotating laser scanner," *J. Field Robot.*, vol. 33, no. 1, pp. 103–132, 2016.

[12] W. Tabib and N. Michael, "Simultaneous localization and mapping of subterranean voids with gaussian mixture models," in *Conf. on Field and Service Robot.*, Tokyo, Japan, Aug. 2019.

[13] M. Gary, N. Fairfield, W. C. Stone, D. Wettergreen, G. Kantor, and J. M. Sharp, *3D Mapping and Characterization of Sistema Zacatón from DEPTHX*.

[14] B. Charrow, S. Liu, V. Kumar, and N. Michael, "Information-theoretic mapping using Cauchy-Schwarz quadratic mutual information," in *Proc. of the IEEE Intl. Conf. on Robot. and Autom.*, Seattle, WA, May 2015.

[15] J. Saarinen, H. Andreasson, T. Stoyanov, J. Ala-Luhtala, and A. J. Lilienthal, "Normal distributions transform occupancy maps: Application to large-scale online 3d mapping," in *Proc. of the IEEE Intl. Conf. on Robot. and Autom.* IEEE, 2013, pp. 2233–2238.

[16] C. OMeadhra, W. Tabib, and N. Michael, "Variable resolution occupancy mapping using gaussian mixture models," *IEEE Robot. Autom. Letters*, vol. 4, no. 2, pp. 2015–2022, April 2019.

[17] S. Srivastava and N. Michael, "Efficient, multifidelity perceptual representations via hierarchical gaussian mixture models," *IEEE Trans. Robotics*, vol. 35, no. 1, pp. 248–260, Feb 2019.

[18] M. Corah, C. OMeadhra, K. Goel, and N. Michael, "Communication-efficient planning and mapping for multi-robot exploration in large environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1715–1721, April 2019.

[19] R. Hosseini and S. Sra, "An alternative to em for gaussian mixture models: Batch and stochastic riemannian optimization," *Mathematical Programming*, pp. 1–37, 2017.

[20] C. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer-Verlag New York, 2007.

[21] J. A. Bilmes *et al.*, "A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models," *Intl. Comput. Sci. Inst.*, vol. 4, no. 510, p. 126, 1998.

[22] B. Eckart, K. Kim, A. Troccoli, A. Kelly, and J. Kautz, "Mlmd: Maximum likelihood mixture decoupling for fast and accurate point cloud registration," in *Intl. Conf. on 3D Vision*. IEEE, 2015, pp. 241–249.

[23] S. Srivastava, "Efficient, multi-fidelity perceptual representations via hierarchical gaussian mixture models," Master's thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh PA, August 2017.

[24] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT press, 2005.

[25] J. Amanatides, A. Woo *et al.*, "A fast voxel traversal algorithm for ray tracing," in *Eurographics*, vol. 87, no. 3, 1987, pp. 3–10.

[26] J. L. Blanco and P. K. Rai, "nanoflann: a C++ header-only fork of FLANN, a library for nearest neighbor (NN) with kd-trees," https://github.com/jlblancoc/nanoflann, 2014.

[27] A. Williams, S. Barrus, R. K. Morley, and P. Shirley, "An efficient and robust ray-box intersection algorithm," in *ACM SIGGRAPH 2005 Courses*. ACM, 2005, p. 9.

[28] E. W. Weisstein, *Tetrahedron*. [Online]. Available: http://mathworld.wolfram.com/Tetrahedron.html

[29] K. Goel, M. Corah, C. Boirum, and N. Michael, "Fast exploration using multirotors: Analysis, planning, and experimentation," in *Conf. on Field and Service Robot.*, Tokyo, Japan, Aug. 2019.

[30] B. J. Julian, S. Karaman, and D. Rus, "On mutual information-based control of range sensing robots for mapping applications," *J. Intl. & Robot. Research*, vol. 33, no. 10, pp. 1357–1392, 2014.

[31] Z. Zhang, T. Henderson, V. Sze, and S. Karaman, "Fsmi: Fast computation of shannon mutual information for information-theoretic mapping," in *2019 International Conference on Robotics and Automation (ICRA)*, May 2019, pp. 6912–6918.

[32] X. Yang, K. Sreenath, and N. Michael, "A framework for efficient teleoperation via online adaptation," in *Proc. of the IEEE Intl. Conf. on Robot. and Autom.* IEEE, 2017, pp. 5948–5953.

[33] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proc. of the IEEE Intl. Conf. on Robot. and Autom.*, Shanghai, China, May 2011.

[34] A. Spitzer, X. Yang, J. Yao, A. Dhawale, K. Goel, M. Dabhi, M. Collins, C. Boirum, and N. Michael, "Fast and agile vision-based flight with teleoperation and collision avoidance on a multirotor," in *Proc. of the Intl. Sym. on Exp. Robot.* Buenos Aires, Argentina: Springer, 2018, to be published.

[35] K. Goel, M. Corah, and N. Michael, "Fast exploration using multirotors: Analysis, planning, and experimentation," The Robotics Institute, Carnegie Mellon University, Tech. Rep. CMU-RI-TR-19-03, 2019.

[36] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proc. of the Intl. Sym. on Comput. Intell. in Robot. and Autom.*, Monterey, CA, Jul. 1997.

[37] W. Tabib, M. Corah, N. Michael, and R. Whittaker, "Computationally efficient information-theoretic exploration of pits and caves," in *Proc. of the IEEE/RSJ Intl. Conf. on Intell. Robots and Syst.*, Daejeon, Korea, Oct. 2016.

[38] M. Corah and N. Michael, "Distributed matroid-constrained submodular maximization for multi-robot exploration: theory and practice," *Auton. Robots*, 2018.

[39] L. Janson, T. Hu, and M. Pavone, "Safe motion planning in unknown environments: Optimality benchmarks and tractable policies," in *Proc. of Robot.: Sci. and Syst.*, Pittsburgh, PA, Jul. 2018.

[40] T. M. Cover and J. A. Thomas, *Elements of information theory*. John Wiley & Sons, 2012.

[41] L. Contin and S. Battista, "Performance evaluation of video coding schemes working at very low bit rates," in *Proceedings of ICASSP '94. IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. v, April 1994, pp. V/409–V/412 vol.5.

[42] M. Masry and S. S. Hemami, "An analysis of subjective quality in low bit rate video," in *Proceedings 2001 International Conference on Image Processing (Cat. No.01CH37205)*, vol. 1, Oct 2001, pp. 465–468 vol.1.

[43] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Trans. Robotics*, vol. 34, no. 4, pp. 1004–1020, August 2018.

[44] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The grasp multiple micro-uav testbed," in *IEEE Robot. Autom. Mag.*, Sep. 2010.

[45] K. J. Patrick, *Pennsylvania Caves and other Rocky Roadside Wonders*. Stackpole Books, 2004.