IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

Multi-Robot Multi-Room Exploration With Geometric Cue Extraction and Circular Decomposition

Seungchan Kim, *Graduate Student Member, IEEE*, Micah Corah, *Member, IEEE*, John Keller, Graeme Best, *Member, IEEE*, and Sebastian Scherer, *Senior Member, IEEE*

***Abstract*—This work proposes an autonomous multi-robot exploration pipeline that coordinates the behaviors of robots in an indoor environment composed of multiple rooms. Contrary to simple frontier-based exploration approaches, we aim to enable robots to methodically explore and observe an unknown set of rooms in a structured building, keeping track of which rooms are already explored and sharing this information among robots to coordinate their behaviors in a distributed manner. To this end, we propose 1) a geometric cue extraction method that processes 3D point cloud data and detects the locations of potential cues such as doors and rooms, and 2) a circular decomposition for free spaces used for target assignment. Using these two components, our pipeline effectively assigns tasks among robots, and enables a methodical exploration of rooms. We evaluate the performance of our pipeline using a team of up to three aerial robots, and show that our method outperforms the baseline by 33.4% in simulation and 26.4% in real-world experiments.**

***Index Terms*—Aerial systems: Perception and autonomy, multi-robot systems, vision-based navigation.**

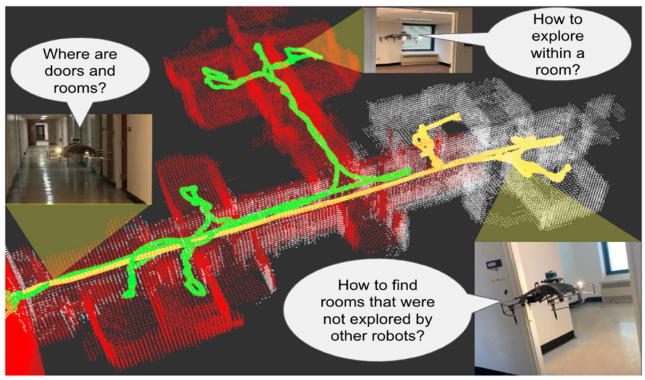

Fig. 1. We present an autonomous multi-robot exploration pipeline that coordinates the behaviors of robots exploring multiple rooms in a building. Using our pipeline, robots explore different rooms in a methodical manner. The traveled trajectory and area covered by each robot is visualized with different color.

I. Introduction

MULTI-ROBOT exploration [1], [2] in unfamiliar, unknown environments has attracted attention in the robotics research community, due to its potential to accomplish duties faster than a single robot, and its wide applicability in tasks including search & rescue operations [3], hazardous source detection in turbulent environments [4], and planetary missions [5]. Recently, in light of the DARPA Subterranean Challenge, there is a growing attention on exploration of large underground and indoor environments by teams of robots, with realistic communication constraints, sensor coverage, and compute conditions [6], [7], [8]. In this work, we aim to develop a more structured multi-robot autonomous exploration pipeline for operation in indoor environments, taking advantage of geometric properties of structures in a building.

Specifically, we aim to build algorithms for multiple robots exploring inside a building composed of multiple rooms, whose locations and sizes are not known in advance. Rather than following a frontier-based approach [9], we explicitly model the geometry of rooms to enable a methodical exploration of structural environments, empowering robots to explore the rooms one by one, observing each room in-turn before moving on to another. Furthermore, we also want coordinated behaviors such that robots select rooms avoid redundant observations by multiple robots (Fig. 1). Why do we want this type of multi-room exploration? First, rooms are usually the parts of the building where meaningful target objects are placed in, compared to corridors and hallways. Thus, they warrant focused attention during exploration. Second, rooms are non-overlapping and structural units that constitute the building; the entire indoor space can be segmented using a room-oriented search. One thing to consider is that, it is inefficient to assign multiple robots to the same small room; thus, we need a coordinated target assignment scheme for multi-robot, multi-room exploration.

To this end, we propose our method, Multi-Robot Multi-Room (MRMR). Unlike learning-based geometry prediction for exploration, our method does not require an offline dataset of representative environments to learn from. Instead, we use a simple yet effective and generalizable technique to extract

Manuscript received 25 July 2023; accepted 27 November 2023. Date of publication 13 December 2023; date of current version 26 December 2023. This letter was recommended for publication by Associate Editor F. Arrichiello and Editor M. A. Hsieh upon evaluation of the reviewers' comments. This work was supported by the Defense Science and Technology Agency Singapore. *(Corresponding author: Seungchan Kim.)*

Seungchan Kim, Micah Corah, John Keller, and Sebastian Scherer are with Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213 USA (e-mail: seungch2@andrew.cmu.edu; micahc@andrew.cmu.edu; jkeller2@andrew.cmu.edu; basti@andrew.cmu.edu).

Graeme Best is with the School of Mechanical and Mechatroninc Engineering, University of Technology Sydney, Ultimo, NSW 2007, Australia (e-mail: graeme.best@uts.edu.au).

Video: https://youtu.be/zUtK1hh2Tpo

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2023.3342553, provided by the authors.

Digital Object Identifier 10.1109/LRA.2023.3342553

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

geometric cues that indicate the potential locations of doors and rooms, only from 3D point cloud data. Our pipeline processes data observed by LiDAR sensors onboard, converting 3D point clouds into a 2D binary occupancy grid map, and then into a 2D distance transform map. Using the distance transform map, robots perform real-time geometric analysis to discover structural cues that signal the doors and rooms. Robots update global plans and execute local planners accordingly, actively searching for unreached doors and unexplored rooms.

We also propose the idea of representing free spaces within rooms with circles. We show that circular decomposition is a compact representation of the environment for robots exploring rooms and sharing necessary information with other robots via communication.

We evaluate our autonomous exploration pipeline using multiple unmanned aerial vehicles (UAV), both in simulation and real-world experiments. Built upon the multi-robot exploration and planning component of the complete autonomy stack [10] of Team Explorer, which showed most successful exploration by aerial robots in the final round of DARPA SubT Challenge competition, our methods record significant performance gain in fast discovery and exploration of multiple rooms, and coordination of behaviors for multiple robots.

In summary, our contributions are:

- An autonomous multi-robot exploration pipeline that extends prior works in [10] to coordinate behaviors of robots in a building composed of multiple rooms.
- Incorporation of two new modules for multi-robot multi-room exploration: 1) a geometric cue extraction method that detects the locations of doors and rooms from 3D LiDAR point cloud data, and 2) circular decomposition of spaces for room representation, target assignments, and communication.
- Empirical validation of our multi-robot exploration pipeline via simulated and real-world experiments.

II. Related Work

A. Multi-Robot Exploration

Multi-robot exploration problems have been studied with various approaches. Many prior works [1], [9] view this problem as an assignment of frontiers (the boundaries between known and unknown space), where robots explore environments by continuously moving toward nearby, unexplored frontiers. Other approaches include sampling-based [11], information-theoretic [12], [13], graph search-based [14], recursive tree-based search [15], and sub-map merging [16]. Recent works include hybrid approaches, such as combining frontier-based approach and graph-search [10].

Multi-robot exploration research can also be categorized by whether decision-making is centralized or decentralized. In centralized schemes [17], [18], a central entity plans out tasks for a team of robots with an access to the global information of the environment, which could hypothetically produce globally optimal solutions. However, a single failure of a robot or communication link could lead to the failure of the entire system. Instead, we follow the decentralized [19], [20] schemes, which are more robust to the single point of failure. Each robot makes decisions and optimizes trajectories based on its own understanding of the environment, with realistic communication constraints among robots.

B. Space Partitioning and Decomposition

In robotic exploration research, space partitioning approaches decompose space into subparts or partitions and seek to cover the whole space by consecutively exploring the partitions. Wu et al. [21] proposes Voronoi-based partitioning to coordinate multi-robot exploration, dividing the entire space into a set of polygons. Solanas and Garcia [22] propose using unsupervised clustering for multi-robot coordination. Hu et al. [23] uses the combination of deep reinforcement learning and Voronoi-based partitions to improve coordination strategies for multi-robot exploration.

In this work, we focus on decomposing the free space into a set of circles. Previously, Gao et al. [24] proposed generating circles from raw point clouds for safe online trajectory optimization in cluttered environments. Ren et al. [25] improves this idea by generating corridors with larger sphere volumes and receding schemes that enable high-speed trajectory planning. Most recently, Musil et al. [26] demonstrates incrementally-built segmented graph of spheres enable safe and flexible flights. [26] also shows the compressed version of the graph structure enables efficient communication in multi-robot setting. While the previous works used the circular or spherical decomposition in the context of safe trajectory planning, we focus on generating circular representations in the context of indoor room exploration, coupled with geometric cue extraction from distance transform map; we also focus on the multi-robot exploration, showing circular representation is an effective information scheme to be shared for decentralized behavior coordination of multiple robots.

C. Room Detection

Division of a floorplan into rooms, or identifying the potential location and size of rooms is essential to room-based search and exploration. Most popular approach to detect rooms is Voronoi-based approach by Thrun [27], which utilizes distance transform and Voronoi graph to find critical points in the map to detect passages to the room. Wurm et al. [28] uses this idea to divide the space map into segments that can correspond to individual rooms, and generate Voronoi graph to assign targets for each robot in the multi-robot teams, in a centralized manner.

Other works include graph-based partitioning which uses topological map to cluster nodes and build higher-level hierarchical map [29], sparse graph-based approach to build 3D skeleton diagrams using 2D Voronoi Diagram [30], or feature-based room segmentation [31] which learns features and geometric shapes that match with the characteristic of rooms. More recently, scene graph-based methods were proposed: Bavle et al. [32] proposed factor-graph with hierarchical layers and room segmentation scheme using 3D LiDAR point clouds; Rosinol et al. [33] proposed 3D dynamics scene graph, with multiple layers in a hierarchy representing different levels of semantic structures including rooms, followed by Hughes et al. [34] which features improved hierarchical scene graph generated real-time. In this work, we focus on LiDAR processing approach that enables high-speed, low-compute processing onboard and robust exploration in light-degraded environments, as opposed to learning-based approaches that often require a large amount of computations and pretraining on offline datasets. We revisit the idea of distance transform [27], but improve this idea by using a different cue detection method and space decompositions that are compatible with state-of-the-art fully autonomous multi-robot exploration algorithms.

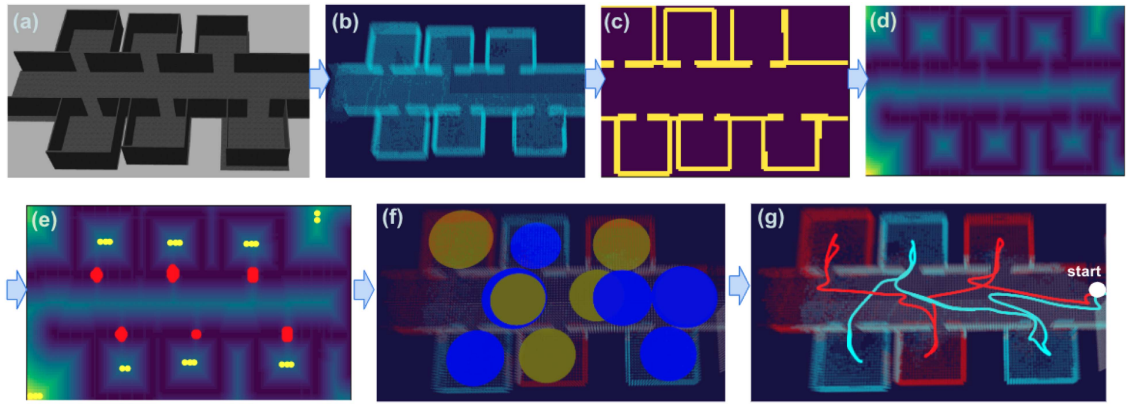

Fig. 2. Overview of our exploration pipeline. (a) An indoor environment composed of multiple rooms. (b) The robot generates 3D occupancy voxel grid map from 3D point cloud data. (c) It flattens 3D voxel gid map into 2D binary map, and applies median filtering. (d) The robot generates 2D distance transform map (e) It computes saddle points (red) and local maxima (yellow) from the distance transform map. (f) Free space is decomposed into circular representations. In multi-robot setting, the robot shares the circles it explored (blue), and the circles that other robots have explored (yellow circles). (g) Trajectories traveled by two robots are visualized with different colors, which are results of the coordinated behaviors via shared circular representations.

III. Problem Definition

Consider a team of n robots ($i = 1, 2, \ldots, n$) that are exploring an indoor environment. Denote the trajectory traveled by each robot as ξ_i. The robots build a map of the environment with LiDAR, which is represented with a 3D occupancy voxel grid in a shared coordinate system (each voxel represents a cube with side length 0.2 m). The robots observe surfaces of the environment with a primary sensor, such as a camera or RGBD sensor with limited field of view. The intersection between the occupied voxels of the LiDAR map and the primary sensor field of view is denoted by the voxel grid O_i (for robot i). The observation model for the primary sensor can be defined based on the specifications of the robot and the application; we model the primary sensor as a forward-facing fish-eye camera with a 5 m detection range, and 170° field of view, modelling occlusions with ray casting, as in [10].

Let us assume that there are total K rooms to be explored in the building, and denote V_j^{rm} ($j = 1, 2, \ldots, K$) as voxels within each room. Then, the total number of voxels in these rooms that are observed by the primary sensor of robot i is

$$\left| \bigcup_{j=1,\ldots,K} (O_i \cap V_j^{\text{rm}}) \right| \tag{1}$$

($|\cdot|$ is the number of voxels). In this work, we will reward robots for exploring rooms in a building rather than increasing total coverage by traversing corridors and hallways.

In the multi-robot setting, our objective is to maximize the union of such voxels, explored by all robots collectively. Each robot finds its own trajectory ξ_i seeking to maximize the union of observed voxels in rooms by all robots,

$$\xi_1^*, \xi_2^*, \ldots, \xi_n^* = \underset{\xi_1,\xi_2,\ldots,\xi_n}{\arg\max} \left| \bigcup_{i=1..n} \Big(\bigcup_{j=1..K} (O_i \cap V_j^{\text{rm}}) \Big) \right| \tag{2}$$

given a fixed time. The robots will solve (2) approximately and in a decentralized manner, so that each robot will plan its own path ξ_i^* while communicating with other robots.

IV. Distributed Exploration Method

In this section, we present our autonomous distributed exploration pipeline, which coordinates the behaviors of multiple robots exploring in a building composed of multiple rooms. The pipeline overview is displayed in Fig. 2. We first explain preliminary background on the autonomous robot exploration baseline [10], which we build upon and improve (Section IV-A). Then, we describe the two main building blocks of our method, geometric cue extraction (Section IV-B, Algorithm 1) and circular decomposition of free space (Section IV-C, Algorithm 2). We also explain the multi-robot communication and target assignment (Section IV-D, Algorithm 3), and finally explain how they all come together to form our Multi-Robot Multi-Room (MRMR) method (Section IV-E, Algorithm 4).

A. Preliminary: Autonomous Exploration Baseline

The baseline method we use is the open source autonomous aerial robot exploration pipeline [10] developed by Team Explorer, to compete in DARPA Subterranean Challenge. This baseline enables robots to navigate in a wide range of challenging underground or indoor environments such as a mine, subway, tunnel, or cave, with limited communication, sensor coverage, and light conditions.

The baseline method leverages LiDAR sensors to discover the geometry of surrounding environments, using OpenVDB [35] as a data structure for representing map occupancy grids, and SLAM solutions generated by Super Odometry [36]. As the robot moves through the environment, it estimates which voxels have been visually observed according to the camera model in Section III. with an OpenVDB map.

The path planning component of the baseline method, which we focus on in this work, can be loosely categorized as a frontier-based exploration approach with graph search and selection heuristics. Using vision and range sensors, the robot generates a set of *viewpoints* at the frontiers. The viewpoints are scored with heuristics, and the viewpoints with high scores are selected. Then, paths are planned to reach the viewpoints. RRT-Connect [37] is used to find a feasible global path to

Algorithm 1: ExtractCues.

Input: Voxel Grid Map Data O

2D binary map $B \leftarrow$ Take O where $z \in [z_{\text{low}}, z_{\text{high}}]$

Distance map $M \leftarrow$ *distanceTransform(filter*(B)*)*

Hessian matrix $H \leftarrow Hessian(M)$

Obtain two lists $P^{\text{sadd}}, P^{\text{max}} \leftarrow$ from H using (4)

$\Delta = [\,]$. $\forall c \in P^{\text{max}}$, distance $\delta \leftarrow M(c), \Delta$.append$(\delta)$

Output: $P^{\text{sadd}}, (P^{\text{max}}, \Delta)$

viewpoints, and A* graph search and motion primitives are used for local path planning.

For multi-robot exploration, onboard communication hardware and DDS networking are used. The plans for robots are coordinated implicitly by sharing knowledge of the world. This includes knowing the take-off locations of each robot, and communicated shared map in a global reference frame.

B. Extracting Geometric Cues of Doors and Rooms

The first component of our method is detection of doors and rooms via geometric cue extraction, as shown in Algorithm 1. The intuition for this algorithm is that *saddle points* on the distance transform are approximately equivalent to the locations of doors. We also extract local maxima and their distances to closest wall to decompose free spaces.

The robot incrementally obtains 3D point cloud observations from onboard sensors which it uses to maintain a voxel grid map. The 3D voxel grid map O is flattened into 2D binary map B (occupied cells: 1, unknown/free cells: 0) by setting the point in 2D as occupied if any voxel within the range of height $z \in [z_{\text{low}}, z_{\text{high}}]$ is occupied. In practice, we set $z_{\text{low}} = 0$ and $z_{\text{high}} = 1.8$. The robot applies median filtering[1] to the binary map B, and then converts B into 2D distance transform map M, which computes the minimal distance to closest occupied cell for all pixels. Then the robot obtains a second-order partial derivative matrix of the distance map, or hessian H, from which it can compute saddle points P^{sadd} and local maxima P^{max}, using the determinant[2]:

$$\det(x, y) = f_{\text{xx}}(x, y) f_{\text{yy}}(x, y) - (f_{\text{xy}}(x, y))^2, \tag{3}$$

and with P^{sadd} and P^{max} defined as

$$\begin{cases} (x, y) \in P^{\text{sadd}} \text{ if } \det(x, y) < -0.1 \\ (x, y) \in P^{\text{max}} \text{ if } \det(x, y) > 0 \;\&\; f_{\text{xx}}(x, y) < -0.1 \end{cases} \tag{4}$$

A local maximum $m \in P^{\text{max}}$ is a point, whose distance $\delta \in \Delta$ to the wall is the highest compared to other neighboring pixels. Pairs of (m, δ) will be used as centers and radii of circles, in the free space decomposition.

The more interesting part is the use of saddle points (Fig. 3). A saddle point refers to a critical point that is not a local extremum; any point that is a relative minimum along one axis and relative maximum along another perpendicular axis is classified as saddle points. In our setting, a door (yellow dot in Fig. 3(a)) can be classified as a saddle point,[3] because it is farthest from occupied cells(walls) compared to other points placed along the wall-axis (red), while it is closest to the occupied cells(walls) compared to other free points placed along the perpendicular axis (blue).

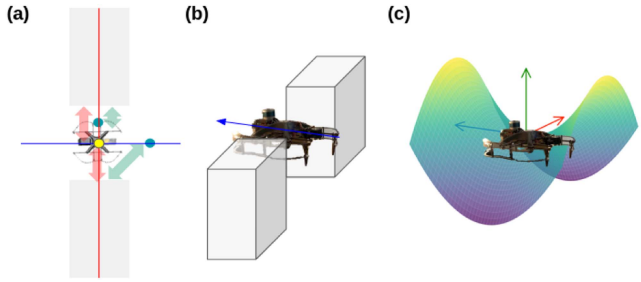

Fig. 3. (a) Saddle point (yellow) is farthest from occupied cells compared to other points along the wall axis (red), while closest to the occupied cells compared to other points along the perpendicular axis (blue). (b) Side-view of drone at a door. (c) Drone at saddle point in distance transform space.

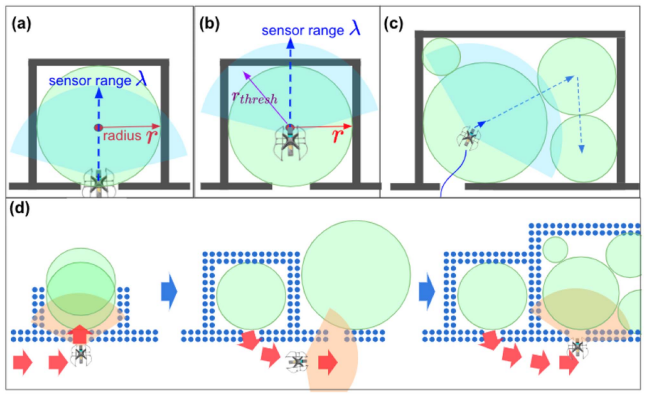

Fig. 4. (a) We represent a room using a circle for robot observation. (b) We let the radius of the circle, r, is smaller than the sensor range λ for effective observations. (c) A larger room is represented by multiple circles. (d) The robot updates, merges, splits the circles with new sensor information.

C. Circular Decomposition of Free Space

We represent the free space in rooms by decomposing it into circles. The motivation behind this is that visiting the centers of the circles will enable the robot to observe most of the room. Due to the limited camera FoV, some parts may be missed. However, in practice, the robot usually turns when going to and from the centers. If full visual coverage is particularly important, a spinning behavior can be added at each center, but we found this unnecessary in the considered scenarios. We note 360° FoV can also help for full coverage.

As in Fig. 4(a), (b), we can generate a circle that is tangent to the inner wall of the room, and make the robot reach the center of the circle to observe the surroundings with a camera of sensor range λ; during this process, it is important to maintain that radius r is smaller than a threshold value. For example, if $r > \lambda$, even if the robot goes to the center, there must exist a part

[1]Median filtering of kernel size [1,3] and [3,1]

[2]We empirically found that the condition $\det(x, y) < -0.1$ (instead of $\det(x, y) < 0$) works well in practice to find saddle points. Similarly, we set $f_{\text{xx}}(x, y) < -0.1$ as a threshold for detecting local maxima.

[3]In practice, multiple saddle points can be detected for a single door; however, points within distance $\epsilon_d = 1.0\tilde{}m$ are handled the same.

Algorithm 2: UpdateCircles.

Input: $C, (P^{\max}, \Delta)$
for each (m, δ) pair: $\triangleright m \in P^{\max}, \delta \in \Delta$
  $c \leftarrow GenerateCircle(m, \delta)$
  for each $c' \in C$:
    $r_1 \leftarrow \delta, r_2 \leftarrow c'.r, d \leftarrow dist(c.center, c'.center)$
    if $d < 0.5(r_1 + r_2)$: $C \leftarrow C \cup \{c\}$ if $r1 > r2$
    elif $d < \epsilon(r_1 + r_2)$:
      $c'' \leftarrow Merge(c, c'); C \leftarrow C \cup \{c''\}$
  if not $c.merged$: $C \leftarrow C \cup \{c\}$
  $\forall c\_ \in C, SplitCircle(c\_)$ if $c\_.r > r_{thresh}$
Output: C

in circle that is not covered by the sensor even within the FoV. In practice, we set the radius threshold for circle-split $r_{thresh} = 2.5$ m ($r \leq r_{thresh}$), which is admissible with our sensor range $\lambda = 5.0$ m. In a larger room like Fig. 4(c), the Algorithm 1 often outputs a longer distance $\delta > \lambda$. In this case, we split the circle into a set of circles with smaller radius r, and let the robot reach the centers of smaller circles one by one.

The circular decomposition of the space is not static, as the robot updates the map by incrementally obtaining more information of the surrounding (e.g. discovers a new wall). As shown in Fig. 4(d), the robot updates the size of the circles so that they adjoin the newly observed walls, merge the circles if they overlap significantly, and split the circles if their radii are above the threshold r_{thresh}. This process of maintaining and updating a set of circles is shown in the Algorithm 2. Given each pair of local maxima and distance, the robot generates a circle c of center m and radius $r_1 = \delta$, which is compared with the circles c' of center c_t and radius r_2 in the previous set of circles C. If they overlap too closely, we save the larger one between the two; if distance d between centers of c and c' are smaller than $\epsilon(r_1 + r_2)$ where $\epsilon = 0.95$, we merge these two into a new circle c''.[4] We simply save c if this isn't merged to any of the previous circles in C. Lastly, we split the circles if the circle's radius is above the threshold $r_{thresh} = 2.5$, by setting the first split circle's radius as 2.5 and the other circle to be adjoining to the first one.

D. Multi-Robot Communication and Target Assignment

As discussed in Section IV-B, the robot extracts geometric cues and detects saddle points in 2D as potential doors, and it also represents rooms with circular decomposition of spaces as discussed in Section IV-C. We send this 2D saddle points and centers of circle to 3D, by simply setting[5] their z value as 1.0, and update the set of doors D and circles C. For distributed target assignment among multiple robots, we utilize the doors and circles as information to be shared. Each robot not only maintains D and C, but also updates a set of doors D_{r} and circles C_{r} it has reached; as shown in Communication() function of Algorithm 3, $C_{\mathrm{r}}, D_{\mathrm{r}}$ are shared with other robots bidirectionally. Receiving $C_{\mathrm{r}}, D_{\mathrm{r}}$ from other robots, the robot updates a set of doors and circles, $C_{\mathrm{o}}, D_{\mathrm{o}}$, that were reached by other robots. The information shared for communication, D_{r} and C_{r}, are just a set of points in 3D coordinates and a set of pairs of point and radius.

[4]Although the merging methods vary, we empirically find that setting new center as $\frac{(r_1 \cdot m + r_2 \cdot c_t)}{(r_1 + r_2)}$ and radius as $\frac{(r_1 + r_2 + d) \cdot d}{2(r_1 + r_2)}$ effective.

[5]Any z around the middle of the range (z_{low}, z_{high}) is fine.

Algorithm 3: TargetDoor, TargetCircle, Communication.

Input: C, D, C_{r}, D_{r}, C_{o}, D_{o}, $\epsilon_d = 1.0, \epsilon_c = 1.5$
$D \leftarrow D \setminus D_{\mathrm{r}}, C \leftarrow C \setminus C_{\mathrm{r}}$
def TargetDoor():
for each $d \in D$:
  if $\exists d' \in D_{\mathrm{o}}$ $s.t.$ $dist(d, d') < \epsilon_d$: **then** $D \leftarrow D \setminus \{d\}$
  $d_{near} = \arg\min_{d \in D} dist(robot, d)$; **return** d_{near}
def TargetCircle():
for each $c \in C$:
  if $\exists c' \in C_o$ $s.t.$ $dist(c.center, c'.center) < \epsilon_c \cdot c'.r$:
    then $C \leftarrow C \setminus \{c\}$
  $c_{near} = \arg\min_{c \in C} dist(robot, c)$; **return** c_{near}
def Communication(C_r, D_r):
  $robot.Publish(C_r, D_r)$
  $C_o, D_o \leftarrow robot.Subscribe(other\_robots'\ C_r, D_r)$
  return C_o, D_o

Algorithm 4: Multi-robot Multi-room (MRMR) exploration. Algorithm is run on each robot at 10 Hz.

Input: Initialize $O, C, D, C_{\mathrm{r}}, D_{\mathrm{r}}, C_{\mathrm{o}}, D_{\mathrm{o}}$ with empty sets $\{\}$
for each timestep:
  $P^{\text{sadd}}, (P^{\max}, \Delta) \leftarrow$ **ExtractCues**(O)
  $D \leftarrow P^{\text{sadd}}$; $C \leftarrow$**UpdateCircles**$(C, (P^{\max}, \Delta))$
  if not $robot.is\_targeting$:
  $d \leftarrow$ **TargetDoor**(); $robot.is\_targeting \leftarrow True$
  $MoveTo(d)$
  if $robot.reached\_the\_target\_door$:
  $D_{\mathrm{r}} \leftarrow D_{\mathrm{r}} \cup \{d\}$
  $c \leftarrow$ **TargetCircle**(); $MoveTo(c.center)$
  while $\exists c'$ $s.t.$ $dist(c'.center, c.center) < \mu(c'.r + c.r)$
    $c \leftarrow$ **TargetCircle**(); $MoveTo(c.center)$
  $robot.is\_targeting \leftarrow False$
  for $\forall c$ $s.t.$ $c.is\_reached$: $C_{\mathrm{r}} \leftarrow C_{\mathrm{r}} \cup \{c\}$
  $C_{\mathrm{o}}, D_{\mathrm{o}} \leftarrow$ **Communication**(C_r, D_r)
Output: Path trajectory traveled by robot

This compact representation enables efficient communication among robots.

When targeting a new door, a robot not only excludes doors D_{r} reached by itself but also doors D_{o} reached by other robots. Any candidate door $d \in D$ that is distanced less than ϵ_d from any of the doors reached by other robots D_{o} is excluded. This is possible because the robots share a common global coordinate frame. Likewise, when targeting a new circle, a robot excludes circles reached by itself and other robots, both C_{r} and C_{o}. The details are explained in TargetDoor() and TargetCircle() methods in Algorithm 3.

E. Multi-Robot Multi-Room (MRMR) Exploration

Here, we explain how the previous algorithms come together and fit into one method, Multi-Robot Multi-Room (MRMR) exploration. The detailed procedure is in Algorithm 4. Each timestep is a fixed timestep of 0.1 s. At every timestep, the robot extracts geometric cues (saddle points, local maxima, and distances to the walls), and updates the circular decomposition of the space. Then the robot targets a door, moves to the door, and explores a room by targeting a circle that composes the

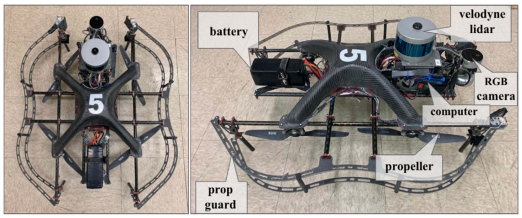

Fig. 5. Aerial robot system hardware. Left: top-view, Right: side-view. Custom-built quadrotor robot equipped with sensors and computer onboard.

TABLE I
DETAILS OF THE SIMULATION TEST ENVIRONMENTS

Name	Images	Rooms	Volume(m^3)	Voxels
Env1		6	462.6	9587
Env2		8	776.7	21313
Env3		5	402.9	11869
Env4		9	729.1	20105
Env5		12	1179.1	33371
Env6		9	1226.9	38807
Env7		10	932.4	23734
Env8		12	1024.9	31082

room. When there exists a circle c' that is adjacent to the current circle c (we set $\mu = 1.1$, and if distance between $c'.center$ and $c.center$ is smaller than $\mu(c'.r + c.r)$, we regard them adjacent), we set it as a next target circle. This is effective when exploring a larger type of room, composed of multiple adjacent circles. After completing this, the robot marks the doors and circles as reached, and shares this information $D_{\mathrm{r}}, C_{\mathrm{r}}$ with other robots to avoid re-targeting of doors and circles that were already explored. This sharing of information through communication enables effective distributed target assignment.

V. EXPERIMENTS AND RESULTS

In this section, we explain the experimental setup and display results to evaluate our algorithm, MRMR. We first elaborate on the experimental setup and discuss results for simulation environments. Then we discuss the experimental setup and results for real-robot experiments. For comparison, we chose the previous frontier-based exploration [10] as a baseline, both in simulation and real-robot experiments.

A. Simulation Experiments

1) Experimental Setup: We used the SubT UAV code released by [10], developed to simulate real subterranean environments. We designed eight indoor simulation environments with different configurations; Table I displays the details of each environment. First five environments (Env1 to Env5) feature simple, idealized settings with doors, rooms and corridors. Env6 features larger sizes of rooms. Env7 includes objects (e.g. table, bookshelf, cabinet) in each room. Lastly, Env8 includes a scenario where the wall-axes are not aligned with x,y-axis of the global coordinate frame.

TABLE II
SUMMARY OF SIMULATION EXPERIMENTS

Method	1 Robot		2 Robots		3 Robots	
	Vxl.	Rm.	Vxl.	Rm.	Vxl.	Rm.
Baseline	27.15%	19.17%	39.08%	41.14%	52.88%	58.76%
MRMR	**35.57%**	**34.58%**	**55.43%**	**51.49%**	**67.4%**	**66.68%**
Improvement	31.01%	80.38%	41.84%	25.16%	27.45%	13.48%

The bold values mean better performance.

We tested with different numbers of robots ($n = 1, 2, 3$) and measured the number of voxels collectively observed by the robots' camera sensors over 2 minutes. Each graph curve is an average of three independent runs. We also measured percentages of observed voxels (out of total voxels in rooms) and number of rooms (out of total number of rooms) by each method, and displayed the results in a separate table.

2) Results: The results of the simulation experiments are displayed in Fig. 6. The plots in the first row of the figure are single-robot case, and the plots in the second and third row are multi-robot cases (2-robots and 3-robots). These plots demonstrate that our Method (MRMR) observes more voxels in the rooms than frontier-based baseline, across different environments and different number of robots. In average, MRMR observes $31.01\%, 41.84\%, 27.45\%$ more room grid voxels than baseline in single-robot, two-robots, and three-robots cases. In the Table II, we reported how MRMR observes more voxels (out of total room voxels; denoted Vxl.) and explore more rooms (out of total number of rooms; denoted Rm.) than baseline; MRMR outperforms the baseline both in terms of voxels and number of rooms. MRMR shows larger outperformance compared to baseline in the case of two-robots than three-robots. While outperformance is still significant, adding more robots to the same space results in necessary overlap of their sensor coverage.

More qualitative explanations of difference between the baseline and MRMR are following: as shown in Fig. 7(a), using the baseline, robots first choose to travel corridors fast by moving straight to increase the coverage area without turning to the doors or rooms. In contrary, as in Fig. 7(b), the robots quickly turn directions to reach the doors and enter rooms, rather than traveling along the corridor. Using the baseline (Fig. 7(c), (e)), multiple robots often enter the same rooms redundantly and sometimes miss exploring rooms; in contrary our method (Fig. 7(d), (f)) enables robots to uniquely visit each room without redundancy.

We briefly report the measured runtime of new modules. The average runtimes of distance transform, geometric cue extraction, and circular decomposition are $480.5\,\mu$s, 1.962 ms, and $35.8\,\mu$s. We also report the accuracy of door detection using saddle points extraction: The average precision and recall over all environments are 0.94 and 0.97.

Lastly, we discuss the behaviors of robots in challenging environments. When objects are in rooms, (Fig. 8(a)), robots generate circles that are tangential to the objects, as these objects serve as occupied cells when generating 2D distance transform map. When wall axes are not aligned with global frame (Fig 8(b)), our saddle point detection is still robust and accurate at detecting doors. In the setting with a larger room (Fig. 8(c)), the robot generates multiple circles and reaches the centers of the circles one by one to cover the room.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

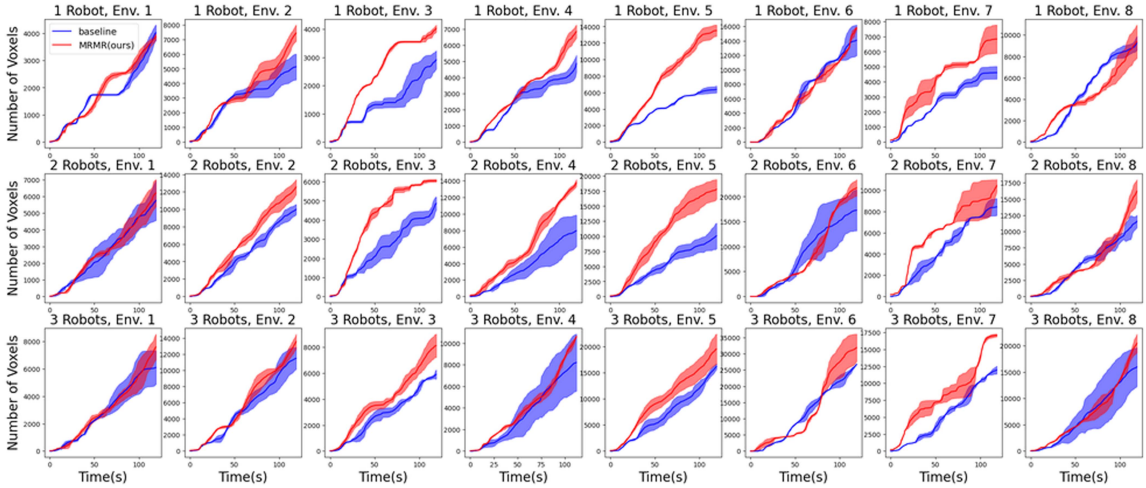

Fig. 6. Comparison of baseline and our method (MRMR). The first, second, and third row is 1-robot, 2-robots, and 3-robots case, respectively, and each column represents each different environment. Results show that our MRMR method observes more voxels in rooms than the baseline.

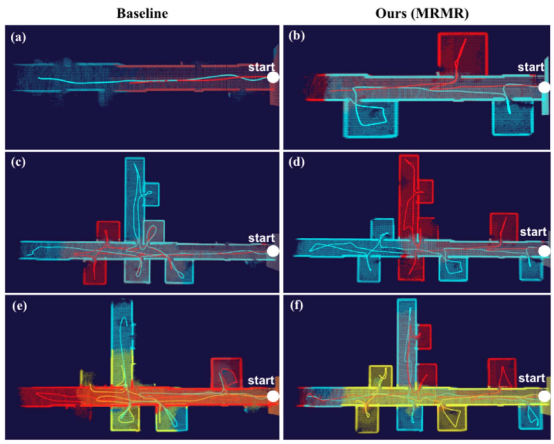

Fig. 7. Visualization of baseline's and our method's behaviors. (a) Using the baseline, the robots travel the corridor first by moving straight, while (b) using our method, robots quickly turn directions to find doors and enter the rooms. (c),(e) Multiple robots often enter the same rooms redundantly. (d),(f) Each room is uniquely visited by each robot using our method.

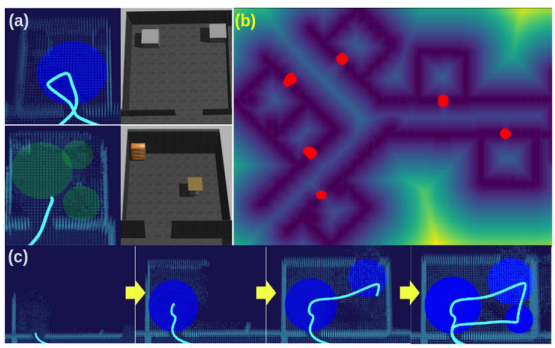

Fig. 8. (a) When objects are in rooms, the robot generates circles that adjoin the objects. (b) Our saddle point detection is robust to the case where door axis is not aligned with global coordinate frame. (c) Robot generates multiple circles while exploring a large room.

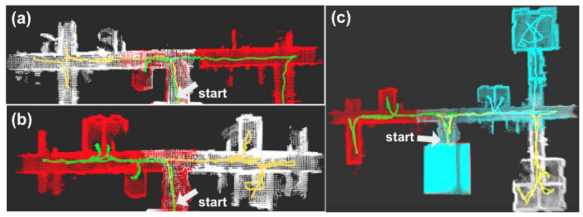

Fig. 9. Visualization of covered areas and traveled trajectories in real-robot experiments: (a) baseline (two-robots); (b) MRMR (two-robots). MRMR observes more rooms than baseline. (c) MRMR (three-robots).

B. Real-Robot Experiments

1) Experimental Setup: For real-world experiments, we use custom quadrotors as shown in Fig. 5. Each drone is 68 cm wide, 81 cm long, weighs 5.2 kg, and is equipped with a Velodyne VLP-16 Lidar, Realsense RGBD sensors, Rajant DX2, Intel NUC computer with Intel Core i7-8550 U CPU. No additional data filtering or clean-ups are required compared to simulation. We run experiments in an abandoned hospital in Pittsburgh, PA. The main results compare our method to the baseline with two to three robots; drones fly sequentially as the mechanism for inter-robot collision avoidance for real robots is limited. Additionally, we report results of a trial with three robots flying simultaneously with our method only. We report the number of voxels (Vxl.) and number of rooms (Rm.) collectively observed by the robots for both experiments.

2) Results: The results are displayed in the Table III. In the comparative tests, both in two-robots and three-robots cases, the robots generally observe more voxels and explore more rooms using MRMR. The exception is trial 2 of the two-robot case where the baseline outperforms MRMR but by an insignificant amount. On average, MRMR observes $30.66\%, 22.09\%$ more voxels in rooms than the baseline, in two-robot and three-robot cases respectively. Fig. 9(a), (b) are the results of running both methods with two robots. Finally, in the simultaneous execution experiment (Fig. 9(c)), the robots successfully explored 8 rooms

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

TABLE III
SUMMARY OF REAL-ROBOT EXPERIMENTS OVER THREE TRIALS

#	2 Robots				3 Robots			
	Baseline		MRMR		Baseline		MRMR	
	Vxl.	Rm.	Vxl.	Rm.	Vxl.	Rm.	Vxl.	Rm.
1.	6204	3	**10285**	**5**	11584	5	**15732**	**8**
2.	**10721**	5	9428	5	14336	7	**17320**	**9**
3.	8437	4	**11724**	**6**	14561	7	**15976**	**8**

The bold values mean better performance.

using MRMR, observing 16793 voxels in the rooms, with three drones.

VI. CONCLUSION AND FUTURE WORK

We proposed a multi-robot multi-room autonomous exploration pipeline (MRMR) that methodically explores rooms in a building and coordinates behaviors of robots. To this end, we presented a geometric cue extraction method that detects locations of doors and rooms from point clouds, and circular decomposition of spaces for target assignment. We validated performance of our method in simulated and real experiments. Some limitations are that the approach is only applicable to single story buildings due to flattening of 3D voxels into 2D distance transform, and that the collision avoidance among aerial robots is implicit, rather than explicit. For future work, we plan to extend our approach to multi-floor buildings and to consider exploration with heterogeneous multi-robot systems and to improve such systems with advanced vision and learning modules.

REFERENCES

[1] W. Burgard, M. Moors, C. Stachniss, and F. E. Schneider, "Coordinated multi-robot exploration," *IEEE Trans. Robot.*, vol. 21, no. 3, pp. 376–386, Jun. 2005.

[2] J. Braga, A. P. Aguiar, and J. B. de Sousa, "Coordinated Multi-UAV exploration strategy for large areas with communication constrains," in *Proc. 3rd Iberian Robot. Conf.*, 2017, pp. 149–160.

[3] J. P. Queralta et al., "Collaborative multi-robot search and rescue: Planning, coordination, perception, and active vision," *IEEE Access*, vol. 8, pp. 191617–191643, 2020.

[4] B. Ristic et al., "Autonomous multi-robot search for a hazardous source in a turbulent environment," *Sensors*, vol. 17, no. 4, 2017, Art. no. 918.

[5] M. J. Schuster et al., "Towards autonomous planetary exploration," *J. Intell. Robotic Syst.*, vol. 93, pp. 461–494, 2019.

[6] M. Kulkarni et al., "Autonomous teamed exploration of subterranean environments using legged and aerial robots," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2022, pp. 3306–3313.

[7] S. Scherer et al., "Resilient and modular subterranean exploration with a team of roving and flying robots," *Field Robot. J.*, vol. 2, pp. 678–734, May 2022.

[8] A. Agha et al., "NeBula: TEAM CoSTAR's robotic autonomy solution that won phase II of DARPA subterranean challenge," *Field Robot.*, vol. 2, pp. 1432–1506, 2022.

[9] B. Yamauchi, "Frontier-based exploration using multiple robots," in *Proc. 2nd Int. Conf. Auton. Agents*, 1998, pp. 47–53.

[10] G. Best, R. Garg, J. Keller, G. A. Hollinger, and S. Scherer, "Resilient multi-sensor exploration of multifarious environments with a team of aerial robots," in *Proc. Robot.: Sci. Syst.*, 2022.

[11] G. A. Hollinger and G. S. Sukhatme, "Sampling-based motion planning for robotic information gathering," in *Proc. Robot.: Sci. Syst.*, 2013, pp. 1–8.

[12] B. Charrow, *Information-Theoretic Active Perception for Multi-Robot Teams*. Philadelphia, PA, USA: Univ. Pennsylvania, 2015.

[13] M. Corah and N. Michael, "Distributed matroid-constrained submodular maximization for multi-robot exploration: Theory and practice," *Auton. Robots*, vol. 43, pp. 485–501, 2019.

[14] T. Dang, M. Tranzatto, S. Khattak, F. Mascarich, K. Alexis, and M. Hutter, "Graph-based subterranean exploration path planning using aerial and legged robots," *J. Field Robot.*, vol. 37, no. 8, pp. 1363–1388, 2020.

[15] Y. Sung and P. Tokekar, "A competitive algorithm for online multi-robot exploration of a translating plume," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 3391–3397.

[16] J. Yan et al., "MUI-TARE: Cooperative multi-agent exploration with unknown initial position," *IEEE Robot. Automat. Lett.*, vol. 8, no. 7, pp. 4299–4306, 2023.

[17] R. Luna and K. E. Bekris, "Efficient and complete centralized multi-robot path planning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2011, pp. 3268–3275.

[18] F. Gul et al., "A centralized strategy for multi-agent exploration," *IEEE Access*, vol. 10, pp. 126871–126884, 2022.

[19] Q. Li, F. Gama, A. Ribeiro, and A. Prorok, "Graph neural networks for decentralized multi-robot path planning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 11785–11792.

[20] Y. Zhai, B. Ding, X. Liu, H. Jia, Y. Zhao, and J. Luo, "Decentralized multi-robot collision avoidance in complex scenarios with selective communication," *IEEE Robot. Automat. Lett.*, vol. 6, no. 4, pp. 8379–8386, Oct. 2021.

[21] L. Wu et al., "Voronoi-based space partitioning for coordinated multi-robot exploration," *J. Phys. Agents*, vol. 1, no. 1, pp. 37–44, Sep. 2007.

[22] A. Solanas and M. A. Garcia, "Coordinated multi-robot exploration through unsupervised clustering of unknown space," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2004, pp. 717–721.

[23] J. Hu, H. Niu, J. Carrasco, B. Lennox, and F. Arvin, "Voronoi-based multi-robot autonomous exploration in unknown environments via deep reinforcement learning," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 14413–14423, Dec. 2020.

[24] F. Gao, W. Wu, W. Gao, and S. Shen, "Flying on point clouds: Online trajectory generation and autonomous navigation for quadrotors in cluttered environments," *J. Field Robot.*, vol. 36, no. 4, pp. 710–733, 2019.

[25] Y. Ren et al., "Bubble planner: Planning high-speed smooth quadrotor trajectories using receding corridors," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 6332–6339.

[26] T. Musil, M. Petrlık, and M. Saska, "SphereMap: Dynamic multi-layer graph structure for rapid safety-aware UAV planning," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 11007–11014, Oct. 2022.

[27] S. Thrun, "Learning metric-topological maps for indoor mobile robot navigation," *Artif. Intell.*, vol. 99, no. 1, pp. 21–71, 1998.

[28] K. M. Wurm, C. Stachniss, and W. Burgard, "Coordinated multi-robot exploration using a segmentation of the environment," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2008, pp. 1160–1165.

[29] E. Brunskill, T. Kollar, and N. Roy, "Topological mapping using spectral clustering and classification," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2007, pp. 3491–3496.

[30] H. Oleynikova, Z. Taylor, R. Siegwart, and J. Nieto, "Sparse 3D topological graphs for micro-aerial vehicle planning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 1–9.

[31] K. Sjöö, "Semantic map segmentation using function-based energy maximization," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2012, pp. 4066–4073.

[32] H. Bavle, J. L. Sánchez-López, M. Shaheer, J. Civera, and H. Voos, "S-graphs+: Real-time localization and mapping leveraging hierarchical representations," *IEEE Robot. Autom. Lett.*, vol. 8, no. 8, pp. 4927–4934, 2023.

[33] A. Rosinol, A. Gupta, M. Abate, J. Shi, and L. Carlone, "3D dynamic scene graphs: Actionable spatial perception with places, objects, and humans," in *Proc. 16th Robot.: Sci. Syst.*, 2020.

[34] N. Hughes, Y. Chang, and L. Carlone, "Hydra: A real-time spatial perception system for 3D scene graph construction and optimization," in *Proc. 18th Robot.: Sci. Syst.*, 2022.

[35] K. Museth, "VDB: High-resolution sparse volumes with dynamic topology," *ACM Trans. Graph.*, vol. 32, no. 3, pp. 1–22, 2013.

[36] S. Zhao, H. Zhang, P. Wang, L. Nogueira, and S. Scherer, "Super odometry: IMU-centric LiDAR-Visual-inertial estimator for challenging environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 8729–8736.

[37] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2000, pp. 995–1001.