Active Autonomous Aerial Exploration for Ground Robot Path Planning

Article *in* IEEE Robotics and Automation Letters · January 2017

DOI: 10.1109/LRA.2017.2651163

4 authors:

Jeffrey Delmerico
University of Zurich
28 PUBLICATIONS **2,181** CITATIONS
SEE PROFILE

Elias Mueggler
University of Zurich
28 PUBLICATIONS **3,148** CITATIONS
SEE PROFILE

Julia Nitsch
ETH Zurich
9 PUBLICATIONS **222** CITATIONS
SEE PROFILE

Davide Scaramuzza
University of Zurich
396 PUBLICATIONS **40,456** CITATIONS
SEE PROFILE

Active Autonomous Aerial Exploration for Ground Robot Path Planning

Jeffrey Delmerico, Elias Mueggler, Julia Nitsch and Davide Scaramuzza

***Abstract*—We address the problem of planning a path for a ground robot through unknown terrain, using observations from a flying robot. In search and rescue missions, which are our target scenarios, the time from arrival at the disaster site to the delivery of aid is critically important. Previous works required exhaustive exploration before path planning, which is time-consuming but eventually leads to an optimal path for the ground robot. Instead, we propose active exploration of the environment, where the flying robot chooses regions to map in a way that optimizes the overall response time of the system, which is the combined time for the air and ground robots to execute their missions. In our approach, we estimate terrain classes throughout our terrain map, and we also add elevation information in areas where the active exploration algorithm has chosen to perform 3D reconstruction. This terrain information is used to estimate feasible and efficient paths for the ground robot. By exploring the environment actively, we achieve superior response times compared to both exhaustive and greedy exploration strategies. We demonstrate the performance and capabilities of the proposed system in simulated and real-world outdoor experiments. To the best of our knowledge, this is the first work to address ground robot path planning using active aerial exploration.**

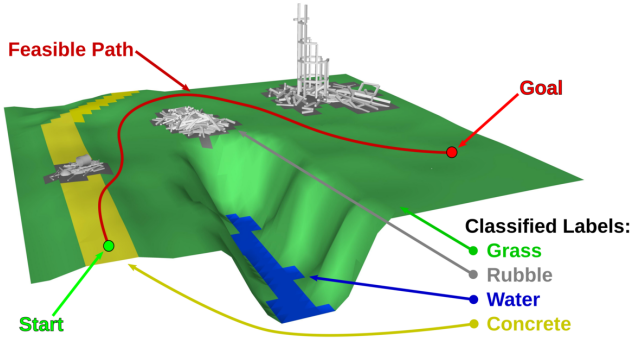
***Index Terms*—Search and Rescue Robots, Motion and Path Planning, Visual-Based Navigation**

Supplementary Material

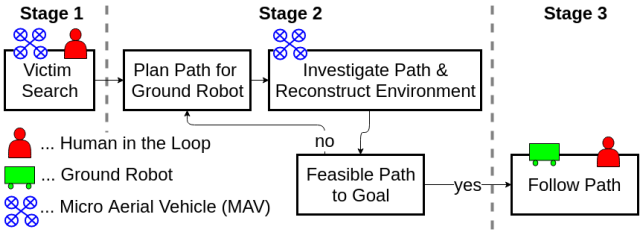
This paper is accompanied by a video illustrating the approach, available at: https://youtu.be/s2v6TICaukQ.

I. Introduction

IN disaster environments or search and rescue scenarios, time is a critical factor in the success of the first responders [1]. These are also scenarios where those same rescue personnel must often put themselves in dangerous situations in order to provide aid. Unmanned systems have the possibility to provide new capabilities for these operators, as well as increase their safety and decrease the response time in delivering that aid. However, one challenge for these scenarios is that the environment may have been altered by the disaster (e.g., an earthquake or a mudslide), potentially invalidating any prior information about the environment, such as maps from aerial surveys or satellite imagery. Consequently, robotic systems that can benefit first responders must be capable of gathering and using data *on demand, without reliance on a priori maps*.



(a) Terrain Map



(b) System Overview

Fig. 1: Overview of the proposed system. Fig. 1a shows a diagram illustrating our intended terrain map output. It includes the elevation information, terrain classification, and the path found for the ground robot. Fig. 1b shows the workflow of the collaborative team.

The approach proposed here is motivated by the need to deliver a fast unmanned response in a previously unexplored environment using a collaborative robot team. We address the problem of exploration of an unknown environment by a flying robot using its onboard sensors from an overhead perspective. Initially, we classify the ground surface using a classifier that is trained "on-the-spot" for the terrain types that are present. We then *actively* explore the environment by iteratively reconstructing segments of the map in 3D, in order to guide the ground robot over only traversable terrain. To minimize the overall response time from system deployment to delivery of aid, this exploration seeks to minimize both the estimated travel time for the ground vehicle and the time required to explore the map from above.

Our system operates in three stages, illustrated in Fig. 1. During all stages of operation, the micro aerial vehicle (MAV) operates fully autonomously or in vision-assisted flight (sta-

Manuscript received: September, 9, 2016; Revised November, 25, 2016; Accepted December, 24, 2016.

This paper was recommended for publication by Editor Wan Kyun Chung upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by the National Centre of Competence in Research Robotics (NCCR) through the Swiss National Science Foundation.

The authors are with the Robotics and Perception Group, University of Zurich, Switzerland—http://rpg.ifi.uzh.ch.

Digital Object Identifier xxxxxxxxxxxxxxxxxxx

bilized and controlled using visual odometry), due to the importance of reducing the attentional load and stress level of the operator by providing autonomous assistance [1].

1) Initial operation involves a human operator flying in vision-assisted mode, commanding the MAV to search for a goal location for the ground robot (e.g. a victim). During this phase, 3D reconstruction of the environment does not take place because the MAV flies too high for precise reconstruction. However, camera imagery from the MAV is used to obtain an initial classification of the terrain for regions that the MAV flies over, while the operator searches for a goal.
2) Once the goal is found, the MAV engages in *autonomous* vision-guided flight to a series of waypoints (3D positions that the MAV should fly to, beginning with the start location) that are chosen *actively*. For each waypoint, 3D reconstruction and ground robot path planning are computed, and the next waypoint is chosen to minimize the estimated response time for the full system (combined aerial exploration and ground path traversal time).
3) Stage 2 repeats until a feasible path is completely explored, and then the ground robot executes that path.

Our proposed system accomplishes these tasks by performing onboard visual odometry for localization and navigation, while mapping the terrain with monocular 3D reconstruction from low altitude in order to provide a precise elevation map. We obtain the terrain class labels in our map through patchwise image classification on the MAV's image stream. Our classifier is trained "on-the-spot" by collecting data and training in flight, in approximately one minute [2]. This allows us to adapt our classification to the terrain that is present in the search and rescue environment, without relying on a priori maps or classifiers. These components provide the elevation and terrain classification inputs to our active exploration algorithm. Their implementations and motivations are described in more detail in Section III.

We validate the performance of our active exploration approach in extensive simulated experiments (see Sec. V-A) and demonstrate our collaborative system in field experiments (see Sec. V-B). We achieve faster response times when compared to both greedy and exhaustive exploration.

A. Contributions

We make the following contributions:

- An active exploration planner that builds a terrain map from an MAV and plans a path for a ground robot.
- A collaborative system that improves the response time of a ground robot delivering aid in an unknown search and rescue environment. It utilizes the active, autonomous exploration planner for the MAV, as well as human-in-the-loop control for other components.

II. Related Work

Other works have combined the capabilities of air and ground robots to perform path planning through a terrain map. An unmanned helicopter is used in [3] to gather multimodal aerial data, which is used to create an a priori terrain classification map. This classified map is used for path planning with the intention of allowing a ground robot to navigate the computed path. Like our proposed system, they utilize the D* algorithm [4] for planning, but unlike our approach there is *no active interaction* between the flying robot and the path planner, and the map is *exhaustively explored* before the path is computed.

Terrain classification for ground vehicle navigation has previously been studied where ground-level data, aerial data, or a combination have been integrated into a traversability map, but without a collaborative robot team. In [5], the authors utilize high altitude aerial imagery, publicly available contour maps, and 3D point clouds from aerial LiDAR surveys. These overhead data are combined with the output of laser and camera sensors onboard a ground robot in a single cost map to enable long-distance path planning and replanning through challenging outdoor terrain. This system requires data from airplane flyovers, which may not be available or may no longer be accurate in a search and rescue or disaster scenario.

In [6], a flying robot was equipped with an RGB camera and used for patchwise terrain classification. Various pictures were taken between $1\,m$ and $5\,m$ altitude and the performance of different features was evaluated using a Random Forest for classification. Unlike our approach, however, the classified patches were not further used for ground robot guidance.

High altitude, high resolution aerial images are frequently used to perform terrain classification [7], [8], [9]. However, none of these approaches consider ground robot guidance or path planning, and do not operate using *active vision*.

A combination of high resolution grayscale and multispectral information is used in [10] to perform a pixelwise classification and 3D reconstruction using a Support Vector Machine for classification. As a postprocessing step, elevation information from the reconstruction is used to refine some classes. While this approach is similar to ours, in that the authors seek to reconstruct both the 3D surface and terrain classes, they utilize high-altitude aerial imagery that was captured a priori and passively, rather than actively. Our motivation for an active approach is that emergency response personnel cannot rely on a priori data that is no longer valid following a disaster.

To determine feasible and efficient paths for the ground robot, we must estimate both the terrain topology and terrain class. Thus, we leverage tools for 3D reconstruction [11] and terrain classification [2]. The latter was specifically designed to be trained rapidly and "on-the-spot".

Unlike existing methods, we propose an active approach for terrain mapping to guide a ground robot in a search and rescue scenario, and we demonstrate superior performance to passive approaches that do not explicitly consider the overall response time of the system. To the best of our knowledge, this is the first work to use active exploration by a flying robot to guide a ground robot through terrain.

III. Terrain Mapping

Our robot team consists of a lightweight MAV and an all-terrain ground vehicle. Our MAV is equipped with a

downward-looking camera, and flies in autonomous and vision-assisted manual flight modes using onboard visual odometry [12], [13]. The images from this camera are additionally used for terrain classification (Sec. III-A) and elevation mapping (Sec. III-B). We use both the estimated terrain class and elevation to determine traversable paths in the map, and estimate their costs in terms of response time. The ground robot in our system is a rugged tracked vehicle that is capable of driving on a variety of different terrain types and climbing both moderate grades and low obstacles. More information about the robots in our team can be found in Section V-B.

We represent the scenario environment as a discretized, rectangular 2D map of size $L \times W$ meters, made up of discrete square cells of size $r \times r$ meters, containing start (y_s) and goal (y_g) locations for the ground robot. For each cell in the map, we record terrain class probabilities, and elevation information if the cell has been mapped during 3D reconstruction. We store these different quantities as layers in a Grid Map data structure [14]. In principle, the map could be dynamically allocated as our robot explores, but within the scope of this paper we use a fixed size that is sufficiently large for our environment and contains the start and goal.

Some existing works that combine semantic classification with 3D mapping utilize other map types. In [15] the environment is decomposed into cubes for classification, but their map representation is actually a 2D grid of these cubes, so ultimately their map contains elevations as a point cloud contained within each cube. While in [16], the authors use forward-facing, ground-level stereo imagery to generate semantically labeled dense 3D maps.

However, we chose a 2.5D elevation map as the most appropriate representation for our task, as it facilitates the use of the D* algorithm in finding paths for the ground robot. It also allows us to represent the elevation of the terrain, since only the top surface of the ground and obstacles are visible from an overhead perspective. A full 3D representation would be unnecessary, since we are unable to observe the sides of any vertical obstacles in the scene as a ground vehicle could. Additionally, our scenario contains an initial phase in which we search for a goal, but do not yet perform 3D reconstruction. We can still leverage the appearance information in our camera images to classify the terrain of any map cells that we fly over during this search.

A. Terrain Classification

We consider the scenario where no map, and no information about the terrain classes, is available a priori. Therefore, our terrain classifier must be trained when the robot system is deployed. In [2], we developed an "on-the-spot" classifier training approach where an operator launches the MAV and labels a few regions of interest on the live image stream from the MAV's downward-looking camera.[1] With these few labeled regions, many training patches are collected by projecting subsequent images to the map and cropping patches that fall on the previously labeled areas. A feature-based classifier can be trained *in flight*, requiring only 60 seconds from launch to begin generating semantic labels for the terrain. We are also able to train a convolutional neural network (CNN) based classifier within 10-15 minutes using the same training data collection procedure. This provides better performance if additional time for training is available.

[1]A video illustrating our rapid training procedure can be found at: https://youtu.be/yVyyhQch6bI.

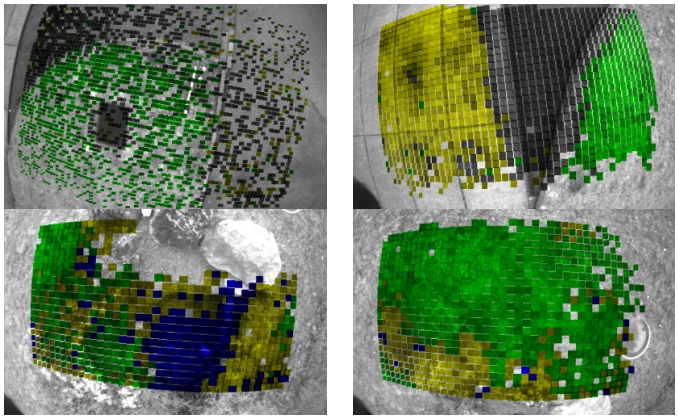


Fig. 2: Classification results on example images from our field experiments, using a classifier trained "on-the-spot". Accumulated probability estimates in our map are projected to the input image, with color representing the class and confidence. Green corresponds to *grass*, blue to *water*, yellow to *concrete/rock*, and gray to *pavement*.

In the work proposed here, we utilize the procedure that we previously developed in [2] when training the classifiers for our system. In the experiments of Sec. V-B, we use a CNN classifier (see Fig. 2 for examples of the classifier output). After training, we begin to survey the environment as we would in a mission scenario. Patches are sampled from the image stream, classified, and then projected to the terrain map. The probabilistic output of the classifier is accumulated for each cell in the map, and averaged over all of the classified patches that have projected to that cell. The most probable class is then used to represent the terrain for each cell.

B. Elevation Mapping

We generate elevation measurements using monocular dense reconstruction, whereby a sequence of images is used to triangulate common 3D points. These images are collected with the camera on our MAV when it flies over a region of the map that is chosen for reconstruction. The 3D reconstruction is computed using REMODE [11]. During exploration, the MAV flies at a known elevation, so we assume an approximately constant field of view of the ground surface. Consequently, we obtain an approximately rectangular patch of elevation data for each waypoint visited by the MAV.

IV. Exploration

Within the map structure defined in Section III, we represent the cost of passing from a cell to its neighbor in terms of *traversal time*. In particular, we compute the time cost of

traversing from cell x_i to cell x_j based on terrain type and local gradient:

$$T_{i,j} = \frac{dist(x_i, x_j)}{v(x_i, x_j)} \tag{1}$$

where $dist(x_i, x_j)$ is the euclidean distance between the centers of the cells, and $v(x_i, x_j)$ is the velocity that the ground robot can drive while traversing this terrain. This function is defined based on the terrain class (C_i) of the current cell and the elevation gradient between the two cells:

$$v(x_i, x_j) = V_{C_i}(1 - L(grad(x_i, x_j))) \tag{2}$$

where V_{C_i} is the speed of the ground robot over a level surface of class C_i, $grad(x_i, x_j)$ is the gradient (slope) determined from the elevation change between the two cells. We propose a function $L(x)$ as a logistic curve that interpolates between V_{C_i} for level ground, and the minimum speed (V_{min}) that the ground robot can travel continuously for its maximum traversable gradient ($grad_{max}$).

$$L(x) = \frac{V_{ratio}}{1 + \exp(-k(x - x_0))} \tag{3}$$

Here $V_{ratio} = V_{min}/V_{C_i}$, k is a free shape parameter that controls the steepness of the logistic curve, and $x_0 = grad_{max}$ is the midpoint of the curve. This logistic function captures the real-world behavior of many ground robots. With small slopes, the speed is minimally affected, but when the incline becomes larger, the operator must take greater care. When climbing inclined surfaces that approach the robot's mechanical limits, the speed will approach zero.

We define a feasible path as a sequence of adjacent cells $\{X = x_0, \ldots, x_N\}$ such that $x_0 = x_s$, $x_N = x_g$, x_{i+1} is adjacent to x_i in an 8-connected neighborhood $\forall x_i$, and the sum of the costs of the cells along that path is finite. We require that elevation data be available for all of the cells along the path, in order to avoid non-traversable discontinuities that would make the path infeasible.

Based on the field of view of the MAV's onboard camera, we discretize the map into potential waypoints for the MAV to visit in order to perform 3D reconstruction there. Each waypoint is centered on a patch of size $l \times w$ meters. One waypoint is located above x_s, and the remaining map is decomposed into a non-overlapping grid of these patches (see Fig. 3).

The problem of efficiently finding a feasible path for a ground robot using observation from a flying robot is partially one of minimizing the number of waypoints visited, and therefore the MAV flight time. In particular, we consider this problem as one of *minimizing overall response time*, which is the sum of the MAV exploration time T_{mav} and the path traversal time of the ground robot T_{gr}. We propose a novel exploration strategy that specifically minimizes this response time in its iterative selection of waypoints, and we compare to several alternative strategies in order to demonstrate its superior performance.

We make the following design choices about our approach to the exploration and mapping task:

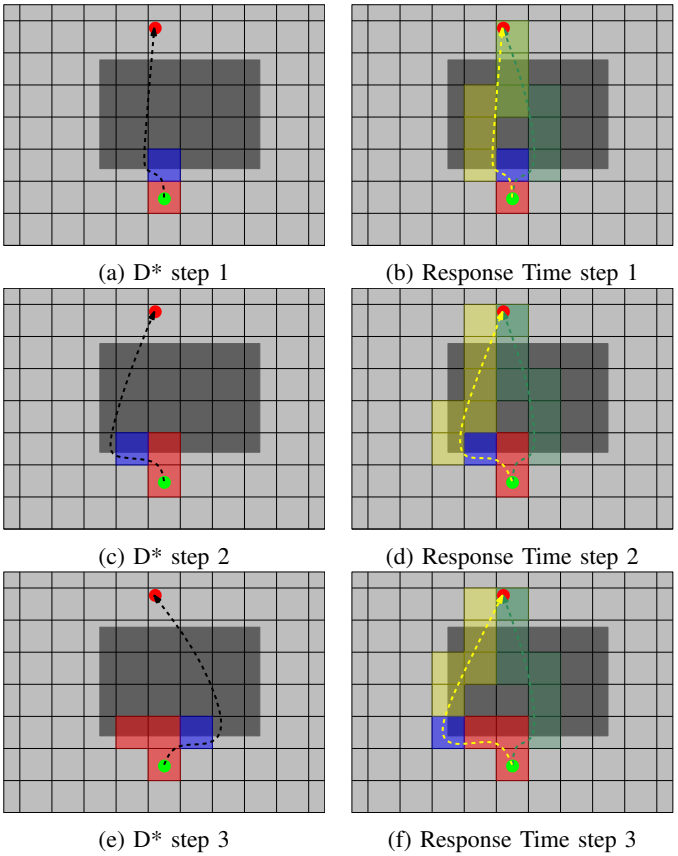
- The size of the environment to be explored is known, and the start and goal locations are contained within it.
- The first visited waypoint is always the one centered over x_s, and only waypoints that are adjacent to previously explored areas are considered at each iteration of the planner. We restrict to these candidates in order to guarantee that our explored area is contiguous by incrementally extending it.
- Consider one map state, where for each map cell, class labels are known with some uncertainty, and elevation information may or may not be available. Using a modified version of the D* algorithm [4], we can compute the cost to reach the goal from any cell in the map using the available information, and therefore construct the optimal path through the current map.

The D* algorithm constructs optimal paths on a discretized map of cells representing possible states for a mobile robot. These states are modeled as nodes in a graph, connected with edges to each of their eight neighbors. The edges have weights associated to them that represent the cost of traversing from one cell to the other. In our system, we define these costs to be the estimated time required to move the ground robot to the neighboring state. With a known robot motion model, these costs can be estimated accurately for different terrain types and inclinations.

The D* algorithm permits updates to these costs, such as when new sensor data is obtained, and will dynamically recompute the optimal path. Normally, D* is used for a robot that re-plans its path as it moves and encounters new obstacles sensed with its *onboard* sensors. In that case, updates must be processed only until they will no longer improve the cost-to-goal from the robot's current position. Our mapping scenario motivates a slight modification, since our ground robot is not following the path as we update it, and we instead want to process updates to the map so that the changes to paths propagate fully in the map. In the terminology of [4], at each update we run the $PROCESS\_STATE$ algorithm until the $OPEN$ list is empty. With this modification, we compute the path and cost-to-goal for every cell in the map, given the state of the map after each update. It should be noted that we can not guarantee the global optimality of the response time, since the elevation in the full map would need to be known a priori. However, given a particular map state, and start and end cells within it, the D* algorithm does guarantee an optimal path between them.

A. Exhaustive Exploration

The naive way to approach the task of finding a feasible path would be to first perform exhaustive mapping of every waypoint in the map by flying a lawnmower pattern over the full environment. Given complete elevation and terrain class estimates, the flying robot could then compute the optimal path for the ground robot. While this approach would produce the optimal path, in terms of traversal time for the ground robot, the exploration time necessary for the MAV to map the full environment would be too large to justify the search for global optimality. The total response time for both exhaustive mapping and ground robot deployment is denoted T_{Ex}. We have described this strategy here as a baseline to motivate our active approach.



(a) D* step 1 (b) Response Time step 1

(c) D* step 2 (d) Response Time step 2

(e) D* step 3 (f) Response Time step 3

Fig. 3: This sequence of diagrams illustrates several steps of the D* and Response Time planners. In this simple environment, with a uniform terrain class and a large rectangular obstacle in the middle (dark gray), the MAV is exploring from the start location (green dot) to the goal location (red dot). The grid represents the coarse set of waypoints/dense reconstruction patches. The finer grid of cells in the elevation map are not shown. The explored patches are in red, and the patch chosen for the next step is in blue. The D* planner (left column) oscillates between following the obstacle to the left and then to the right. The Response Time planner (right column) evaluates multiple candidate paths (one for each boundary segment, but only two shown here for simplicity) to compute an estimate of the remaining time to explore that path. In this case, continuing to follow the yellow path requires exploring an extra patch. But the time required for the MAV to fly from the current location to the next waypoint along the green path would make its total response time greater, so it continues exploring the yellow path.

B. D* Path Exploration

A more efficient strategy utilizes a greedy approach to exploration, where the waypoint that is chosen for exploration at each iteration is the one that follows the current D* optimal path from y_s to y_g. In particular, this strategy extends the explored area by following the optimal path to the boundary of the previously explored waypoints and chooses the waypoint on the unexplored side of that boundary segment. This strategy is presented as another performance reference. Although the greedy approach will eventually produce the same optimal path as in the exhaustive search, it does not consider the flight time of the MAV in its choice. This can result in visiting many unnecessary waypoints when the D* optimal path changes due to new information in the map. See Fig. 3 for an example of this behavior. We denote as T_{D^*} the total response time using this exploration strategy.

C. Response Time Minimized Path Exploration

Our proposed exploration strategy utilizes an exhaustive search over candidate paths in order to explicitly minimize the *total response time* of the system, not just the path cost. More concretely, we consider the set B of boundary line segments that separate the explored region from the unexplored region. For each $b_i \in B$, we find the lowest cost path that exits the explored area through this segment, at cell x_{b_i}. This path is a sequence of states in the map $\{X_i | x_{b_i} \in X\}$, such that a sequence of waypoints $\{W_i\}$ would extend the contiguous explored region from x_s to x_g along $\{X_i\}$. For each cell on b, extending the explored area by mapping the waypoint on the unexplored side of b would require the same flight time (i.e. cost) to explore, so we only consider the best path through each b_i. The boundary segment whose corresponding waypoint is chosen next for exploration is the one that minimizes:

$$b_{next} = \underset{b_i \in B}{\arg\min}\ T_{x_s,x_{b_i}} + T_{x_{b_i},x_g} + \sum_{w_j \in \{W_i\}} T_{w_{j-1},w_j} \quad (4)$$

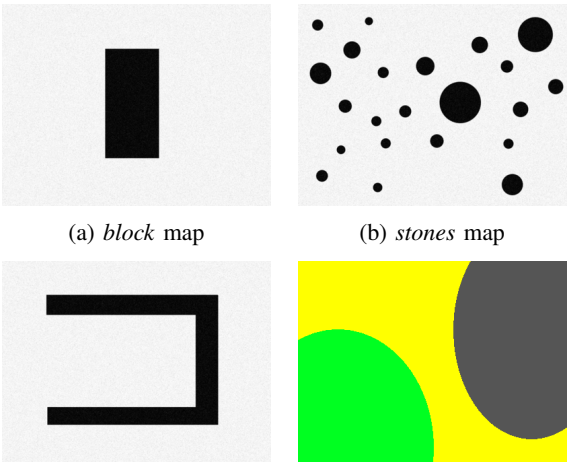
where $T_{x_s,x_{b_i}}$ represents the estimated time for the ground robot to reach x_{b_i} from x_s, $T_{x_{b_i},x_g}$ is the estimated time for the ground robot to reach x_g from x_{b_i}, and each T_{w_{j-1},w_j} is the time necessary for the flying robot to reach waypoint j from waypoint $j-1$ plus the time to capture elevation data at waypoint j. This expression is the sum of the time remaining to map a candidate path plus the time required to drive that path with the ground robot. By minimizing this quantity over all candidate paths that cross our known boundary, we select the next waypoint as the one that minimizes the overall response time for the ground robot to reach its goal. We call this combined aerial robot exploration and ground robot path following time T_{RT} for the Response Time planner.

V. Experimental Results

We conducted simulated experiments to assess the quantitative performance of our *Response Time* planner with respect to the reference *D** and *Exhaustive* approaches. We also executed several real-world experiments to demonstrate the performance of our approach in mock search and rescue scenarios.

A. Simulated Experiments

We designed several scenarios in which to test the exploration planners described in Section IV. Each scenario is a $30\,m \times 40\,m$ map with large, untraversable obstacles in different arrangements. The rest of each map has level elevation that is corrupted with zero-mean Gaussian noise with a standard deviation of $3\,cm$. Several terrain classes are defined in blobs that cover approximately one third of the map each:



(a) *block* map (b) *stones* map

(c) *dead_end* map (d) Terrain class labels

Fig. 4: Simulated experiment scenario maps. For each map, the black areas are untraversable obstacles and the rest of the map has level but noisy elevation. Fig. 4d shows the terrain classes used for all of the maps. Yellow represents *concrete*, green represents *grass*, and gray represents *gravel*.

concrete, *gravel*, and *grass*. The maps, named *block*, *stones*, and *dead_end*, along with the terrain class arrangement are shown in Fig. 4.

In both our real and simulated experiments, the maps are discretized to a cell resolution of $0.1\,m$, which was chosen as a reasonable granularity for our robot size ($\sim 40\,cm \times 80\,cm$) and the resolution of our camera imagery from mapping height ($2-3\,m$). Our waypoints are spaced such that our map patches are of size $2.0\,m \times 1.5\,m$, which is slightly smaller than the field of view of our depth maps from mapping altitude, in order to ensure continuity in our elevation map. The base speeds associated to each terrain class were $V_{concrete} = 0.5\,m/s$, $V_{grass} = 0.25\,m/s$, and $V_{gravel} = 0.33\,m/s$. The MAV speed between waypoints is set to $0.6\,m/s$, and the time to compute elevation at a chosen waypoint is fixed at $10.0\,s$. These parameters are all set empirically from the real-world performance of our MAV and ground robot during operation.

For each map, 100 trials were performed, where for each trial we randomly generated a start position and a goal position, and then ran both the D* and Response Time exploration planners. Initially, the robot's terrain map begins with only the terrain class label for each cell, but no elevation information. Beginning with the waypoint over the start location, we add that waypoint's rectangular patch of elevation from the corresponding map in Fig. 4 to the robot's terrain map. This approach to simulating the elevation assumes that the 3D reconstruction is perfect and converges for all pixels in the reference image. Clearly this is not the case in field scenarios, but these experiments are intended to evaluate the choices made by the exploration planners, and not the system as a whole.

Since each trial has a different path length, resulting in a wide range in the magnitude of the response times, we compute the *speedup* of the Response Time planner over the D* planner ($S = \frac{T_{D^*}}{T_{RT}}$) and exhaustive exploration ($S = \frac{T_{Ex}}{T_{RT}}$), which is unitless and allows direct comparison of trials with different start and goal locations (see Table I).

TABLE I: Average Speedup of Response Time Exploration Planner w.r.t Greedy (D*) and Exhaustive Strategies

Map Name	Block	Stones	Dead End
Speedup: RT to D* $\frac{T_{D^*}}{T_{RT}}$	1.24	1.189	1.275
Speedup: RT to Exhaustive $\frac{T_{Ex}}{T_{RT}}$	10.15	9.00	7.58

B. Field Experiments

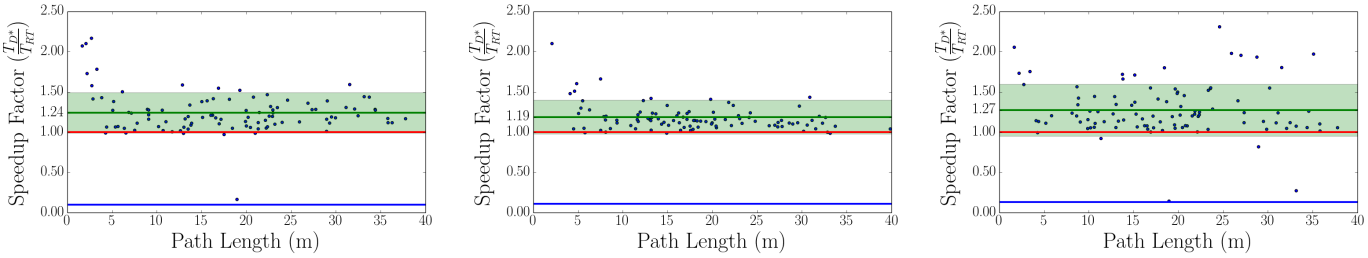
We successfully tested our systems in two outdoor scenarios, which we name *driveway* and *canyon*, using a quadrotor MAV and a ground robot. We chose these scenarios in order to demonstrate the main capabilities of our system. In particular, while the *driveway* scenario has little elevation change, it serves as a proof of concept. The *canyon* scenario, however, consists of a culvert and creek, with an elevation difference between the water level and the top of the culvert of almost 2m. The purpose of this testing environment was to verify that our path planner could avoid untraversable terrain classes and handle significant elevation changes.

The stages of operation are as described in Section I and visualized in Fig. 1b. In both scenarios, the terrain classes were *pavement*, *concrete*, *grass*, and *water*. The base speeds used for path planning are the same as in our simulated experiments, with the addition of the *pavement* class, where $V_{pavement} = 0.5\,m/s$. Unlike the other classes, with empirically-determined velocities in Sec. V-A, we mark all cells that are classified as *water* as impassable, since our robot cannot traverse it. We first describe the deployed robots and then detail the two scenarios and achieved results.

1) Robot Description: The MAV, a custom built quadrotor based on off-the-shelf components (see Fig. 6a), is equipped with a downward-looking camera, an inertial measurement unit (IMU), and an onboard computer. Our state estimation and control pipeline integrates pose estimates from the visual odometry algorithm SVO [13] with accelerometer and gyroscope measurements from the IMU to stabilize itself [12]. All computations necessary for autonomous flight are performed on the single board computer (an Odroid U3) onboard the MAV, but a subset of the images is streamed to a laptop computer that runs both the terrain classification and dense 3D reconstruction. Our exploration algorithm also runs on this computer and sends waypoints to the quadrotor.

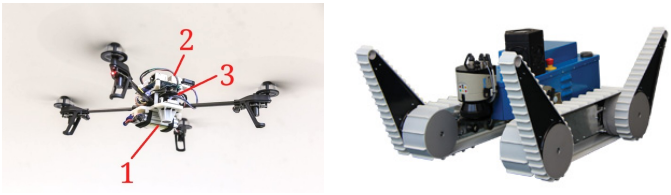
The ground robot is an "Absolem" from Bluebotics (see Fig. 6b) that is capable of driving over rough terrain. During the experiments, it was remote-controlled to follow the computed path. We must note here that while a flying robot with an onboard camera and a somewhat rugged ground robot are assumed for our proposed system, the particular vehicles used in our experiments are not in any way definitive, and each could be replaced with any platform having similar basic capabilities.

2) Driveway Experiment: The first scenario (see Fig. 7a) demonstrates the system's capability to distinguish different



(a) Speedup on *block* map. (b) Speedup on *stones* map. (c) Speedup on *dead_end* map.

Fig. 5: Quantitative results of the simulated experiments. For each map, 100 trials were run with randomized start and goal locations. The D* and Response Time planners were executed for each set of start and goal locations, and the ratio of their overall response times computed (speedup factor). The individual trials (dots), mean speedup (green line), and ± 1 standard deviation (light green band) are shown. The red line represents no improvement with respect to the D* planner, and the blue line represents no improvement with respect to exhaustive exploration.



(a) Quadrotor (b) Ground Robot

Fig. 6: A closeup of our quadrotor (left): 1) down-looking camera, 2) Odroid U3 quad-core computer, 3) PIXHAWK autopilot. At right is the ground robot, a Bluebotics Absolem.

terrains: although a straight line path through the grass might seem faster (and is feasible for the ground robot), it is beneficial to stay on concrete since the ground robot can drive twice as fast on a hard surface. Fig. 7b shows the classification result after vision-assisted manual flight and Fig. 7c shows the final result. When driving the path with the ground robot, we found that the path on concrete is in fact 50% faster than the straight path over grass.

3) Canyon Experiment: The second, more challenging scenario (see Fig. 8a) includes steep terrain and water, which cannot be traversed by the ground robot. The terrain classification safely distinguished pavement, grass, concrete, and water, leading to a feasible path for the ground robot (see Fig. 8b). The elevation mapping assured that the path remains in regions that are flat enough for the ground robot to traverse (see Fig. 8c). Finally, we could successfully follow the path with the ground robot.

VI. Discussion

Simulated experiments demonstrated a significant improvement in response time when comparing our proposed path planner to the greedy D* approach and to exhaustive exploration (see Fig. 5). With the exception of a small number of outlier trials ($\sim 1.7\%$ of our 300 trials) where the Response Time planner performed worse than the D* planner ($S < 1$), the RT planner consistently provides a speedup between 1 and 1.5, with an average factor of 1.23 across all of the maps. The response time when using our proposed active approach is also 8 to 10 times faster than the exhaustive mapping strategy. The average speedup factors relative to the D* planner and the exhaustive approach are collected in Table I.

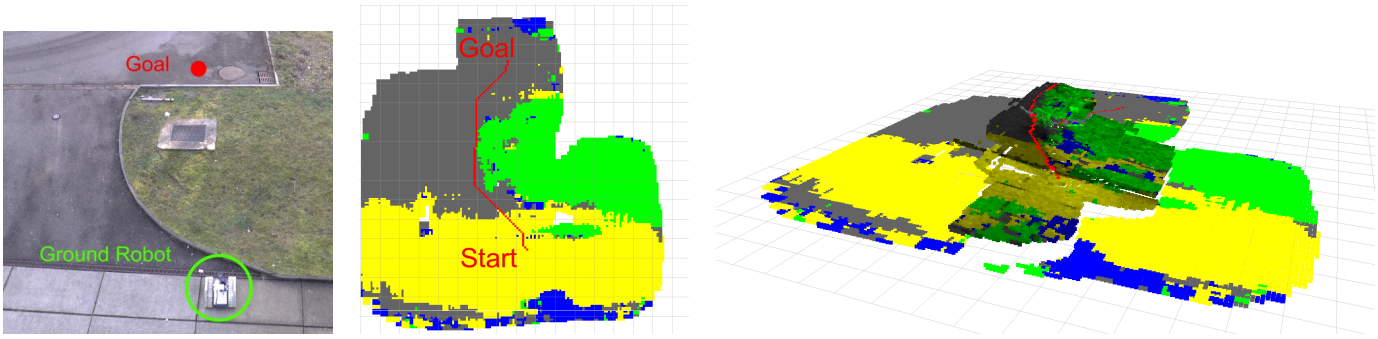
We also successfully deployed our collaborative system and used it to test the proposed Response Time exploration planner in two real-world scenarios. These two field experiments demonstrate that the proposed path planning algorithm is capable of exploring a feasible and efficient path for the ground robot in environments with multiple terrain classes, elevation changes, and untraversable terrain.

VII. Conclusion

In this paper, we proposed a collaborative search and rescue system consisting of a flying robot that explores and maps the environment, in order to find a traversable path for a ground robot. Our primary innovation is an *active* exploration strategy for the MAV. Rather than exhaustively mapping the environment, or following the current best path to the goal, our Response Time planner explicitly minimizes the overall time for the system to operate. By considering the time necessary for the MAV to map the waypoints along the ground robot's path, in addition to the time required to drive that path, we have demonstrated that the combined time for exploration and path traversal can be significantly reduced. Our focus on minimizing response time is motivated by the desire to provide a robot system that both puts robots in dangerous situations instead of rescuers, and provides aid as fast as possible.

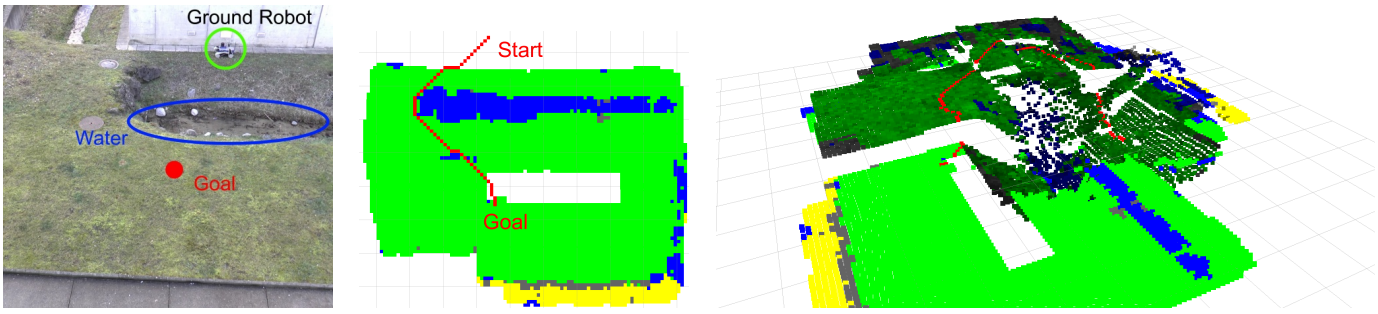
In accomplishing this goal, we also demonstrated the use of a convolutional neural network image classifier for terrain classification, which we used in mapping the terrain and finding traversable terrain for the ground robot. In further minimizing response time, we utilized an "on-the-spot training" procedure that allowed us to train the classifier on demand with the terrain that is present at a disaster site, where pre-trained classifiers would not be available.

Our proposed Response Time planner and overall system have been validated in simulated and real-world scenarios, and in the future we intend to expand the scope of these field deployments to large-scale disaster sites. This work represents a step toward a field-deployable system that can extend the



(a) Overview (goal in red) (b) Initial Classification (c) Elevation along path

Fig. 7: Driveway Experiment: While a direct path would be shorter, the chosen path (in red) would have lower response time due to the speed of the ground robot on different terrains. The terrain classes are *concrete* (yellow), *pavement* (gray), *grass* (green), and *water* (blue). Elevation mapping is only performed along the path for the ground robot, as visible in (c).



(a) Overview (goal in red) (b) Initial Classification (c) Final Result

Fig. 8: Canyon Experiment: The ground robot cannot cross the water. Our system finds a feasible path for the ground robot that stays on the grass. The terrain classes are *concrete* (yellow), *pavement* (gray), *grass* (green), and *water* (blue). The path is shown in red in (b) and (c). Elevation mapping is only performed along the path for the ground robot, as visible in (c).

capabilities of first responders and reduce the response time of their aid in real disaster situations.

ACKNOWLEDGEMENTS

Special thanks to Renaud Dubé and the TRADR project team of the Autonomous Systems Lab, ETH Zurich, for providing the ground robot and participating in the experiments.

REFERENCES

[1] R. R. Murphy, *Disaster Robotics*. The MIT Press, Cambridge, MA, 2014.

[2] J. Delmerico, A. Giusti, E. Mueggler, L. Gambardella, and D. Scaramuzza, ""on-the-spot training" for terrain classification in autonomous air-ground collaborative teams," in *Int. Sym. on Experimental Robotics (ISER)*, 2016.

[3] B. Sofman, J. A. Bagnell, A. Stentz, and N. Vandapel, "Terrain classification from aerial data to support ground vehicle navigation," Carnegie Mellon University, Tech. Rep. CMURI-TR-05-39, Jan 2006.

[4] A. Stentz, "Optimal and efficient path planning for partially-known environments," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 1994, pp. 3310–3317.

[5] D. Silver, B. Sofman, N. Vandapel, J. A. Bagnell, and A. Stentz, "Experimental analysis of overhead data processing to support long range navigation," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2006, pp. 2443–2450.

[6] Y. N. Khan, A. Masselli, and A. Zell, "Visual terrain classification by flying robots," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*. IEEE, 2012, pp. 498–503.

[7] E. Sali and H. Wolfson, "Texture classification in aerial photographs and satellite data," *Int. J. of Remote Sensing*, vol. 13, no. 18, pp. 3395–3408, 1992.

[8] M. Azimi-Sadjadi, S. Ghaloum, and R. Zoughi, "Terrain classification in SAR images using principal components analysis and neural networks," *IEEE Trans. Geoscience and Remote Sensing*, vol. 31, no. 2, pp. 511–515, 1993.

[9] J. A. Montoya-Zegarra, J. D. Wegner, L. Ladicky, and K. Schindler, "Semantic segmentation of aerial images in urban areas with class-specific higher-order cliques," in *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2015, pp. 127–133.

[10] B. Gruber-Geymayer, A. Klaus, and K. Karner, "Data fusion for classification and object extraction," in *Proceedings of the ISPRS Workshop CMRT 2005*, 2005, pp. 125–130.

[11] M. Pizzoli, C. Forster, and D. Scaramuzza, "REMODE: Probabilistic, monocular dense reconstruction in real time," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2014, pp. 2609–2616.

[12] M. Faessler, F. Fontana, C. Forster, E. Mueggler, M. Pizzoli, and D. Scaramuzza, "Autonomous, vision-based flight and live dense 3D mapping with a quadrotor MAV," *J. of Field Robotics*, pp. 1556–4967, 2015.

[13] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2014, pp. 15–22.

[14] P. Fankhauser and M. Hutter, "A universal grid map library: Implementation and use case for rough terrain navigation," in *Robot Operating System (ROS) – The Complete Reference (Volume 1)*, A. Koubaa, Ed. Springer, 2016, ch. 5.

[15] D. Maturana and S. Scherer, "3d convolutional neural networks for landing zone detection from lidar," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2015, pp. 3471–3478.

[16] V. Vineet, O. Miksik, M. Lidegaard, M. Nießner, S. Golodetz, V. A. Prisacariu, O. Kähler, D. W. Murray, S. Izadi, P. Pérez, and P. H. S. Torr, "Incremental dense semantic stereo fusion for large-scale semantic scene reconstruction," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2015, pp. 75–82.