Reactive obstacle avoidance for multicopter UAVs via evaluation of depth maps

Luca Di Stefano[1] Eliseo Clementini[2] Enrico Stagnini[3]

[1] Gran Sasso Science Institute (GSSI), L'Aquila, Italy [2] University of L'Aquila, L'Aquila, Italy [3] DroniAbruzzo, L'Aquila, Italy

Abstract

In the context of Unmanned Aerial Vehicles (UAVs) navigation, an "obstacle" is any object that stands between the UAV and its destination. Reacting to unforeseen obstacles is a major issue in the field of autonomous navigation.
We consider obstacle detection to be a problem of evaluating the visibility of the destination from the UAV viewpoint. Data acquired from an on-board depth camera are used to assess the visibility of the destination in a qualitative framework, and to plan a new route when obstacles are detected.

Introduction

A vast amount of research has addressed the problem of autonomous navigation for multicopter UAVs [1, 2]. However, relatively little work has been done to apply qualitative spatial reasoning to this field.
We focus on the field of goal-oriented obstacle avoidance, where the intent is to guide the UAV towards a destination while avoiding unforeseen obstacles.
We propose an obstacle avoidance algorithm that can achieve such results in outdoors scenarios.

UAV equipment

We assume the UAV is equipped with:

- A depth camera, i.e. a device that can sense the distance of objects from the viewpoint of the UAV and encode it as a raster image. Such an image is commonly known as a depth map.
- A GPS sensor that provides the position of the UAV (also known as a GPS fix).

Assessing visibility

We express the problem of checking whether the destination P is reachable by flying in a straight line as a *visibility* problem. However, to account for the volume of the UAV we have to extend the definition of point-to-point visibility to solid shapes.
We adopt the definition from [3]: we say that an obstacle O occludes a solid A from an observer B if any segment $\overline{ab}, a \in A, b \in B$ intersects O. Otherwise, A and B are visible from each other with respect to O.
A visibility check can also reveal that P lies too close to an obstacle, meaning that P is unreachable.

To evaluate the visibility of P:

1. Point the depth camera so that the destination P is near the center of the viewpoint.
2. Acquire a depth map I.
3. Project P onto I via a geometric transformation, obtaining a pixel $P' \in I$ (Figure 3).
4. Compare the current distance from P to the value of I in a neighborhood of P', whose radius depends on the volume of the drone and a fixed safety margin.

If any pixel stores a value smaller than d, the destination is occluded.

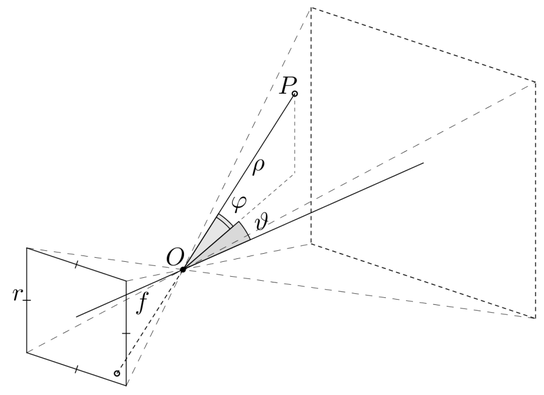

Figure 3: Projection of a point in space onto the depth map. We model the depth sensor as a pinhole camera [4, Ch. 6]. r, f are known parameters of the camera, while O is its position. ρ, ϑ, φ can be deduced from P, O, and the orientation of the UAV.

Finding an escape point

If an obstacle occludes the destination, the algorithm directs the UAV to an escape point. This is a visible point in space from which the UAV should be able to reach its destination.
To find an escape waypoint:

1. Dilate the depth map I. The amount of the dilation of a pixel p depends on its intensity: the closer an obstacle, the highest the radius of dilation. Call I' the dilated depth map.
2. Threshold I' and apply a distance transform. The pixels with distance equal to 1 will be the candidate escape points.
3. Choose the candidate E' according to a cost function that evaluates the distance from P' and the altitude difference between the two points.
4. Project E' to 3D space, obtaining E.

Figure 2 exemplifies the procedure. The three square pictures are I, I', and a representation of the candidate escape points (the solid black line) with the chosen one highlighted.

Experimental setup

We implemented the algorithm as a set of Python 3 scripts, which interact with a simulation running in the V-REP platform [5].
Figure 1 shows the simulation scenario. We modeled an uneven terrain cluttered with trees, in order to verify the reaction of the algorithm to both the obstacles and the ground.
We ran a total of 12 simulations, measuring the total distance covered by the UAV and the duration of the flight. We used different depth map resolutions to evaluate the robustness of the algorithm to the lack of detailed information about the environment.
Table 1 exposes the measurements. To evaluate the quality of the chosen paths, each flight distance has been normalized to the distance between the starting position of the UAV and the destination.

	Resolution	Reached Waypoints	Dropped Waypoints	Absolute distance	Normalized distance	Time
1	256	2	0	24.16	1.04	**00:58.6**
2	256	2	1	23.83	1.03	01:12.0
3	256	3	0	**23.43**	**1.01**	01:22.5
4	128	2	1	23.62	1.02	01:13.0
5	128	4	1	24.08	1.03	01:32.0
6	128	3	1	23.54	1.01	01:12.1
7	64	4	5	**24.99**	**1.07**	**02:09.6**
8	64	3	2	24.19	1.04	01:20.7
9	64	3	1	23.68	1.02	01:20.3
10	32	4	1	23.73	1.02	01:25.8
11	32	3	2	24.05	1.03	01:46.2
12	32	3	2	23.79	1.02	01:18.3

Table 1: Experimental results on the simulated scenario. Minima and maxima are highlighted.

Conclusions and future work

The simulation shows some promising results:

- Decreasing the depth map resolution does not affect the flight distance in a relevant way;
- however, lower resolutions may raise the chance of picking less optimal escape points, which in turn can negatively impact on the flight time.

Further development should address the following issues:

- We currently ignore GPS inaccuracies. In a realistic setting, the algorithm should adapt the safety margins to the estimated accuracy of the fix or put the mission on hold until a minimum accuracy is achieved.
- The flight time could be improved by optimizing the UAV speed, which at the moment is constant; this would require some additional safety checks to guarantee that the UAV is able to stop before a collision with unforeseen obstacles occurs.

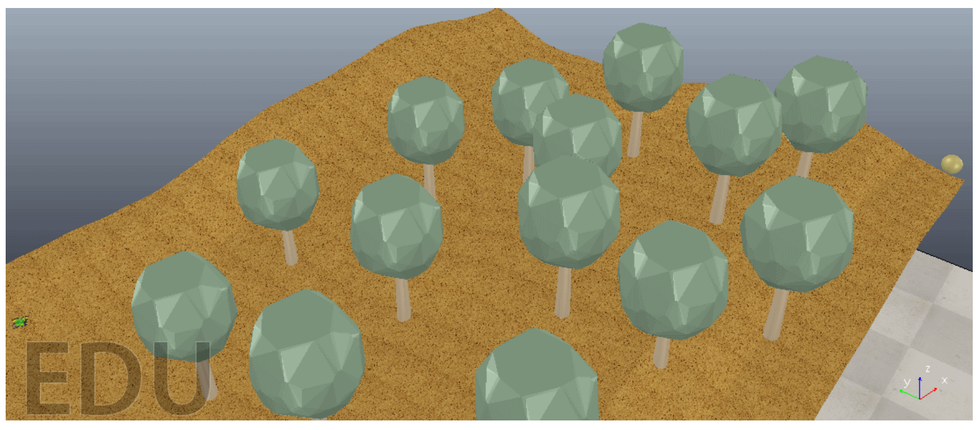

Figure 1: The simulation scenario. The yellow globe is the destination.

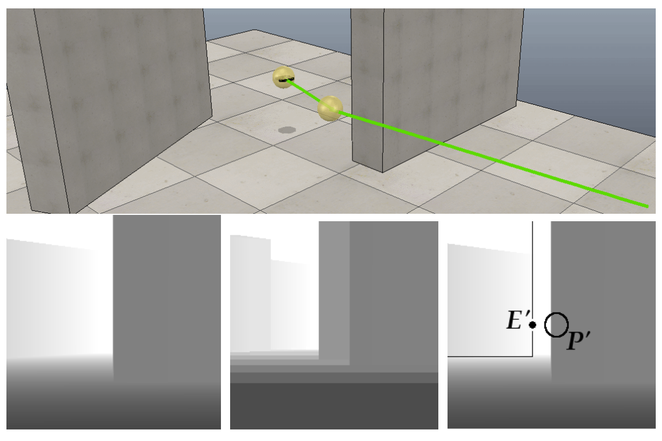

Figure 2: Obstacle avoidance through an escape point E.

References

[1] C. Goerzen, Z. Kong, and B. Mettler, "A Survey of Motion Planning Algorithms from the Perspective of Autonomous UAV Guidance," *Journal of Intelligent and Robotic Systems*, vol. 57, pp. 65–100, jan 2010.

[2] F. Kendoul, "Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems," *Journal of Field Robotics*, vol. 29, pp. 315–378, mar 2012.

[3] P. Fogliaroni and E. Clementini, "Modeling Visibility in 3D Space: A Qualitative Frame of Reference," in *9th International 3DGeoInfo 2014 - Lecture Note in Geoinformation and Cartography*, pp. 243–258, 2015.

[4] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd ed., 2004.

[5] E. Rohmer, S. P. N. Singh, and M. Freese, "V-REP: A versatile and scalable robot simulation framework," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1321–1326, IEEE, nov 2013.

Contact Information

git github.com/lou1306/localpathplanner
luca.distefano@gssi.it
eliseo.clementini@univaq.it
info@droniabruzzo.it