Towards 3D Cave Mapping with UAVs

Hunor Laczkó*, Bálint Jánossy*, Tamás Zsedrovits*
* Pazmany Peter Catholic University, Faculty of Information Technology and Bionics, Budapest, Hungary

***Abstract* — Cave mapping is far beyond surface mapping technologies because of the rugged terrain and the limited availability of sensors (for example, no GPS). Most of the existing maps were hand-drawn and are inaccurate. The mapping technologies are outdated or require expensive tools, or they are too complicated to use by anyone. For this, we propose a drone-based system where with the use of its basics sensors and laser range finders, and monocular cameras, we will be able to create 3D maps in real-time.**

***Index Terms* — vision-aided navigation, SLAM, cave mapping**

I. Introduction

A cave mapping system could be an excellent tool for exploring previously unknown places but could also be helpful for different touristic and recreational activities. We propose a system that can be installed on an Unmanned Aerial Vehicle (UAV) because it would provide several advantages in this environment. For instance, it would be ideal for long chimneys for which manual mapping would be hard or even impossible. Such a drone is relatively small and can travel fast, so it is suitable for caves. Of course, it cannot fly into narrow corridors where a person could fit, but every technique has its limitations.

The current prototype consists of an offboard model with a laser rangefinder and an inertial measurement unit capable of creating 2D maps of the environment. For this, we used the Robot Operating System (ROS) as the framework in which we could integrate the necessary parts. The goal was to create a model which could be easily transferred to a real drone. For this, we used an Odroid minicomputer [1] as the main unit, a Pixhawk Mini [2] flight controller with the IMU, and a Hokuyo laser rangefinder [3]. We wanted to create a map and navigate the drone with its help, too, so we chose a SLAM (Simultaneous Localization and Mapping) algorithm. We used Google's Cartographer implementation, which was easy to integrate into the ROS framework and provided a complete customization option. The final map was created on a remote computer in real-time because we needed more performance for better accuracy than the Odroid could provide. With this system, we were able to make different measurements and create maps based on these.

II. Related Work

The traditional method for mapping is by manual measurements in which by using manual tools like tape measures, cartographers build polygons from the measured lines and draw the map based on these. With the appearance of modern software and tools, this process became more manageable.

There are 3D imaging software where one can visualize these polygons in 3D and create the 2D maps with projection [4]. Using laser distance measure equipment made the procedure even faster since these could record the slope angle and the magnetic azimuth. One such solution is DistoX [5], which along with the previously mentioned measures, records multiple points around the breaking points, making it easier to fit planes on the point cloud.

There are other active cave mapping projects, too, like Caveatron [6] or Zebedee [7]. The main differences between these are whether they can be used on the move [6] or require static measurements [5]. There are also approaches using no laser, but they could not achieve significant results yet.

III. System Overview

We created an offboard model of the system for the prototype, which replicates the final, drone-mounted set up as close as possible; it would only require slight modifications to integrate into a drone. For better usability, we created a 3D printed holder for these illustrated in Figure 1.

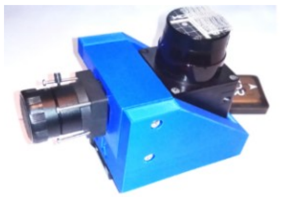


Figure 1. 3D printed holder with the camera at the front, LiDAR on the top, and the Pixhawk Mini on the bottom

The main part is an Odroid XU4 minicomputer on which the ROS framework ran. The main task of this unit is to gather the data from the different sensors and relay them to another computer with a bit of pre-processing if necessary. Later on, after optimizing the mapping algorithm, it could be possible to do all the calculations on this unit.

There are two main sensors. The first one being the Hokuyo urg-04lx-ug01 lightweight 2D laser rangefinder, also called LiDAR [3]. The LiDAR uses infrared laser beams, and it can take measurements with a 10 Hz frequency, 240 degrees viewing angle, and 0.36 angular resolution. It can record data from 20mm to 4000mm with a maximum of 3% error. The resolution and the distance are suitable for cave mapping.

The other main sensor is the camera, a daA1280-54uc Basler monocular camera. This global shutter camera has 1.2 megapixels resolution and can record 54 fps at a 64-degree viewing angle. Like with the LiDAR, the main reasons for this choice were the low price and the small weight. Also, the global shutter is essential to be resilient to onboard vibrations.

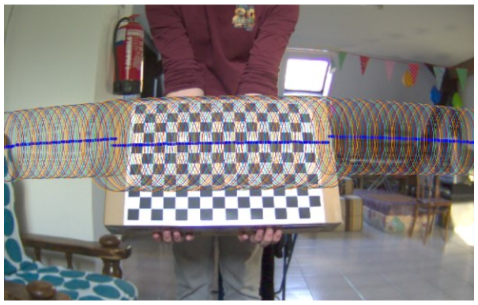


Figure 2. The points from the laser measurement projected back to the picture after the transformations.

The camera and LiDAR each record data in their relative coordinate system. RADLOCC [8], [9] calibration was used to calculate a translation and orientation vector between the two sensors. For verification, the points were projected to the camera's image (Figure 2).

The final system will be implemented on a 3DR Iris+ [10] drone. It has the 3DR Pixhawk flight controller, which is compatible with the Pixhawk Mini used in the prototype system. This module also provides acceleration and gyroscope data.

Figure 3. 3DR Iris+ drone.

An Optitrack coordinate reference system was used for accurate motion tracking [11]. Its purpose was to check the result of the measurements and calculations and verify their error. It is based on eight infrared cameras, each capable of 240 fps, and with this, we were able to localize the given object with 6 DOF.

As mentioned before, the ROS framework was used to build up the system. Both the camera and the LiDAR had their corresponding ROS nodes. The Optitrack was operated by the *vrpn_client_ros* node, which received data from the streaming engine running in Motive, Optitrack's software. The *mavros* node was used to communicate with the flight controller, the Pixhawk Mini, and receive the IMU data. The Cartographer node was running on another computer with higher performance to receive as accurate results as possible. An overview of the connections can be seen in Figure 4.

IV. Cartographer

The mapping algorithm was chosen considering the available data, the computational requirements, and ROS compatibility. Gmapping [12], which uses the OpenSLAM algorithm, Hector SLAM [13] which realized a 3D localization based on two 2D maps, and Google's Cartographer [14] were tested, and the Cartographer was the best option. This method does not require odometry data; it only needs inertial measurement data for 3D mapping and realizes a two-level graph-based SLAM algorithm.

The first level is the local SLAM responsible for building submaps, which are locally correct for a short time window from the laser data. The second layer is the Global SLAM which runs in the background and is responsible for creating a globally correct map from the submaps. If available, it could also incorporate for the constraint matching other sensory data like IMU or odometry. A high-level overview of the system can be seen in Figure 5.

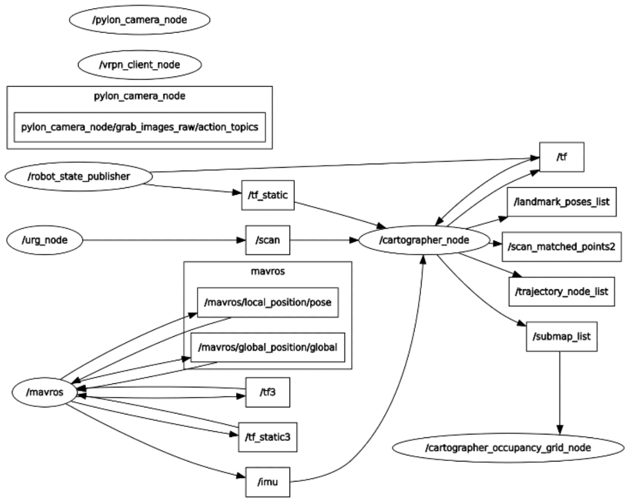
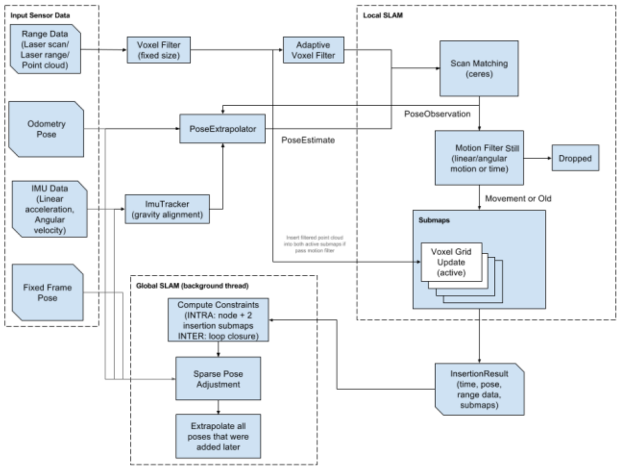


Figure 4: rqt graph of the ROS connections during the mapping process


Input Sensor Data
Range Data (Laser scan/ Laser range/ Point cloud)
Odometry Pose
IMU Data (Linear acceleration, Angular velocity)
Fixed Frame Pose
Voxel Filter (fixed size)
Adaptive Voxel Filter
PoseExtrapolator
ImuTracker (gravity alignment)
PoseEstimate
PoseObservation
Local SLAM
Scan Matching (ceres)
Motion Filter Still (linear/angular motion or time)
Dropped
Movement or Old
Submaps
Voxel Grid Update (active)
Insert filtered point cloud into both active submaps if pass motion filter
Global SLAM (background thread)
Compute Constraints (INTRA: node + 2 insertion submaps INTER: loop closure)
Sparse Pose Adjustment
Extrapolate all poses that were added later
InsertionResult (time, pose, range data, submaps)
Figure 5. Cartographer system overview.

V. Mapping Results

The tests were done on two different measurement types. First, we tested the system in a simulated environment inside a building, then took measurements with the real sensors and used that for the map creation.

The simulated environment was created with the Gazebo simulator, and it consisted of corridors, smaller rooms, and a bigger open space. We conducted tests on different parts of this environment, but they all gave similar results. The place's layout mainly consisted of straight lines, which were easy to create, and the individual rooms are visible on the map. Nonetheless, the final results were inaccurate because the straight lines also represented a problem. We did not use any additional sensory data, like IMU. When the LiDAR was traveling along a long wall, the algorithm could not decide if it was standing in a place or moving since it only saw a straight line without other waypoints. As a result of this, even though the individual rooms are correctly represented, their relative positions are incorrect. In most cases, they overlap, as can be seen in Figure 6.

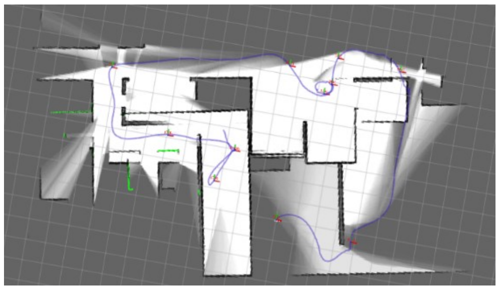


Figure 6. The smaller parts overlap because of the drifting.

The following tests were made with the actual sensors in a student dorm's corridors. This was a U-shaped corridor with a few room-like open spaces along with it. During the first measurements, the sensors (LiDAR, IMU) were held in hand, resulting in some swinging of the modules. The algorithm could use the data from the IMU to correct some of the swinging and project the measurements back to the horizontal plane, but it could only compensate for smaller movements (Figure 7.). In the following test, the sensors were fixed to a rolling platform to exclude this swinging factor.

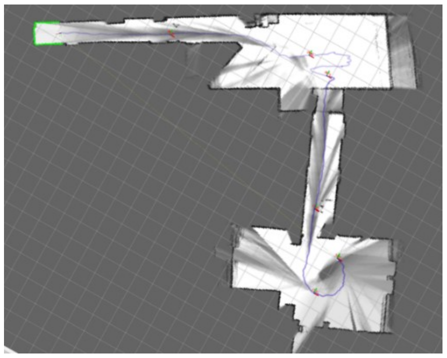


Figure 7. The corridors were narrow, so the swinging was smaller, correctable. In the large spaces, this was more noticeable and could not be entirely corrected.

At first, the LIDAR was at a 2-meter height, which meant that it only saw the walls and door frames. While the results were mostly accurate, there was drifting in this case, too, like in the simulated environment.

The final measurements were also made on the rolling platform but at the height of 40 centimeters. It meant that many objects appeared on the scans, such as chairs, tables, shelves, and couches. This introduced some noise, but it also created more waypoints that the algorithm could use and keep track of its position. The sensors were also brought into some connecting rooms along the corridor, and the algorithm was able to keep track of its position, and no drifting was present (see Figure 8).

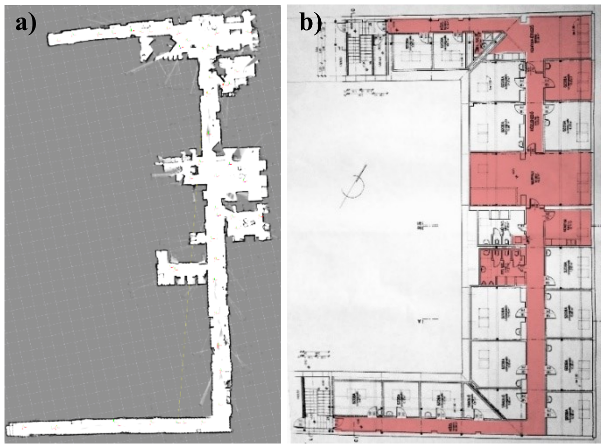


Figure 8. a) The final map, with more noise but also more accurate b) The floorplan of the student hostel, marked with red the areas traversed during the final test.

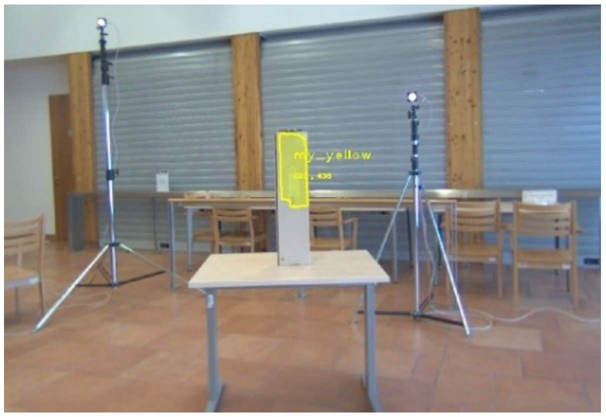


Figure 9. The recognized contours in the image.

VI. Camera module

It could be possible to create a 3D map using only laser rangefinders, but using a camera (or more) can also improve accuracy and provide additional information. For this purpose, we experimented with the camera and tried to determine the distance of a given object on the image based on the laser data. The experiment was done using the Optitrack reference system to check the error of our measurements. For the measurements, there was a given object of a predefined value, yellow, so it would be easy to localize in the image. The following tasks were implemented using the OpenCV library. To find the yellow object, a threshold was applied in the HSV color space. Based on the polygon given by the contours of the object, the centroid was calculated. (Figure 9.)

The centroid point had to be undistorted because the camera had a default distortion. Another ROS node was used to acquire the distortion parameters that were later used for the OpenCV's *undistortPoints* function. With this point, for the simplicity of the experiment, an assumption was made that the laser's points were in the same plane as this centroid point, meaning only the corresponding laser measurement had to found. The camera and LiDAR calibration described earlier can be used to validate this assumption. The laser rangefinder's data covered 180 degrees viewing angle with 512 measured points, while the camera covered 64 degrees with 1280 points. Based on this, the center portion of the laser data was taken and interpolated to find the one matching the centroid point. There were some errors because of the measurement noise, but the overall results were satisfactory, as shown in Figure 10.

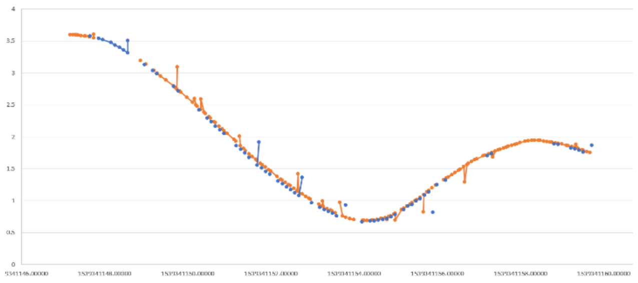


Figure 10. The calculated (blue) and Optitrack data (orange) closely follow each other; only a few errors occur. The y-axis shows the distance of the object in meters, and the x-axis shows the Unix time.

VII. Conclusion

The work so far is a proof of concept. The modules presented can be used to map an environment and all of it could be implemented onboard a drone. So far, we have a laser-based mapping system capable of creating 2D maps and a camera that can recognize a given object and determine its distance based on the laser data, all of this on a small 3D printed module that can easily be mounted on a drone.

The maps are created only in 2D and need to be extended to 3D. From the technology currently available, two viable solutions exist for an onboard sensor setup 1) solid-state 3D LiDAR and 2) two 2D LiDAR sensors rotated 90 degrees relative to each other. We are currently testing an Intel RealSense L515 camera [15].

The project's final goal is to create an autonomous system, meaning no user interaction would be needed for the drone to map the cave. While this is a distant goal, it requires that a map would be available in real-time. Instead of the offline map creation, a lower resolution map can be created in real-time onboard the drone simultaneously. The measurements can be recorded for offline high-resolution map creation.

Acknowledgment

This work has received funding from Pázmány Péter Catholic University KAP R&D and equipment grants (KAP20-51005-1.1-ITK, KAP20-53019-3.6-ITK, and KAP20-53019-3.6-ITK). The research was partially funded by the Hungarian National Research, Development and Innovation Office through the research grant TUDFO/51757-1/2019-ITM and PD 128699. The authors thank Levente Halmai for supplying the measurements from the simulated environment.

References

[1] "Odroid XU4." [Online]. Available: https://www.odroid.co.uk/hardkernel-odroid-xu4/odroid-xu4.

[2] "Pixhawk Mini." [Online]. Available: https://docs.px4.io/en/flight_controller/pixhawk_mini.html

[3] "URG-04LX-UG01: Specifications." [Online]. Available: https://www.hokuyo-aut.jp/dl/URG-04LX_UG01_Specification.pdf.

[4] P. Zsolt, "Polygon barlangtérképező program." [Online]. Available: http://www.barlang.hu/polygon/polygon/polygon.html.

[5] "Paperless Cave Surveying." [Online]. Available: https://paperless.bheeb.ch/.

[6] J. Mitchell, "Caveatron description," 2017. [Online]. Available: http://caveatron.com/.

[7] R. Zlot and M. Bosse, "Three-Dimensional Mobile Mapping of Caves," *J. Cave Karst Stud.*, vol. 76, no. 3, pp. 191–206, Dec. 2014.

[8] Qilong Zhang and R. Pless, "Extrinsic calibration of a camera and laser range finder (improves camera calibration)," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, 2004, vol. 3, pp. 2301–2306.

[9] P. T. Kassir Abdallah, "Reliable Automatic Camera-Laser Calibration," in *Proceedings of the 2010 Australasian Conference on Robotics & Automation*, 2010.

[10] "Iris - The Ready to Fly UAV Quadcopter." [Online]. Available: http://www.arducopter.co.uk/iris-quadcopter-uav.html#.

[11] "Prime 13." [Online]. Available: https://optitrack.com/products/prime-13/specs.html.

[12] G. Grisetti, C. Stachniss, and W. Burgard, "Improved Techniques for Grid Mapping With Rao-Blackwellized Particle Filters," *IEEE Trans. Robot.*, vol. 23, no. 1, pp. 34–46, Feb. 2007.

[13] S. Kohlbrecher, O. von Stryk, J. Meyer, and U. Klingauf, "A flexible and scalable SLAM system with full 3D motion estimation," in *2011 IEEE International Symposium on Safety, Security, and Rescue Robotics*, 2011, pp. 155–160.

[14] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2D LIDAR SLAM," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 1271–1278.

[15] "LiDAR Camera L515" [Online]. Available: https://www.intelrealsense.com/lidar-camera-l515/