Simulation of an Indoor Autonomous Drone for Assisting Police in Mass Shootings

Massachusetts Academy of Math and Science

Rumaisa Abdulhai

rabdulhai@wpi.edu

Table of Contents

I. Abstract

Mass shootings in schools and places of worship have claimed hundreds of lives in the United States during the last several years as there are no effective mechanisms to prevent them. This project investigates the use of autonomous indoor aerial vehicles to assist police in critical emergency tasks during school shootings. The Robot Operating System (ROS) and Gazebo software platform were used to successfully simulate the navigation of an unmanned aerial vehicle (UAV) from a start position to a goal position where the perpetrator and potential victims of a school shooting are likely to be found. First, a 2D occupancy map of a typical US middle school's indoor environment was achieved using a quadcopter, a 2D laser sensor, and the grid-mapping algorithm. Next, Adaptive Monte Carlo Localization (AMCL) was used with Dijikstra's global path planning and the Dynamic Window Approach (DWA) local planning algorithms to successfully navigate from a source point to a destination while avoiding obstacles. The mapping process took approximately 7 hours to map a middle school with a main office, media center, auditorium, cafeteria, courtyard, 4 hallways, and 12 classrooms. The drone navigated autonomously from the courtyard of the school to a hallway in about 110 seconds at an average speed of 0.14 m/s a distance of about 15 meters. Finally, an onboard camera on the UAV was used to successfully capture and transmit video from the scene in real time to a web server. This project serves as proof of concept that unmanned aerial vehicles or autonomous drones could be used to inform law enforcement of the potential dangers during mass school shootings thereby enabling them to take immediate and appropriate actions to save human lives.

II. Acknowledgements

I would like to thank my Beaver Works Summer Institute (BWSI) Instructors Dr. Mark Mazumder and Dr. Ross Allen for motivating me to embark on this project. I would like to thank my STEM advisors Dr. Kevin Crowthers and Dr. Raghavendra Cowlagi for their ontime guidance and holding me to the phases/deadlines throughout my project. I would also like to thank WPI graduate student Hanqing Zhang for introducing me to Gazebo and answering technical questions that arose from time to time. Finally, I would like to thank my parents for mental support and staying up late alongside me as I completed software intensive mapping and path planning tasks.

III. Literature Review

// Insert Literature Review here.

IV. **Introduction**

During the last several years, school shootings have claimed hundreds of precious student's lives in the United States. One can never tell when and where the next mass shooting will happen, inciting fear among parents, students, and school staff as well as worshippers in churches, synagogues, and mosques. With lawmakers sharply divided on gun control, there are no effective laws or procedures to prevent mass shootings. After the Parkland High School shooting in Florida where 17 students were fatally shot by a lone 21-year-old gunman [reference required], I have been thinking about how technology can be used to protect my school if a shooting were to happen there. The first potential solution that came to my mind is airport style checkpoints consisting of metal detectors and scanners at school entrances that can deter potential shooters. However, these are prohibitively expensive in the equipment, operations, and time delays that they would cause, especially during the morning opening hours of the school. They would also negatively impact the open character of a typical American public school. Terrestrial or ground robots are a second alternative solution that could be used to tackle potential shooters once a shooting has taken place. Terrestrial robots can approach the shooter and may try to disarm him without fear of losing a human life. But terrestrial robots are slow, expensive, and can be disengaged or tumbled over by a potential shooter relatively easily. Aerial robots such as drones are a third alternative that could be used in an indoor environment to deter potential shooters as they are fast, relatively inexpensive, and are being increasingly used in many outdoor applications such as agriculture and cinematography. Therefore, this research project investigates the usage of indoor autonomous drones to fight mass shootings.

Robots have evolved greatly from their initial use in assembly lines where they performed deterministic, repetitive actions to modern complex robots that can perform actions in a highly uncertain and uncontrolled environment. An autonomous drone is a highly complex aerial robot also known as an Unmanned Aerial Vehicle (UAV) that has found applications in agriculture, delivery, and cinematography. Drones have reduced the time for crop and soil inspection. One drone, namely the Agras DJI octocopter, can spray fertilizer with its 4 nozzles on up to 6000 square meters of land in 10 minutes (Puri, Nayyar, & Raja, 2017). The Zipline Company uses drones to deliver blood packages to hospitals in Rwanda cutting normal delivery time from hours to minutes thereby saving the lives of patients who suffer from blood loss during surgery (Ackerman & Strickland, 2018). The true feat of autonomous drone technology can be seen in the new Skydio 2, which can follow an individual participating in extreme outdoor sports at high speeds while avoiding obstacles and recording high-quality footage at the same time (Skydio Inc, 2019). Outdoor drone applications have been relatively easier to design and test due to the availability of GPS technology that works quite well in outdoor environments. Outdoor drones can localize themselves relatively easily and navigate to a specified point using GPS technology.

While there have been outdoor drone applications in agriculture, cinematography and delivery outlined above, building an indoor drone application has unique challenges: the drone must navigate through narrow and tight spaces while avoiding obstacles and fly in a GPS-denied environment (Khosiawan & Nielsen, 2016). An alternative to GPS is therefore required. Simultaneous Localization & Mapping (SLAM), is one such alternative, that has been used for indoor drone navigation. SLAM is the methodology of estimating the location of a robot in an environment while trying to form a map of the environment, using sensors such as lasers, monocular cameras, or binocular cameras (Li et al., 2016). Once a map is established and the drone is aware of its position in the environment, a path planning algorithm is necessary for the drone to efficiently navigate autonomously from a source point to a destination point while avoiding any obstacles in its path.

The following provides a vision of how autonomous drones or UAVs could be used to fight shooters and protect potential victims in mass shootings. When an intrusion is detected or a gunshot is fired at a school, one or more drones stationed indoors could fly to the scene within minutes or even seconds. The drone's onboard cameras can capture and relay video in real-time to law enforcement. This way, police are well aware of the scene before they arrive, which will help them greatly in taking appropriate action. One drone could track the perpetrator, while another could inform potential victims the safest route to exit the building. It is okay to lose a few inexpensive drones rather than losing precious human lives.

Although indoor drone mapping and navigation are much more technically challenging than outdoor, indoor drones hold great promise in applications such as warehousing, hotels, arenas, hospitals and schools for delivery and security as they are compact, fast and are able to record/transmit video from scenes. This project therefore investigates how an autonomous indoor drone application could be developed and used to assist law enforcement to fight mass shootings thereby saving precious human lives. It will investigate the use of an autonomous indoor drone to collect real-time information about the precise location and perpetrator(s) of a mass shooting in schools or places of worship. The specific goals of this project are to simulate mapping, path planning, and navigation of an indoor drone in a school environment to a point of emergency, collect real-time information about the location of the perpetrators involved, and relay it to a trusted group of individuals such as law enforcement who then can take appropriate action to save precious human lives. When a gunshot is fired or an intruder is detected, the drone will autonomously navigate to the point of disturbance and serve as a first responder to the scene. The drone's onboard cameras will be able to capture and relay live footage and provide a valuable window to critical pieces of information during a mass shooting to law enforcement fulfilling the needs of emergency responders. This way, police and emergency responders are well aware of the scene before they arrive, which is expected to reduce setup time and increase communication between the victims and responders thereby saving precious human lives.

The following provides an organization of this project report. The Materials and Methods section describes the software packages, methods, and algorithms used in the project. The Software Application Architecture section describes the ROS mapping and navigation nodes, topics and key parameters that are used in the development of this autonomous drone application. The Results section highlights the successful mapping of the model school and navigation of the drone between any two points achieved using the specified ROS nodes and algorithms. The Discussion/Conclusion section; analyzes the results achieved and places it into the context of the application in real life. The Future work section outlines the improvements and extensions that are possible for this project. The References section includes all the sources used during the project. The Appendices section includes the code developed for the indoor drone application throughout the project.

V. Materials and Methods

This section covers the software tools, methods, and procedures used in this project to build the autonomous indoor drone application that allows the robot to map an indoor environment and navigate from a source point to a destination point.

Software Tools

A Linux machine running ROS Kinetic and Ubuntu 16.04 were used for the project. ROS, known as the Robot Operating System, provides the infrastructure for building autonomous robots. It is open source, available in multiple programming languages, and contains libraries for building and writing code for robots whether terrestrial or aerial. With a subscriber/publisher system, multiple parts of the robot can listen to sensor output by subscribing to topics where the information is published and take appropriate action based on these measurements. There are various ROS versions, but the most commonly used version is ROS Kinetic Kame. The programming language Python was used with ROS Kinetic to write the code along with an Integrated Development Environment (IDE) called Visual Studio Code. The code developed is not guaranteed to work for other ROS versions, but many packages used in this project are available in other ROS distros.

Along with a full installation of ROS, the Gazebo Simulator is included and used to develop the autonomous drone application in the project. The Gazebo Simulator is a well-known industrial platform for developing and testing realistic 3D simulations of robots in both indoor and outdoor environments. It provides support for real sensors in the market such as monocular or stereo cameras, and 2D or 3D lasers. Gazebo environments are known as worlds. One of the first steps of this project involved using the Gazebo Simulator to construct a simple indoor environment with walls as obstacles. The standard wall model provided by Gazebo was modified in size to produce a 7.5x20x20 m^3 environment. A drone was instantiated in the indoor environment in Gazebo, and

was able to map the simple environment using the gmapping SLAM algorithm and autonomously navigate from one point to another using local path planning and global path planning methods which will be explained more in detail later.

Installation

The first step in the entire process was to set up the development environment and download the necessary packages. A Lenovo laptop running Windows was set up with a Linux machine. Windows was uninstalled and Ubuntu 16.04 was installed as it is compatible with ROS Kinetic. The first step after setting up the Laptop's OS was installing the full version of ROS on the Linux machine. Next, a catkin_ws workspace was initialized with a src subfolder. The hector quadrotor stack was cloned into the ~/catkin_ws/src folder. The hector_quadrotor stack provides the drone model used for this project which was a quadcopter with the Hokuyo 2D lidar sensor. The specific package within the stack that provides the drone model is called hector_quadrotor_description. In addition, the gmapping package, the amcl package, and the move_base packages were installed into the opt folder on the Linux machine. These three packages were used for mapping, localization, and navigation, respectively. The role of these packages will be explained more in detail later on. The instructions for installing ROS and the packages can be found on the GitHub page for the project.

Creating Packages

The second step was developing a package for the application. The first subpackage created in this package was the quadcopter_gazebo package. Inside this package is the world file which contains the code for the indoor environment. The world was developed directly on the simulator, and the code for the specific configuration of the environment was stored in the world file. The launch file for the rviz visualization tool and Gazebo world was also created in that package. Rviz is a visualization tool used in ROS to view the information being broadcasted by nodes of the program. Rviz is used to view the laser scans and the map being updated in real time. The other subpackages created were the quadcopter_navigation and takeoff_land packages, respectively, for navigation and hovering of the drone. The quadcopter_navigation package includes the saved map of the environment, the configured rviz file for visualization of ROS topics, as well as the launch files for the amcl, gmapping, and move_base nodes. The takeoff_land package contains two python files. One python file is for hovering the drone at a specified height above the ground while another python file is for teleoperating, or controlling the drone. The teleoperated code allows the drone with 2 controllable degrees of freedom out of the 6 degrees of freedom: y axis motion and the yaw motion.

Mapping Phase

After developing a package of the application and building the indoor environment, the code for mapping the indoor environment using the gmapping SLAM algorithm was executed. The gmapping node is an implementation of Simultaneous Localization & Mapping (SLAM). The node's input is the laser scan data from the Hokuyo 2D LIDAR sensor and the output is a 2D probabilistic occupancy grid map of the environment. The occupancy grid map continuously updates as the drone explores the environment (teleoperated). The white space on the map means free space, the black area means obstacles, and the grey area means the area is unexplored. Each square on the discrete map represents the probability that it is an obstacle. The instructions for executing the code are provided in the Appendices Section.

Navigation Phase

For the navigation phase, the amcl and move_base nodes are mainly used. AMCL stands for Adaptive Monte-Carlo Localization. It is a localization method which uses a particle filter to estimate the pose (position and orientation) of the drone in the environment. The node subscribes to the current laser scans and the occupancy grid map which are compared against each other to localize the drone. The node outputs or publishes the particle filter which contains all the estimates of the robot's pose.

The move_base node allows for the 2D navigation goal to be used in rviz. It listens for the goal to be set in rviz and outputs the necessary velocities needed for the drone to navigate from its current pose to the goal pose. The node contains the global and local path planners. The global planner makes a path from the start point to the goal point. Some global path planning methods include A*, RRT*, and Dijikstra's Algorithm, but this code uses Dijikstra's Algorithm. The global planner may be changed through the yaml parameter files. The local path planning method used is called the dwa_local_planner (Dynamic Window Approach). For each time step, the local planner generates sample velocities that the robot can take. It predicts the possible trajectory of the robot with each of the sample velocities and finds the optimal trajectory based on the distance to the goal, the speed of the robot, distance to obstacles, and the robot's distance to the global path. Both nodes are required for the drone to successfully navigate from the source point to the destination point. Rviz was used to view the local path planning trajectory or the particle filters in real time. The instructions for executing the code are provided in the Appendices Section.

VI. Software Application & Architecture

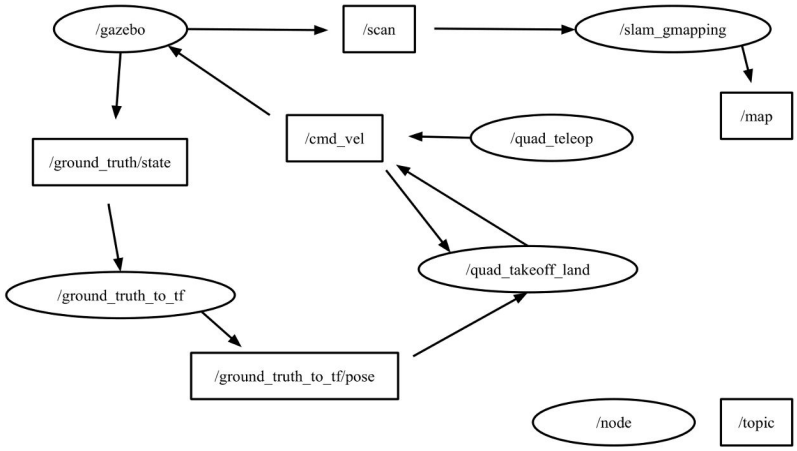


Figure X: ROS Mapping Architecture

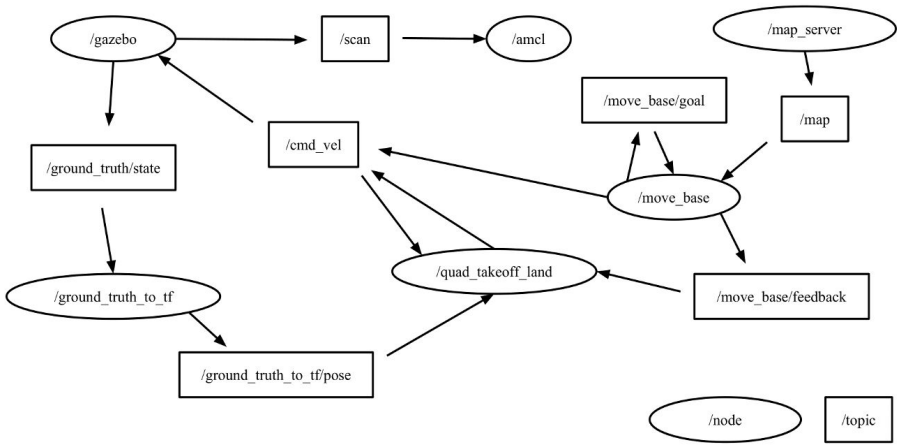


Figure Y: ROS Navigation Architecture

VII. Results

This section presents the results obtained using the developed autonomous indoor drone application. The results are divided into creation of an indoor environment, mapping the environment, and finally navigating the environment.

Creation of an Indoor Environment

One of the first steps of this project involved using the Gazebo Simulator to construct a simple indoor environment with walls as obstacles. The standard wall model provided by Gazebo was modified in size to produce a 7.5m x 20m x 20m environment. Several of these walls were used and placed in certain orientations to produce the desired environment. The sample indoor environment

consisting of two school hallways that was created is shown in Figure 1. This indoor environment is used for the subsequent mapping and navigation phases.

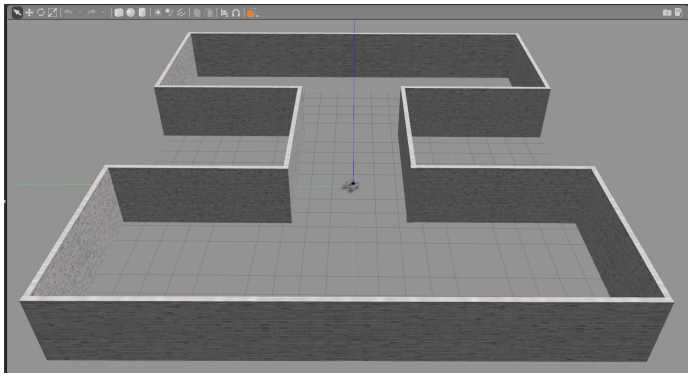


Figure 1: The Simple Gazebo Indoor Environment

Mapping Phase

After building the indoor environment, a quadrotor with a Hokuyo 2D lidar was instantiated inside at the center of the environment. The quadrotor is shown in Figure 2 below. Next, the quadrotor was commanded to take off and hover at a specified height using the created takeoff_land node. The teleoperated node for controlling the drone around the indoor environment in the x and y plane using keyboard keys was initiated. Next, the gmapping node for 2D mapping of the indoor environment was initiated and the drone began collecting the LaserScan data. The node's input is the laser scan data from the Hokuyo 2D LIDAR sensor and the output is a 2D probabilistic occupancy grid map of the environment. The occupancy grid map continuously updates as the drone explores the environment (teleoperated). A snapshot of the mapping phase and an intermediate image of the resulting probabilistic map is shown in Figures 3 below. The white space on the map means free space, the black area means obstacles, and the grey area means the area is unexplored. The final completed map is shown in Figure 4. The complete process of mapping was recorded and can be found at the link below.

https://drive.google.com/drive/folders/1zvveXdScOxd2hiGnyvaLpDm5zXCi5mQ_

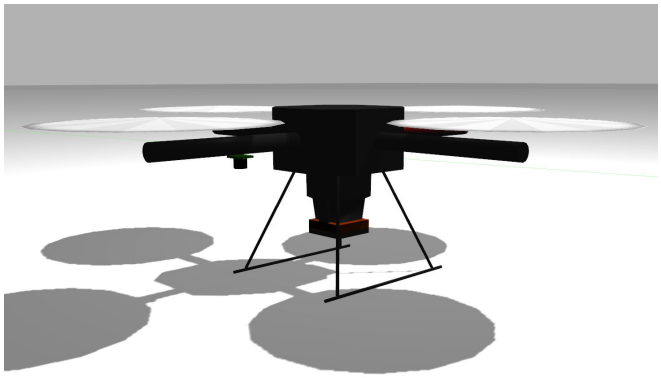


Figure 2: Instantiated quadrotor with a Hokuyo 2D Lidar

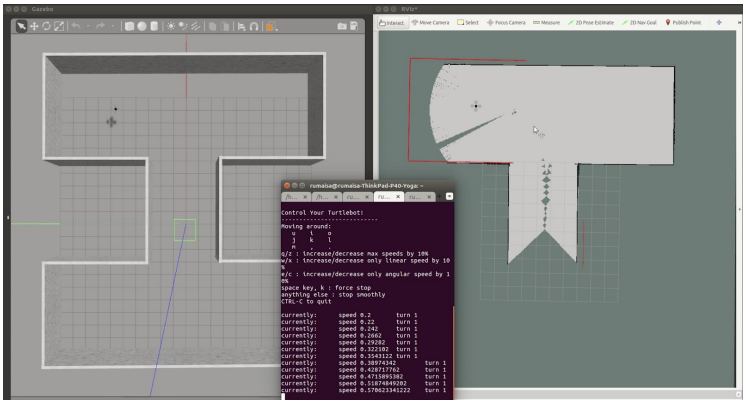


Figure 3: Snapshot of the Mapping Phase of the Environment

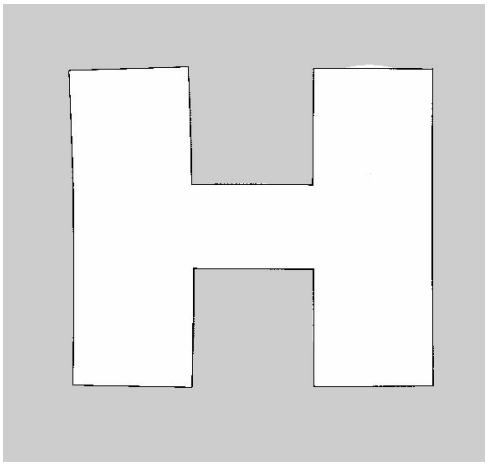


Figure 4: The Completed Probabilistic Map

Navigation Phase

A drone was instantiated in the indoor environment in Gazebo at the center of the indoor environment and hovered to a specified height using the takeoff_node. The Rviz visualization tool was then launched for commanding the drone to a certain

destination point using the 2D navigation goal. The move_base and amcl nodes for path planning and localization were launched. The 2D navigation goal was specified, and the drone was able to autonomously navigate from the source point to the specified destination point using Dijikstra's algorithm and the DWA Local Planner. The time for the drone to complete each 2D navigation goal was dependent on the distance between the source and destination points and the specified speed of the drone. A snapshot of the navigation process in the environment is shown in Figure 5 below. The entire navigation process was recorded and the video can be viewed at the link below.

https://drive.google.com/drive/folders/1zvveXdScOxd2hiGnyvaLpDm5zXCi5mQ_

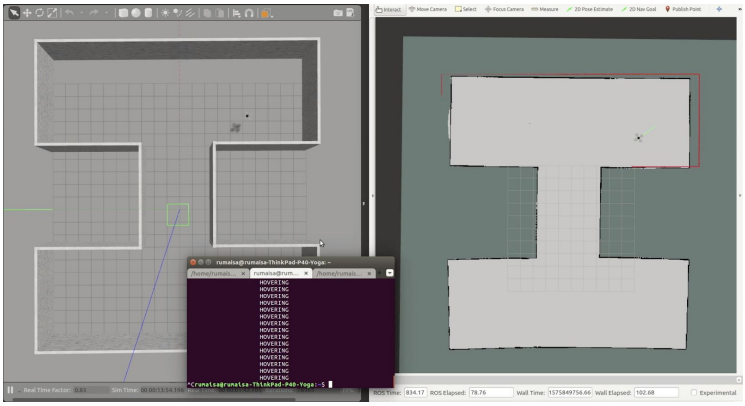


Figure 5: Snapshot of the Navigation Process using Dijikstra's Algorithm and DWA

Final Results

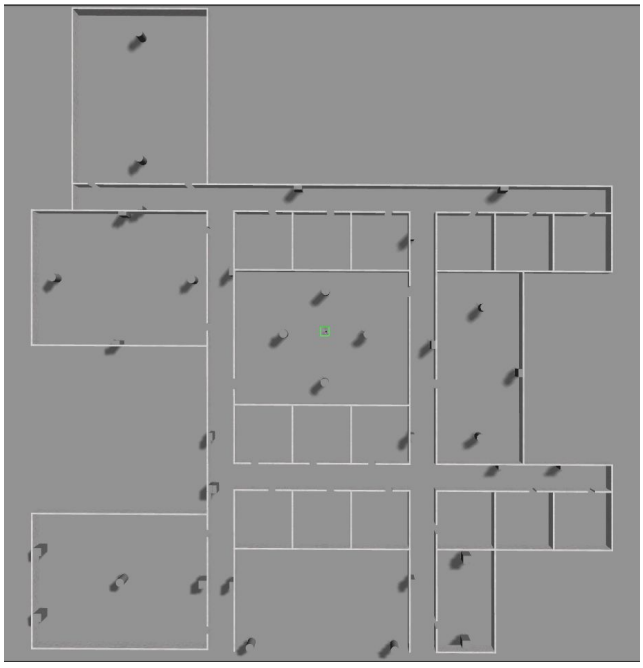


Figure 6: Top-Down View of the School Environment in Gazebo

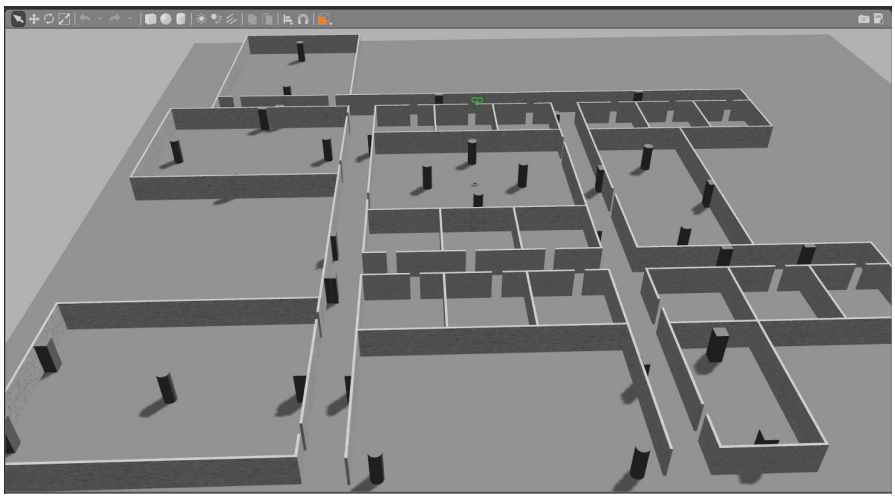


Figure 7: Birds-Eye View of the School Environment in Gazebo

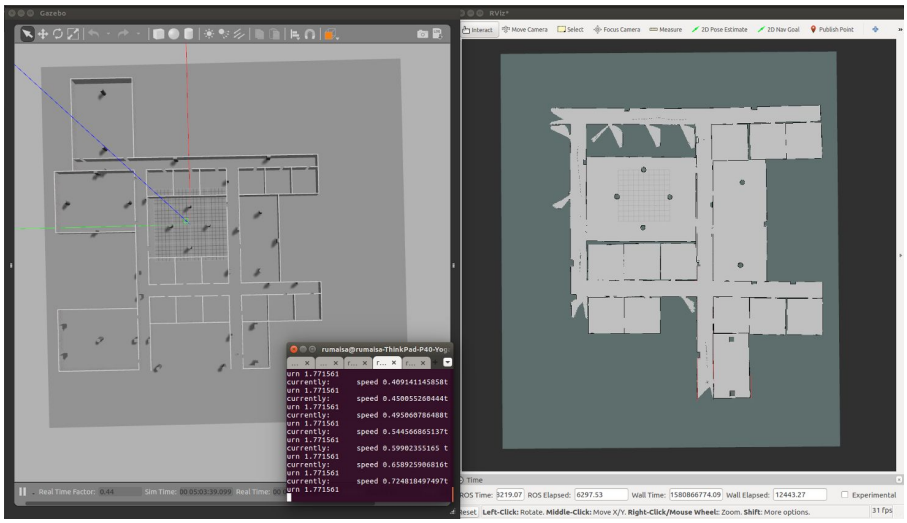
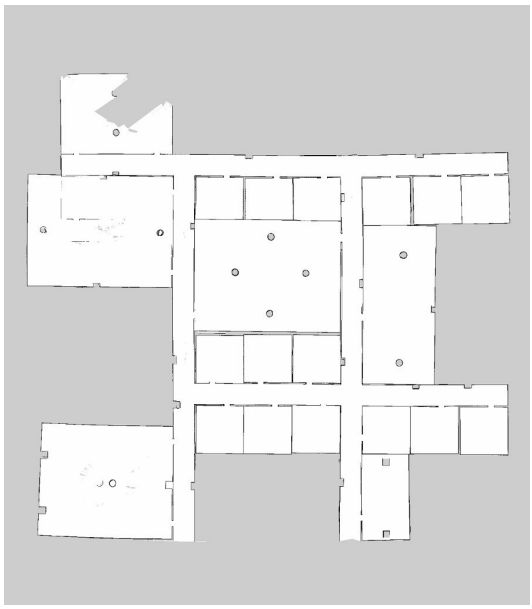


Figure Z: Navigation of the quadcopter in the school environment-0



VIII. **Discussion/Conclusion**

This project investigated the use of autonomous or unmanned aerial vehicles (UAVs) to assist police in critical emergency tasks during school shootings. The Robot Operating System (ROS) and Gazebo software platform were used to successfully simulate and showcase a proof of concept software system that demonstrated how such an indoor UAV application could be built and used during mass shootings in an indoor school environment. First the computer assisted design (CAD) features of Gazebo simulator were used to build an indoor environment of a typical US middle school consisting of the school main office, media center, auditorium, cafeteria, courtyard, 4 hallways, and 12 classrooms. This process took approximately 4 hours over the course of a week. Second, ROS, a quadcopter with a Hokuyo laser and gmapping, an implementation of the SLAM method, were used to successfully map the designed school environment. The resulting 2D occupancy grid map captured the free spaces and obstacle spaces of the school's indoor environment and clearly delineates the boundaries between the two spaces. Next, Adaptive Monte Carlo Localization (AMCL) was used with Dijikstra's global path planning and the Dynamic Window Approach (DWA) local planning algorithms to enable the drone to successfully navigate from a starting position to a destination position while avoiding obstacles. The 2D occupancy grid map is essential for an indoor drone application to avoid obstacles and navigate the tight spaces of the GPS denied indoor school environment unlike an outdoor drone application where the availability of GPS makes navigation a lot easier. The mapping process took approximately 7 hours to build the 2D occupancy grid map of a moderately complex middle school. Finally, an onboard camera on the UAV was used to successfully capture and transmit video from the scene in real time to a web server. In conclusion, this project built a successful proof of concept indoor drone application that demonstrated how an unmanned aerial vehicle or autonomous drone could be used to inform law enforcement of the potential dangers during mass school shootings thereby enabling them to take immediate and appropriate actions to save human lives.

Several improvements are possible with this project. First, although the CAD features of Gazebo were used to build the school environment from scratch in this project, the process could be automated and the time required can be shortened by building tools that can import a 2D map layout and convert the layout into a real 3D indoor environment. This addition may result in a CAD project in itself. Second, the time required for the mapping process could be shortened from the current 7 hours for a medium sized middle school to about 2-3 hours. This improvement would require extensive time and effort to optimize the software modules or nodes in ROS that implement Simultaneous Location and Mapping (SLAM). In addition, more realistic obstacles related to the indoor environment could be placed in order to improve the quality of the 2D occupancy grid map and more accurately depict the real-life environments. Furthermore, although a Hokuyo lidar was used to build the 2D grid map during the mapping process, lidar devices for

detection of obstacles are bulky and typically used in outdoor applications such as self driving cars. Lidar devices with smaller form factor are coming on the market and could be used in drones in the future. Instead of using lidar scans to build the 2D occupancy grid map, a stereo camera such as an Intel Realsense r200 mounted on the front of the drone could be used to obtain camera scans and build a map of the environment. Real sense cameras are cheaper and lighter than lidars and are readily available in research drones and therefore could be leveraged for indoor drone applications. These improvements in turn would make the project more realistic and feasible for application in real indoor environments such as schools and places of worship. Although the autonomous indoor drone proof of concept application was specifically built for an indoor school environment, the concept could be generalized relatively easily and the software application could be adapted for a variety of indoor environments such as a church, synagogue, mosque, a hotel or a convention center.

IX. References

Abdulhai, R. (2019). rumaisaabdulhai/PathPlanning. Retrieved from https://github.com/rumaisaabdulhai/PathPlanning/blob/master/rrt_modified.py.

Ackerman, E. , & Strickland, E. (2018, Jan). Medical delivery drones take flight in east africa. *IEEE Spectrum, 55*, 34-35. doi:10. 1109/MSPEC. 2018. 8241731 Retrieved from https://ieeexplore. ieee. org/document/8241731

Chin, T. (2019, February 26). Robotic Path Planning: RRT and RRT*. Retrieved from https://medium. com/@theclassytim/robotic-path-planning-rrt-and-rrt-212319121378.

Correll, N. (2014, October 14). Introduction to Robotics #4: Path-Planning. Retrieved from http://correll. cs. colorado.edu/?p=965.

Guermonprez, P. (2017, November 13). Autonomous Drone Engineer - A1 - Intel Aero in 5mn. Retrieved from https://www. youtube. com/watch?v=7t7l885g8dI.

Khosiawan, Y. , & Nielsen, I. (2016). A system of UAV application in indoor environment. *Production & Manufacturing Research, 4*(1), 2-22. doi:10. 1080/21693277. 2016. 1195304

Kudan. (2016, May 23). An Introduction to Simultaneous Localisation and Mapping. Retrieved from https://www. kudan. io/post/an-introduction-to-simultaneous-localisation-and-mapping.

Matarić Maja J. (2008). *The Robotics Primer*. Cambridge, Mass: The MIT Press.

Melissa Yan (2014). Dijkstra's Algorithm [PDF file]. Retrieved from https://math. mit. edu/~rothvoss/18. 304. 3PM/Presentations/1-Melissa. pdf

Li, J. , Bi, Y. , Lan, M. , Qin, H. , Shan , M. , Lin, F. , & Chen, B. M. (2016). *Real-time Simultaneous Localization and Mapping for Uav: A Survey*. *Real-time Simultaneous Localization and Mapping for UAV: A Survey*.

OSRF. (n. d.). Beginner: Overview. Retrieved from http://gazebosim. org/tutorials?tut=guided_b1&cat=.

Puri, V. , Nayyar, A. , & Raja, L. (2017). Agriculture drones: A modern breakthrough in precision agriculture. *Journal of Statistics and Management Systems, 20*(4), 507-518. doi:10. 1080/09720510. 2017. 1395171

ROS Wiki. (n. d.). Retrieved from http://wiki. ros. org/ROS/Introduction.

Skydio Inc. (2019). How does Skydio 2 work? (n. d.). Retrieved from https://support. skydio. com/hc/en-us/articles/360036116834-How-does-Skydio-2-work-.

Swift, N. (2017, March 1). Easy A* (star) Pathfinding. Retrieved from https://medium. com/@nicholas. w. swift/easy-a-star-pathfinding-7e6689c7f7b2.

Thrun, S. , Burgard, W. , & Fox, D. (2010). *Probabilistic Robotics*. Retrieved from https://docs. ufpr. br/~danielsantos/ProbabilisticRobotics. pdf

Wenderlich, R. (2011, September 29). Introduction to A* Pathfinding. Retrieved from https://www. raywenderlich. com/3016-introduction-to-a-pathfinding.

Williams, B. , & Frazzoli, E. (2010). *16. 410 Principles of Autonomy and Decision Making.* Massachusetts Institute of Technology: MIT OpenCourseWare. Retrieved from https://ocw. mit. edu.

X. Appendices

Mapping Phase

In the first Terminal Tab, the launch file for running the Gazebo world and Rviz tool was executed:

```
roslaunch quadcopter_gazebo quadcopter.launch
```

In the second Terminal Tab, the launch file for running the gmapping node which creates a map of the environment was executed:

```
roslaunch quadrotor_navigation quadrotor_mapping.launch
```

In the third Terminal Tab, the code for hovering the drone at a constant height was executed:

```
rosrun takeoff_land takeoff_code.py
```

In the fourth Terminal Tab 4, the code for teleoperating the drone using keyboard keys was executed:

```
rosrun takeoff_land quad_teleop.py
```

In the fifth Terminal Tab after mapping is complete, the map is saved to the desired directory on the computer:

```
rosrun map_server mapsaver -f /home/<username>/catkin_ws/src/quadrotor_navigation/maps/new_map
```

After the map has been saved, close all terminal tabs.

Navigation Phase

In Terminal Tab 1:

```
# Runs the Gazebo world and rviz visualization tool
roslaunch quadcopter_gazebo quadcopter.launch
```

In Terminal Tab 2

```
# Runs the code for hovering the drone
```

```
rosrun takeoff_land takeoff_code.py
```

In Terminal Tab 3

```
# Allows 2D nav goal for navigation in Rviz

roslaunch quadrotor_navigation quadrotor_move_base.launch
```