Drone Express

TELEGRID

DRONE EXPRESS - LOCAL OBSTACLE AVOIDANCE

1 INTRODUCTION

Obstacle avoidance is a crucial element in Autonomous Unmanned Aerial Vehicles (UAVs) operations. Drone collisions are capable of not only destroying the drone but also causing damage to objects and people on the ground. Obstacles, in general, fall into two categories - local obstacles and global obstacles. Local obstacles are defined as dynamic objects which suddenly appear in the UAV's path such as birds in flight, other drones, or unknown objects. Global obstacle are defined as stationary objects that do not change their location frequently such as tall trees or cell towers. They also include areas that are known and categorized as prohibited or potentially dangerous for UAV flight including restricted air space as well as places on the ground that would incur significant damage in case of a crash.

Obstacle avoidance is accomplished by including, in the UAV's overall navigation system, Obstacle Avoidance Algorithms that prevent collision with both global and local obstacles. They mitigate potential collisions by changing flight path in real time and re-route the UAV as necessary while in flight. The algorithms are highly sophisticated and require considerable computer processing power which is not usually available onboard standard UAVs. Also the extra weight and power requirements of the processor impact to some degree both flight time and payload capabilities. For that reason most UAV designers choose to process these algorithms on the ground portion of the overall navigation system or the Cloud, not on the UAV itself. The problem with this approach is that it relies on having a constant, low latency wireless connection between the UAV and the ground network. Low latency is critical because once an obstacle is identified even the smallest amount of latency in notifying the UAV could result in a collision.

Drone Express has, instead, chosen to process both global and local obstacle avoidance algorithms on the UAV itself. For that purpose each of its drones includes a high-computational embedded system-on-module (SOM) co-processor with GPU. Analysis has shown that the extra weight and power consumption of the SOM will be almost negated by the reduction in traditional obstacle sense and detect hardware that is normally needed on a UAV.

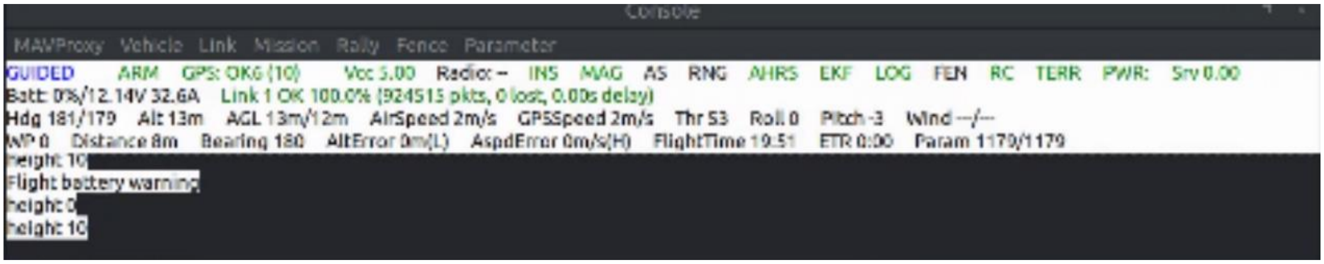
2 LOCAL OBSTACLE AVOIDANCE ALGORITHM

The most common strategy for local obstacle avoidance relies on the Vector Field Histogram (VFH) technology where a sonar element is used to generate a 2D occupancy grid of the environment which is then mapped to a 1D polar histogram. To date VFH algorithms have been used in a 2D environment mostly in ground robots. Drone Express has chosen the 3DVFH* algorithm where a global map of the environment is built in the form of an Octomap. This global map is used to extract local information which produces a bounding box around the UAV in order to perform the histogramic obstacle avoidance. This obstacle avoidance strategy falls somewhere between the global and local obstacle avoidance approaches.

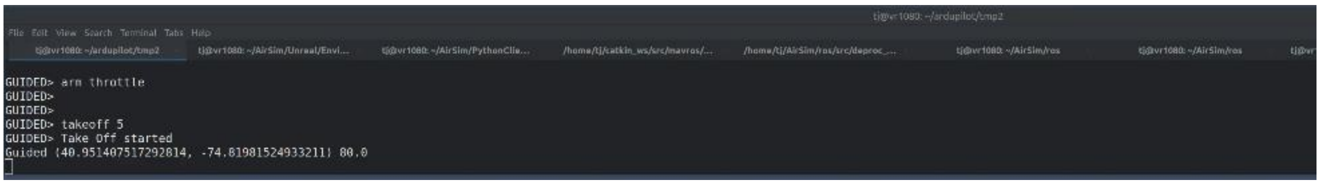
3 OBSTACLE AVOIDANCE SIMULATION

To demonstrate this obstacle avoidance technique a drone simulator developed by AirSim was selected. AirSim is an open-source simulator for drones, cars and other vehicles, built on Unreal Engine and supports hardware-in-loop with the PX4 flight controller for physically and visually realistic simulations. AirSim is configured to simulate a single stereo depth camera, same specs as the intel realsense d435i, and outputs depth frames to a ROS topic at 30fps

The flight controller firmware, Ardupilot, is configured as a multirotor controller and is used to control the actual simulation.



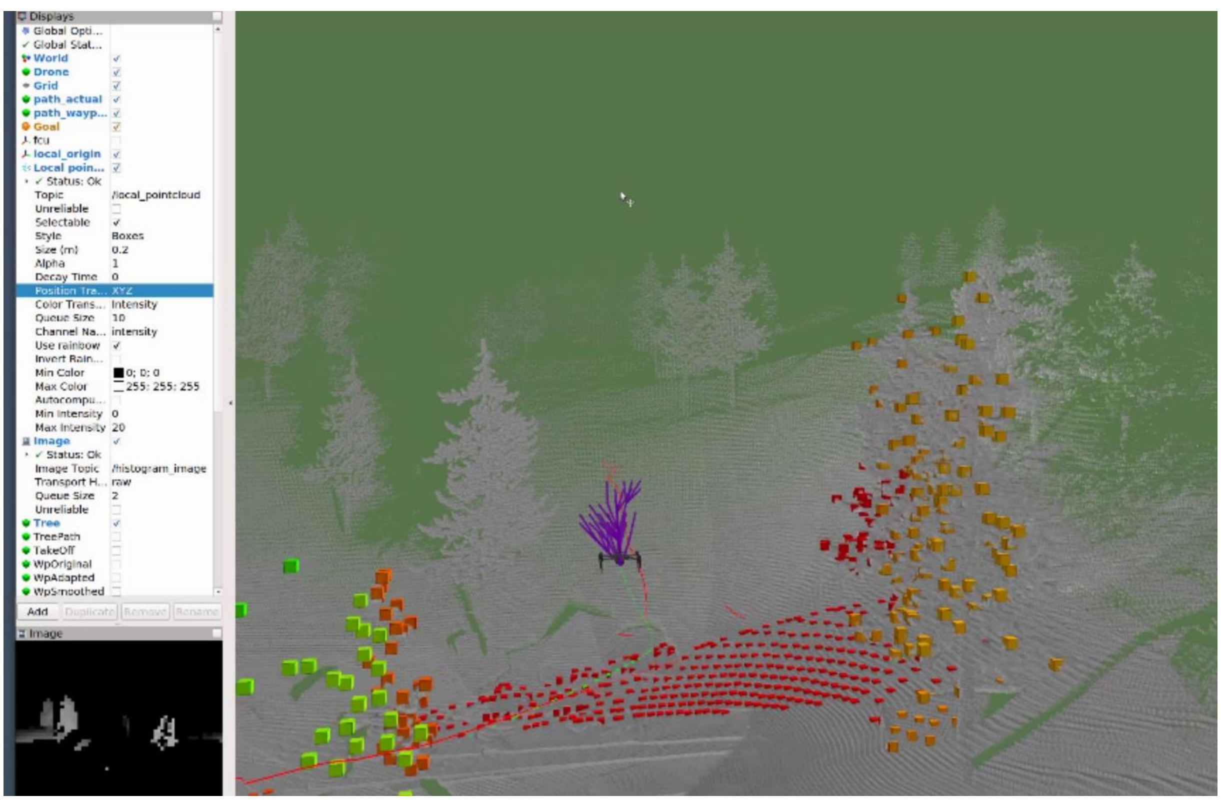
Ardupilot Output



Mavproxy Output

MAVROS is used to connect the mavlink bus on the Ardupilot that is sending imu/position information to a ROS topic and receives MAVROS commands from ROS to Ardupilot (like setpoint).

A ROS based local planning algorithm takes the imu/position information + depth camera frames and outputs back to MAVROS in the form of continuous stream of 'setpoint' commands to modify the drone's path. It also outputs the local pointcloud the algorithm is currently using + nav path to ROS.



RViz Scene

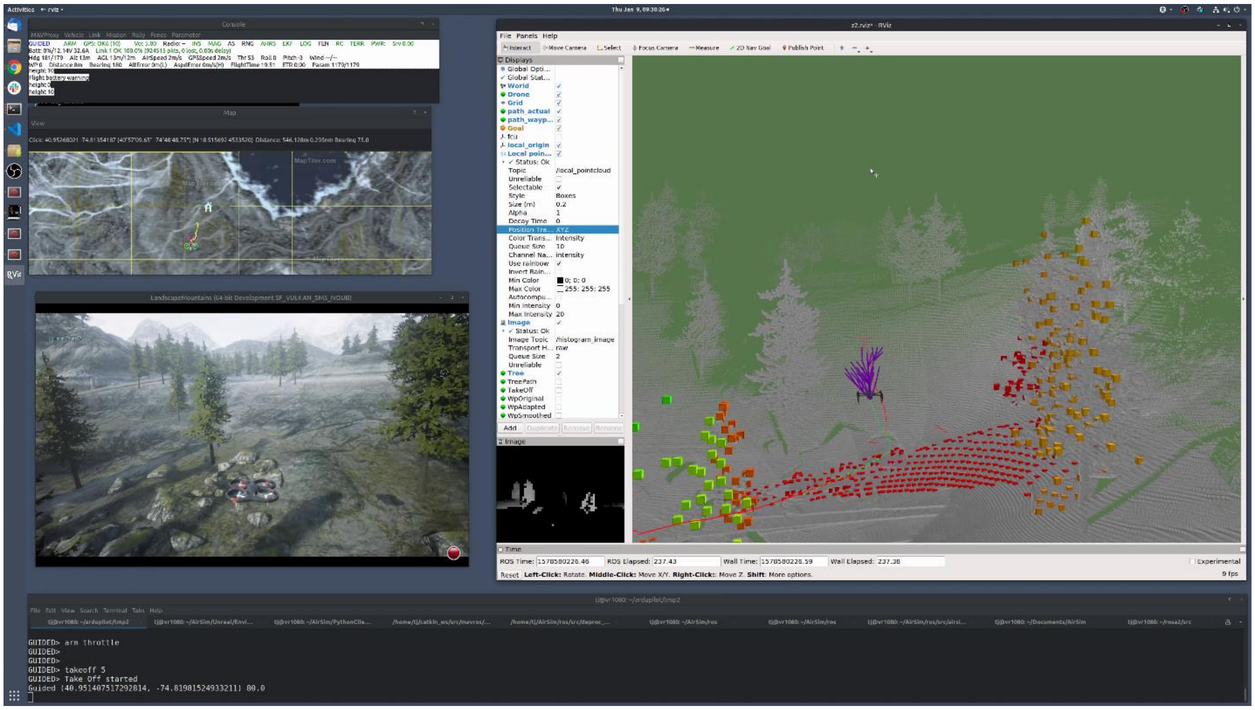
RViz is a 3D visualizer for the Robot Operating System (ROS) framework. Extended depth information is plotted in RVIZ (grey part of the RViz scene) - not used by the algorithm though as the depth range of the camera is only ~10m- but useful for debugging/visualization. RViz also shows the current drone imu/position data in space and also the theoretical (red line) path coming out of the planner and (green line) actual path.

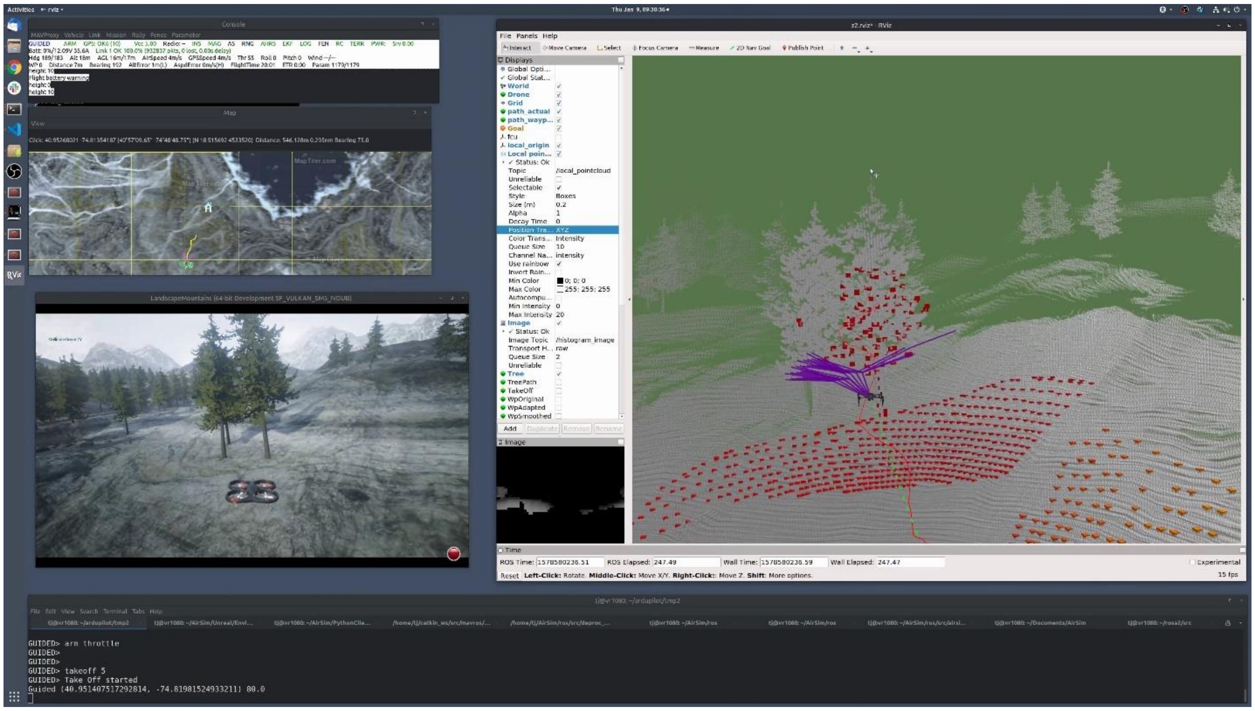
3 CONCLUSION

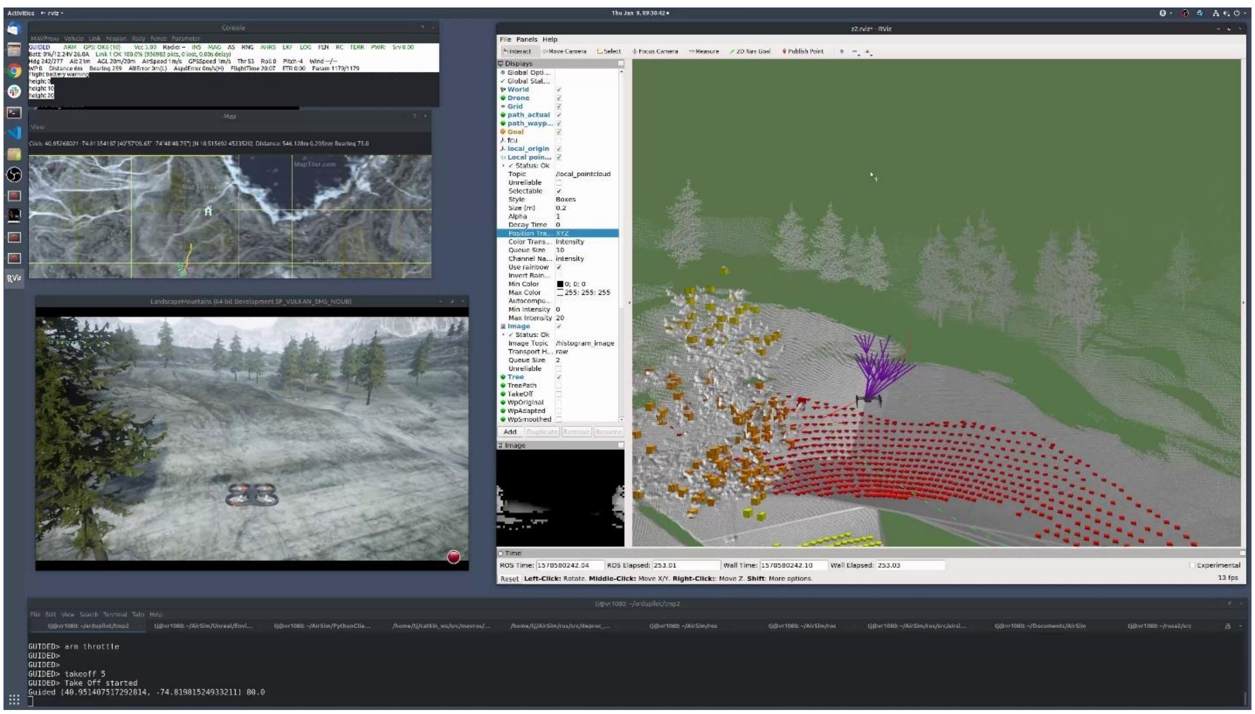
To date, results of simulating the 3DVFH* algorithm using this setup have been very promising. They show that the algorithm is capable of navigating around obstacles efficiently given the type of obstacles expected to be encounter in an outdoor-delivery scenario.

The screen shots below show a number of different cases where a waypoint is assigned (not obvious from the screen shots) about 500m ahead of where the drone is traveling. In all the cases, with only a single front-facing stereo depth camera, the algorithm is able to both maintain drone’s distance from the mountainous environment and at the same time provide adequate navigation points for the drone to move around trees as it approaches them within about 10m.

The purple lines show the actual tree paths that are being calculated in real time and the path finding logic decides which is best given various weights applied (distance/velocity cost weights).







A global obstacle avoidance algorithm will be used to establish a preliminary flight plan to the destination that avoid stationary or known obstacles. The local obstacle avoidance algorithm will be used only when the drone encounters an obstacle in-flight in order to navigate around it. Because this algorithm is performed locally it cannot do a proper lowest-cost path plan so it's only used in cases where it needs to get around transient obstacles.

This simulation is performed using a single depth camera, however, it will be expanded to include an additional camera on the bottom and top. The point clouds will then be taken from all the cameras and merged into a single point cloud input to the algorithm. This will provide a more complete cloud for the local obstacle avoidance algorithm to use.