Final documentation: MobGuardian

Arjun Singh, Jorge Navarro, Edgar Dasilva

Abstract

We believe we can create a novel solution for performing “surveillance” tasks, such as identifying the act of drowning in an individual, using an autonomous drone. The goal of the project is to implement a Proof of Concept (PoC) for drones to seek and identify certain “behaviors” in action figures. By combining pose estimation and object detection with autonomous aerial systems, we will perform this behavior categorization on the action figures. We were able to identify different postures, while flying the drone and generate signals that would help the drone reposition based on the action or position captured.

1 Introduction

Our team will take the next step in surveillance technology by building a generalized computer vision framework for drones to search for target “behaviors” in a specified area.

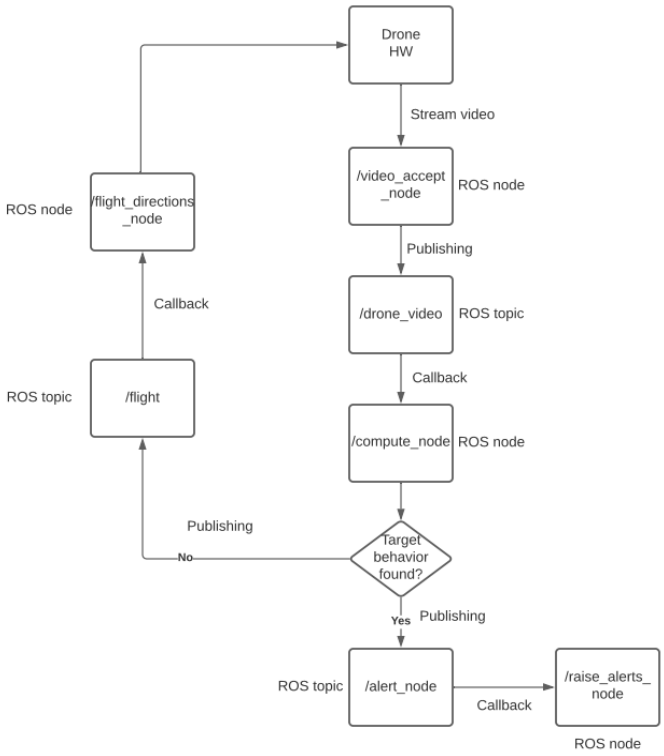
1.1 Goal of the Project

The goal of the project is to implement a Proof of Concept (PoC) for drones to seek and identify certain “behaviors” in action figures. By combining pose estimation and object detection with autonomous aerial systems, we will perform this behavior categorization on the action figures. Here, a “behavior” is some object, pose, or combination defined by the stakeholder. Similarly, action figures represent humans.

1.2 Level of Autonomy

We will develop an autonomous level 5 system for searching and identifying behaviors. Use cases include drowning detection, active shooter detection, and theft prevention. A fully autonomous system will send a signal to a stakeholder (coast guard, police officer, night guard, etc.) for further investigation if a behavior is detected.

1.3 Description



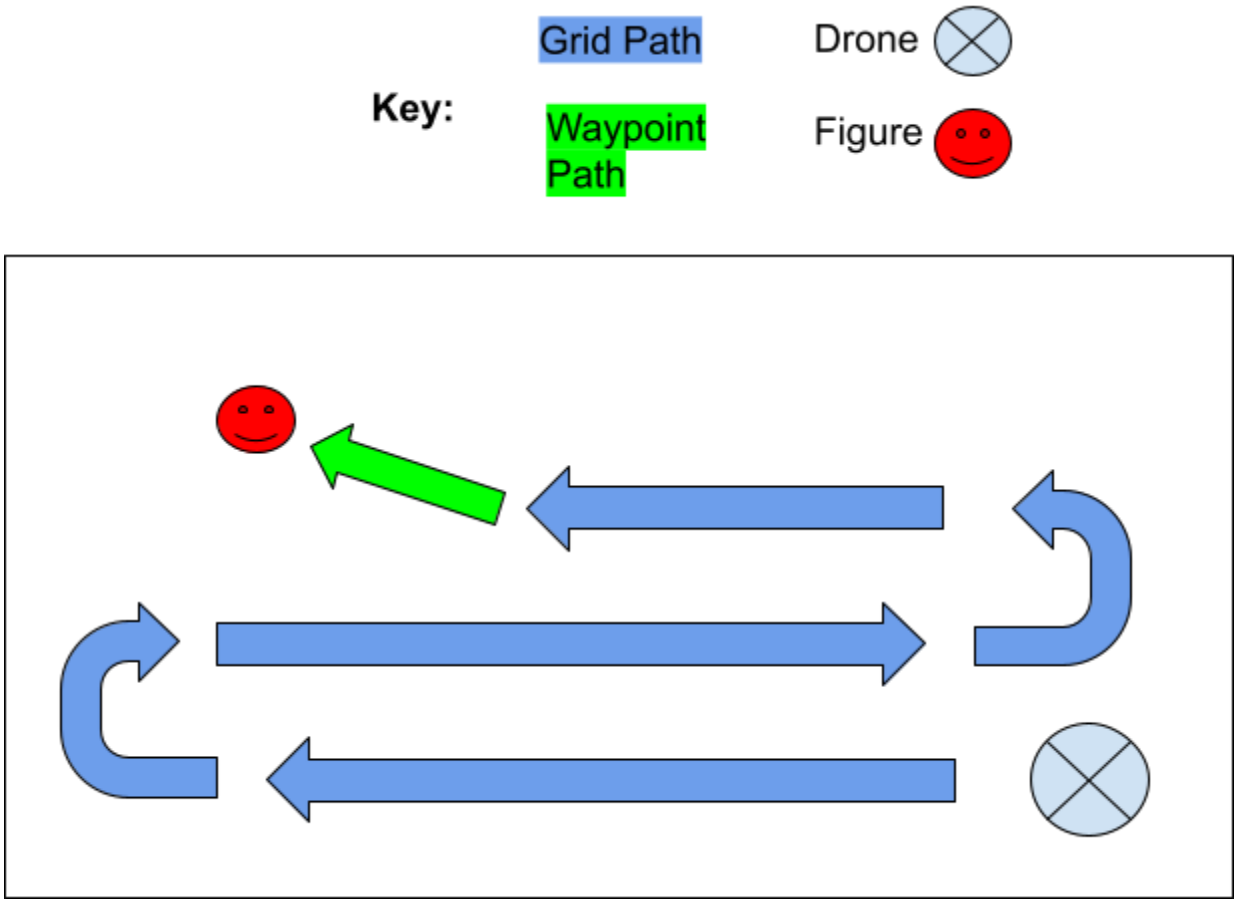
The overall architecture depends on edge computing. The programmable drone transmits images and meta data to the edge and the edge uses this information to determine a course of action and the following set of instructions for the drone. This feedback loop will not involve humans beyond activating and deactivating the system.

The image detection system will be trained on images taken of the action figures. We will take pictures of action figures in various positions, perform image transformation, pose estimation, and feed the results to train a Deep Neural Network (DNN). We will also leverage transfer learning on image net to build a basic detector for finding the action figures.

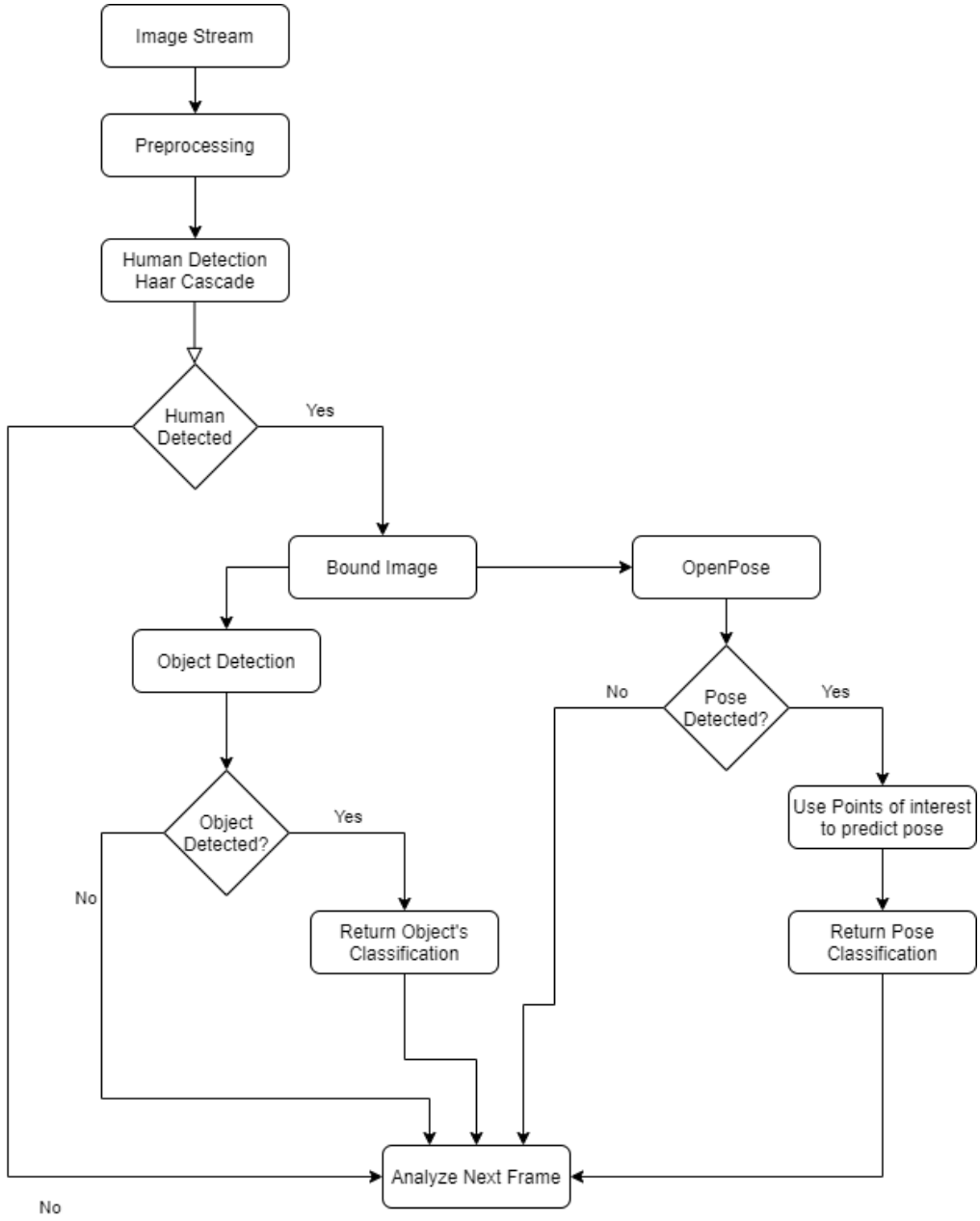
The repositioning program will be in charge of moving the drone in a specific manner in order to clearly capture the position of the figures. The drone must be able to capture images from different angles and perspectives until it is able to successfully identify the specific behavior the figure is conveying. As of the moment, the idea is to create a loop or boolean variable that constantly checks for available positions, and the drone will keep on moving until this boolean variable is true, or the loop is exited.

The area the drone will survey will be hardcoded into the system. This scene will be composed of action figures in various positions with various objects. Our drone system will fly around the area until an action figure is detected. If the figure is detected and the

angle is optimal, the system will perform pose estimation. Otherwise, the drone will reposition itself to get a more optimal picture. If a certain pose is detected, the edge system (laptop) will display an indicator on the screen.



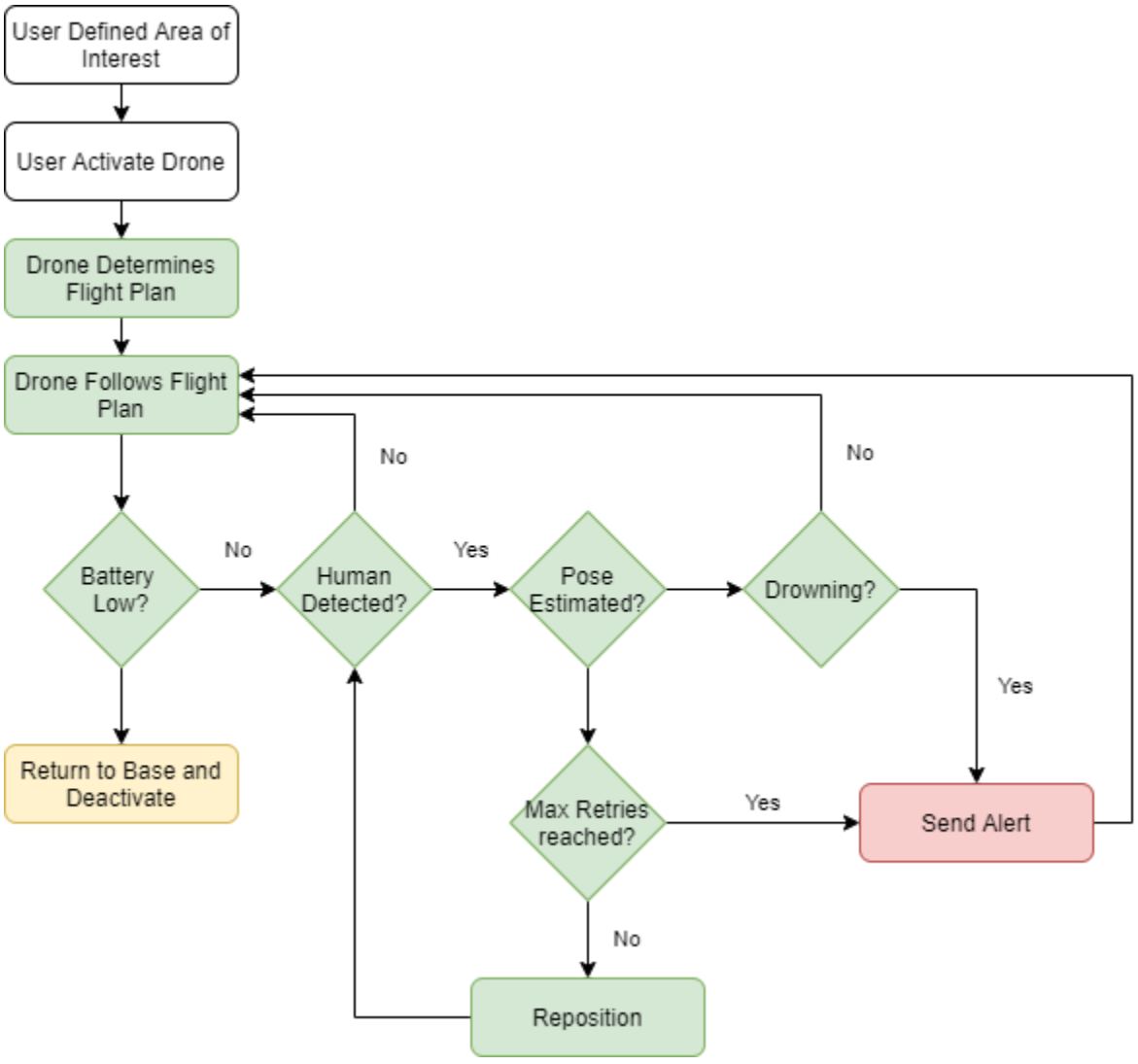
In the world of drone mapping software, there are two particular techniques that are most commonly used. **Drone grid paths** follow a grid pattern. They are best used for mapping missions designed to collect imagery for processing into 2D and 3D data products. **Drone waypoint paths** follow an irregular pattern based on the unique characteristics of the flight area/space of interest, and are best used for linear missions designed for inspections, project progress tracking, surveillance and security, etc. Our drone will implement a grid path flight plan, until it recognizes an irregular object in the ground, and initializes a waypoint path to investigate it.



1.4 Example

A quick Google search will show that drowning is the 3rd leading cause of unintentional death in the world with an estimated 236,000 incidents annually. As such, surveillance for drowning is of the utmost importance, especially at beaches, pools, and other popular bodies of water. Sometimes, these areas become too chaotic for lifeguards to effectively patrol.

At the beginning of the day, a lifeguard will define the drone's search area and deploy the drone. The drone will create a flight plan for surveying the area. As the drone flies overhead, it will search for humans. When a human is found, it will determine the pose of the human to see if they are drowning. If a pose cannot be determined, the drone will reposition itself to get a better angle of the human until a state can be determined. If the drone determines the human is drowning, it will signal the lifeguard (possibly via a loud sound and/or transmission to the lifeguard's radio) of the event. If a pose cannot be determined, it will call attention in a less alarming manner. After sending the signal, the drone will continue on its flight plan.



The overall idea can be extended to other surveillance tasks such as active shooter detection or theft detection. This project is, in part, inspired by Arjun's previous research in Active Classification for behaviors via Deep Q Learning.

3.1 Team Members and Roles

Team Member	Background	Contribution
Edgar Dasilva	B.S. Electronic Engineer & MBA 2 years of Python experience	Robot Operating System (ROS) architecture. Gazebo simulation.
Jorge Navarro	Current Occupation: High School Student 1 year python and c experience Worked in robotics as part of the First Lego League, worked mainly in the mechanical aspect and architectural design.	Autonomous Drone repositioning
Arjun Singh	B.S. Electrical & Computer Engineering Current Occupation: Data Engineer within Business Analytics System 6 years Python Experience Some experience with DNN, Tensorflow/Keras, OpenCV	Computer Vision and system integration Research and Develop tools for CV. Help integrate CV pipeline with the overall system. Expose CV capabilities for use in aerial mauvering, target acquisition, and pose estimation.

4 Software and Developing Tools

4.1 Software

- Python
- TensorFlow/Keras - Neural Network for computer vision
- OpenCV - image preprocessing
- Docker - run ROS docker image (if needed)
- ROS - orchestrate nodes for image pipeline and system control

- DJI API - Image streaming and movement command streaming. The images streamed from the DJI API will feed the computer vision pipeline.

4.2 Laptop/Desktop Setup

Ubuntu VM/native Ubuntu

Arjun Singh Laptop: Windows Surface Book with GTX 1080 M

Jorge Navarro Laptop: MacBook Air

Edgar Dasilva Laptop: Windows Surface Pro

4.3 Hardware

Ryze Tello Drone

4.4 Simulator

Gazebo

5 List of Milestones

Week 1: Project Exploration and acquiring hardware by team members

Week 2: Proof of Concepts

Week 3: Programmatically Control Drone, ROS first pass, Aerial Vehicle flight pattern research

Week 4: Train DNNs/Haar Cascades/Other CV tools, repositioning first pass, refine ROS and role into flight planning

Week 5: Refine CV tools, End to end integration, sophisticated repositioning

Week 6: Testing, data collection, refinement

Week 7: More fine tuning, Presentation Prep

Results and Discussion

- Multiple poses: we were able to detect multiple poses with the drone
- Object detection to identify action figure's equipment: we were able to detect objects such as guns
- Temporal pose:
- Signaling system (gui) for occurrence of specific poses: We were able to signal the GUI of the specific poses
- Sophisticated spatial awareness: Needs improvement
- Introduction of static obstacles
- Repositioning to counter collision: Still needs to work to do
- Dark conditions: camera resolution affects the visibility and the identification of objects.

Drone surveying the area

Drone identifying behaviors

Conclusions

To conclude, throughout the duration of the course, and the past month, we have created a product that proves how real time object detection via drone for the purposes of surveillance is possible. By combining a capable drone, computer vision, machine learning, and a software that binds them all together, drone surveillance technology proves to be a potential future breakthrough in world health and security. The next major step is to apply this technology in real world spaces...

Future Work

- Testing on real humans
- Accurate position tracking for better response system (GPS)
    - Improvements of the hardware of the drone could greatly increase its capacity, efficiency and performance. Integration of technology such as GPS tracking. This would help the drone be more aware about its position, make it able to learn about its environment and how to navigate it more accurately.
- More advanced drone with controllable camera
    - A more advanced drone, equipped with a higher camera quality, would likewise improve its performance. These would make it able to traverse its area more efficiently, and be able to sense its surroundings with more clarity.
- Improved Object Detection and Pose Estimation
    - Improved object detection and pose estimation could be added to machine learning in order to make the drone be able to recognize more potential threats. Behaviors such as thefts, fights, or mobs could make the drone be more capable of analyzing its surroundings and the dangers posed within them.
- Repositioning to combat object occlusion
    - Drone repositioning would make the drone be more competent in handling object occlusion. If the objects it were sensing somehow had obstacles within its line of sight, the drone would need to reposition itself in order to get a better perspective of the object. This would help it become more fool proof, and be ready for anything in its way.

References

Drone: Tello (ryzerobotics.com)
Drone Python SDK: Tello SDK 2.0 User Guide.pdf (ryzerobotics.com)

Drone Connection: dji sdk - Unable to programmatically connect to Tello drone via code - Stack Overflow
Drone Programming: dji-sdk/Tello-Python: This is a collection of python modules that interact with the Ryze Tello drone. (github.com)
Drone Example Repo: raymondlo84/nvidia-jetson-ai-monitor: An example development repository for using Nvidia Jetson Nano or Xavier as health monitor using computer vision. (github.com)

OpenPose: CMU-Perceptual-Computing-Lab/openpose: OpenPose: Real-time multi-person keypoint detection library for body, face, hands, and foot estimation (github.com)
OpenPose ROS: openpose/0_index.md at master · CMU-Perceptual-Computing-Lab/openpose (github.com)
OpenPose Example Repo: raymondlo84/nvidia-jetson-ai-monitor: An example development repository for using Nvidia Jetson Nano or Xavier as health monitor using computer vision. (github.com)
Another Pose Estimation Example: NVIDIA-AI-IOT/trt_pose: Real-time pose estimation accelerated with NVIDIA TensorRT (github.com)

ROS-Tello Driver: appie-17/tello_driver: ROS driver for DJI/Ryze Tello drones (github.com)
OpenCV Object Tracking Drone: Easy Programming of Tello Drone | Python OpenCV Object Tracking - YouTube
AI Tracking Drone: I create AI tracking drone using DJI Tello - YouTube
Source: Jetson Community Projects | NVIDIA Developer

The first set of links describe the drone and contain information about controlling the drone. The drone communicates with an edge device by streaming video to the edge and receiving movement commands from the edge. The second set of links contain information about pose estimation. In the feedback loop between the drone and the edge device, the edge will use pose estimation on the video to try to determine if a human is present and the pose of that human. Finally, the last set of links describe integration with ROS and projects with the drone. In sum, the information here can be assembled to complete this project's objective.