Project Report

Face Tracking Drone

Neha Dubey

1. Abstract

This project aims at creating a drone that would be able to identify a human face and then track it as the face moves. The drone would be able to track the face in four possible directions; front, back, right and left. It won't be able to move up and down. This project has been created as a stepping stone towards creating an aerial photography drone. Due to the time restrictions in the current session I have decided to just work on a face tracking drone and then extend on the idea later to achieve a fully functional aerial photographer drone. The motivation for the project came from situations when on vacation we don't have anyone to click our pictures and we might miss some good spots to get clicked at.

Taking this project as an opportunity I was able to implement the face tracking drone using opencv and haar cascade. The architecture used was ROS Foxy which makes it quite flexible and scalable in future. I have also defined future scope to be worked on later. As a result of the project I was able to recognize how powerful and simple opencv and haar cascade is. I also realized how efficient ROS architecture is that helps us design robot architecture.

2. Introduction

This project is a drone that is capable of detecting a human face in front of it and once detected it will follow the face towards left, right, forward and backwards directions. In the absence of a face it would remain still (suspended) in the air waiting for a human face to appear for a couple seconds else it will land.

2.1. Goal of the project

Goal of the project is to be able to identify a human face out of all exposed objects to the drone. Once identified, the drone should be able to detect movement of the face and follow it accordingly. If multiple faces are detected by the drone then the face closest to the drone would be the one followed by it.
This project is a stepping stone that could be extended to be an aerial photography drone. Due to the limited time in this class I plan on achieving the first stage of a face tracking drone. Aerial photography is out-of-scope for the purpose of this project.

2.2. Level of autonomy

I am planning to develop a level 4 autonomous drone which would need no external support or directions once it starts the flight. There would be checks regarding drone battery and any time it reaches a predetermined threshold level, the drone would land safely (built in functionality in drone). As a safety feature I will let the human operator interrupt in case of emergency and hit a key ‘q’ (quit) from the keyboard for emergency landing if required at all. Because of the presence of this human override it would be a level 4 autonomy system.

2.3. Description

This project aimed at creating a drone that is able to detect a human face amongst various objects that the drone has been exposed to. If more than one human face is exposed then the drone will identify the closest face by comparing the area of the face. This closest face is the one that would be followed by the drone. The image capture and processing is done by using openCV. I have built project architecture using ROS2 Foxy.

As the scope has been defined for the project, the drone is controlled by laptop, which implies the controlling program (ROS nodes) runs on a laptop. As soon as all the controlling nodes start and the drone is powered on, it would take off and would try to detect a human face in front of it. I implemented Haar Classifier which is a machine learning object detection program that identifies objects in images and videos. Specifically for the purpose of the project I used ‘haarcascade_frontalface_default.xml’ to identify frontal human faces.

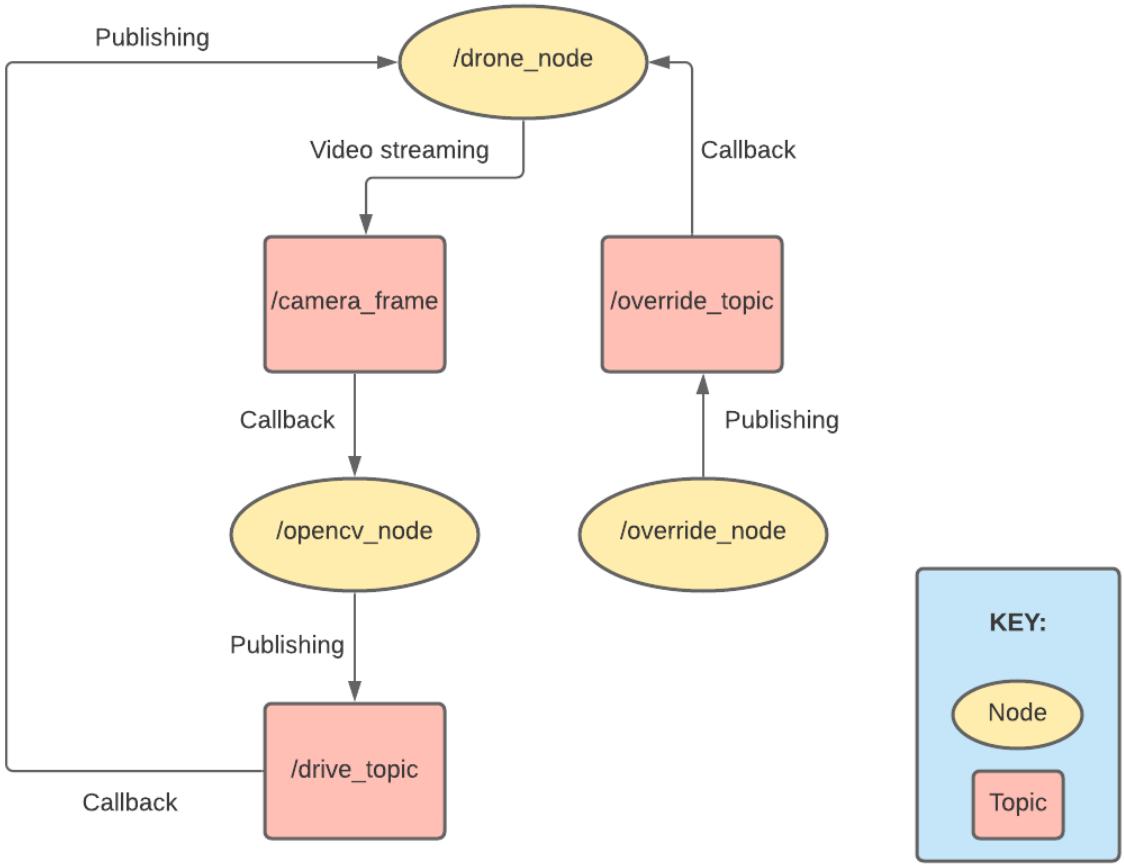
Below is an architectural diagram explaining the structure of ROS architecture.

There would be four nodes and two topics as given below. Nodes and topics would function as:

1. **drone_node:** This is the node that connects to the drone directly and accepts video frames which are then published to the camera_frame topic. This node also subscribes to drive_topic and then sends driving instructions to drone as per video processing done by opencv_node. This node also subscribes to override_topic which is supposed to send emergency landing directions to drone if desired via keyboard interrupts.
2. **camera_frame:** This is a topic that accepts video frames as published by drone.
3. **opencv_node:** This is the node subscribing to camera_frame topic and accepts video frames as relayed by drone. This is going to be a compute node that would try and detect a human face in the video frame and once detected would track the face by identifying any movement in the face and hence publishing driving directions for the drone to the drive_topic accordingly. This node would also put a

maximum speed limit so that the drone does not go out of control and cause potential damage to the environment.

4. **drive_topic:** This is going to be a topic accepting driving directions from opencv_node.
5. **override_node:** This is a node that would keep listening to keyboard interruptions. If needed for a human override to land the drone keyboard press for key ‘q’ could be used. When hitting the key ‘q’ (quit) this node would publish a message on override_topic indicating drone node to land the drone.
6. **override_topic:** This is the topic that accepts messages from override_node and these messages would be subscribed by drone_node to take desired actions.



PID control was used to control the motion of the drone while tracking/following a face. The PID controller calculation (algorithm) involves three separate parameters; the proportional, the integral and derivative values. The proportional value determines the reaction to the current error, the integral value determines the reaction based on the sum of recent errors, and the derivative value determines the reaction based on the rate at which the error has been changing. The weighted sum of these three actions is used to adjust the process via a control

element such as the position of a control valve or the power supply of a heating element. In the project I have only used Proportional and Derivative of PID control to determine drone motion.

3. Related Work

Area	References
Intro to drones and drone programming	https://www.youtube.com/watch?v=LmEcyQnfpDA
OpenCV in ROS	https://automaticaddison.com/getting-started-with-opencv-in-ros-2-foxy-fitzroy-python/

4. Team Organization

I did this as an individual project.

5. Software and Developing Tools

5.1. Software

- Ubuntu 20.04
- Python virtual environments
- ROS 2 Foxy
- Python 3
- djitellopy
- openCV
- Numpy

5.2. Laptop/Desktop setup

- I am using a Lenovo ThinkPad with below configurations:
- Intel core i7, 2.80 GHz
- 16G RAM
- 512GB

5.3. Hardware needed

Ryze DJI Tello Drone

5.4. Simulator needed

Did not use any simulation

6. List of Milestones

Week/Date	Milestone	Deliverables
1 (6/22 - 6/27)	• Brainstorm project ideas	• Project idea
2 (6/28 -7/4)	• Acquire hardware • Select softwares to be used	• Hardware acquired • Software selected and installed
3 (7/5 - 7/11)	• Learn ROS architecture • ROS PoC for publishing and subscribing to video streams via laptop's in-built camera	• Working ROS PoC for pub sub model transferring video stream
4 (7/12 - 7/18)	• Explore programmatically controlling drone • openCV PoC for detecting human face and tracking it	• Drone could be controlled by laptop • Working openCV PoC detecting and tracking human face
5 (7/19 - 7/25)	• Integrate openCV PoC with ROS architecture	• ROS nodes tracking human face on drone HW
6 (7/26 - 8/1)	• Test drone for positive and negative scenarios • Refine algorithm based on testing	• Program tested on drone • Any feedback implemented
7 (8/2 - 8/5)	• More testing • Prepare project report • Prepare presentation	• Testing completed • Project report • Project presentation

7. Results and Discussions

I was able to get the drone to detect the human face. I did try with two faces in front of it and the drone identified my face correctly as that was the one closest and then followed it.

8. Issues Faced

I did face a lot of challenges while working on the project. Few of them are as stated below:

- This was my first exposure to ROS and it's architecture designing was one of the biggest hurdles

- Tuning PID parameter for drone turned out to be quite tricky and time consuming
- Faced issues with the same node acting as a subscriber and publisher. Had a huge lag with drone rendering camera frames to laptop. Finally figured out multi-threaded nodes to resolve the issue. Writing a multi-threaded program was my first ever attempt.
- Drone was not stable at a specific altitude
- Override_node didn't work within launch file

9. Conclusions

- Opencv is very powerful for image processing
- Haar cascade is very efficient and does identify human faces with accuracy
- Simple techniques of image processing could be done to build interesting and helpful projects (this project is a stepping stone for aerial photography)
- Multi-threaded nodes were integral part

10. Future Work

Due to the time limitation of the class I did not extend the project and have built just the basic features. I plan on adding more features like:

- PID parameters further tuning
- Examine issue with override_node inside launch file
- Extending this as an aerial photography drone in the absence of anyone else to click picture
- Controlling drone with gesture – this would let the future aerial drone to click picture when people are ready
- Use of Neural Networks to have drone identify just one person's face and be his/her personal aerial photographer

11. GitHub Code

Below is the url for project code on GitHub:
https://github.com/nehadubey2311/openCVProject/tree/master/finalProject