Hawkeye

ECE Capstone 2021, Group B9
Alvin Shek, Vedant Parekh, Siddesh Nageswaran

Advisors: Marios Savvides, Rashmi Anil

Introduction and Motivation

- What is Hawkeye?
  a. An automatic drone tracking system with live aerial footage
  b. Shoots aerial video that does not require human control, eliminating human error

- Use Cases:
  a. Useful for recreational filming, rescue missions etc.
  b. Imagine shooting exciting videos of sports events (your Turkey Bowl game), or having a hands-free vlogging experience!

Basic Components

- Our solution involves:
  a. A drone connected to a camera / microcontroller that is able to track a user wearing a brightly colored shirt and capture video of them
  b. A wearable wrist display that streams the live video captured from the drone
  c. A compute board located on the user that performs most of the tracking computation and interfaces with the microcontroller on the drone and the wrist display

- Areas of ECE included in our project are:
  a. Software systems (computer vision, flight planning, communication protocols)
  b. Hardware systems (wearable device, button controls)
  c. Signals (computer vision)

Project Scope

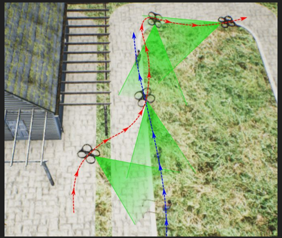
- We will operate under the assumptions:
  - The drone will operate in open field without obstacles
  - There will be WiFi access in test environment
  - There will be little to no wind
  - The drone will operate during daytime
  - The drone will be limited to tracking one person
  - The target being tracked may be amongst 3 other people, but not more

Technical Challenges

Target Tracking

- Detect target person in image with bounding box
- Estimate person’s motion

Motion Planning



- Plan future trajectory of waypoints to follow drone under acceleration limits

Safety

- Well-defined protocols for handling unexpected events on drone and on user display:
  - Large obstacles (ex: trees)
  - Low battery
  - Loss of visual tracking
  - Inability to keep up with target in speed

Communication

- Stream video compression
- Wifi bandwidth limits
- Sending video to display using hardware protocol (ex. I2C)

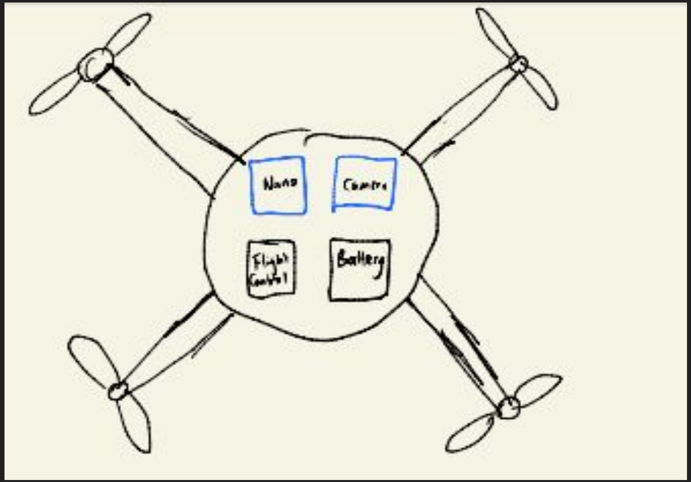
Requirements

Requirement	Intuitive Explanation	Quantitative Specification
Drone Stability	Captured video should not be very jittery	Standard deviation of target's distance to center of frame should not exceed ⅙ the width of the frame
Tracking Accuracy	Video should be centered to the target	Center of target should be located within a centered bounding box ⅓ the size of the frame
Quality of Streamed Video	Requirements for video on wrist display	Unsure, dependent on bandwidth limitations
Quality of Captured Video	Requirements for video saved to SD card	At least 720p, 30fps

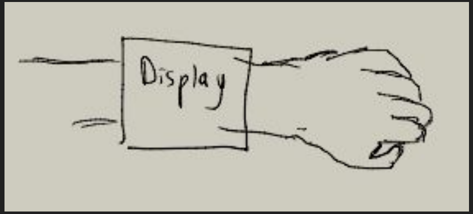
Requirements

Requirement	Intuitive Explanation	Quantitative Specification
Run Time	How long the drone runs on a single battery charge	5 - 10 minutes
Power Consumption		Unsure
Weight of Compute	The weight of the wrist display and the compute board that is on the user	2kg. at most
Size of SD Card		16MB

Solution Design- Hardware



- Iris 3D+ drone
- Camera for recording
- Computer on board
    - Raspberry Pi Zero
- Computer on the user for more computationally intensive tasks
    - Jetson nano
- Wearable display for showing live video
- Buttons to control start/stop of flight

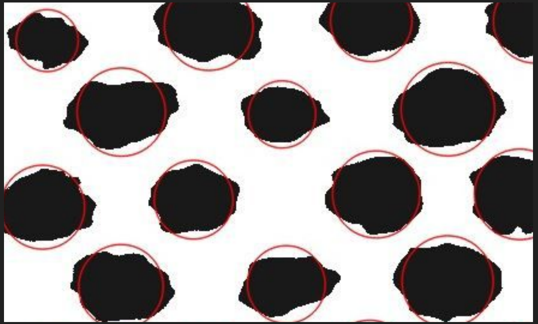


Solution Design- Software

- Target Tracking (done on Jetson Nano)
  - Detect target with CV: RGB color filter + blob detection
  - Estimate/Predict state of target in future

- Drone flight control (done on Jetson Nano)
  - Need to compute desired yaw and x, y and z positions relative to current drone position

- Communication between board and drone (handled on Rpi Zero)
  - Need to communicate the yaw and coordinates to the drone's flight controller and send the streamed video to the Jetson Nano



Testing, Verification & Metrics

- Most reqs. (flight time, weight, video quality) are self-explanatory to test

- The more complicated tests are for:
  - **Drone Stability**
    - Calculate the standard deviation in distance from center of target to center of frame across 30 frame windows
    - Calculate % of erroneous windows relative to total windows in video
  - **Tracking**
    - Go through the captured video frame by frame, find the center of the target and calculate whether it lies within the desired bounding box
    - Calculate % of erroneous frames relative to total frames in video

Division of Labor

- Alvin:
  - Drone motion planning
  - Flight controller interface

- Vedant:
  - Target identification
  - Hardware communication protocols
  - Design circuitry for wearable

- Siddesh:
  - Target state prediction / estimation
  - Designing the housing for the display / drone

Schedule

		February	March					April				May
		Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13	Week 14
1	**Phase 1: Design**											
1.1	Make overall block diagram											
1.2	Choose/finalize components											
1.3	Order components											
1.4	Familiarize ourselves with drone API and simulator											
1.5	Familiarize ourselves with Jetson Nano API											
1.6	Familiarize ourselves with wearable display communication protocol											
1.7	Design Presentation											
1.8	Design Report											
2	**Phase 2: Pre-integration**											
2.1	Implement color filtering and blob detection											
2.2	Test detection on static images											
2.3	From target data across multiple frames, calculate velocity vector											
2.4	Use calculated velocity along with a model for target movement to accurately predict future movements											
2.5	Test detection on live video (long distance with smaller target since blob detection parameters need to be tuned)											
2.6	Successfully send sample motion commands to drone in simulation, then on physical drone											
2.7	Design a general motion planning stack; design to easily integrate with target tracking											
2.8	Test and debug the motion planning stack											
2.9	Implement communication between camera and RPi Zero, film sample video											
2.10	Implement video streaming over Wifi between RPi Zero and nano											
2.11	Implement hardware protocol (ex. UART, I2C) to interface between nano and display											
3	**Phase 3: Integration**											
3.1	Map target's motion in video to desired motion of drone											
3.2	Design circuitry for the wearable device (ex. display, buttons, power supply etc.)											
3.3	Hook up buttons to the Nano and configure them to send start and stop instruction to RPi											
3.4	Integrate RPi start stop signals received from nano with the drone flight controller											
3.5	Design safety fallback behavior and validate on simulation and on drone											
3.6	Integrate the flight parameter data generated by Nano with the flight controller											
3.7	Create housing for Jetson nano and display so that they can easily be worn by the user											
3.8	Create chassis to house RPi and camera onto the drone and verify camera angle											
4	**Phase 4: Performance testing**											
4.1	Verify that video streaming performs with a stationary target											
4.2	Verify drone can continuously track target during flight w/o streaming video II											
4.3	Verify that drone can match speed of target that is varying between speedsII											
4.4	Verify live video from drone can be seen on wearable display for a moving targetII											
4.5	Iterate and repeat tests II											
4.6	Slack											
5	**Final Report**											
5.1	Record demo video II											
5.2	Edit video II											
5.3	Final presentation II											

Legend:
- Vedant
- Alvin
- Siddesh
- Together