Week 1 (9/9): Full Stack Overview

1. What software is on our current drone? (Summary of the next 5 weeks)

- Object Detection (Week 3)
    - historically done w/ OpenCV
    - we switched to YOLO (manually supervised learning), but considering switching back for competition
- Aerial Imagery/Stitching (Week 2)
    - we've tried OpenCV, but better results w/ OpenDromeMap so far
- Flight controller (Week 4)
    - runs Ardupilot
    - keeps drone stable and flying
    - has little autonomy
    - takes commands via MavLink over WiFi from onboard computer
- On-board computer (Week 5)
    - NVIDIA Jetson Orin Nano running Ubuntu
    - "brain" of system, tells FC what to do
    - handles large streams of data from camera

2. What kinds of challenges/projects can YOU work on as a Software team member?

- improving Flight controller
    - Reinforcement learning research
- CV
    - design a better object detection system
- Applications only, no coding
    - 3d mapping research

3. What's the "industry standard" for drone software in 2025?

Hobbyist Software vs Commercial Drones

Feature	Hobbyist Stack (good starting point)	Commercial Stack (what we're working towards)
Flight software	Ardupilot/Betaflight	ROS 2-based, highly modular. PX4 for FC

Real-Time Autonomy	Basic (e.g., YOLO)	Advanced SLAM, path planning, AI inference
Compliance & Safety	RFID device + B4UFLY	Automated risk assessments, airspace alerts
Data Integration	Local storage	Cloud stuff

Pushes in BVLOS and multiple drones
BVLOS (Beyond Visual Line of Sight), multiple drones controlled by one operator (swarming)

4. How does one learn about drones effectively in 2025?

- AI
    - good for conceptual questions, helping with code
    - not great for things beyond human knowledge
- experience

1. Talk back and forth with AI to make sure you understand it conceptually
2. Set goals, do projects
3. Track progress
4. For individual details, a combination of AI and fact-checking on individual-sites/personal experience is best, AI may get specifics wrong
5. Reach out

Week 2 (9/16): Computer Vision/Aerial Imagery

1. How are drones used for aerial imagery purposes?

- search and rescue, agriculture, law enforcement, environmentalism
- competitions
- types of aerial imagery:
    - Oblique vs. Nadir
    - RGB-D /multi-spectral/ plain RGB

2. How is OpenCV used for computer vision purposes? How do we use it as a club?

- includes feature detection, 3d reconstruction, motion analysis, optical flow, etc
- Parts of OpenCV that we've used:
    - Feature detection (SIFT, ORB)
    - Feature matching
    - image manipulation

3. What are some pre-made alternatives to writing our own aerial imagery code, and how do they work? What are the drawbacks?

- OpenDroneMap
    - Benefits: simpler to use, web app
    - Drawbacks: not optimized for real-time flight, requires accurate GPS tags, runs poorly w/out GPS

4. What's the next steps forward as a club?

- projects:
    - custom **real-time** image stitching/SFM (structure from motion) algorithm with OpenCV
    - real-time stitching w/ OpenDroneMap
    - SLAM (Simultaneous Localization and Mapping)
    - Optical Flow
    - applying drone imagery (research)

Week 3 (9/23): Object Detection

1. What are some methods used for blob detection?

LoG - Laplacian of Gaussian

2 steps

1. Gaussian Blur (G)
    - Preprocessing step
    - reduces noise for Laplacian
    - stdevs increase blur
2. Laplacian Operator
    - Look at 2nd order derivatives
    - edge detection
    - Edges occur at zero crossings

- YOLO Object Detection

2. How can blob detection be used to find objects in a field? What are its benefits and limitations?

Deep Neural Networks

Adds hidden layers - non linear activation functions
Now can infer on more complex/nonlinear relationships

Proven: A large enough deep network can model any function

Common Activation Functions

Sigmoid, Hyperbolic Tangent, Rectified Linear Unit (ReLU)

Learning Weights

We have a loss function which we want to minimize using gradient

Improving a NN

1. Add more layers/inputs/neurons
2. Train with more data
3. Innovate
    - Add convolutions
    - add pooling
    - what makes YOLO specialized for object detection

Convolutions and Pooling

Convolutions - learned kernels that can extract patterns(edges, curves, etc)
(max) pooling - reduces dimensionality, lessens overfitting, noise reduction

3. What are the underlying mechanics of the convolutional neural network used in the YOLO model?

YOLO is a one-shot CNN
Two-shot vs one-shot
Two-shot:

- first pass is to propose object locations, the next for prediction
- slower but more accurate

One-shot:

- only one pass of input image to make prediction
- faster but less accurate

- Train/Validation/Test
    - training data - data that changes weights
    - validation data- use to tune hyperparameters (learning rate, when to stop)
    - test data - don't use until end, evaluates how good the model is
    - Generally about 70/20/10 split is good
- Overfitting
    - too much training is dangerous

4. How can our club leverage our manpower to label many images in a short amount of time on roboflow? (Teach them how to use roboflow so that when we need to label a bunch of images quickly, we can use the built in roboflow tools to mobilize quickly)

5. What are some upsides and downsides to using YOLO for our competition vehicle?

- less accurate but faster

6. What other techniques might there be for object detection?

Week 4 (9/30): Flight Controllers

1. What is the role of a FC in a drone system, and how does it differ from an onboard computer (like a Jetson)?

Flight Controller (FC)

- maintains drone stability using sensor data (IMU, GPS, barometer, etc.)
- processes radio inputs and executes control loops
- communicates via protocols such as MAVLink, often over UART (Serial) or CAN

Onboard Computer (e.g. Jetson)

- handles high-level tasks: CV, path planning, autonomy
- Runs ROS nodes, SLAM, object detection, mission logic
- send commands to FC and receives telemetry for decision-making
- connects via serial, Ethernet, or USB depending on setup

2. What sensors are typically connected to a flight controller?

Basic Sensors: IMUs (Internal Measurement Unit) and Barometers
IMU

- combines accelerometers and gyroscopes to measure orientation, angular velocity, linear acceleration
- essential for flight stabilization and altitude control

Barometer

- Measures atmospheric pressure to estimate altitude
- used for altitude hold and smooth vertical control

Magnetometers (Compass)

- detects Earth's magnetic field to determine heading
- helps correct yaw drift and supports navigation in GPS-denied environments

GPS module(s)

- provides global position, velocity, time data
- Enables waypoint navigation, return-to-home, geofencing (keeping drone from flying too far away)

Drone Cameras

- we are using standard RGB cameras
- can send optical data to FC and onboard computer
- capture full-color images and video
- used for mapping, inspection, aerial photography
- often mounted on a gimbal for stabilization

Key Specs to Consider

- Sensor size (larger=better low-light performance)
- resolution (megapixels or video-format)
- Field of view (FOV), frame rate, dynamic range
- Interface type (HDMI, USB, Ethernet, UART)
- digital vs optical zoom

RC Transmitters & Receivers

RC Transmitter (TX)

- handheld device used by pilot to send commands

RC Receiver (RX)

- mounted on drone, receives signals from transmitter
- passes commands to flight controller via protocols like SBUS (Serial Bus)

3. What are key Communication Protocols between flight controllers and peripherals

Serial (UART)

- direct, basic communication channel between 2 devices
- used for communication between flight controller and onboard computer
- simple 2-wire setup: TX (transmit) TX and RX (receive)

CAN (Controller Area Network)

- sort of like a device group chat
- used in cars and high-end drones

I2C/SPI

I2C (Inter-Integrated Circuit)
- used for low-speed communication with devices like barometers, magnetometers, and IMUs
- shares two wires: SDA (data) and SCL (clock)

SPI (Serial Peripheral Interface)

- similar to UART, but w/ multiple devices listening to one "master" device

MAVLink (Micro Air Vehicle Link)

- a message format, not physical protocol that runs over serial, UDP, or CAN
- send commands and receive data

DDS (Data Distribution Service)

- good for real-time performance in contrast to MavLink

4. How does firmware (e.g. Ardupilot) influence the capabilities of the flight controller?

- low level software which runs on the flight computer
- ex: Ardupilot, Betaflight, PIX4
- can be configured through settings and has different “flight modes” e.g. auto-stabilize, full stick control (acro), etc

PID Control (Proportional-Integral-Derivative)

- used to keep drone pitch stable

Options for FC Firmware

Firmware	Typical Use Case	Strengths	Limitations
ArduPilot	Autonomous missions, mapping, R&D drones	Rich feature set (GPS, geofencing, RTL), supports fixed-wing, VTOL, rovers, boats	Heavier codebase, slower tuning workflow
Betafli	FPV racing,	Ultra-responsive manual control,	Limited autonomy, no

ght	freestyle acro drones	fast PID loops, optimized for agility	GPS-based missions
PX4	Research platforms, modular drone dev, industry standard	Clean architecture, SITL/HITL support, ROS2 integration, flexible hardware configs	Slightly steeper learning curve for beginners

Week 5 (10/7): On-board computers and telecommunication protocols

1. How do we communicate with the drone in the air?

- radio and ground station

Internet Protocols:

- UDP: no reliability, packet loss, fast and lightweight (one-way)
- TCP: checking for a 'session' on both sides, reliable (two-way)

2. What is the role of the onboard computer system and how does it complement the flight controller?

FC

- acts like automatic nervous system (drone stability)

Onboard Computer

- acts like somatic nervous system (high-level tasks: computer vision, autonomy)

Jetson Orin Nano

- lots of connectors
    - USB-C for data transfer
    - Ethernet
    - 40 extra pins (GPIO, I2C, SPI, UART)
    - SD card, wifi
- good for running complex tasks
    - built-in AI accelerator for AI inference

Applications

- linux, ROS
- autonomous robotics, cv
    - runs ROS2 nodes for SLAM, path planning, control
    - real-time object tracking

3. Trade-offs between different data links (WiFi, radio, LTE)?

WiFi
pros:
- high bandwidth (good for streaming, telemetry)
cons:
- short-range
- interference at commonly used bands
- line of sight required
Radio
pros:
- long-range
- reliable
cons:
- lower bandwidth
- not built in to most computers
- regulations on frequencies
LTE (Cellular)
pros:
- high bandwidth, wide coverage
cons:

- higher latency than WiFi
- requires SIM card/data plan
- dead zones

4. How does the OC connect to the ground station or other devices over WiFI? (SSH)

Week 6 (10/14): Autonomy and Advanced Topics (ROS, Machine Learning, Research)

1. What are the benefits of simulating a drone with software/hardware-in-the-loop before flying?

- cost savings
- software-in-the-loop (running simulation on pc)
    - can model real-life scenarios in physics simulators (e.g. Gazebo)
- Hardware-in-the-loop (running simulation on drone hardware)

Examples of Simulators
Commonly Used:

- Gazebo: works great with PX4/ROS

- Isaac Sim by NVIDIA

Less common:

- ArduPilot SITL
- AirSim (Microsoft): built on Unreal Engine
- Flightmare: GPU accelerated sim

2. How is machine learning used to improve drone autonomy (navigation, obstacle avoidance, etc.)? What benefits does it have over traditional guidance systems like PID loops?

- Types of ML:
    - Supervised: Learn from labeled flight data (e.g. object detection)
    - Unsupervised: Discover patterns in sensor data (e.g. terrain clustering)
    - Reinforcement Learning: Learn by trail and error in simulation (e.g. agile maneuvers)
- Applications:
    - navigation/path planning in cluttered environments
    - real-time obstacle avoidance using vision models

Challenge: Sim2Real transfer

- Zeroshot: can we transfer to reality and have it work first try? (much harder than it looks)
- Gap between reality: Simulators simplify physics, sensor noise
- simulated sensors may behave differently than real sensors
- can the policy run on the processing power present onboard?

3. What is ROS, and why is it universal in autonomous robotics?

- not an operating system, but middleware that helps different parts of a robot communicate

Features:

- Modularity: node-based system for reusable code
- Communication: topics, services, actions to pass messages between nodes
- Ecosystem: lots of ROS packages for existing sensors
- Simulation support: works w/ Gazebo/Isaac Sim
- ROS Vocab:
    - node: a process that performs a specific function (e.g. camera driver, path planner, etc.)
    - Topic: A named place where nodes can **publish** messages or **subscribe** to listen for messages
    - Messages: Data structure sent between nodes (e.g. velocity, commands, sensor readings, etc.)

    - Service: A request/response communication (i.e. function call) directly between nodes
    - Action: Service that supports long-running tasks (e.g. fly to waypoint)
    - Package: Collections of ROS code, nodes, config files
    - Launch File: Script that initializes nodes

4. What are the steps/challenges of deploying ROS on embedded systems like the drone?

- compatibility