ARDUINO PROGRAMMING FOR DRONE SYSTEMS

Presented by: Nikoloz Bokolishvili

Invited Lecturer and Engineer

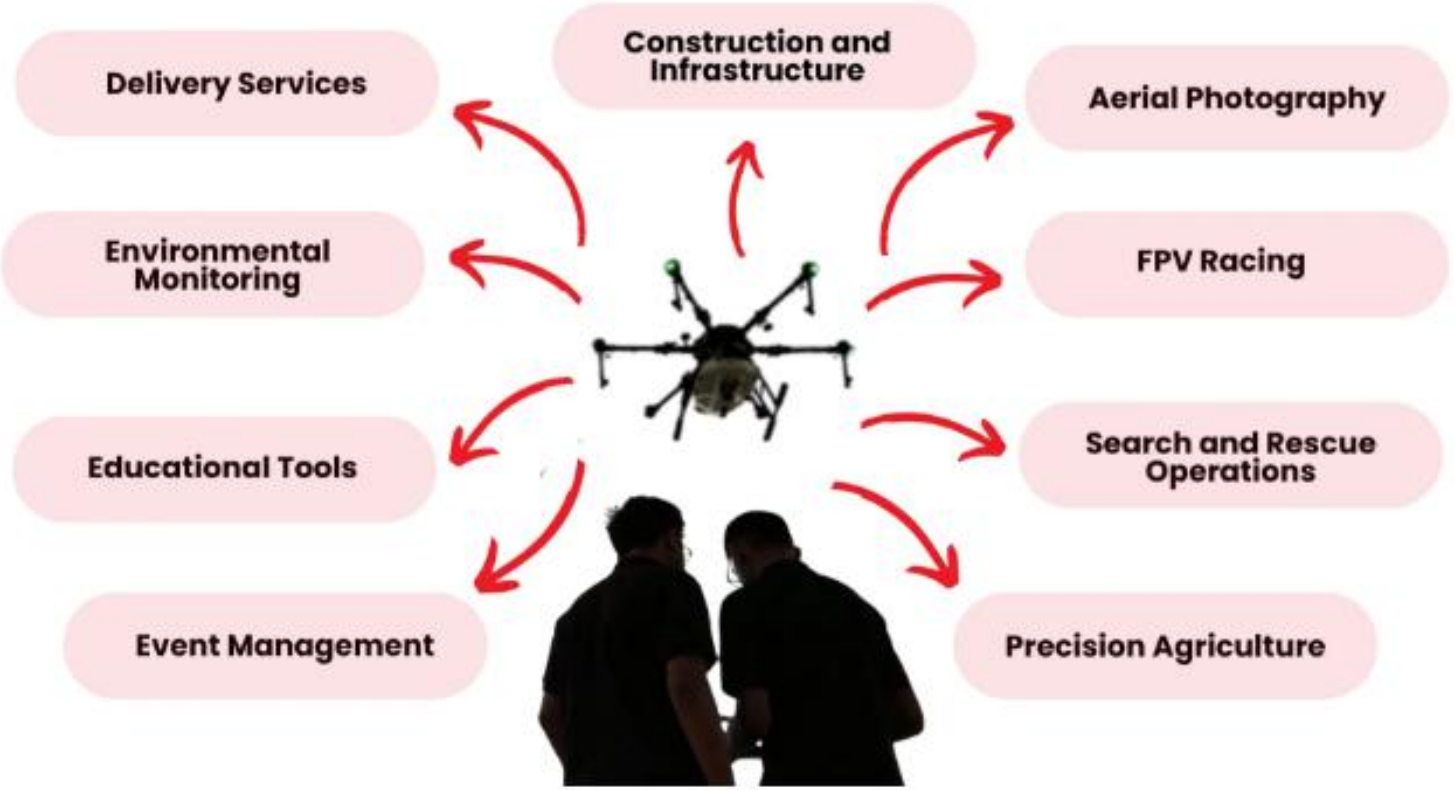
Department of Engineering

University of Georgia

TABLE OF CONTENTS

- Course Overview
- Course Objectives
- Learning Outcomes
- Modules
- Practical Component
- Mode of Delivery
- Learning Activities
- Assessment
- Duration & Credits
- Target Participants
- Prerequisites
- Rationale

INTRODUCTION



INTRODUCTION

- The global small-drones market was valued at **USD 9.03 billion in 2024**. It is expected to grow at roughly **10.36% CAGR** from 2025 to 2032, reaching nearly **USD 19.88 billion by 2032**.
- The broader commercial drone market was valued at **USD 13.86 billion in 2024**. Forecasts estimate it will grow to **USD 65.25 billion by 2032**.
- The overall UAV (drone) market (including OEM + aftermarket) is estimated to reach **USD 26.12 billion in 2025**, and is projected to grow to **USD 40.56 billion by 2030**.
- Projections over the next several years show a global drone market CAGR (Compound Annual Growth Rate) between ~9% and ~14%, depending on the segment (small drones, commercial drones, general UAV market) and report.

COURSE OVERVIEW

The *Arduino Programming for Drones* micro-credential provides learners with a complete progression from foundational electronics and coding skills to advanced drone automation and flight-control programming.

COURSE OBJECTIVES

- Introduce the fundamentals of Arduino programming and microcontroller operation.
- Teach integration of drone hardware components including ESCs, BLDC motors, IMU sensors, GPS modules, ultrasonic sensors, and wireless communication units.
- Develop skills in writing stable flight-control firmware including PID loops and sensor fusion.
- Enable analysis and interpretation of sensor data for drone navigation.
- Build competency in implementing safety protocols, failsafe logic, and real-time error handling.
- Introduce autonomous behaviors such as waypoint navigation, altitude hold, and environmental sensing.

LEARNING OUTCOMES

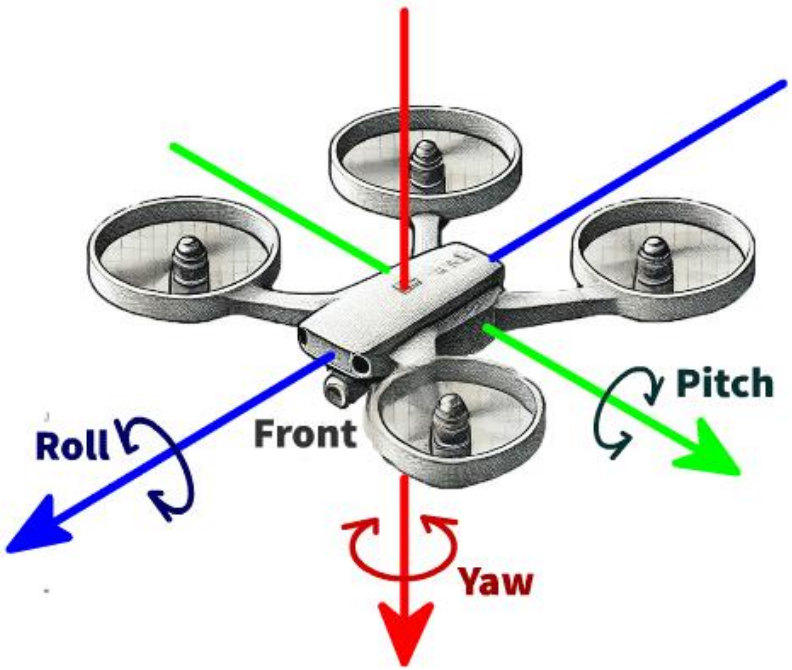
- Program Arduino for drones
- Integrate and calibrate sensors
- Apply PID flight control
- Read and analyze sensor data
- Build simple autonomous functions
- Follow safe UAV practices

MODULE 1 – INTRODUCTION TO DRONES & ARDUINO

- Drone systems & flight basics
- Arduino boards overview
- Intro to C/C++ coding
- ESCs, BLDC motors & power
- Safety protocols
- Hands-on: LEDs, PWM, motor control

DRONE SYSTEMS AND FLIGHT BASICS

- Types of drones (multirotors, quadcopters, hexacopters).
- Basic flight mechanics: lift, thrust, roll, pitch, yaw.
- How drones maintain stability in the air.
- Understanding center of gravity, frame design, and propeller orientation.
- Introduction to flight terminology used in drone engineering.



ARDUINO BOARDS OVERVIEW

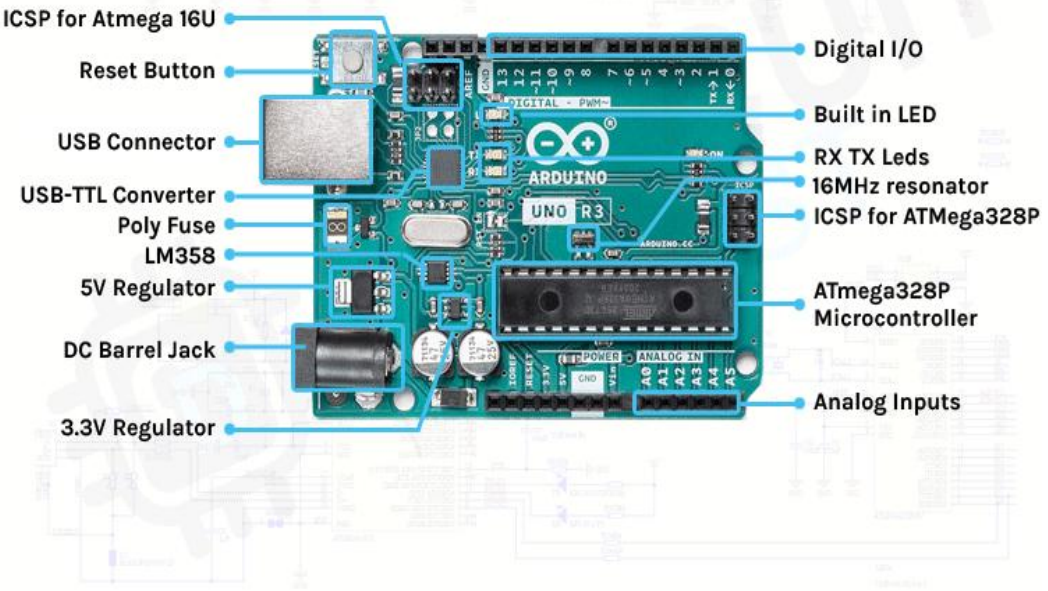
What Arduino is and why it’s used in prototyping.

Differences between Uno, Mega, Nano (memory, pins, capabilities).

How microcontrollers receive, process, and output signals.

Understanding pin types: digital, analog, PWM, power pins.

Overview of Arduino IDE and board setup.



INTRODUCTION TO C/C++ CODING

Basic Arduino C/C++ syntax and structure (setup() and loop()).

Variables, data types, and simple operations.

Control structures: if/else, loops (for, while).

Writing basic functions and understanding how code interacts with hardware.

Uploading code to the Arduino board.

ESCS, BLDC MOTORS & POWER

What ESCs (Electronic Speed Controllers) do.

How ESCs receive PWM signals from Arduino.

How BLDC motors work (brushless, 3-phase motor design).

Power distribution basics: battery voltage, current, connectors, safety.

Calibrating ESCs and understanding motor throttle ranges.

SAFETY PROTOCOLS

SAFE HANDLING OF LI-PO BATTERIES (CHARGING, STORAGE, RISK PREVENTION).

SAFE WORKING AROUND PROPELLERS (PROP-OFF TESTING).

SHORT-CIRCUIT PREVENTION AND PROPER WIRING PRACTICES.

ELECTRICAL SAFETY AND HEAT MANAGEMENT.

UNDERSTANDING NO-FLY ZONES AND LOCAL UAV REGULATIONS (BASIC AWARENESS).

MODULE 2 - SENSORS & DATA ACQUISITION

- IMU, GPS, barometer integration
- Ultrasonic/IR sensing
- Data filtering basics
- Wireless modules (Bluetooth, NRF24)
- Sensor calibration & readings
- Hands-on: IMU data, telemetry

IMU, GPS, BAROMETER INTEGRATION

IMU (Inertial Measurement Unit) components:

- Accelerometer → measures linear movement
- Gyroscope → measures rotation
- Magnetometer → measures orientation using Earth's magnetic field
- Reading raw sensor data via I²C or SPI communication.
- Understanding drift, noise, and why calibration is needed.

GPS modules:

- Getting location coordinates
- Understanding satellites, accuracy, update rates

Barometers (BMP180, BMP280):

- Measuring air pressure
- Converting pressure to altitude

ULTRASONIC/IR OBSTACLE SENSING

How ultrasonic sensors measure distance using sound waves.

How infrared sensors detect nearby objects.

Wiring and reading distance values in Arduino.

Limitations: blind spots, environmental interference, accuracy issues.

Use cases: simple obstacle detection, landing assistance.

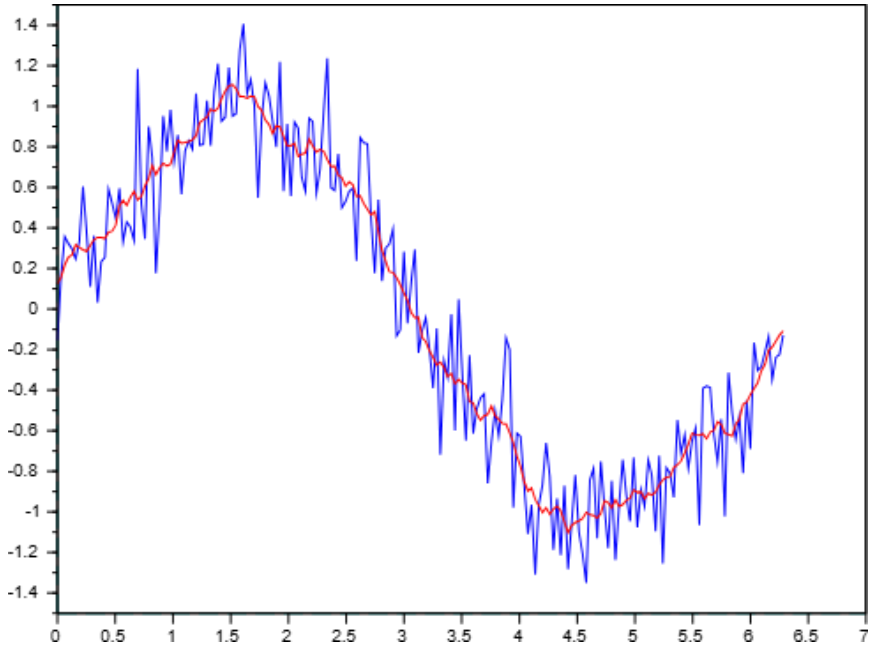
DATA FILTERING BASICS

Why raw sensor data is noisy and unstable.

Introduction to smoothing techniques:

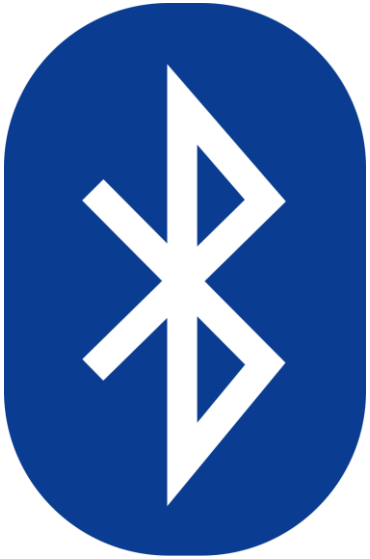
- Moving average
- Complementary filter
- *Basic* concept of Kalman filtering (not full math).

Combining accelerometer + gyroscope data for more stable orientation.



WIRELESS MODULES (BLUETOOTH, NRF24)

- How drones communicate with ground controllers or computers.
- Bluetooth: simple, short-range communication for mobile control.
- NRF24L01: long-range, low-power communication for custom transmitters.
- How to send commands or receive telemetry via Arduino.
- Understanding latency, bandwidth, and packet loss issues.



SENSOR CALIBRATION & READINGS

- Understanding sensor offset and drift.
- Performing calibration routines (for IMUs, magnetometers, barometers).
- How to ensure reliable and repeatable data.
- How environmental factors affect sensor accuracy.
- Plotting sensor readings to understand motion and orientation.

HANDS-ON: IMU DATA, TELEMETRY

- Reading accelerometer and gyro values in real-time.
- Getting GPS coordinates and plotting them.
- Displaying altitude from barometer readings.
- Sending sensor data wirelessly to a computer or mobile device.
- reating a basic telemetry dashboard (serial monitor or simple GUI).

MODULE 3 – FLIGHT CONTROL & PID (12 HRS)

- PWM motor/ESC control
- Sensor fusion for stability
- PID tuning (roll, pitch, yaw)
- Manual flight inputs
- Failsafe & error handling
- Hands-on: stabilization testing

PWM MOTOR/ESC CONTROL

- How Arduino generates **PWM signals** to control motor speed.
- Understanding ESC signal ranges
- Calibrating ESCs to match throttle range.
- Mapping control inputs to motor output.
- How small changes in PWM greatly influence drone stability.

SENSOR FUSION FOR STABILITY

- Combining accelerometer + gyroscope data for accurate orientation.
- Why drones cannot rely on a single sensor (noise, drift).
- Complementary filter concept for blending fast and slow signals.
- Understanding roll, pitch, and yaw angles from IMU readings.

PID TUNING (ROLL, PITCH, YAW)

What PID stands for: **Proportional–Integral–Derivative**.

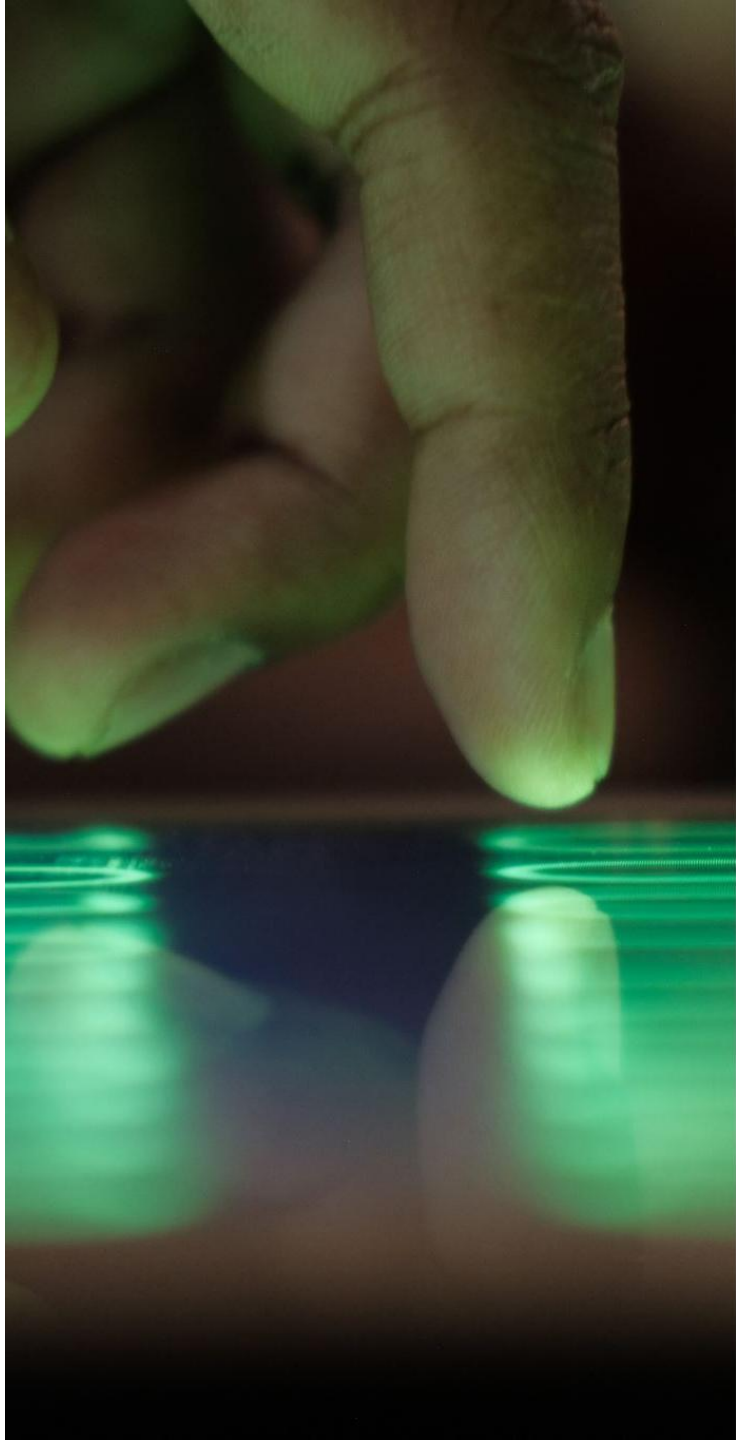
How each component affects drone stability

Understanding oscillations, overshoot, sluggish response.

Tuning PID values manually to achieve stable hover behavior.

MANUAL FLIGHT INPUT OPTIONS

- Using Bluetooth for smartphone control.
- Using NRF24 for custom RC-transmitter control.
- Mapping input commands to throttle, pitch, roll, and yaw.
- Data-validation techniques to prevent extreme or unsafe commands.



FAILSAFE & ERROR HANDLING

Detecting	Detecting sensor failure or loss of communication.
Implementing	Implementing safe motor shutdown routines.
Using	Using timeouts, watchdog timers, and voltage checks.
Preventing	Preventing runaway motors or unstable control loops.

HANDS-ON: STABILIZATION TESTING

- Building a simple drone test rig (non-flying) to test stabilization.
- Running motors using PID output.
- Observing IMU readings during movement.
- Adjusting PID values and seeing immediate changes in behavior.
- Logging data to evaluate control-loop performance.

MODULE 4 – AUTONOMOUS DRONE PROGRAMMING (14 HRS)

- Autonomous flight principles
- GPS waypoint navigation
- Altitude hold logic
- Obstacle avoidance basics
- Telemetry dashboards
- Final project: autonomous mission

AUTONOMOUS FLIGHT PRINCIPLES

Understanding what makes a drone autonomous (sensor-based decision making).

How drones interpret their environment without manual control.

Basics of autonomous behaviors: holding position, changing direction, following predefined rules.

Using sensor feedback (IMU, GPS, barometer, ultrasonic) to guide motion.

Safety considerations in autonomous flight: area limits, return-to-start, emergency landing logic.

GPS WAYPOINT NAVIGATION

How to read latitude/longitude from GPS modules.

Converting GPS coordinates into target movement.

Calculating direction and distance to next waypoint.

Using simple algorithms to move from one point to another.

Handling issues like GPS drift, slow update rate, and accuracy limits.

ALTITUDE HOLD LOGIC

- Using barometer data to measure altitude changes.
- Implementing PID control to maintain a stable height.
- Filtering noisy pressure readings for smooth altitude control.
- Understanding how vertical thrust and motor speed adjustments regulate height.

OBSTACLE AVOIDANCE BASICS

- Detecting nearby objects using ultrasonic or IR sensors.
- Creating simple decision rules: stop, change direction, slow down.
- Understanding sensor limitations (blind spots, reflection issues).
- Integrating obstacle detection with forward movement logic.

TELEMETRY DASHBOARDS

- Sending real-time flight data from Arduino to a computer or mobile device.
- Displaying sensor values, altitude, motor output, and GPS position.
- Using Serial Monitor or building a simple digital dashboard.
- Understanding how telemetry supports debugging and flight analysis.

FINAL PROJECT: AUTONOMOUS MISSION

- Designing a small autonomous task
- Combining IMU, barometer, GPS, and motor-control logic.
- Writing integrated Arduino code that coordinates all behaviors.
- Testing the mission on a controlled rig or simulation environment.
- Preparing a short technical report or demonstration.

MODE OF DELIVERY

Lectures and seminars on drone systems and programming

Hands-on Arduino and sensor-integration labs

Online modules with videos, readings, and quizzes

Discussion forums for collaboration and problem-solving

Tutor support and mentoring for coding and hardware tasks

LEARNING ACTIVITIES

- Lectures and seminars on drone systems and programming fundamentals
- Hands-on labs for coding, sensor integration, and flight-control tasks
- Case studies on real UAV challenges and problem-solving
- Group projects to build and test drone subsystems
- Online modules with videos, readings, and quizzes
- Mini-assignments for sensor data analysis and autonomous functions

ASSESSMENT

- **Quizzes (20%)** – After each module
- **Practical Tasks (40%)** – Motor control, sensors, PID, autonomy
- **Mini-Project (40%)** – Build and test an autonomous drone feature
- **Feedback** – Continuous support in labs and online
- **Submission** – Code, diagrams, and reports via LMS

DURATION & CREDITS

Total Duration: 44 hours

Credits: 2 ECTS

Workload Breakdown:

- Lectures/Seminars – 8 hrs
- Labs/Workshops – 16 hrs
- Online Learning – 8 hrs
- Project & Assessment – 12 hrs

TARGET PARTICIPANTS

- Engineering students and early-career technologists
- Robotics and automation enthusiasts
- STEM educators integrating drone technology
- Hobbyists, makers, and drone operators
- Professionals in surveying, monitoring, and related fields

PREREQUISITES

- Basic computer literacy required
- No prior programming needed
- Technical background recommended but not mandatory
- RPL accepted for experienced learners
- Sufficient English for technical materials


THANK YOU

ANY QUESTIONS ?