FLORIDA TECH

FLORIDA'S **STEM** UNIVERSITY®

S.H.I.E.L.D.

System for **H**igh **I**ntegrity **E**limination of **L**ow-Altitude **D**rones

First Semester Plan Presentatoin

Bryson Noble, Avionics Lead

Faculty Advisor: *Dr. Firat Irmak*

Goal and Motivation

- Team S.H.I.E.L.D. is challenged to design, build, and test a counter-UAS system for small, low-altitude aerial targets. This system shall be capable of autonomously detecting, tracking, and intercepting an aerial target using onboard sensing and computing.

Approach

- Keep a designated area under constant watch, and count on SHIELD to respond the moment an unauthorized drone comes within range. Once a target is spotted, SHIELD locks on, follows its every move, and intercepts it within 20 meters. No manual aiming or tracking required.
- Decide exactly how much control you want. Engage manual override to personally direct the response for peace of mind, or let SHIELD's onboard AI handle detection and engagement entirely on its own. Stay hands-on or hands-off depending on the situation.
- Once SHIELD locks onto a target, it stays locked on, tracking through brief obstructions or awkward angles instead of losing the target and having to reacquire it from scratch.

Novel Features/Functionalities

- The semi-autonomous detection and engagement is a novel functionality of SHIELD. SHIELD automatically detects and prompts an operator to engage. Once the human approves the engagement, the drone can track and intercept a target using onboard edge computing, rather than relying on a ground operator for control.
- SHIELD is small enough to carry on a person, and it is made with cost-effective materials to allow for wide distribution. SHIELD's versatility allows for both public safety and defense applications.

Avionics Hardware

[1] https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-nano/product-development/

[2] https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-orin/nano-super-developer-kit/

[3] https://www.raspberrypi.com/products/ai-hat/

[4] https://www.raspberrypi.com/products/camera-module-3/

[5] https://www.e-consystems.com/nvidia-cameras/jetson-orin-nx-cameras/full-hd-ar0234-global-shutter-camera.asp

[6] https://docs.arducam.com/Nvidia-Jetson-Camera/Jetvariety-Camera/AR0234/

Edge Computer Comparison			
	NVIDIA Jetson Nano [1]	**NVIDIA Orin Nano [2]**	Raspberry Pi 5 + AI HAT [3]
Compute Power	472 GFLOPS	**67 TOPS**	26 TOPS
RAM	4 GB	**8 GB**	8 GB
Power draw	5-10 W	**7-25 W**	3-15 W
AI accelerator	128-core Maxwell GPU	**Ampere GPU + Tensor cores**	Hailo-8 NPU

Camera Comparison			
	Raspberry Pi Camera Module 3 [4]	**e-CAM25_CUONX [5]**	Arducam AR0234 [6]
Interface	MIPI CSI-2	**MIPI CSI-2**	MIPI CSI
Shutter type	Rolling	**Global**	Global
Resolution	11.9 MP 4608*2592	**4K 3840*2160**	2.3 MP 1920*1200
Frame rate capability	720p120	**1080p120**	720p120

Algorithms and Tools

- OpenCV: video frame preprocessing and per-track Kalman filtering (smoothing + motion prediction)
- Ultralytics YOLO (yolo11n.pt): real-time object detection
- ByteTrack (bundled with Ultralytics): persistent multi-object track IDs
- Unity (Editor 6000.3.11f1): "SHIELD Virtual Camera" stand-in camera feed for PC-based emulation, streamed to the Python pipeline over TCP

Technical Challenges

- Running detection and tracking efficiently on the NVIDIA Orin Nano edge hardware: the pipeline is currently only validated CPU-only on a Windows PC, and moving to the power- and thermal-constrained onboard hardware while keeping frame rate high enough to track a moving target in real time is still an open problem.
- Detection and tracking need to hold up consistently on the embedded hardware and in real-world conditions, not just in ideal PC-based emulation, before the system can be trusted to run on its own.
- Integrating all the code that runs on emulation onto the drone prototype once it is created will be a challenge, as will accurately simulating the drone's physics in Unity for testing

Letterbox Resizing Bottleneck

Camera Input: 1920 x 1080

After Preprocessing: 640 x 640

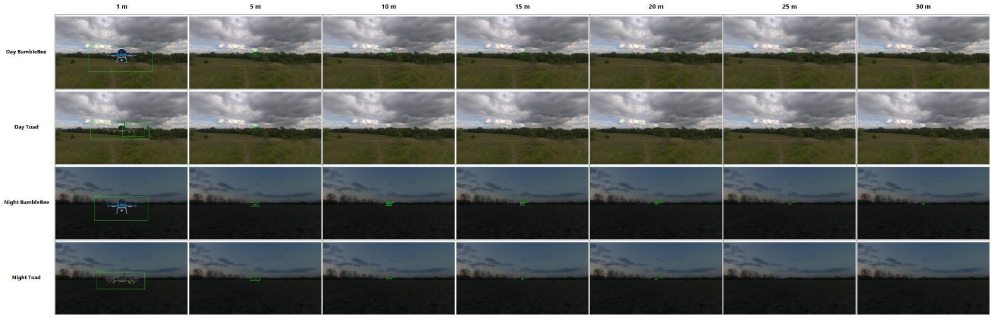
YOLO Letterbox Resize [7]

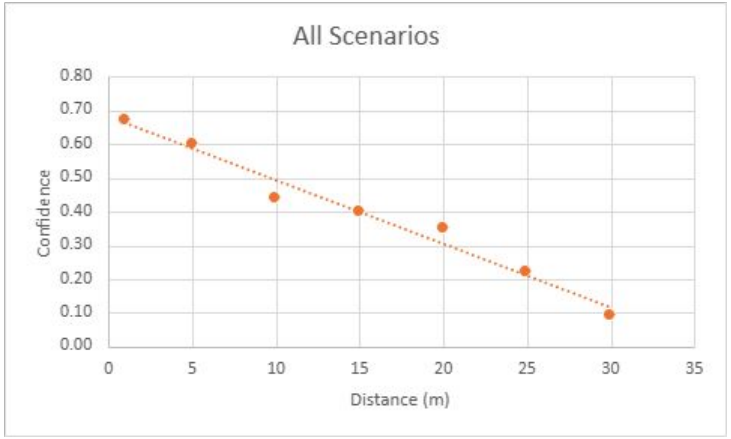
Dataset	Images
Drone Dataset (UAV), Mehdi Özel [8]	1359
Drone YOLO Detection, sshikamaru [9]	4014
Unity Drone Dataset, Bryson Noble	2000

[7] https://docs.ultralytics.com/guides/preprocessing-annotated-data
[8] https://www.kaggle.com/datasets/dasmehdixtr/drone-dataset-uav
[9] https://www.kaggle.com/datasets/sshikamaru/drone-yolo-detection

Target Engagement System Limiting Distance

- Verify distance/confidence requirements
- Find Limiting Distance for TES





[10] "Limiting Distance Experiment," Bryson Noble

Milestone 1 TODOs

- Compare and select technical tools for detection, tracking, and (once defined) interception subsystems.
- Provide "hello world" demos evaluating candidate tools - already partially done: the repo's emulation pipeline runs YOLO detection + ByteTrack + Kalman smoothing against a webcam or the Unity virtual camera.
- Resolve initial technical challenges identified above.
- Compare and select collaboration tools for software development, documents/presentations, communication, and task scheduling.
- Create Requirement Document.
- Create Design Document.
- Create Test Plan.

Milestone 2 TODOs

- Complete development on STE (SHIELD Test Environment) program.
- Find, combine, and train a custom drone/balloon detection model.
- Add moving target GameObject instantiation to testlib and motion scripting to the Unity test scenes.
- Write and implement STE test scripts to verify system requirements and initial requirements in requirement document.
- Design and begin implementing the manual override control interface.

Milestone 3 TODOs

- Integrate detection/tracking output with the interception subsystem (emulation only) once it exists.
- Implement "Hardware" platform mode (NVIDIA Orin Nano) and test on drone prototype, if built.
- Run the custom-trained model against the full set of Unity lighting/environment scenes and evaluate confidence against the threshold in the requirements document.
- Have all existing requirements traced to STE test scripts.
- Conduct full end-to-end system test and demo on emulation.

Questions?