

**CSE0408 Programming III**

### Department of Computer Engineering

Instructor : Yusuf Altunel, PhD

Academic Year : 2023 - 2024 Semester: Fall  Spring  Summer

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Student No : | Name & Surname : | E-mail Address : | Signature : |
| Student-1: | 2000004855 | Emrehan Gökçay | emrehan\_gokcay@hotmail.com | karanlık, sanat içeren bir resim  Açıklama otomatik olarak oluşturuldu |
| Student-2: |  |  |  |  |
| Student-3: |  |  |  |  |

|  |
| --- |
| **Project is:** A survey  An Implementation  Code in Python: Yes  No  **Project Name** : UAV Lock-On System using Image Processing in Python |
| **Project Statement**: A Lock-On system used in UCAV’s (Unmanned Combat Aerial Vehicles) and UAV’s (Unmanned Aerial Vehicles) using to help these aerial vehicles main algorithm of optical operations to follow up targets, and shooting the tracked targets down. We’re going to create the similar but the less violent one in Python. I’m going to use one of Python’s already existing libraries, the OpenCV Library for optical operations such as live tracking the enemy vehicles which were trained by the data of existing Unmanned Aerial Vehicle teams’ vehicle bodies of Teknofest using YOLOV8. The project will have the implementation of the code which will actively track an enemy UAV through its main camera and locks on to them actively if found.  Complete List of Submissions:   1. **Resources**: The followings are inseparable submission items:    1. Project proposal (this file) to explain the project    2. Source code file    3. Database/file to keep permanent data used in application    4. Design document to explain the details of the implementation, data description, algorithms and libraries. If there any external references, the exact position in source code of those references.    5. User documents to describe how the code is being run.    6. Guide to show how to compile, deploy and run the application    7. GitHub page to access and download the source code    8. Other submissions if any 2. **Functions**: The following list of modules/features/requirements/functions/user stories    1. Computer Vision: This function/module is responsible for implementing the basis of our project and is responsible for processing images or video frames captured by the drone's camera and identifying and tracking targets. For example, we can use techniques such as edge detection and template matching to identify objects in the camera's field of view.    2. Live Tracking: This module focuses on live tracking of defined targets. Live tracking involves the process of constantly updating the position and status of targets as they move within the field of view. This may include methods that predict the future position of objects based on their past motion, for example using optical flow.    3. Dataset training with YOLOV8: YOLOV8 refers to the YOLO (You Only Look Once) deep learning object detection model. YOLOV8 is a deep learning algorithm that can be used for real-time object detection. Training with YOLOV8 involves teaching the system using a labeled dataset to recognize the objects you want to recognize. This process includes data preparation, model training, and evaluation.    4. Machine Learning: This function includes the application of machine learning algorithms to enhance the system's capabilities, such as target recognition, decision-making, or predictive analysis. We can use machine learning to classify the type of objects, predict their future position, or make decisions based on visual data.    5. Simulated on ROS/Gazebo: This module implies that we will simulate the functions of the drone and its camera in the Robot Operating System (ROS) and Gazebo environment. ROS is a framework for writing software to control robots and interact with various hardware devices. Gazebo allows us to simulate robots and their interactions with their environment. In this context, it will be used to simulate the drone and its camera to create a simulation to test and validate your visual processing algorithms before deploying them on real hardware. |
| **Team Roles**   1. Developers:    1. Emrehan Gökçay -> Whole developer. 2. **Project Manager** to define and follow up project planning: Emrehan Gökçay 3. **Product Owner** to manage Product Backlog Items & Product roadmap: Emrehan Gökçay 4. **SCRUM/Agile Master** to support the team and guarantee execution of responsibilities: Emrehan Gökçay |
| **Resources** (libraries and resources to use in implementation):   * OpenCV Python Library * <https://www.youtube.com/playlist?list=PLRwrEtlfdUVBlEbYHb_y1AABvR4z_GjO5> * <https://www.youtube.com/playlist?list=PLkJUWWxr1XL7ZKq5oL2DtpljgQO4mLVA1> * JetBrains PyCharm * <https://www.youtube.com/playlist?list=PLzcys7whQ6eSJdSHJh-xdOLvWPP24J76x> * <https://docs.opencv.org/4.x/d6/d00/tutorial_py_root.html> * My notebook notes about OpenCV |

#### I already have an active roadmap with me which has these parts each week starting from this week:

* Learning OpenCV
* Create an example face recognition program through the example datasets of Haar-Cascade for putting the skillsets to the test that I’ve learned before.
* Starting the main code of UAV Lock-On System.
* Finding example images for training data & training the code with datasets.
* Running tests of vehicle for testing the code whether it is running properly of not.
* If correct, collecting test videos while actively running the code on UAV.
* Preparing documentations for explaining the project in detail.
* Final Presentation.

#### Links

IEEE Formatted: https://template-selector.ieee.org/secure/templateSelector/downloadTemplate?publicationTypeId=3&titleId=1&articleId=1&fileId=550