Department of Computer Science & Engineering
The University of Texas at Arlington

Detailed Design Specification
CSE 4317: Senior Design II
Fall 2019

Nebuloo
Falcon

Alejandro Waumman
Juan Barragan
Sunil Niroula
Luis Antonio Gonzalez Trejo

Revision History

Revision	Date	Author(s)	Description
0.1	9.15.2019	LAGT	document creation
0.2	11.21.2019	LAGT	small updates
0.3	12.06.2019	LAGT	Final Draft

Contents

1 Introduction **5**

2 System Overview **5**

3 AI Agent Layer Subsystem **6**

3.1 Layer Operating System 7

3.2 Layer Software Dependencies 7

3.3 Object Detection Subsystem 7

4 Controller Layer Subsystem **9**

4.1 Layer Operating System 9

4.2 Layer Software Dependencies 9

4.3 Data Subsystem 9

5 Simulation Engine Layer Subsystem **10**

5.1 Layer Operating System 10

5.2 Layer Software Dependencies 10

5.3 Simulation State Subsystem 10

6 Appendix **12**

List of Figures

1 System architecture 6
2 Example subsystem description diagram 7
3 Example subsystem description diagram 9
4 Example subsystem description diagram 10

List of Tables

1 Introduction

Falcon is an artificial intelligence (AI) system that controls a anti-aircraft ground-mounted turret. It will be used to demo the use of Dream Shader, a simulation engine that can be used to train an AI system and then verify and validate it through various test scenarios. Those will be the primary components of the system, along with a controller. The controller will take in data from the AI agent layer and the simulation engine layer to decide what actions the turrent should take. This can be information about the environment being simulated that the system is expected to receive in regular field tests as well as action suggestions from the AI agent.

2 System Overview

The simulation engine layer will feed the AI agent layer a framebuffer. It will also feed the controller layer information about the state of the simulation and settings that the user may have set that should influence the move evaluation. The ai agent will give the simulation engine a modified framebuffer since it will act as a filter that add bounding boxes and label to objects that it recognizes. It will also send its move suggestion to the controller layer. The controller layer will only send data to the simulation engine layer to alter the state of the simulation.

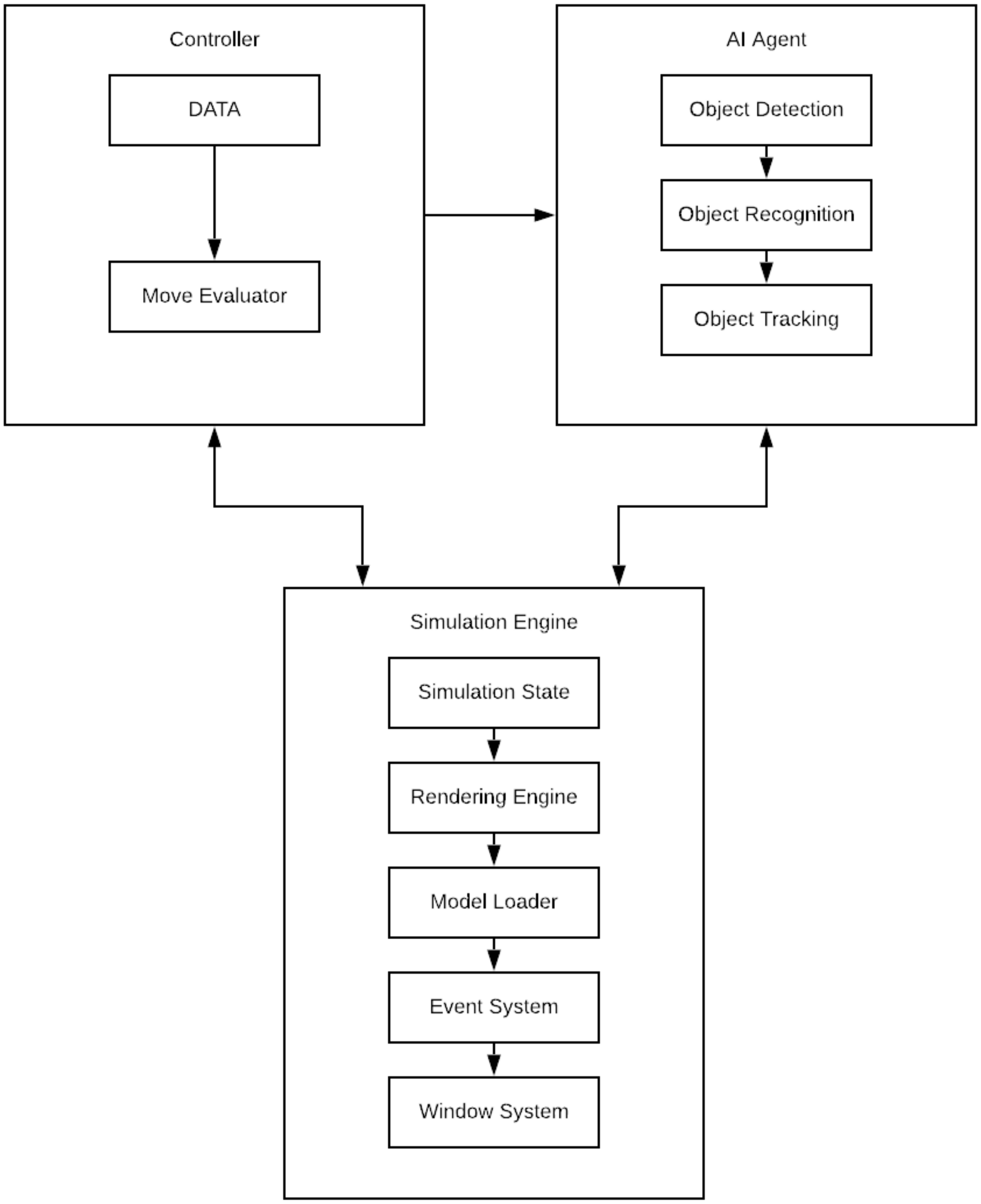


Figure 1: System architecture

3 AI Agent Layer Subsystem

The AI agent layer will consist of three subsystems. There will be an object detection subsystem, an object recognition subsystem, and an object tracking subsystem. This layer will receive a frame buffer and alter it to change the simulation state and send its move evaluation to the controller layer.

3.1 Layer Operating System

Linux

3.2 Layer Software Dependencies

OpenCV, YOLOv3, Darknet, and possibly TensorFlow if we have the time to train a new model from scratch.

3.3 Object Detection Subsystem

This layer is particularly responsible for object detection, object recognition and object tracking. The first subsystem under AI agent is going to be object detection which will use the YOLO v3 algorithm to detect the object through the video feed that is provided to the algorithm.

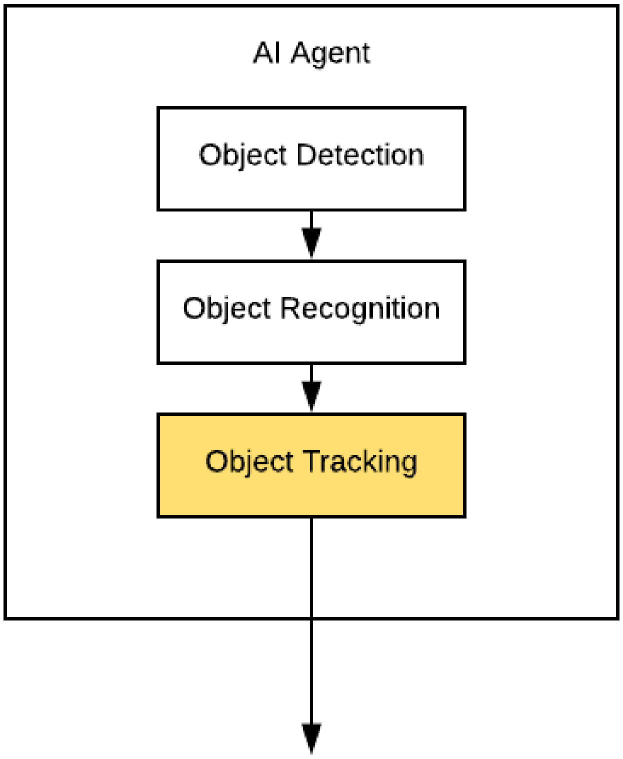


Figure 2: Example subsystem description diagram

3.3.1 Subsystem Operating System

This subsystem is running on Linux

3.3.2 Subsystem Software Dependencies

YOLOv3 and Darknet as of now, but we will try to create our own model from scratch.

3.3.3 Subsystem Programming Languages

C++17

3.3.4 Subsystem Data Processing

This subsystem uses YOLO v3 algorithm to detect the objects in an image. This interacts with the object recognition subsystem under the AI agent system. This subsystem will take the video feed from the camera and give the input as an image to the algorithm and the algorithm will detect if the object we are looking for is present in the image or not. This is a continuous process as in our project we are looking for a plane that is flying so it has to check all the possible feeds and find the object and shoot it down.

4 Controller Layer Subsystem

The layer that is used to make decisions based on the information provided by the user and the other layers in the system. The subsystems in the Controller layer are "Data" and the "Move Evaluator."

4.1 Layer Operating System

Linux

4.2 Layer Software Dependencies

This layer depends on the proper functionality of the Object Detection Subsystem in order to work as desired. OpenCV and YOLOv3 are vital for this.

4.3 Data Subsystem

This subsystem collects the move information given to it by the AI Agent and any other information given to it from the Simulated Engine.

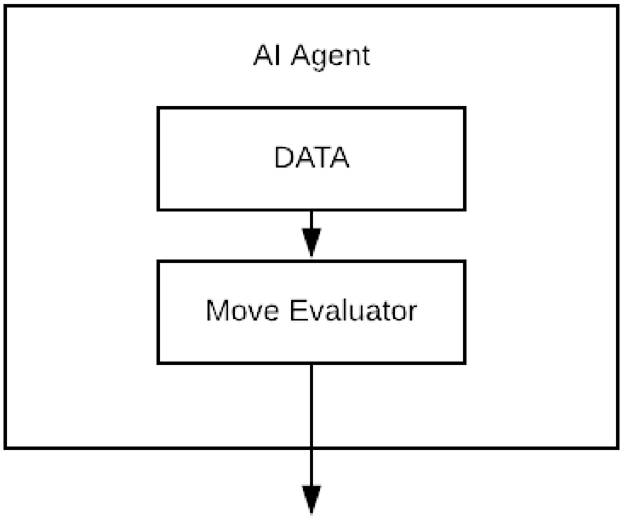


Figure 3: Example subsystem description diagram

4.3.1 Subsystem Operating System

Linux

4.3.2 Subsystem Programming Languages

C++17

4.3.3 Subsystem Data Structures

Data will mostly flow through the system as Objects.

5 Simulation Engine Layer Subsystem

The simulation engine is the heart of the system, without it, the project would be basically non-existent. The simulation engine will render scenes and then the other layers will extract useful data from them in order to perform object detection, tracking, and actions to take.

5.1 Layer Operating System

Linux

5.2 Layer Software Dependencies

It depends on the AI Agent software layer which consists of OpenCV, darknet and our graphics engine.

5.3 Simulation State Subsystem

The simulation state subsystem holds all of the data related to the state of the simulation. This controls how the renderer will draw things, how the controller layer will make decisions, and many other things that rely on the state of the simulation.

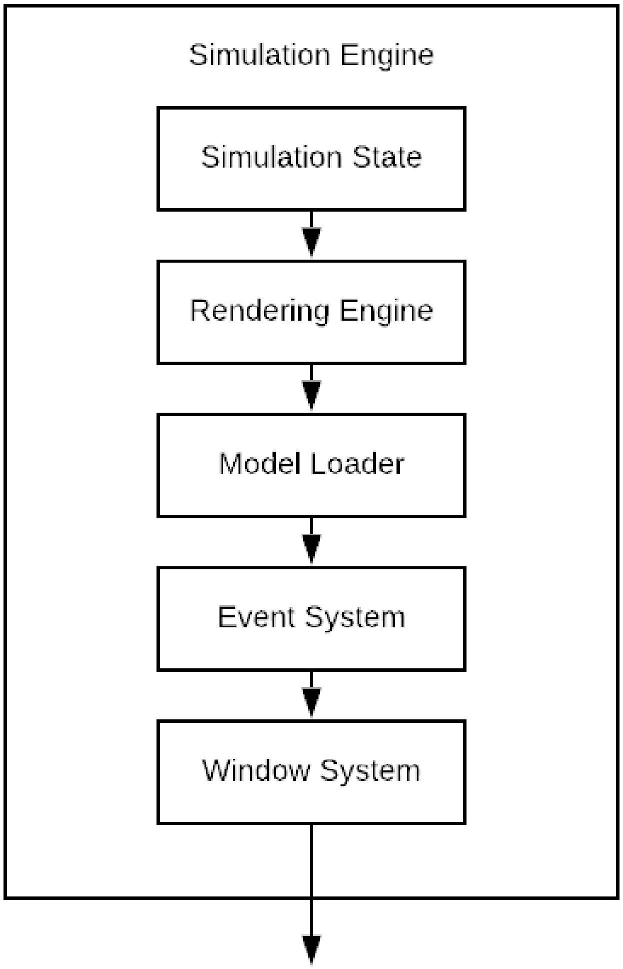


Figure 4: Example subsystem description diagram

5.3.1 Subsystem Operating System

Linux

5.3.2 Subsystem Software Dependencies

The simulation engine is basically the most important part of the entire system. This layer has a few dependencies of its own such as STB, GLFW, and Vulkan. However, this layer also depends on the other layers in order to function correctly in unison, which is where Darknet and YOLOv3.

5.3.3 Subsystem Programming Languages

C++17

5.3.4 Subsystem Data Structures

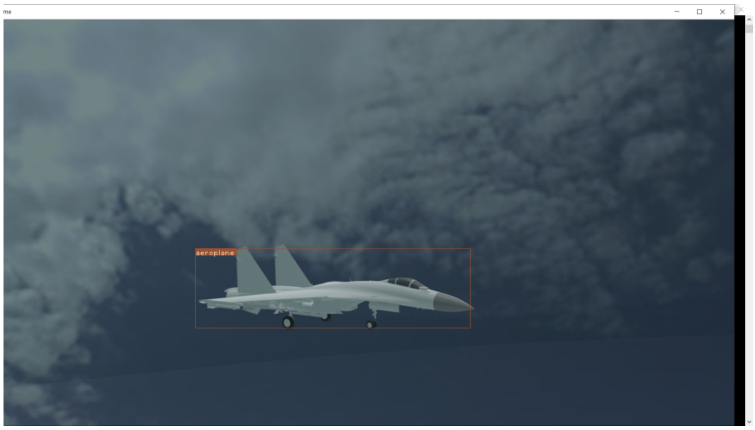
Data moving back and forth between layers as objects or just as data streams.

5.3.5 Subsystem Data Processing

Ideally, data moves through this layer in a top down fashion, starting with the window system, model loader, rendering and displaying.

6 Appendix

Screen captures and logos will be added eventually as soon as they are produced.


me
aeroplane