Evidence 2

Mission: Person of interest

Modeling of Multi-Agent Systems with Computer Graphics

Grupo 101

Ana Elena Velasco García	**A01639866**
Baltazar Servín Riveroll	**A01643496**
Emilio Pardo Gutiérrez	**A01644781**
Jozef David Hernández Campos	**A01644644**
Maria José Medina Calderón	**A01639205**

Iván Axel Dounce Nava

Jesús Israel Hernández Hernández

Oscar Guadalupe Hernández Calderon

Fecha de entrega:

12 sept 2025

Index

1. Description

1.1 - Main problem

The project addresses the complex challenge of coordinating multiple autonomous agents in emergency search and rescue scenarios. Specifically, it tackles the deployment of three independent drones in a dynamic 3D environment to efficiently locate and assist scattered individuals.

The core problems include implementing realistic vision-based detection that mimics real-world sensor limitations rather than using direct coordinate systems. This ensures the simulation reflects actual deployment constraints where drones must actively search and identify targets through their own perception systems.

Additionally, the system must prevent resource conflicts through intelligent multi-agent coordination to ensure each person receives exactly one drone. This requires sophisticated assignment protocols that avoid overlapping missions while maintaining autonomous operation.

Environmental challenges include handling obstacles that block line-of-sight and require adaptive navigation strategies. The drones must demonstrate realistic behavior when trees, buildings, or terrain interfere with their detection capabilities.

Finally, the system must maintain autonomous decision-making while providing human operators with real-time monitoring and control capabilities. This scenario simulates critical real-world applications where multiple autonomous units must work together efficiently without central micromanagement.

1.2 - Solution

The solution implements a sophisticated multi-agent system architecture featuring three autonomous drone agents with individual vision systems and a centralized coordination

layer. Each drone operates independently using a realistic perception model that combines SphereCast detection (1m radius, 100m range) with raycast line-of-sight verification to simulate actual sensor limitations.

The vision system continuously scans 360° at 45°/second, filtering for "Person" tagged objects while respecting physical obstacles that block detection. This creates realistic behavior where drones must actively search and cannot detect targets through walls or dense vegetation.

A centralized DroneTargetAssigner prevents conflicts by maintaining a global registry of assigned targets, ensuring one-to-one drone-person mapping. This coordination layer operates without requiring direct inter-agent communication, demonstrating emergent collaborative behavior.

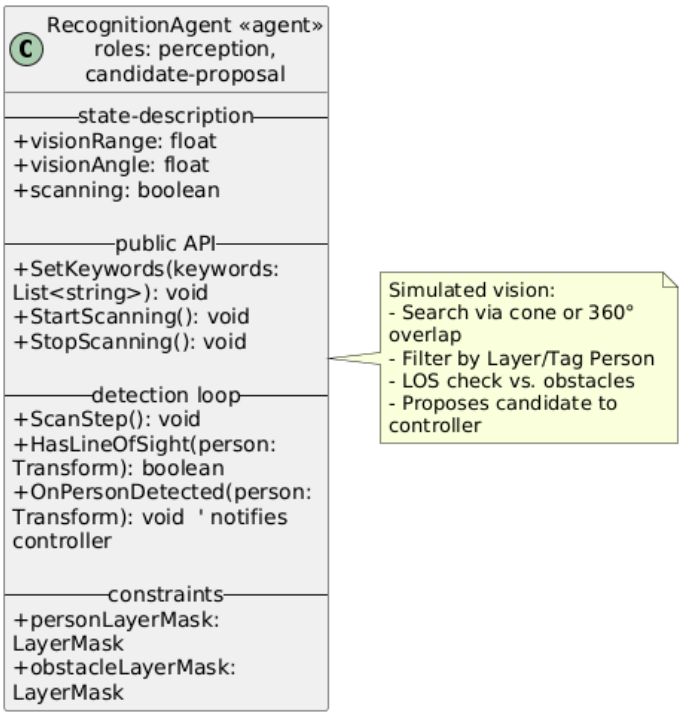
The system features intelligent obstacle handling through multi-angle raycast attempts and memory-based search patterns around last known positions. When a drone loses sight of its target due to obstacles, it attempts alternative viewing angles and searches the area where the person was last detected.

A comprehensive Unity based interface provides real-time monitoring with color-coded status indicators ([SEARCHING], [FLYING TO TARGET], [LANDING], [LANDED]) and simulation controls (pause/resume, restart, speed adjustment).

The architecture demonstrates emergent behavior where individual autonomous agents achieve coordinated results through intelligent local decision-making.

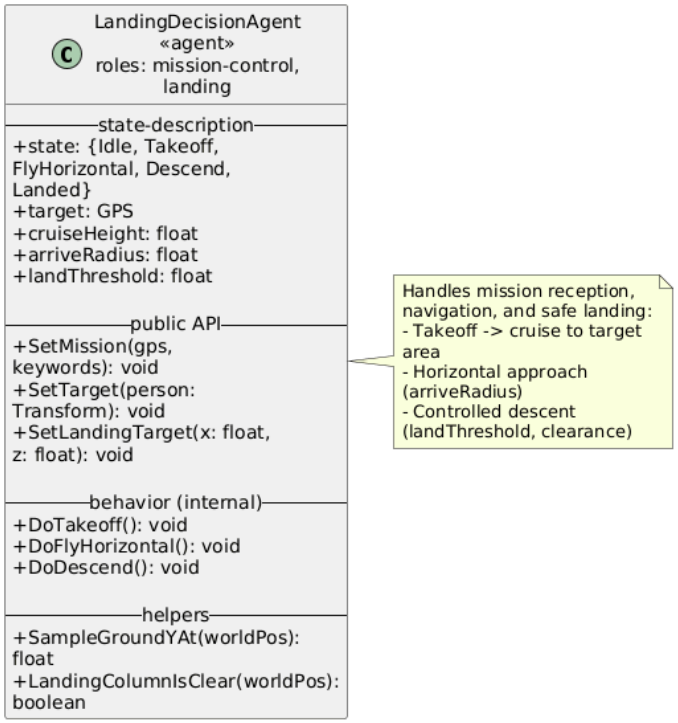
2. Agent UML diagrams

Figure 2.1 - Recognition Agent Class Diagram



Description: The Recognition Agent perceives and interprets the environment through raycasts, as it detects objects tagged as "Person". After continuously scanning the environment, the detected candidates are proposed to the Landing-Decision Agent.

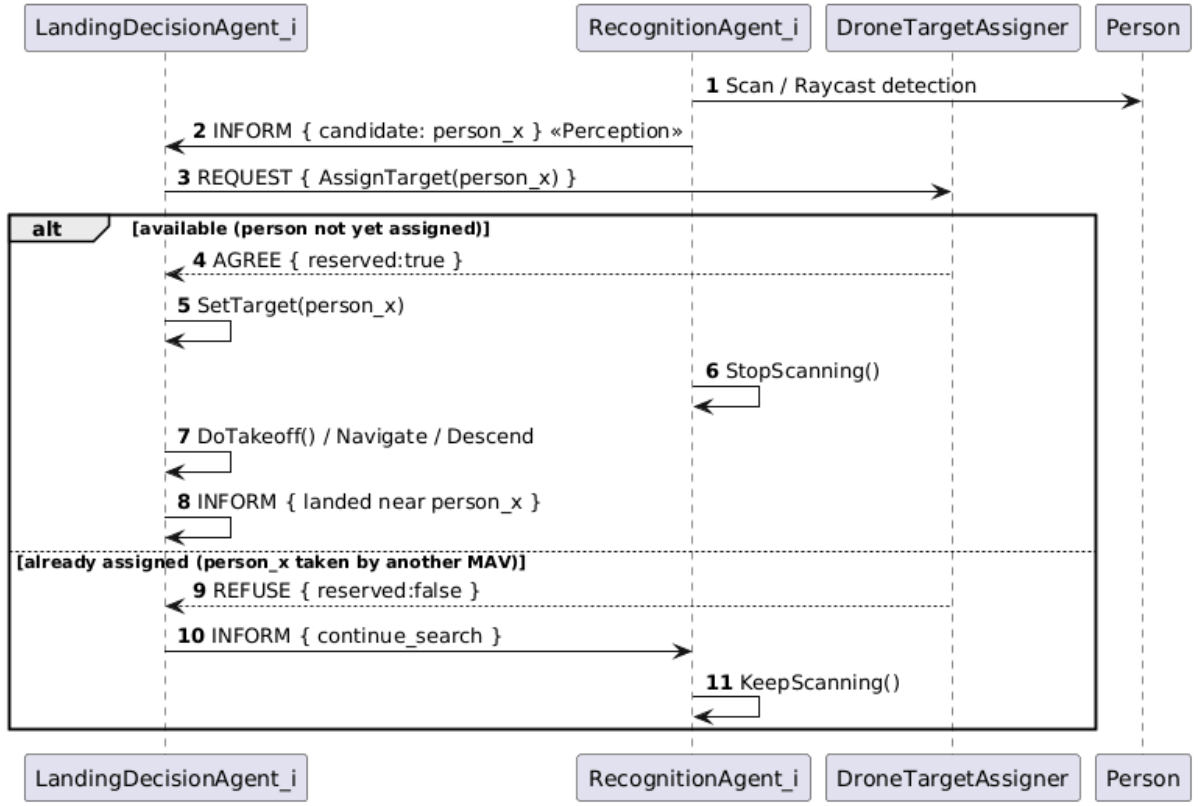
Figure 2.2 - Landing-Decision Agent Class Diagram



Description:

The Landing-Decision Agent manages the overall logic of the MAV. It sets the mission and is the source of information. It also ensures that the MAV can perform a safe landing near the person of interest.

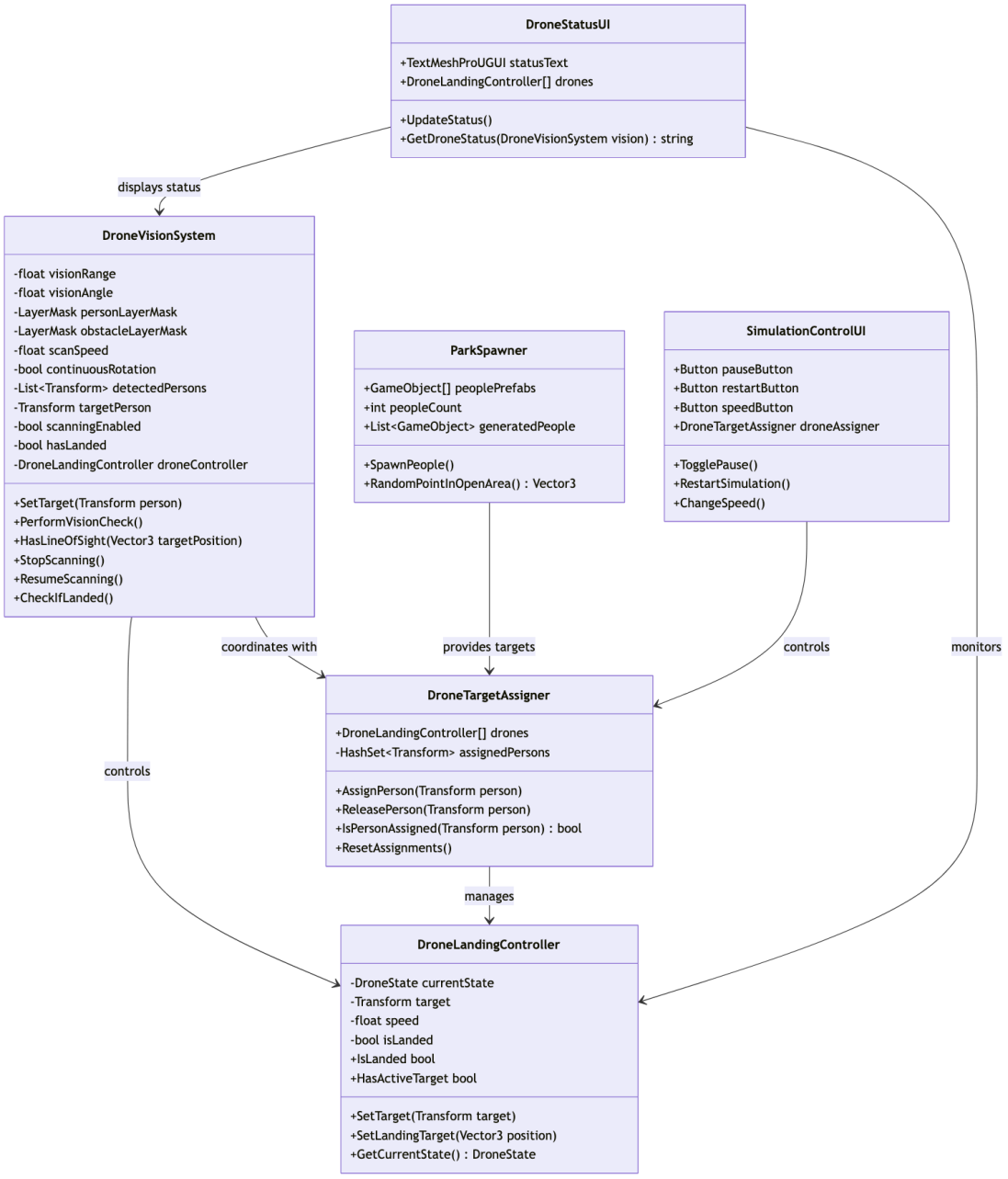
Figure 2.3 - Agent Interaction Protocol Diagram



Description: The Agent Interaction Protocol diagram models how the two agents exchange communicative acts. It shows the sequence from start to finish.

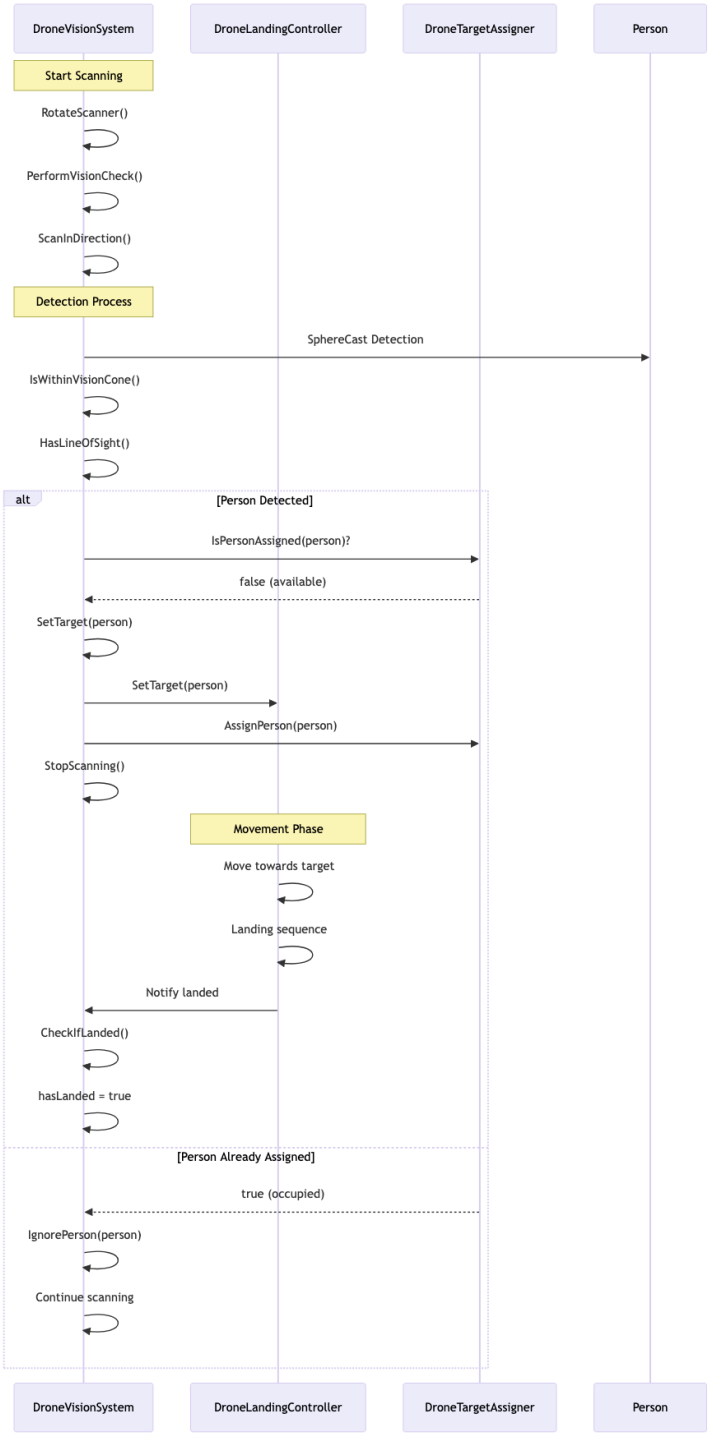
3. Simulation UML diagrams

Figure 3.1 - Simulation UML class diagram



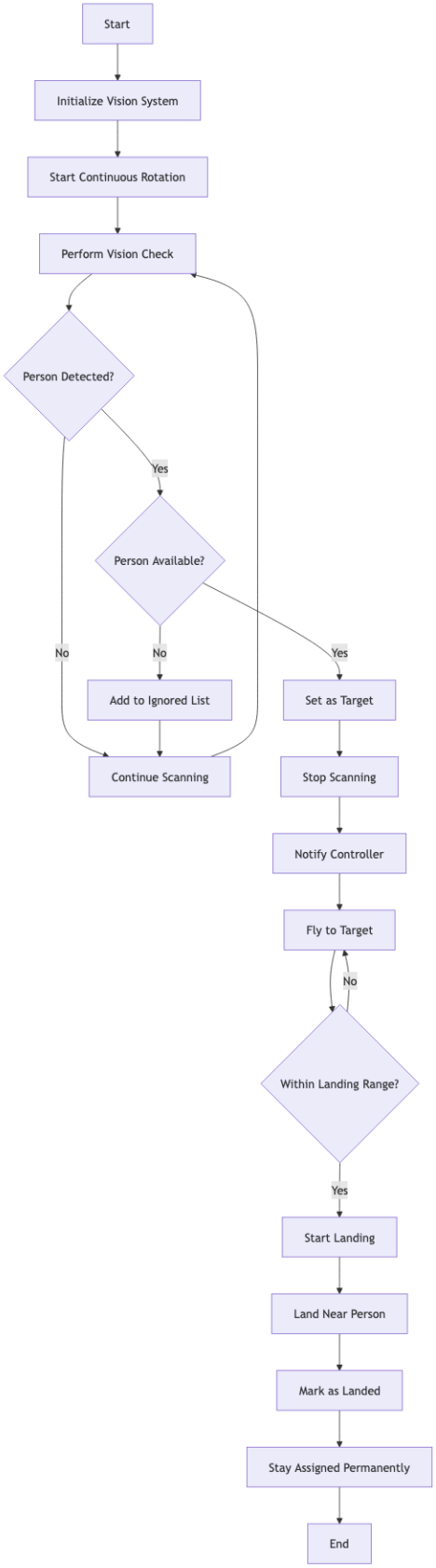
Description: The Simulation UML Class Diagram illustrates an overview of the simulation. It includes the MAV and its main agents along with the park spawner in charge of providing the target, and the drone-simulation UI.

Figure 3.2 - Drone detection process UML Sequence Diagram



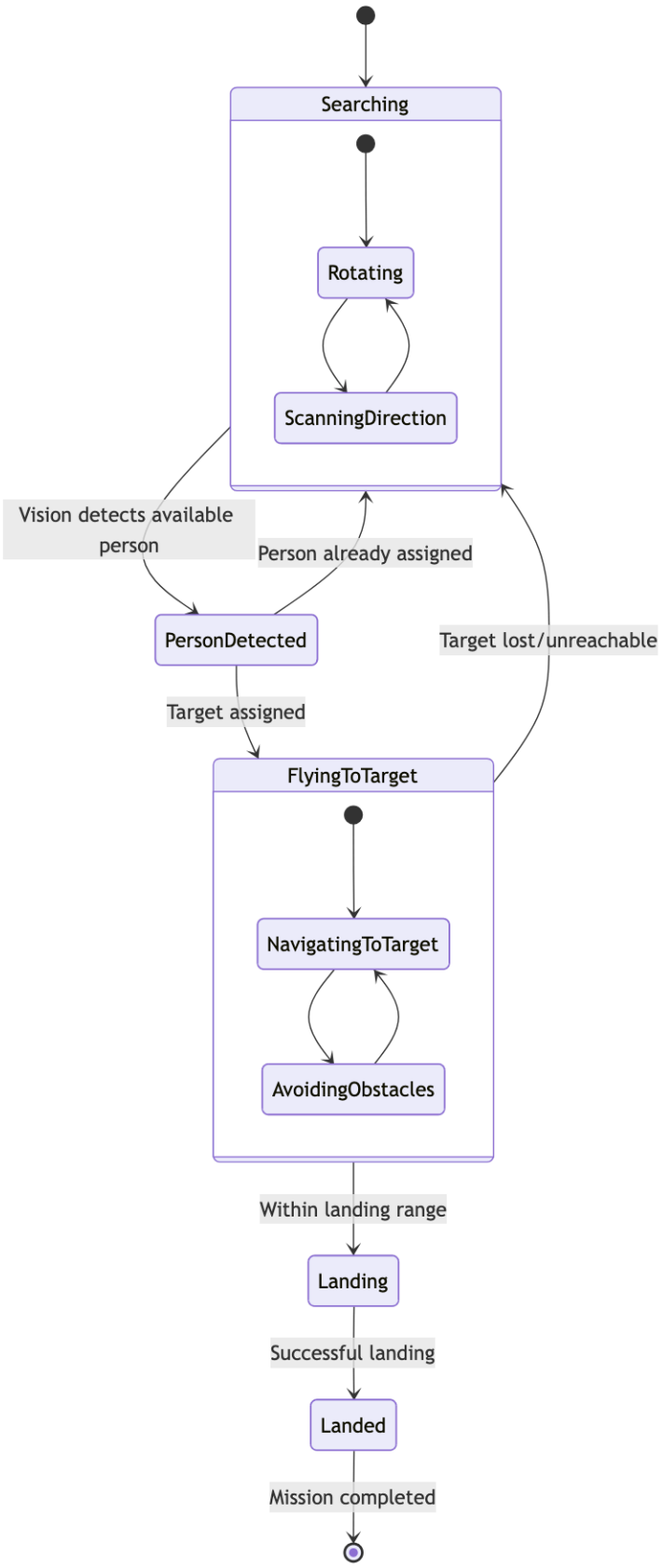
Description: The Drone detection process diagram shows the inner workings of the drone and it highlights the interactions between perception, coordination, and control, that makes the MAV able to autonomously identify and land near a target.

Figure 3.3 - Drone Behavior Flow UML Activity Diagram



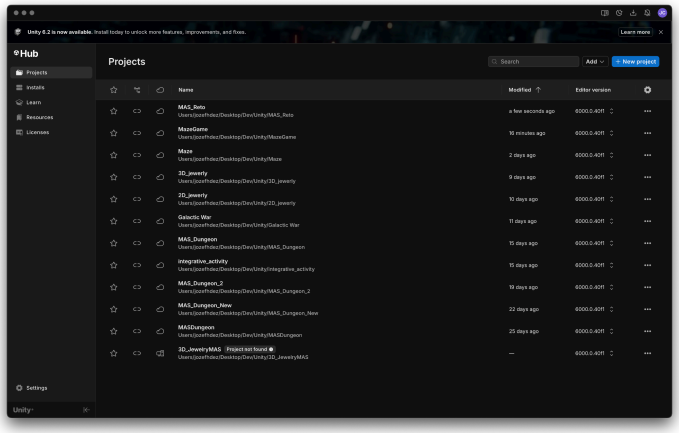
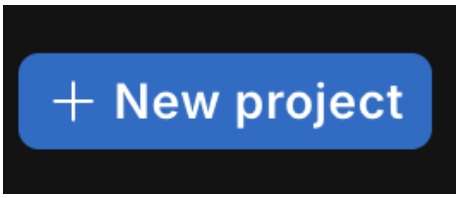
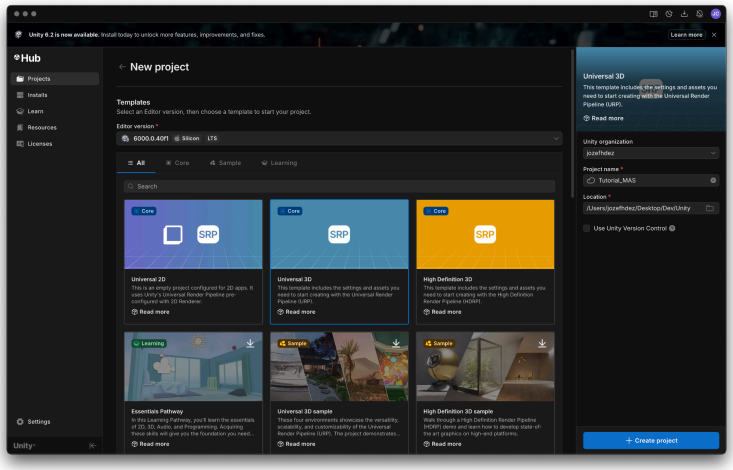
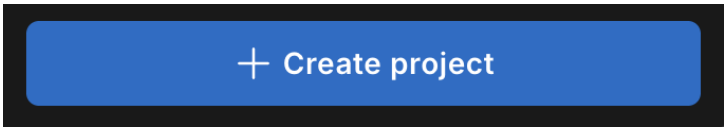
Description: The Drone Behavior Flow Diagram models the actions and responses of the drone during a mission, with an emphasis in the sequential decision-making steps that enable the autonomous functions of the MAV, such as detection, assignment, and landing.

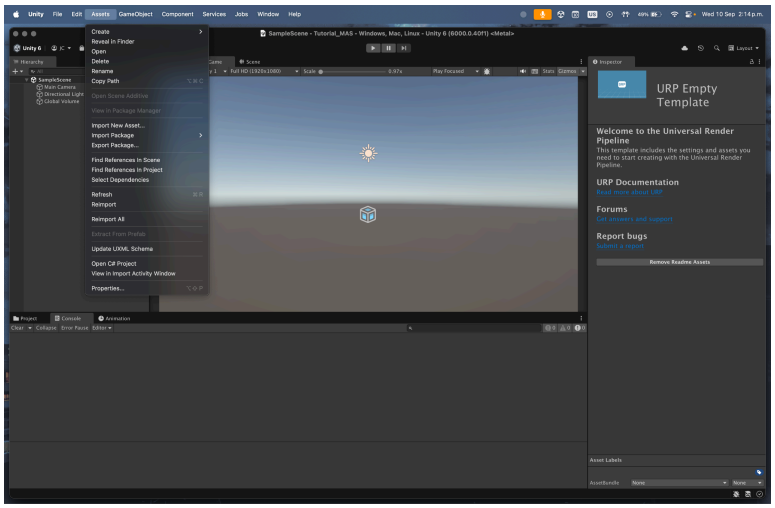
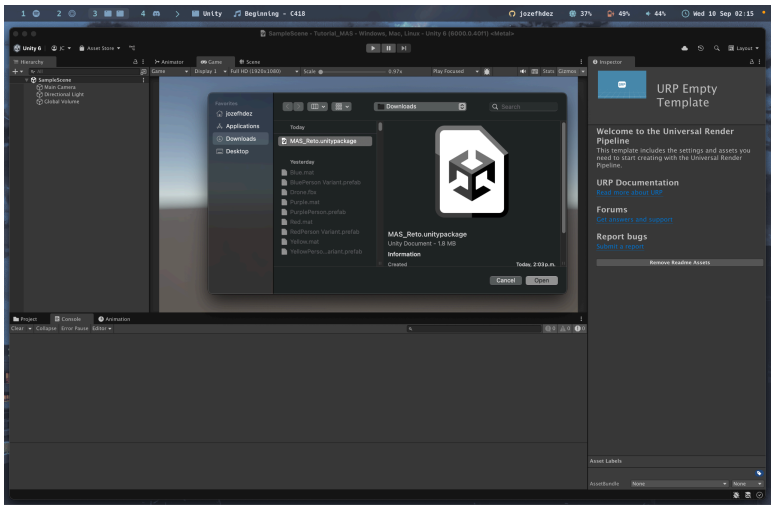
Figure 3.4 - Drone UML State Diagram

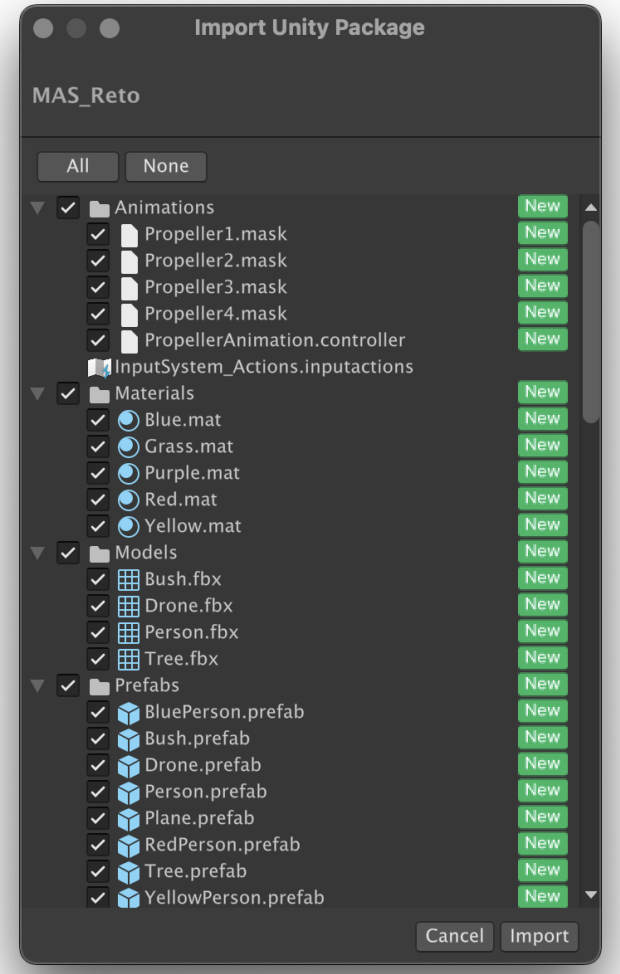


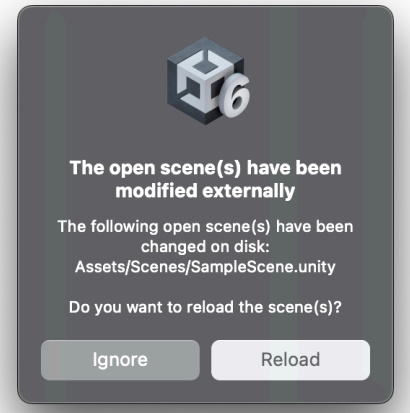
Description: The Drone State Diagram represents the different operational modes of the MAV throughout its mission. This diagram models how the drone cycles between perception, navigation, and landing states until the objective is achieved.

4. Installation process documentation

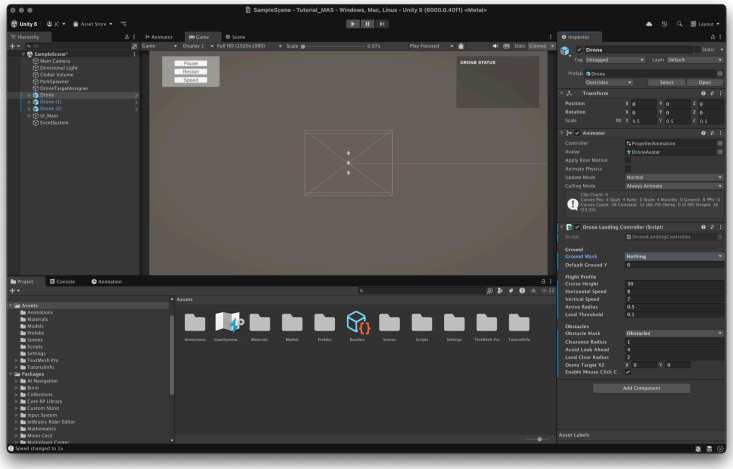
	Open Unity Hub
  	Create new Universal 3D project

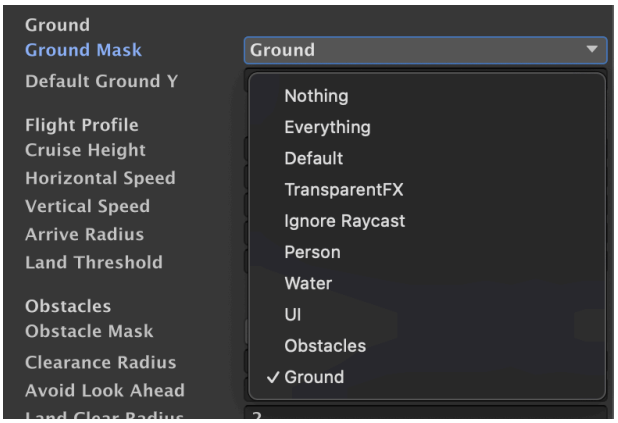
	Open the Assets tab and choose the Import Package option
	Look for the unity package and open it

	Choose the “All” option and import it

	When this pop up shows up press Reload
	Now you will need to add a couple of layers to the unity project

	Add them in the correct index and identical name as shown in the picture

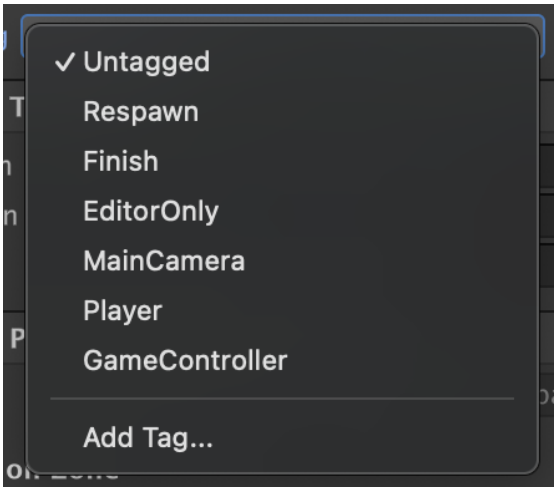


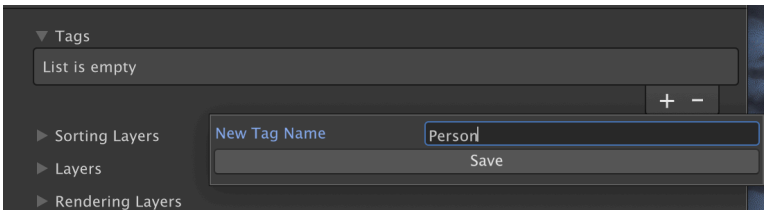


It is needed to add the layers to the Drone objects, choose them in Hierarchy and in the inspector modify the next:

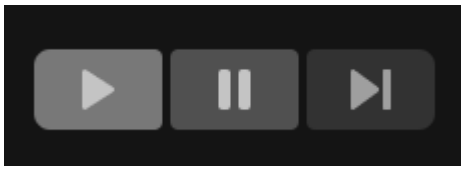
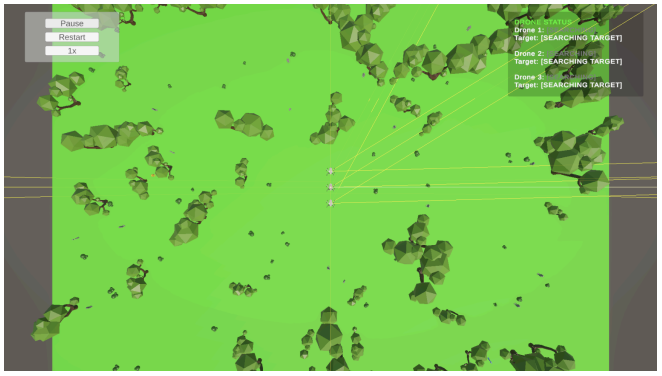
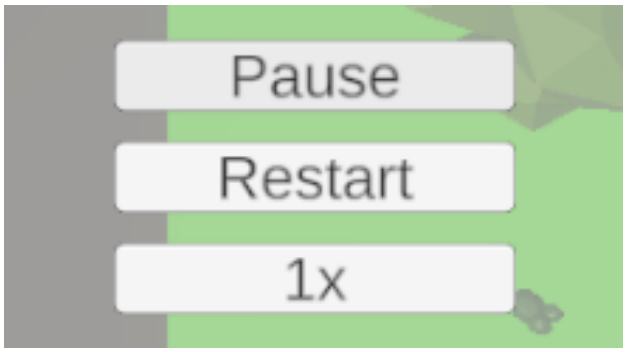
Ground Mask = Ground.

Repeat for each drone.





Now let’s create a tag, call it “Person”

	You can start the simulation
	This is the main view you will see
 	The graphical interface for our simulation. **Control panel:** Here you'll be able to pause, restart and change the simulation speed **Drone status panel:** This section will show you the drone status, which will change accordingly to the drone state.

5. Implementation of the agents

None
Code can be found in Canvas and Github

5.1 - Link to DroneLandingController.cs

5.2 - Link to DroneVisionSystem.cs

5.3 - Link to DroneTargetAssigner.cs

6. Implementation of the graphical-control interface

None
Code can be found in Canvas and Github

6.1 - Link to ParkSpawner.cs

6.2 - Link to SimulationControlUI.cs

6.3- Link to DroneStatusUI.cs

7. Repository

7.1 - Link to Repository

8. Video execution of the system

8.1 - Link to Video

9. Individual documentation

9.1 - Questions

1. Why did you select the multi-agent model?
2. What variables were considered when making this decision?
3. What is the interaction of these variables with respect to the mission results?
4. Why did you choose the presented system architecture?
5. What are the advantages of the final solution?
6. What are the disadvantages of the presented solution?
7. What modifications could be made to reduce or eliminate the mentioned disadvantages?
8. A reflection on your learning process. To do this, review the original document that contains your expectations at the beginning of the block and compare it with the experiences you had throughout these weeks.

9.2 - Ana Elena Velasco García

1. We chose a multi-agent model because the mission required several drones working together instead of just one. A single drone would not have been efficient or reliable in such a large environment. With multiple agents, we could simulate how real autonomous systems operate independently while still coordinating to achieve the same goal.
2. We considered the vision range and accuracy of detection, how well the drones could avoid obstacles, the coordination needed to prevent overlap, the size of the environment, and the possibility of false detections.
3. These variables directly affect the results. Better vision and scanning mean faster detections, obstacle avoidance keeps drones on track, and coordination ensures that no

two drones waste time on the same target. All of these together make the mission more efficient and reliable.

4. We chose a modular system where each agent had a specific role: one for recognition, one for decision-making, and one for landing. This separation made the project easier to manage, easier to debug, and closer to how real-world systems are designed.
5. The final solution was reliable and realistic. The drones were able to scan, detect, and land near the target without manual input. Central coordination helped keep everything organized, and the interface made it easy to follow the progress of the mission.
6. One disadvantage was that the system sometimes failed when conditions were not ideal, since the vision system was simplified and could make mistakes. Another limitation was that we could not fully implement advanced AI recognition in the short time we had.
7. To improve the system, the drones could use more dynamic search patterns, changing position and angle to cover blind spots. In the future, real AI-based recognition could be added to make detections more accurate and reliable in complex situations.
8. At the beginning of the block, I expected we would work with something closer to real AI, with natural language commands and advanced recognition. As the project progressed, I learned how important it is to build a strong foundation first and then add complexity gradually. I also learned how design diagrams connect to the actual implementation in Unity, and how much testing and adjusting is required. My biggest lesson was understanding the difference between theory and practice, and how teamwork and iteration are essential to making the system work.

9.3 - Baltazar Servín Riveroll

1. The mission naturally needed entities that could operate independently and autonomously, capable of analyzing the world and reaching a plan to satisfy their objective, which is why this MAS system was perfect to simulate the communication between drones and the situations they would encounter.
2. Vision range, scanning accuracy, obstacle detection and avoidance and target assignment; all these variables were needed to simulate a real life situation and not a perfect world where everything works perfectly.
3. Obstacle detection and avoidance make the drones pick in real time a different path to their destination, the communication between drones prevent drones from being assigned the same objective and the characteristics of the vision system result in accuracy levels and efficacy of the object detection.
4. We chose the modular architecture to keep every system of the simulation separated so debugging would be easier and not create incompatibilities.
5. It resulted in a very reliable and realistic simulation of drones using computer vision to identify targets as well as to avoid obstacles and choose other routes.
6. As it could happen in the real world, the computer vision cannot see through objects so if a person is behind a tree the drone will not be able to see it.
7. After a certain time has passed without a target the drone could start a seeking pattern in the sky to have different angles and find the person of interest.
8. The main difference I can find with the expectations at the beginning of the block and now is the use of AI, the project was designed to implement real computer vision processed in a different server to locate person using natural language prompts like “find the person with the orange vest”, we can see now that it was not possible due to the short time and vague subject approach of the block.

9.4 - Emilio Pardo Gutiérrez

1. A multi-agent model framework was favored for this type of task over a more traditional one because we tried to ensure that the drones have complete autonomy without losing reliability. We figured that having autonomous agents, each specialized in a task, would make up for a more efficient and error-proof solution.
2. The key variables considered to make the simulation not only prioritized the safety of the target but also the integrity of the drone were vision range, scanning accuracy, obstacle detection, avoidance, and target assignment.
3. Obstacle detection and avoidance makes it possible for the drone to perform the given task without putting its integrity at stake, but it would not be possible without the variables related to the recognition, such as vision range and scanning accuracy, which are also essential for the drone to identify the assigned target.
4. The modular architecture has great qualities such as scalability, compatibility and ease of maintenance for the system, making it a natural choice for our project.
5. One of the greatest advantages of our solution is that not only does it have a high success rate; it also is realistic and very close to the systems used in real-life drones.
6. Just like its real-life counterpart, without a clear field of vision, it won't be able to identify the target.
7. One solution that came to mind during development is that if this was a real-world mission and the drone were struggling to find the target, it could use alternative methods to vision, like sound or movement detection.
8. When starting the project I was not really sure what to expect, but had the idea that it would be way closer to the knowledge I already have about autonomous systems, like generative AI; I was gladly surprised that it was much simpler than that, that way we

had a chance to really get involved with the design of the system. I think that made me learn a lot from the related topics.

9.5 - Jozef David Hernández Campos

1. Because it naturally fits the real-world scenario of multiple autonomous drones operating independently to find and assist different people. Each drone acts as an independent agent with its own vision system, decision-making capabilities, and target assignment, which mirrors how actual real systems work.
2. The key variables considered were: vision range and angle for realistic detection capabilities, scanning speed for efficient area coverage, obstacle detection layers, target assignment coordination to prevent conflicts between agents. These variables directly impact the system's effectiveness and add realism.
3. Vision range and scanning speed determine how quickly drones find targets, while vision angle affects detection accuracy around obstacles. The coordination system prevents multiple drones from targeting the same person, maximizing efficiency.
4. We chose a modular architecture to ensure maintainability, scalability, and clear separation.
5. The solution provides realistic behavior with vision based detection, intelligent obstacle handling and automatic coordination between agents to prevent conflicts.
6. The main disadvantage of our solution is that, like in real world environments, vision can fail due to obstacles, leading to incomplete target detection.
7. We could implement dynamic movement patterns during the searching phase where drones actively navigate to different elevated positions and patrol routes to maximize area coverage, rather than staying stationary while rotating.
8. At the beginning of this block, I thought we would be building complex communication systems where drones would constantly talk to each other and to

ground control stations. I was expecting to implement sophisticated protocols and maybe even some AI decision-making algorithms between multiple agents. What I actually ended up working on was making individual drones that could "see" and navigate properly. Most of my time went into figuring out how to make the vision system work with Unity's raycast system, dealing with obstacles blocking the drone's view, and making sure the UI actually displayed the right information.

9.6 - Maria José Medina Calderón

1. We chose MAS systems because the mission needed a multi-agent model that could operate independently and autonomously, having each its own vision system, decision-making capabilities, and target assignment. Also, the simulation must mirror how real systems work.
2. The variables considered were: vision range, scanning accuracy, obstacle detection and avoidance, and target assignment.
3. Vision range and scanning speed determine how quickly drones find targets, obstacle detection and avoidance make the drones pick in real time a different path to their destination, the coordination system prevents multiple drones from targeting the same person, maximizing efficiency.
4. We chose the modular architecture to ensure every system of the simulation is separated, so debugging would be easier and will not create incompatibilities.
5. The advantages of the final solution are that it give us a realistic behavior with vision based detection, it can avoid obstacles and choose other routes.
6. The disadvantages of the solution are: The computer vision can not see through objects, leading to possible incomplete target detection.

7. To reduce or eliminate the mentioned disadvantage we could implement dynamic movement patterns, when searching from the target the drone will change position and angle until the target is found.
8. The difference from my expectations are that I thought I will be using IA for the analysis and detection of targets, as it was initially described to us. We got to the point that we had to change the project since it was not possible to do everything we wanted to achieve in the 5 weeks and the vague subject approach of the block.