Tecnológico de Monterrey

Evidencia 2-Review 3

Eugenio Loeza | A01067846

Diego Angulo | A01643797

Gael Castillo Zepeda | A01638638

Diego Alejandro Ibarra Flores | A01644350

21 de Agosto del 2025

Modelación de sistemas multiagentes con gráficas computacionales

Ivan Axel Dounce Nava

Repositorio de GitHub

1. Descripción del Reto

El reto consiste en desarrollar una simulación en la que un micro vehículo aéreo (MAV) debe ejecutar de manera totalmente autónoma, una misión de búsqueda y localización de una persona a partir de una descripción textual de una persona, por ejemplo: “Encuentra a la persona con chaqueta naranja y casco amarillo”, y una posición GPS donde es probable que se encuentre la persona. En la simulación el MAV debe volar desde una zona designada o zona de despegue, hasta la posición GPS objetivo, situada al menos 150 metros de distancia.

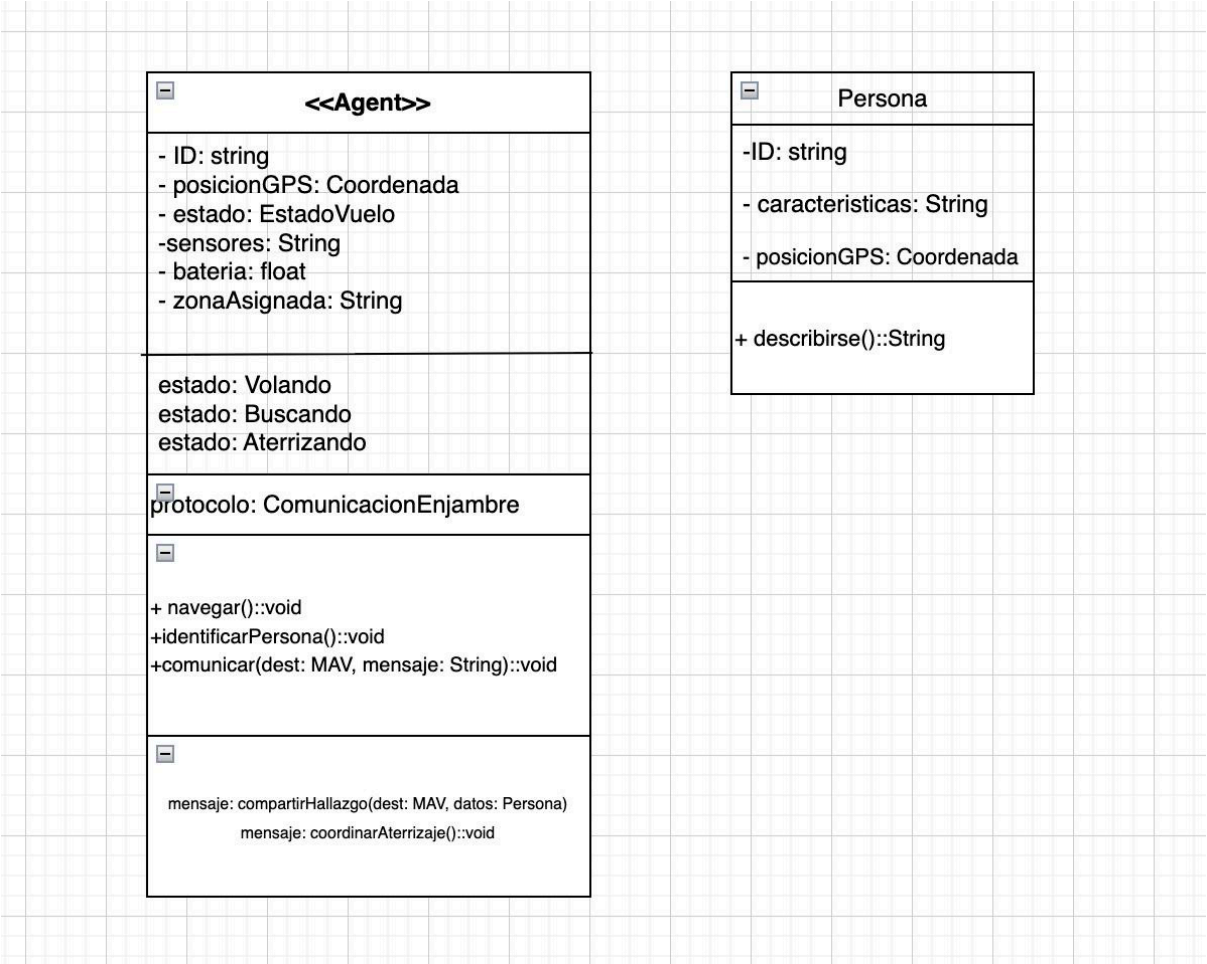
Una vez en el área, y dentro de un radio de 20 metros alrededor de la posición GPS, el MAV deberá identificar correctamente a la persona que cumpla con la descripción textual proporcionada, aun cuando haya otras personas presentes con características distintas. Finalmente, la simulación debe contemplar que el MAV realice un aterrizaje seguro a un radio o distancia de 2 metros de la persona, sin golpearlo. El objetivo principal del reto es que el MAV logre completar de forma autónoma el desplazamiento, la identificación y el aterrizaje bajo los criterios establecidos.

2. Identificación de los agentes involucrados

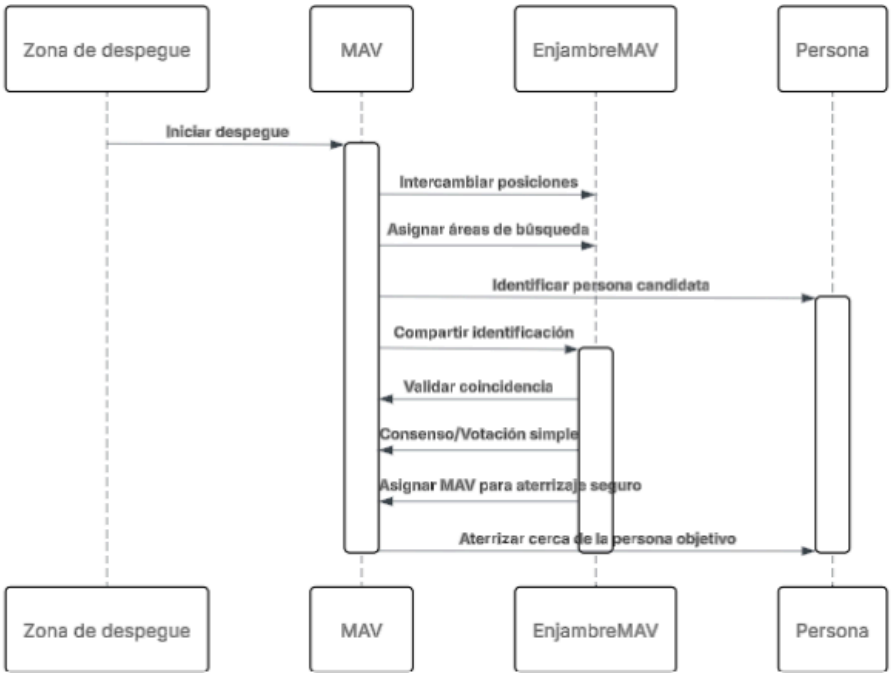
En la simulación propuesta se consideran los siguientes agentes autónomos:

- **Agente MAV (1-3):** Son 3 drones cada uno funciona como un agente autónomo con la capacidad para comunicarse, navegar y tomar decisiones. Estos tienen la capacidad de compartir información de búsqueda, en caso de encontrar a la persona y a la vez puede negociar entre ellos respecto a quien busca por cual zona, quien aterriza, etc.
- **Agente persona:** Es la representación de la descripción textual, y su rol es guiar a los agentes MAV a tomar decisiones. Es un objetivo de búsqueda el cual los drones deben identificar y aproximarse de manera segura.
- **Agentes distractores:** Otras entidades simuladas que dificultan la identificación y obligan al MAV a validar correctamente la descripción.
- **Agente control en tierra:** Supervisión mínima, sólo para monitoreo de métricas de desempeño del enjambre de MAVs.

2.1. Primera iteración del diagrama de Clases de Agentes presentando los diferentes agentes involucrados



2.2. Primera iteración del diagrama de protocolo de Interacción de Agentes.



3. Plan de trabajo inicial y aprendizaje adquirido

Actividad	Responsables	Fecha estimada	Esfuerzo	Estado
Creación de repositorio GitHub	Eugenio	21 de agosto	10 min	pendiente
Diseño de diagramas UML	Eugenio y Gael	21 de agosto	30 min	pendiente
Desarrollo de prototipo de navegación autónoma	Diego A y Diego I	30 de Agosto	3 horas	pendiente
Implementación de protocolos de comunicación entre MAVs	Todo el equipo	5 de Septiembre	8 horas	pendiente
Prueba inicial de simulación	Todo el equipo	10 de Septiembre	3 horas	pendiente
Revisión y ajuste final	Todo el equipo	15 de Septiembre	3 horas	pendiente

- **Eugenio Loeza:** Experiencia en programación orientada a objetos y estructuras de datos en coordinación y diseño de arquitecturas. Área de oportunidad,: optimización de simulaciones en tiempo real
- **Diego Angulo:** Habilidades en modelación matemáti*ca.* Fortaleza en planeación de trayectorias. Área de oportunidad: implementación en código.
- **Gael Castillo:** Experiencia en trabajo con agentes y comunicación. Fortaleza en diseño de protocolos de interacción. Área de oportunidad: documentación.
- **Diego Ibarra:** Experiencia en programación y optimización fortaleza, en algoritmos de búsqueda. Área de oportunidad: Aumentar habilidades en entornos gráficos (Unity).

Expectativas y compromisos del equipo:

- Expectativas:

- Desarrollar un prototipo funcional de simulación multiagente
- Aprender a modelar, interacciones colaborativa entre MAVs
- Integrar buenas prácticas de trabajo con GitHub

- Compromisos:
  - Mantener comunicación constante
  - Actualizar el repositorio después de cada avance individual
  - Respetar tiempos y responsabilidades asignadas

4. Breve descripción de cómo los MAVs recibirán una instrucción y la interpretarán.

El MAV debe recibir una orden desde el control en tierra. Por ejemplo “Busca a la persona que tiene chaqueta azul y casco negro en este punto del GPS”. Después de esto el dron lee el GPS y ya con esto sabe hacia donde debe de volar. El dron traduce la descripción textual en atributos para que su cámara pueda detectar, los colores y las prendas de la persona indicada. Cada uno de los drones se reparte en zonas para no tener que buscar todos en el mismo lugar.

De esta manera se asegura que cada MAV entienda claramente la misión y que el trabajo en equipo sea más eficiente.

5. Breve descripción de la estrategia principal para que los MAVs naveguen a una ubicación GPS específica.

El MAV genera una ruta para llegar desde la zona de despegue hasta el punto GPS indicado. Durante el trayecto sigue waypoints intermedios que lo guían en la dirección correcta y ajusta su camino en caso de obstáculos. Una vez que llega al área de búsqueda, el dron comienza un patrón de vuelo en zig-zag o de barrido para cubrir toda la zona asignada y asegurarse de no dejar espacios sin revisar. Esto permite que los drones exploren el área de forma ordenada y sin dejar puntos ciegos.

6. Breve descripción de la estrategia principal para que los MAVs identifiquen características específicas de una persona (suponga que tiene una cámara con IA "imperfecta" (es decir, hay una pequeña probabilidad de reconocimiento erróneo), una vista desde arriba y una cierta altitud de vuelo).

El MAV utiliza su cámara con inteligencia artificial para comparar lo que ve con la descripción dada. Por ejemplo, si debe encontrar a una persona con casco negro y chaqueta azul, busca esos atributos en las imágenes que captura. Como la IA no es perfecta y puede equivocarse, el dron repite el análisis varias veces para aumentar la confianza y, si sigue en duda, pide a otro dron que confirme la detección. De esta forma, evitan confundir a la persona objetivo con los distractores. Con este proceso se aumenta la precisión del reconocimiento y se reduce el riesgo de error en la misión.

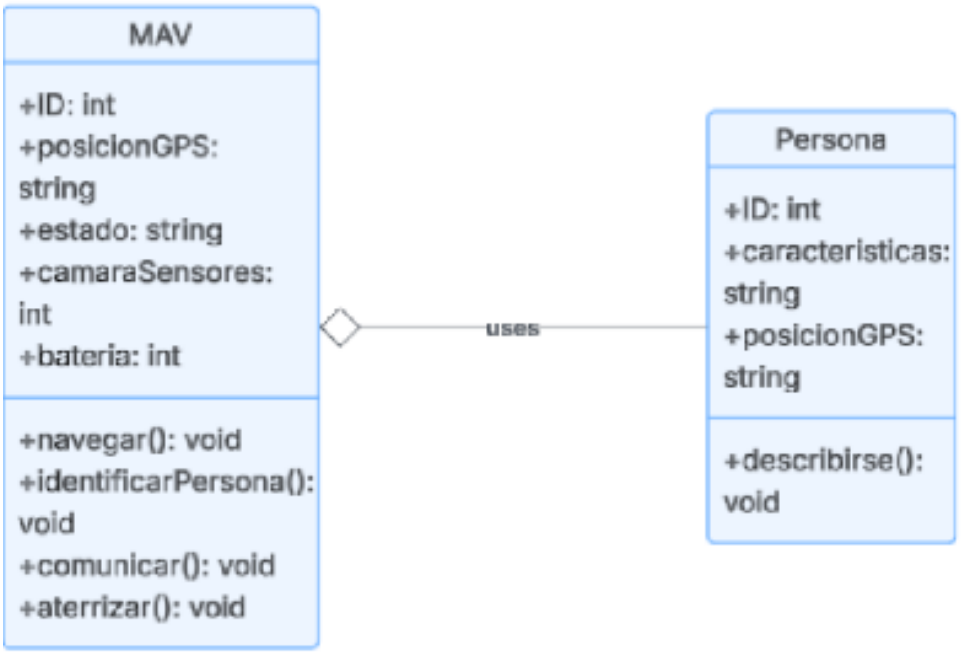
7. Breve descripción de la estrategia principal para que los MAVs realicen el aterrizaje cerca de la persona.

Una vez que el MAV confirma a la persona objetivo, comienza una aproximación controlada. Se mantiene a una altura segura mientras calcula la trayectoria de descenso. Luego baja poco a poco hasta quedar a una distancia máxima de 2 metros de la persona, cuidando siempre de no golpearla. Si detecta algún riesgo durante el descenso, como estar demasiado cerca o perder de vista al objetivo, interrumpe el aterrizaje y lo intenta de nuevo de manera más segura. Así se garantiza un aterrizaje seguro que cumple con la condición de estar cerca de la persona sin ponerla en peligro.

8. Breve descripción de la estrategia principal para que los MAVs se comuniquen y colaboren para encontrar a una persona.

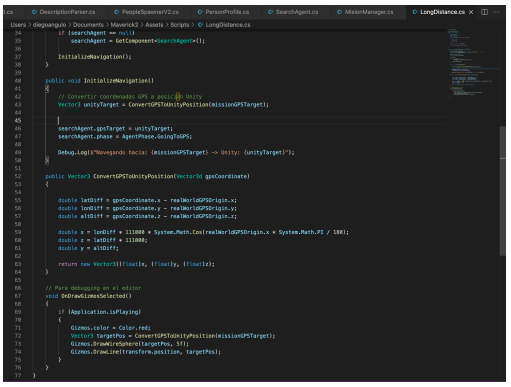
Los MAVs se comunican desde el inicio para repartirse las zonas de búsqueda. Cada uno le dice al otro qué sector cubrirá para no duplicar esfuerzos. Cuando un MAV cree haber encontrado a la persona, envía un reporte de detección a los demás. Otro dron puede acercarse para confirmar la información y evitar errores. Además, si dos drones quieren aterrizar al mismo tiempo, se comunican para ver cuál tiene prioridad según quién esté más cerca o tenga mayor confianza en la detección. Si uno se queda sin batería o falla, otro cubre su sector para que el objetivo se finalice con éxito. Con esta colaboración, el equipo de MAVs asegura que la misión no dependa de un solo dron y se complete con mayor rapidez y fiabilidad.

9. Diagrama UML de la simulación y sus elementos



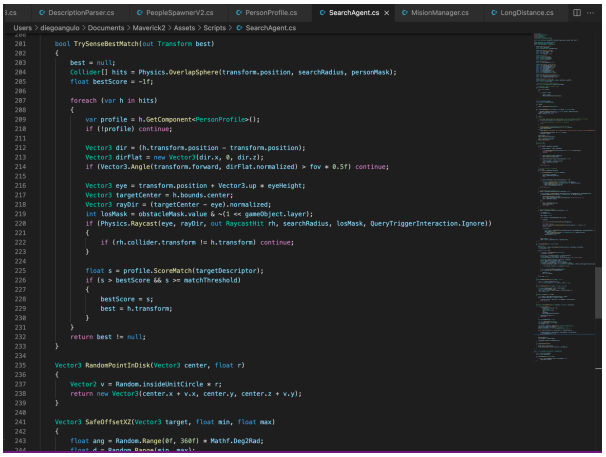
10. Código para implementar los agentes

Navegación a coordenadas

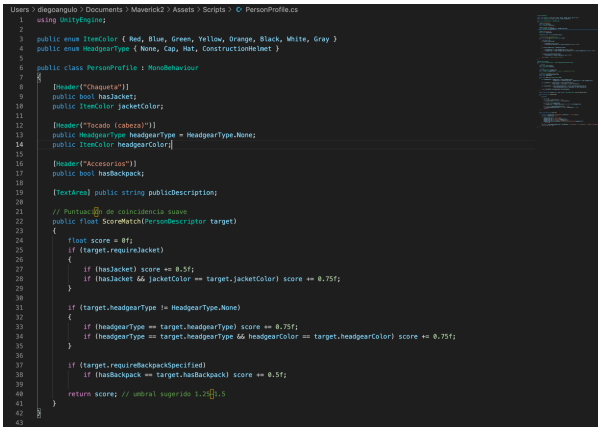


```
        if (searchAgent == null)
            searchAgent = GetComponent<SearchAgent>();

        InitializeNavigation();
    }

    public void InitializeNavigation()
    {
        // Convertir coordenadas GPS a posición Unity
        Vector3 unityTarget = ConvertGPSToUnityPosition(missionGPSTarget);

        // Configurar el agente
        searchAgent.gpsTarget = unityTarget;
        searchAgent.phase = AgentPhase.GoingToGPS;

        Debug.Log($"Navegando hacia: {missionGPSTarget} -> Unity: {unityTarget}");
    }

    public Vector3 ConvertGPSToUnityPosition(Vector3d gpsCoordinate)
    {
        // Diferencias desde el origen
        double latDiff = gpsCoordinate.x - realWorldGPSOrigin.x;
        double lonDiff = gpsCoordinate.y - realWorldGPSOrigin.y;
        double altDiff = gpsCoordinate.z - realWorldGPSOrigin.z;

        // Conversión a metros (aproximada)
        double x = lonDiff * 111000 * System.Math.Cos(realWorldGPSOrigin.x * System.Math.PI / 180);
        double z = latDiff * 111000;
        double y = altDiff;

        return new Vector3((float)x, (float)y, (float)z);
    }

    // Para debugging en el editor
    void OnDrawGizmosSelected()
    {
        if (Application.isPlaying)
        {
            Gizmos.color = Color.red;
            Vector3 targetPos = ConvertGPSToUnityPosition(missionGPSTarget);
            Gizmos.DrawWireSphere(targetPos, 5f);
            Gizmos.DrawLine(transform.position, targetPos);
        }
    }
}
```

Percepción y reconocimiento

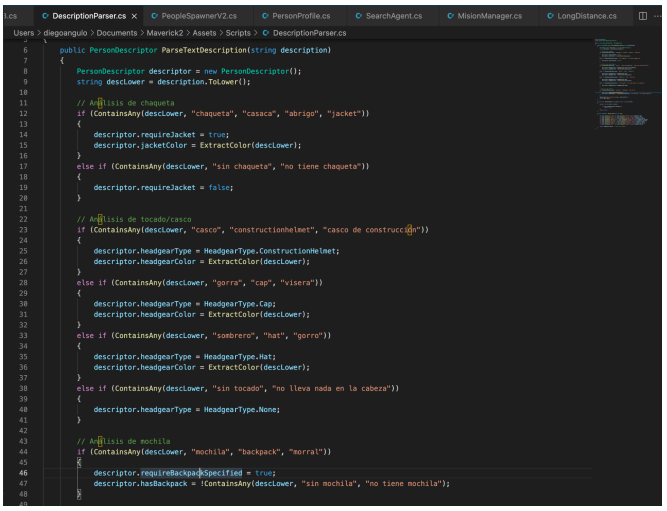


```
bool TrySenseBestMatch(out Transform best)
{
    best = null;
    Collider[] hits = Physics.OverlapSphere(transform.position, searchRadius, personMask);
    float bestScore = -1f;

    foreach (var h in hits)
    {
        var profile = h.GetComponent<PersonProfile>();
        if (!profile) continue;

        Vector3 dir = (h.transform.position - transform.position);
        Vector3 dirFlat = new Vector3(dir.x, 0, dir.z);
        if (Vector3.Angle(transform.forward, dirFlat.normalized) > fov * 0.5f) continue;

        Vector3 eye = transform.position + Vector3.up * eyeHeight;
        Vector3 targetCenter = h.bounds.center;
        Vector3 rayDir = (targetCenter - eye).normalized;
        int losMask = obstacleMask.value & ~(1 << gameObject.layer);
        if (Physics.Raycast(eye, rayDir, out RaycastHit rh, searchRadius, losMask, QueryTriggerInteraction.Ignore))
        {
            if (rh.collider.transform != h.transform) continue;
        }

        float s = profile.ScoreMatch(targetDescriptor);
        if (s > bestScore && s >= matchThreshold)
        {
            bestScore = s;
            best = h.transform;
        }
    }
    return best != null;
}

Vector3 RandomPointInDisk(Vector3 center, float r)
{
    Vector2 v = Random.insideUnitCircle * r;
    return new Vector3(center.x + v.x, center.y, center.z + v.y);
}

Vector3 SafeOffsetXZ(Vector3 target, float min, float max)
{
    float ang = Random.Range(0f, 360f) * Mathf.Deg2Rad;
```



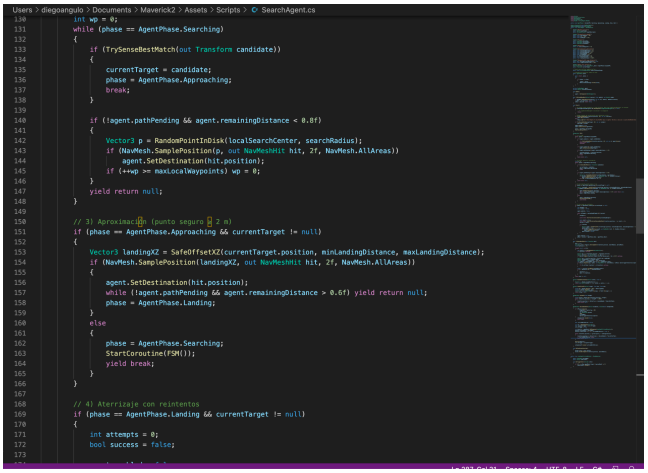
```
using UnityEngine;

public enum ItemColor { Red, Blue, Green, Yellow, Orange, Black, White, Gray }
public enum HeadgearType { None, Cap, Hat, ConstructionHelmet }

public class PersonProfile : MonoBehaviour
{
    [Header("Chaqueta")]
    public bool hasJacket;
    public ItemColor jacketColor;

    [Header("Tocado (cabeza)")]
    public HeadgearType headgearType = HeadgearType.None;
    public ItemColor headgearColor;

    [Header("Accesorios")]
    public bool hasBackpack;

    [TextArea] public string publicDescription;

    // Puntuación de coincidencia suave
    public float ScoreMatch(PersonDescriptor target)
    {
        float score = 0f;
        if (target.requireJacket)
        {
            if (hasJacket) score += 0.5f;
            if (hasJacket && jacketColor == target.jacketColor) score += 0.75f;
        }

        if (target.headgearType != HeadgearType.None)
        {
            if (headgearType == target.headgearType) score += 0.75f;
            if (headgearType == target.headgearType && headgearColor == target.headgearColor) score += 0.75f;
        }

        if (target.requireBackpackSpecified)
            if (hasBackpack == target.hasBackpack) score += 0.5f;

        return score; // umbral sugerido 1.25–1.5
    }
}
```

Procesamiento de descripción

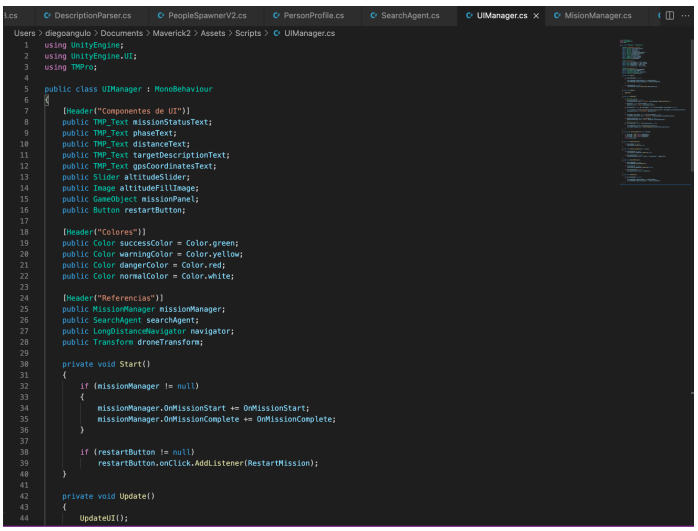


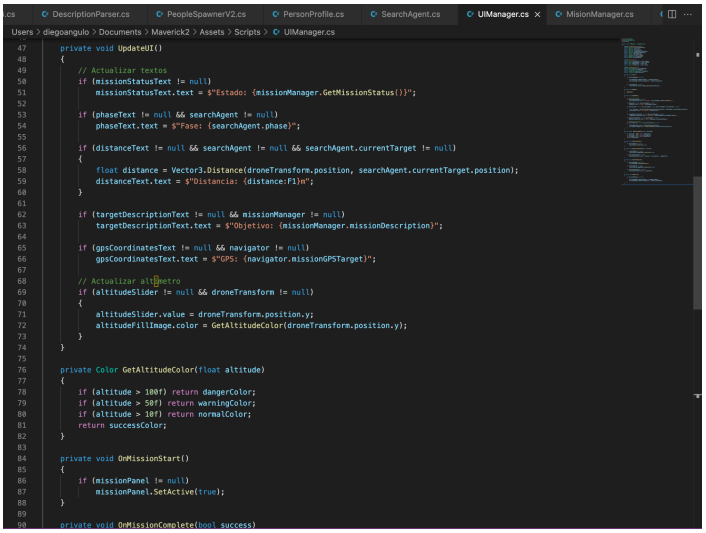
```
public PersonDescriptor ParseTextDescription(string description)
{
    PersonDescriptor descriptor = new PersonDescriptor();
    string descLower = description.ToLower();

    // Análisis de chaqueta
    if (ContainsAny(descLower, "chaqueta", "casaca", "abrigo", "jacket"))
    {
        descriptor.requireJacket = true;
        descriptor.jacketColor = ExtractColor(descLower);
    }
    else if (ContainsAny(descLower, "sin chaqueta", "no tiene chaqueta"))
    {
        descriptor.requireJacket = false;
    }

    // Análisis de tocado/casco
    if (ContainsAny(descLower, "casco", "constructionhelmet", "casco de construcción"))
    {
        descriptor.headgearType = HeadgearType.ConstructionHelmet;
        descriptor.headgearColor = ExtractColor(descLower);
    }
    else if (ContainsAny(descLower, "gorra", "cap", "visera"))
    {
        descriptor.headgearType = HeadgearType.Cap;
        descriptor.headgearColor = ExtractColor(descLower);
    }
    else if (ContainsAny(descLower, "sombrero", "hat", "gorro"))
    {
        descriptor.headgearType = HeadgearType.Hat;
        descriptor.headgearColor = ExtractColor(descLower);
    }
    else if (ContainsAny(descLower, "sin tocado", "no lleva nada en la cabeza"))
    {
        descriptor.headgearType = HeadgearType.None;
    }

    // Análisis de mochila
    if (ContainsAny(descLower, "mochila", "backpack", "morral"))
    {
        descriptor.requireBackpackSpecified = true;
        descriptor.hasBackpack = !ContainsAny(descLower, "sin mochila", "no tiene mochila");
    }
```

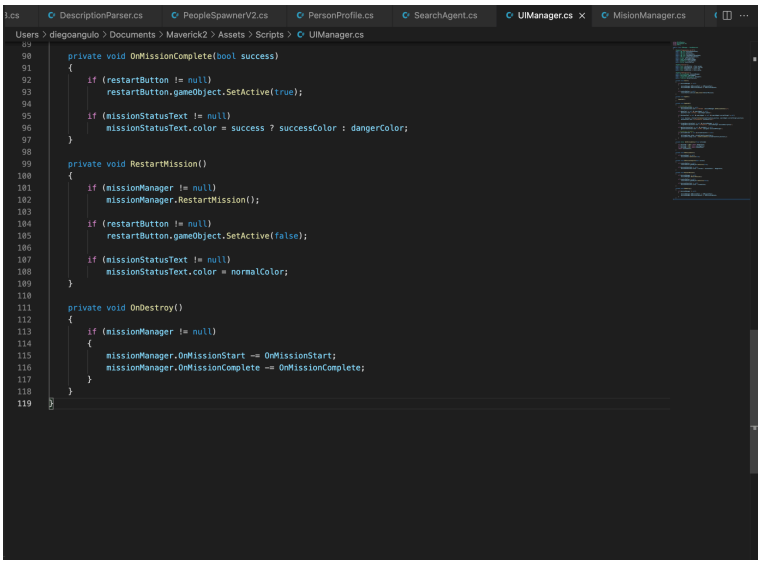
Toma de decisiones y patron de busqueda



11. Código para implementar la parte gráfica de la simulación







12. Plan de trabajo actual

Actividad	Responsables	Fecha estimada	Esfuerzo	Estado
Creación de repositorio GitHub	Eugenio	21 de agosto	5 min	Completada
Diseño de diagramas UML	Eugenio y Gael	21 de agosto	2 horas	Completada
Desarrollo de prototipo de navegación autónoma	Diego A y Diego I	31 de Agosto	1 hora	Completada
Desarrollo e implementación de lógica del prototipo de persona	Gael y Eugenio	1 de Septiembre	3 horas	Completada
Implementación del movimiento del prototipo de navegación	Todo el equipo	4 de Septiembre	4 horas	Completada
Implementación de lógica de colisiones	Diego I y Gael	8 de Septiembre	3 horas	En proceso
Implementación de lógica de verificación de personas	Todo el equipo	8 de Septiembre	8 horas	En proceso
Implementación de protocolos de comunicación entre MAVs	Todo el equipo	10 de Septiembre	8 horas	Pendiente
Aterrizaje autónomo con precisión	Todo el equipo	10 de Septiembre	4 horas	Pendiente
Prueba inicial de simulación	Todo el equipo	10 de Septiembre	3 horas	Pendiente
Revisión y ajuste final	Todo el equipo	15 de Septiembre	3 horas	Pendiente

Aprendizaje: Durante este proyecto, como equipo aprendimos a usar mejor las herramientas y los recursos visuales de Unity, a organizar el trabajo en módulos y etapas claras, y a dividir problemas complejos en partes simples y abordables o solucionables. También fortalecimos el pensamiento lógico durante la implementación, y comprobamos la complejidad de ejecutar un proyecto realista con márgenes de seguridad, lo que supuso un gran reto para convertir una idea en una solución funcional.