Reporte Técnico - Reto - TC2008B - Avance 1

Modelación de sistemas multiagentes con gráficas computacionales
(Gpo 605)

Profesores:
Dr. Luis Alberto Muñoz Ubando
Dr. Raúl V. Ramírez Velarde

Equipo 5

Emilio Barragán	**A01286583**
Jordy Granados	**A00998753**
Humberto Cañedo	**A01255389**
Diego Saldaña	**A01571609**
Angel Peña	**A00838418**

Noviembre 24, 2025

Índice

1. Introducción y descripción del reto a desarrollar

Nuestro reto consiste en simular varios tractores operando simultáneamente dentro de un campo agrícola, donde cada uno debe recorrer rutas eficientes sin generar sobrelapados en su trabajo. Los tractores son representados mediante un sistema multiagente en Python a través de una arquitectura que permite la creación, registro y análisis de los datos obtenidos en la simulación.

La simulación está diseñada para integrarse próximamente con Unity 3D, buscando un entorno visual más completo, dinámico y real. La simulación debe modelar:

- Rutas óptimas para la siembra y aplicación de herbicida
- Evitar el sobrelapado en zonas ya plantadas o fumigadas
- Mantener la separación correcta entre semillas
- Evitar colisiones entre tractores

Con este reto de John Deere, se busca reducir el desperdicio de recurso y materia prima, además de aumentar la eficiencia en la siembra y control de maleza en los campos agrícolas. Al implementar sistemas y tecnologías como Python, Flocking Behaviour, Swarm Intelligence y un planificador de rutas usando el algoritmo A*, se busca mejorar y facilitar las condiciones de trabajo de los trabajadores agrícolas, optimizando el rendimiento de los tractores y aumentar la precisión de las labores realizadas.

2. Créditos

Integrantes del equipo y roles:

- Emilio Barragán - Programador, comunicación SMA-Unity
- Jordy Granados - Programador, Implementación de algoritmos y simulación
- Humberto Cañedo - Programador, Integración con Unity y diseñador 3D
- Diego Saldaña - Programador, Diseñador de arquitectura y documentación técnica
- Angel Peña - Programador, Diseñador de comportamiento y optimización

3. Contexto y Problema

El reto con John Deere establece simular tractores autónomos capaces de realizar al mismo tiempo tareas de siembra y/o fumigación mientras se evitan zonas previamente sembradas y se mantienen condiciones óptimas de la operación como la distancia correcta entre semillas, la ausencia de colisiones con otros tractores y, posiblemente, compensar curvas cuando el terreno no es recto, lo cual aseguraría una cobertura eficiente del terreno con la mínima pérdida posible.

La resolución de este problema tiene aplicaciones prácticas pues en la agricultura moderna, ayuda a reducir costos operativos, evitar desperdicios y mejorar el rendimiento de los cultivos.

Dicho lo anterior, la simulación que realizaremos modelará el comportamiento de varias máquinas autónomas (agentes) trabajando de manera coordinada dentro del mismo campo, permitiendo

analizar el impacto de la planeación, la interacción entre agentes, los protocolos de comunicación y la dinámica colectiva que resulte de este comportamiento.

4. Objetivos generales

Los objetivos de este proyecto son los siguientes:

1. Simular el movimiento coordinado de 4 o más tractores autónomos en un entorno agrícola, evaluando cómo distintas configuraciones de navegación afectan la eficiencia, la cobertura del terreno y la interacción entre agentes, evitando colisiones y sobrelapado.
2. Implementar y experimentar con diferentes variantes del algoritmo A* para generar rutas óptimas hacia objetivos predefinidos, comparando heurísticas, costos de movimiento y estrategias de planeación para identificar cuáles producen mejores resultados en términos de eficiencia y estabilidad del sistema.
3. Modelar un sistema multiagente funcional basado en Swarm Inteligencia y Flocking Behaviour, ajustando parámetros como distancia mínima, reglas de separación, alineación, cohesión y percepción del entorno, con el objetivo de analizar su impacto en la coordinación y desempeño colectivo de los tractores.
4. Probar, comparar y evaluar diferentes configuraciones de siembra y cobertura del campo, determinando cuáles reducen el solapamiento, mejoran la distribución del recorrido y optimizan el uso del espacio agrícola.
5. Medir métricas de desempeño clave, incluyendo:
    a. Número de celdas visitadas
    b. Porcentaje de solapamiento
    c. Distancia total recorrida
    d. Número de replaneaciones
    e. Tiempo total de operaciones
    f. Eventos de proximidad o riesgo de colisión
6. Integrar la simulación con Unity 3D para generar una representación visual clara y dinámica que permita analizar de manera comparativa el comportamiento de los agentes bajo distintos parámetros y configuraciones.

Consideramos que cada uno de estos objetivos son específicos, medibles, alcanzables, relevantes y acotados al alcance de este proyecto, asimismo, está orientado a la investigación y evaluación sistemática del comportamiento de los tractores autónomos en un escenario agrícola simulado.

5. Herramientas de trabajo colaborativo

Para el presente proyecto, nos encontramos utilizando tres herramientas de trabajo colaborativo. La primera de ella es GitHub, donde se encuentra nuestro repositorio principal de código. El link a dicho repositorio es el siguiente:

https://github.com/TheGhostJB/Equipo5_Multiagentes

Adicionalmente, para la presentación que en la que estaremos trabajando y perfeccionando en cada iteración de nuestros sprints, nos encontramos trabajando con canva, bajo la siguiente dirección:

https://www.canva.com/design/DAG5ceFvkUU/9M2Z9eTAEug5KVq3gCqp_g/edit

Por último, para el diagramado y la generación de UMLs, utilizamos la herramienta LucidChart, y el link específico del proyecto se encuentra aquí:

https://lucid.app/lucidchart/74349ec8-93fa-45f5-862e-d9561b7302cd/edit?viewport_loc=-11%2C-11%2C2149%2C1112%2C0_0&invitationId=inv_ffb8d140-f611-4648-88ed-f6933649d1d0

6. Restricciones

El sistema presenta las siguientes restricciones:

- El espacio es un grid discreto (matriz de celdas) con límites fijos.
- Los tractores no pueden ocupar la misma celda simultáneamente.
- Los tractores no pueden moverse fuera del terreno modelado.
- Algunos tractores serán estáticos o tendrán movimiento aleatorio limitado.
- El planificador A* solo puede moverse en 8 direcciones.
- No se realizan curvas libres; el movimiento es celda por celda.
- El algoritmo A* debe recalcular la ruta cuando detecta un bloqueo o nuevo obstáculo.
- El sistema debe ejecutarse en tiempo real para sincronizarse con Unity.

7. Historias de usuario

Al no ser un producto como una aplicación web o móvil, las historias de usuario serán específicas del producto y la simulación que le pueda ser de utilidad al Socio Formador.

HU01 - Como usuario quiero poder visualizar y planear rutas eficientes para que un tractor llegue a su objetivo, evitando sobrelapado y minimizando distancia total.

HU02 - Como usuario quiero que los caminos que tomen los tractores eviten colisiones o choques entre otros tractores.

HU03 - Como usuario quiero ver la simulación de Unity en 3D para visualizar y cotizar el movimiento y comportamiento de los tractores.

HU04 - Como usuario quiero que se genere un reporte o resumen con los datos de un tractor.

HU05 - Como usuario quiero que se genere un reporte o resumen con los datos de todos los tractores y el campo en general.

HU06 - Como usuario quiero modificar parámetros de los tractores para ajustar la simulación y configuraciones.

HU07 - Como usuario, quiero que el sistema coordine simultáneamente las tareas de varios tractores, para evitar conflictos en áreas compartidas.

HU08 - Como usuario, quiero que cada tractor/agente detecte otros agentes y obstáculos, para reaccionar a su entorno de forma autónoma.

HU09 - Como usuario, quiero modificar la vista del entorno y espacio para obtener una representación más clara o diferente del escenario.

HU10 - Como usuario, quiero poder pausar, continuar y ajustar la simulación para analizar momentos específicos.

8. Descripción del sistema Multiagente

Como se menciona anteriormente, los agentes ocupan el lugar de los tractores y actúan en el entorno, el cual es el campo donde hay obstáculos como otros tractores, pero también están las tareas principales, como el control de profundidad y evitar sobrelapado. Los tractores tienen la capacidad de navegar, entender el entorno y tomar decisiones en base al mismo. El comportamiento de los agentes está basado en un sistema de comunicación de Swarm Intelligence, en el que cada agente sigue 4 reglas:

1. Moverse dentro del espacio delimitado y compartido.
2. Evitar Colisiones.
3. Respetar zonas ya trabajadas.
4. Cooperar implícitamente al no invadir áreas de otros tractores.

El tractor principal utiliza el algoritmo de A* para planear y alcanzar el objetivo, además de que replantea la ruta cuando aparece un obstáculo, los otros agentes o tractores influyen en la decisión del tractor principal buscando llegar a un “equilibrio”.

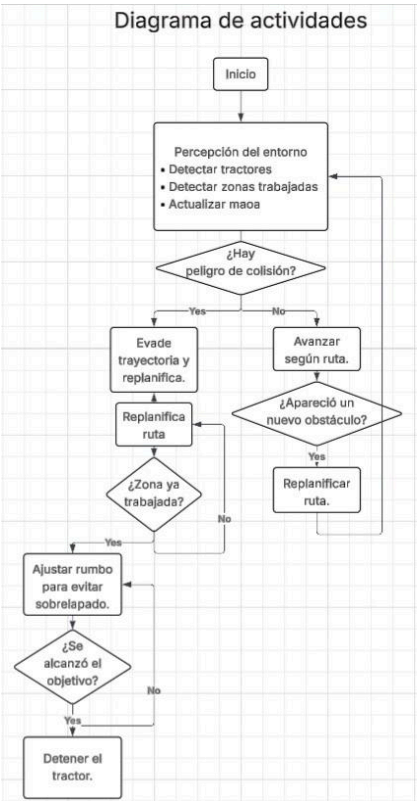

Diagrama 1.1 Diagrama de Actividades

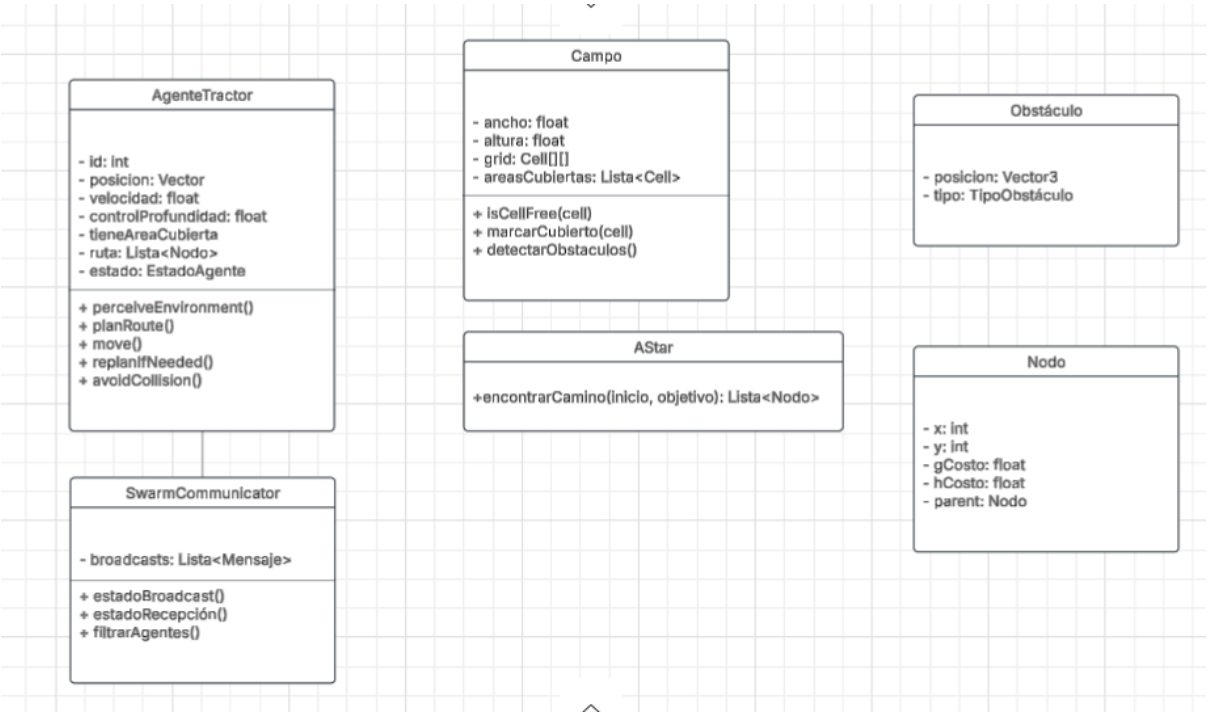

Diagrama 1.2 Diagrama de Clases del Modelo Multiagente

8.1 Modelo de los Agentes

1. **MainTractor (Agente Principal)**
    a. **Tipo:** Agente cognitivo reactivo
    b. **Rol:** Realiza la siembra o la fumigación optimizada
    c. **Características:**
        - **Creencias**
            - Mapa (Grid)
            - Posiciones de tractores cercanos
            - Celda destino y la ruta actual
            - Historial de celdas visitadas
        - **Planes**
            - Planear la ruta más óptima con A*
            - Adaptarse cuando exista un bloqueo
            - Evitar zonas previamente trabajadas
            - Evitar agentes cercanos
        - **Cooperación**
            - Se adapta a la posición de otros tractores
            - Coopera evitando áreas ya asignadas
        - **Aprendizaje**
            - Registro de métricas para hacer análisis después del trabajo

2. **ObstacleTractor (Agente Secundario)**
    a. **Tipo:** Agente reactivo simple
    b. **Rol:** Representa otros tractores operando en el campo como obstáculos
    c. **Características:**
        - Posibles estados: pueden estar estáticos o tener movimientos aleatorios
        - Influyen directamente en la ruta del agente principal
        - Simulan condiciones reales de cómo sería un campo agrícola con varios tractores

3. **TractorModel (Administrador del entorno)**
    a. **Tipo:** Agente estructural
    b. **Rol:** Maneja el espacio simulado y ayuda a coordinar la adaptación de los agentes
    c. **Funciones:**
        - Avanza la simulación paso a paso
        - Registra el historial de las posiciones de los tractores
        - Calcula las métricas como el solapamiento o la distancia recorrida

8.2 Modelo del Entorno

El entorno que se va a simular es el de un campo agrícola modelado como un grid discreto (una matriz NxM). Se caracteriza este grid por lo siguiente:

- **Parcialmente observable:** Cada tractor solo percibe su entorno
- **Determinista:** Mismas acciones -> mismos resultados
- **Secuencial:** Cada acción depende del estado previo
- **Dinámico:** Los agentes cambian el entorno constantemente
- **Discreto:** El espacio se divide en celdas
- **Multiagente:** Varios tractores interactúan simultáneamente

Manejo del tiempo

- La simulación avanza en steps (ticks)
- En cada tick:
    a. Se actualizan los agentes que son obstáculos
    b. El tractor principal evalúa su ruta
    c. Se adapta si es necesario

Modelado del espacio

El campo es un grid donde cada celda puede ser:

- Libre
- Visitada
- Visitada previamente (riesgo de sobrelapado)
- Ocupada por un tractor

Con estas características se facilita el uso del algoritmo A* y la detección de colisiones

8.3 Modelo de la Negociación

En este sistema, la negociación entre agentes se basa en una negociación implícita, a través del uso compartido del espacio, buscando llegar a un equilibrio asegurando que se use la mayor cantidad de suelo posible y, a su vez, evitando el sobrelapado.

- Cada agente cede su espacio si otro agente lo ocupa primero.
- El tractor principal replanifica su ruta si su camino no es atravesable.
- Los agentes respetan zonas ya trabajadas, lo cual evita sobrelapado y uso innecesario del recurso, ya sean semillas o herbicida.

Esta manera en la que trabajan los agentes representa un equilibrio similar al Equilibrio de Nash, donde ningún tractor puede mejorar su desempeño sin afectar a otro.

8.4 Modelo de la Interacción

La interacción entre el sistema multiagente y Unity se lleva a cabo mediante un canal de comunicación, API-REST/WEB Sockets, que envía en tiempo real las posiciones y estados de cada uno de los tractores.

Proceso

1. Python ejecuta el modelo multiagente.
2. En cada paso se genera un JSON con las coordenadas de todos los tractores, la ruta actual, las celdas visitadas y el estado de cada agente.
3. Unity recibe esta información constantemente, empaquetada en el archivo JSON, mediante uno de los dos métodos, los servicios REST o los WEB-SOCKETS, por ahora se planea hacer el desarrollo con WEB-SOCKETS dada su eficiencia para el envío de datos en tiempo real.
4. Unity actualiza la posición de cada tractor, la animación del movimiento, las rutas visuales y las zonas trabajadas.

9. Registro de prompts utilizados y modificaciones a los mismos

Prompt 1.

“Necesito investigar cómo los sistemas de simulación multiagentes pueden ayudar a resolver problemas de movilidad urbana y agricultura de precisión. Específicamente, quiero entender cómo técnicas como Swarm Intelligence y Flocking Behaviour se aplican a la coordinación de vehículos autónomos, qué ventajas ofrecen sobre sistemas centralizados, y qué ejemplos reales existen de implementaciones exitosas en agricultura autónoma. También necesito conocer qué métricas se usan típicamente para evaluar el desempeño de estos sistemas en términos de eficiencia operativa, prevención de colisiones y optimización de recursos.”

Modificación realizada al resultado del prompt 1

La IA proporcionó información general sobre sistemas multiagentes en contextos urbanos como gestión de tráfico y transporte público. Nosotros adaptamos y enfocamos esa información hacia el contexto específico de agricultura de precisión, extrayendo los principios aplicables a la coordinación de maquinaria agrícola autónoma y añadiendo referencias específicas a casos de uso en el sector de

agrotecnología y los desafíos particulares de campos agrícolas como terrenos irregulares y condiciones variables.

Prompt 2

"Necesito diseñar un sistema multiagente para simular tractores autónomos en un campo agrícola. Los tractores deben coordinar tareas de siembra y fumigación evitando colisiones entre ellos y sin sobrelapar zonas ya trabajadas. El entorno es un grid discreto de celdas y cada tractor debe poder navegar de forma autónoma. ¿Qué tipo de arquitectura de agentes me recomiendas y qué algoritmos de pathfinding y coordinación serían apropiados para este escenario?"

Modificación realizada al resultado del prompt 2

Inicialmente sugirió varias arquitecturas incluyendo sistemas centralizados con un controlador maestro. Sin embargo, nosotros modificamos el enfoque hacia un sistema más distribuido basado en Swarm Intelligence y Flocking Behaviour, donde cada agente toma decisiones locales basándose en sus vecinos cercanos, lo cual pensamos que es más conveniente para nuestro caso de uso.

Prompt 3

"Estoy implementando el algoritmo A en Python para planificar rutas de tractores en un campo agrícola representado como una matriz de celdas. Necesito que el algoritmo permita movimiento en 8 direcciones, considere otros tractores como obstáculos dinámicos y pueda replanificar la ruta cuando detecte bloqueos. ¿Podrías mostrarme una implementación base del A* que incluya la estructura de datos para el grid, la función heurística apropiada y el manejo de obstáculos dinámicos?"*

Modificación realizada al resultado del prompt 3

En el prompt 3, el código generado inicialmente usaba una heurística euclidiana simple. La adaptamos para incluir costos adicionales en celdas previamente visitadas, penalizando el sobrelapado, y agregamos lógica para priorizar rutas que mantuvieran la distancia óptima entre líneas de siembra según los requerimientos del socioformador.

Prompt 4

"Tengo una simulación de sistema multiagente desarrollada en Python que modela el comportamiento de varios tractores autónomos. Necesito integrar esta simulación con Unity 3D para visualización en tiempo real. ¿Cuál es la mejor forma de establecer comunicación entre Python y Unity? Considera que necesito enviar posiciones actualizadas de múltiples agentes cada frame y recibir información del estado de la simulación. ¿Debería usar Flask API, sockets, o alguna otra alternativa?"

Modificación realizada al resultado del prompt 4

Para este prompt, la IA recomendó usar Flask con peticiones HTTP. Nosotros optamos por implementar un sistema de comunicación más eficiente usando sockets para reducir latencia, ya que necesitábamos actualizar posiciones de 4+ tractores en tiempo real sin demoras perceptibles en la visualización de Unity.

Prompt 5

"*Crea un video explicativo de 3-5 minutos sobre la simulación de tractores agrícolas autónomos para profesionales que utilizan tecnología agrícola. El video debe cubrir: introducción al desafío de*

eficiencia en agricultura moderna, explicación del sistema multiagente con Swarm Intelligence y Flocking Behaviour para coordinación de tractores, demostración de navegación usando algoritmo A, y análisis de costos mostrando que los sistemas autónomos eléctricos son competitivos versus diésel tradicional. Tono profesional y técnico con visualizaciones de tractores en ambiente agrícola.*

Modificación realizada al resultado del prompt 5

El prompt original que generamos era extremadamente detallado y estructurado con tiempos específicos por sección, lo cual resultó demasiado rígido para la herramienta de generación de video. La IA intentó seguir literalmente cada punto con transiciones forzadas entre secciones.

Simplificamos el prompt eliminando los tiempos específicos de cada sección y condensando los puntos clave en objetivos generales, permitiendo que la herramienta tuviera más flexibilidad narrativa. También removimos referencias técnicas específicas como historias de usuario que no aportaban valor al video final y confundían a la IA.

Además, ajustamos el enfoque económico para que no fuera tan detallado en costos específicos de baterías e infraestructura, y en su lugar pedimos un mensaje más general sobre viabilidad económica. Finalmente, añadimos énfasis en mostrar la coordinación visual entre múltiples tractores operando simultáneamente, que era el aspecto más impactante para la audiencia pero no estaba suficientemente destacado en el prompt original.

Prompt 6.

"Hola Claude, en el proyecto de un sistema multiagentes estamos trabajando con movilidad de tractores en un campo de cosecha, en el que se deben llevar a cabo dichas tareas como siembra con dispersión de semillas y aspersión de herbicida para eliminar malezas. En pocas palabras lo que queremos hacer es que a través de la implementación del sistema multiagente los tractores logren dar con la ruta óptima para desarrollar dichas tareas, pues buscamos potenciar el uso del suelo al máximo posible y reducir el sobrelapado o solapamiento al mínimo creando esta ruta óptima. Por favor, con fines de investigar la factibilidad de los agentes en este contexto, cuéntame cuáles son los beneficios de estos y cómo la simulación de unity 3d conectada al sistema multiagente puede ayudar a solucionar estos problemas de optimización en el campo".

Modificación realizada al resultado del prompt 6

El modelo de Claude Sonnet 4.5, a pesar de que se le proporcionó contexto del entorno, generó una respuesta general para un sistema que incluye simulación en Unity 3D con un sistema multiagente (SMA). El trabajo realizado por el equipo fue enfocar la respuesta obtenida hacia la problemática que buscamos abordar, mediante la especificación del cálculo de la ruta óptima para minimizar el sobrelapado (en sembrado) y el herbicida utilizado (en aspersión), así como maximizar el uso correcto y aprovechamiento del suelo, la especificación del algoritmo de búsqueda avanzada que se usó (A*) y los métodos de comunicación y comportamiento que se establecieron desde el inicio de la clase, Flock behaviour y Swarm communication.

10. Aportaciones individuales

Emilio Barragán: Desarrollo e implementación de las clases y modelos de los Agentes, para una fuente de datos (API) para alimentar la simulación en Unity hecha en Python y desplegada con Render. Así como el desarrollo de las siguientes secciones del documento: 7. Historias de Usuario, 8. Descripción del Sistema Multiagente.

Jordy Granados: Desarrollo de las siguientes secciones del documento: 3. Contexto y Problema, 4. Objetivos Generales, 5. Herramientas de Trabajo Colaborativo, 6. Restricciones, 10. Registro de prompts utilizados, 13. Conclusiones sobre el uso de la IA. De igual manera, apoyé recabando todas las fuentes consultadas por los miembros del equipo, para el desarrollo de la sección 14. Referencias del presente documento. Por otro lado, redacté los prompts para la generación de nuestro video y su consecuente ejecución de los mismos en NotebookLM de Gemini.

Humberto Cañedo: Obtención, selección y adaptación de prefabs 3D para la construcción del entorno agrícola en Unity 3D, configurándolos como tractores-agentes y obstáculos coherentes con la simulación multiagente, realizando ajustes ligeros en SketchUp para optimizar su uso visual y funcional dentro de la escena.

Diego Saldaña: Creación y estructuración del documento y repositorio de GitHub, desarrollo de las partes siguientes del documento: 1. Introducción, 2. Créditos, 8.1 y 8.2 Descripción del Agente.

Angel Peña: Desarrollo y adaptación del material para la investigación de eficiencia de implementación del sistema multiagente y simulación con gráficas computacionales a través de Unity 3D en la presentación de Canva, generada con el modelo de Claude Sonnet 4.5, desarrollo de los puntos 8.3: *“Modelo de la Negociación”* y 8.4: *“Modelo de la Interacción”* del documento de reporte técnico presente y elaboración del prompt 6 con su respectiva modificación.

11. Adaptación del output de la IA

El modelo de Claude Sonnet 4.5, a pesar de que se le proporcionó contexto del entorno, generó una respuesta general para un sistema que incluye simulación en Unity 3D con un sistema multiagente (SMA).

El trabajo realizado por el equipo fue enfocar la respuesta obtenida, la cual fue una respuesta muy general, hacia la problemática que buscamos abordar, mediante la especificación del cálculo de la ruta óptima para minimizar él sobrelapado (en sembrado) y el herbicida utilizado (en aspersión), así como maximizar el uso correcto y aprovechamiento del suelo, la especificación del algoritmo de búsqueda avanzada que se usó (A*) y los métodos de comunicación y comportamiento que se establecieron desde el inicio de la clase, Flock behaviour y Swarm communication.

12. Conclusiones sobre el uso de la IA

El uso de herramientas de inteligencia artificial durante el desarrollo de este proyecto nos da una perspectiva interesante, ya que, por un lado, la IA puede ser bastante útil, pero definitivamente no es la solución mágica que hace todo por nosotros, se necesita mucho pensamiento crítico para identificar qué sí y qué no.

Lo bueno: nos ayudó mucho como punto de partida para estructurar el proyecto. Cuando le dimos el contexto completo y específico sobre los sistemas multiagentes y sobre esta aplicación agro tecnológica, el modelo generó una base coherente sobre la que trabajamos y que nos ayudó a guiarnos durante el proceso de documentación inicial. En vez de empezar desde cero la estructura, pudimos enfocarnos más rápido en los aspectos técnicos que son los que más importan para un proyecto de este tipo. Por otro lado, para la generación del video también sirvió como una gran herramienta. Habiendo estructurado detalladamente el prompt, logramos crear material de presentación profesional en muchísimo menos tiempo del que nos hubiera tomado hacerlo nosotros de manera "manual", la IA logró tomar conceptos técnicos complejos como Swarm Intelligence y el algoritmo de A* y presentarlos de forma accesible.

Lo no tan bueno: Al inicio, las respuestas del modelo eran demasiado genéricas. Nos dio una estructura para un proyecto general de multiagentes, pero nosotros necesitábamos algo más específico al uso de agentes en tractores autónomos para agricultura. Reconocemos que en esta parte haber hecho más a detalle el prompt y enfocado a nuestro caso de uso, hubiera generado resultados más cercanos a lo que necesitábamos. La IA no especificó cosas importantes para nuestro proyecto como por qué elegimos A* sobre otros algoritmos, los protocolos de comunicación entre los tractores o qué métricas podrían ser relevantes para nosotros, todo esto último lo tuvimos que agregar nosotros basándonos en nuestra investigación y entendimiento del problema.

El aprendizaje: La IA funciona bien como un asistente, pero no como sustituto de un miembro del equipo. Nuestro conocimiento del problema y el entendimiento sobre qué necesitaba John Deere son irremplazables y es justo lo que nos ayudó a transformar las respuestas genéricas en soluciones aplicables.

13. Referencias

N. Sharp, "10 major challenges in bringing autonomous farming solutions to market," *ESCATEC Blog*, Jul. 23, 2024. [Online]. Available: https://www.escatec.com/blog/challenges-bringing-autonomous-farming-solutions-to-market

O. Lagnelöv, S. Dhillon, G. Larsson, D. Nilsson, A. Larsolle, and P.-A. Hansson, "Cost analysis of autonomous battery electric field tractors in agriculture," *Biosystems Engineering*, vol. 194, pp. 121–137, Mar. 2021, doi: 10.1016/j.biosystemseng.2021.02.005.

J. Conesa-Muñoz, M. Gonzalez-de-Soto, P. Gonzalez-de-Santos, and A. Ribeiro Seijas, "Distributed multi-level supervision to effectively monitor the operations of a fleet of autonomous vehicles in agricultural tasks," *Sensors*, vol. 15, pp. 5402–5428, Mar. 2015, doi: 10.3390/s150305402.

Discover Agriculture, “Fully autonomous tractor in action! SwarmFarm Robotics changing farming forever,” *YouTube*, Jul. 2, 2025. [Online]. Available: https://www.youtube.com/watch?v=Wi6w6j6xzEQ

C. Ju and H. I. Son, “Modeling and control of heterogeneous agricultural field robots based on Ramadge–Wonham theory,” *IEEE Robotics and Automation Letters*, vol. 5, no. 1, pp. 48–55, 2020, doi: 10.1109/LRA.2019.2941178.

M. Lujak, E. Sklar, and F. Semet, “On multi-agent coordination of agri-robot fleets,” in *Proc. CEUR Workshop Proc.*, vol. 2701, pp. 16–23, Oct. 2020. [Online]. Available: https://ceur-ws.org/Vol-2701/paper_12.pdf

A. Pandit, A. Njattuvetty, and A. K. M. Mulla, “ROS-based multi-agent systems control simulation testbed (MASCOT),” *arXiv*, Dec. 2022. [Online]. Available: https://arxiv.org/abs/2212.12657

H. Veluvolu, G. Y., C. Parthasarathy, P. Akkipeddi, and S. Patil, “Simulation of autonomous tractor using ROS,” *ARAI Journal of Mobility Technology*, vol. 5, no. 1, pp. 1464–1473, Jan. 2025, doi: 10.37285/ajmt.5.1.7.

Milvus, “What are the best tools for simulating multi-agent systems?,” *Milvus Blog*, 2025. [Online]. Available: https://milvus.io/ai-quick-reference/how-do-multiagent-systems-work

M.-E. Pérez-Pons, J. Parra-Domínguez, J. M. Corchado, J. Meira, and G. Marreiros, “Review on the applications of multi-agent systems in agriculture,” in *Proc. IV Workshop on Disruptive Information and Communication Technologies for Innovation and Digital Transformation*, Salamanca, Spain, pp. 49–57, 2020, doi: 10.14201/0AQ03154957.