Parcial Inteligencia Artificial

Fecha 02/10/2025

El parcial constara de una demostración en Image y defensa del código in situ.

Las implementaciones deben hacerse con las técnicas utilizadas en clase, no utilizando librerías de terceros.

Se tendrá en cuenta la calidad del código entregado, incluyendo el buen uso de tipos genéricos, interfaces y namespaces.

El proyecto hace uso de una buena arquitectura de proyecto, separando la arquitectura lógica en un assembly independiente al engine; y dejando la lectura de inputs y renderizado en otro que si contenga referencia al engine.

Se corregirá el código que este pusheado al repositorio, hasta el dia del examen a las 18:00Hs (Hora estandard de Argentina UTF - 3)

Todo lo entregado deberá estar alojado en un repositorio GIT privado creado explícitamente para este parcial, de lo contrario, la calificación será 0.

Deben tener una build subida como release en su repositorio para la corrección.

Condición mínima de aprobación (0 - 4)

- El parcial deberá constar de de una demo jugable estilo RTS 2D con las siguientes características:
    - Crear un mapa de superficie finita con ancho y alto modificable por el usuario antes de su creación.
    - El mapa debe estar poblado por nodos equidistantes en grilla cuadrada circunnavegable, la distancia entre nodos también debe ser modificable por el usuario antes de la creación del mapa.
    - El mapa cuenta con minas de oro dispersas en nodos aleatorios al momentos de generarse el mapa, la cantidad es modificable por el usuario.
    - En un punto del mapa hay un “Centro urbano” que es el encargado de generar dos tipos de agentes, “aldeanos” y “caravanas”.
    - Al iniciar la simulación, los aldeanos van a recolectar oro a la mina más cercana, extraen uno de oro cada cierta cantidad de tiempo, y cada tres de oro extraído necesitan comer una unidad de comida para poder seguir trabajando.
    - La comida es llevada por las caravanas desde el centro urbano hasta las minas a razón de diez por viaje.
    - En el momento de que el aldeano tenga 15 de oro en su inventario, regresará al centro urbano a depositarlo.
    - Si un aldeano no puede comer, esperará a que haya comida disponible para seguir trabajando.
    - En caso de que la mina en la que el aldeano estaba trabajando o a la que se estaba dirigiendo se agote irá a la mina que esté más cercana a su posición.
    - Las caravanas sólo proveen de comida a las minas donde un aldeano está trabajando.

    - En cualquier momento puede sonar una alarma (un botón en la UI) que hace que todos los agentes abandonen lo que están haciendo y regresen a refugiarse al centro urbano.
    - En cualquier momento, si la alarma está dada, se puede cancelar la alarma (con otro botón en la UI) que haga que:
        - Los agentes que estaban dentro del centro urbano salgan y regresen al trabajo.
        - Los que no habían llegado a refugiarse aún retoman sus labores
- Los agentes manejan su comportamiento con una FSM.
- Cada uno de los tipos de agentes es capaz de navegar por el mapa utilizando el algoritmo de A*. Cada tipo de agente aplica distintos pesos a la transición entre tipos de nodos y puede o no atravesar distintos tipos de nodos.
- Los agentes tienen calculado un Diagrama de Voronoi (o polígono de thiessen) para saber cual es la mina de oro más cercana desde cualquier punto del mapa. Este no tiene en cuenta la dificultad de transitar los nodos dentro de las áreas de los polígonos.
- El código de los patrones y técnicas utilizadas (FSM, Pathfinding, ECS, Flocking) no deben tener referencial al engine de ningún tipo.
- Todo aquello que pueda ser paralelizable, debe ser paralelizado.

Condición mínima para promoción (4 - 7)

- Los agentes al calcular un path pueden optimizarlo utilizando el algoritmo de Theeta*, siempre y cuando los nodos que descarte y por los que fuese a atravesar sean todos del mismo tipo.
- Si varios agentes del mismo tipo se están moviendo cerca, evitan superponerse utilizando el algoritmo de Flocking, los parámetros de alignment, cohesion, separation y direction de cada boid son calculados en un sistema de ECS.
- El Diagrama de Voronoi (o polígono de thiessen) de los mineros tiene en cuenta el costo de transitar los nodos del mapa para delimitar las áreas.
- Todos los agentes y entidades en escena se renderizan utilizando llamadas directas a GPU, no por componentes de renderer.

Examen completo (7 - 10)

- El código de los patrones y técnicas utilizadas (FSM, Pathfinding, ECS, Flocking) debe estar en una dll.
- Se puede cambiar el algoritmo de pathfinding que utilizan cada agente en particular en tiempo real.
- Todos los comportamientos de los agentes funcionan con ECS, pero manteniendo la separación de estados de la FSM.