Spatial Minery Simulation

Simulación de robots recolectores en una mina de Marte
Proyecto de Sistemas e Interfaces Inteligentes

Índice

Miembros

López Garnier, Antonio Jesús
Echeverría González, Ernesto
Díaz Rodríguez, Sebastián José
Pescador Barreto, Germán Andrés

Repositorio - Spatial-Minery-Simulation

Introducción

Descripción

Spatial Minery Simulation es, como bien dice su nombre, una simulación de expedición minera en el espacio, más específicamente ambientado en Marte. En esta simulación se muestra el funcionamiento un sistema multiagente encargado de la búsqueda de vetas de minerales y su recolección y gestión en una planta de minería en un planeta. El usuario puede experimentar la vida de un gestor de la expedición, trabajador en la empresa minera, donde supervisa la recolección de minerales por parte de los robots. El comportamiento de estos vendrá especificado detalladamente más adelante. La vista común del empleado se podrá supervisar mediante un dron de vigilancia volador.

El sistema está compuesto por dos tipos de agentes: la nave nodriza y los robots recolectores. El primero de estos se encarga de obtener información de los obstáculos de la superficie mediante imágenes infrarrojas y entregar dicha información a los recolectores. Sin embargo, la mayor carga de trabajo y el objeto de esta simulación se centra en los segundos, pues deben registrar el entorno mapeado sin las marcas de recursos en busca de estos, comunicándose a través de rastros de radiación de las vetas excavadas.

Objetivo

El principio de la simulación es la representación de un caso de minería espacial, basado en un sistema multiagentes, compuesto por robots recolectores y nave nodriza. Sin embargo, mediante esta representación también buscamos estudiar una aproximación al modelo presentado por Luc Steels en 1990, una comunicación entre agentes mediante el uso del entorno bajo incertidumbre. Entre las funcionalidades que buscamos está la implementación de nuevas tecnologías como reconocimiento de voz y realidad virtual para atraer a más usuarios al estudio de agentes y sus posibilidades.

Proyectos similares

Como antes hemos citado, nuestro proyecto se basa principalmente en la simulación de sistema de agentes de Luc Steels[simulación de 2010], la recolección de materiales en Marte. En esta recolección, los robots van desde la nave nodriza, caminando aleatoriamente, hasta toparse con una roca de minerales, desde la que camina prácticamente recto hacia su nave nodriza, donde dejará el recurso. La comunicación quedaba establecida en un rastro de migas con las que conectaban el resto de robots, guiados por el pensamiento de que tras el rastro habría materiales que recoger.

La implementación difiere en la cantidad de conocimiento que contienen los robots mineros, pues estos se encuentran en un entorno de incertidumbre parcial y combinan el uso de estados con algoritmos deterministas.

Herramientas utilizadas

Para lograr la simulación, se han utilizado Unity3D, MonoDevelop, Blender, Android Speech Recognizer Plugin y otros assets obtenibles en la Asset Store del propio programa Unity3D.

Unity3D es un motor gráfico multiplataforma de edición 3D y programación para la creación de videojuegos. Junto a MonoDevelop, su editor de texto integrado y diseñado para programación, se utiliza como base del proyecto, pues ofrece un entorno de modelado cómodo y flexible. Permite la integración de plugin externos al programa y acepta muchísimos tipos de modelos 3D. Este último aspecto nos ha permitido exportar modelos propios de SketchUp y Blender para dar una mejor sensación de inmersión.

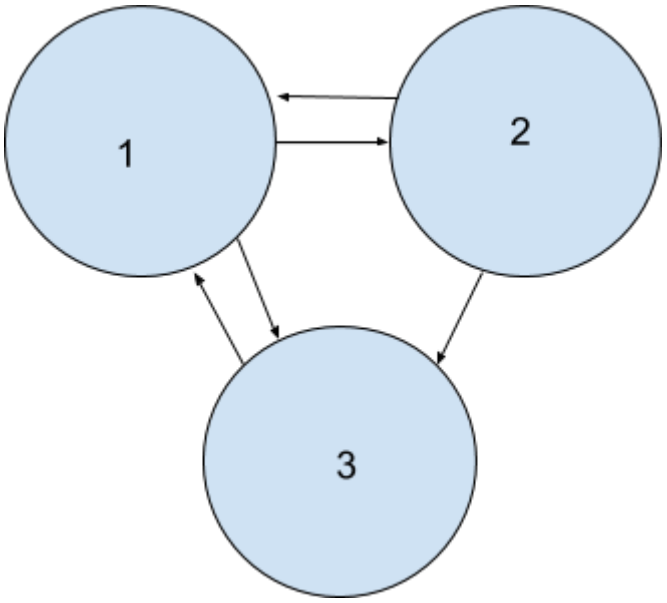
Blender es una herramienta de modelado 3D de alta definición, con un rendimiento asombroso y con gran profundización en el texturizado de estos modelos. Se ha aplicado para la creación de recursos básicos de la simulación. Por otro lado está SketchUp, más especializado en la creación de estructuras. Este no ha sido finalmente aplicado a la simulación gracias a la avanzada tecnología de Blender, así como su mayor acercamiento al objetivo previsto para estas herramientas.

Android Speech Recognizer Plugin es un asset en venta en la Asset Store de Unity3D que permite la detección de palabras y el lanzamiento de comandos de voz. Fue utilizado para ofrecer una vista aérea de los bots durante la simulación, lo que nos permite un mejor análisis situacional de la simulación.

Dejando de lado la tecnología adquirida externamente, se han aplicado técnicas y conocimientos propios de la informática para dar un paso hacia adelante en la simulación. Estas tecnología, centradas en la inteligencia de las máquinas, son la aplicación del algoritmo A* para el mapeado básico del entorno de los robots recolectores, el cálculo de movimiento aleatorio para la exploración profunda del mapa y la aplicación de una jerarquía de estados acompañada de un sistema de reglas de comportamiento para dotar de reacción a los agentes ante las situaciones abarcadas por el proyecto. Detallados en el siguiente apartado.

Sistema basado en reglas para el comportamiento

Como es habitual en sistemas basados en agentes reactivos o híbridos, nuestros agentes cuentan con sistemas basados en reglas que les fuerzan a transitar entre estados cuando se desata un evento en su entorno. Este evento, además de desencadenar una acción del agente, también cambia el estado interno del mismo.



Transición de estados del agente recolector

Por ejemplo, nuestro robot recolector cuenta con tres estados:

1. Robot perdido. En este estado, el robot realiza un movimiento aleatorio en busca del siguiente evento.
2. El robot encuentra una miga dejada por un compañero o por él mismo. Avanza en la dirección indicada por las migas (que señalan a la miga siguiente, ya que son colocadas en orden inverso). Cuando colisione con una mena, avanzará al estado siguiente, pero si la mena a la que llevaban las minas ya se ha agotado, volverá a estar perdido.
3. El robot encuentra una mena y tras recolectar su recurso avanza en dirección a la nave (obviamos para esta simulación el proceso completo de recolecta). Avanza en la dirección indicada por el algoritmo estrella hasta encontrar el siguiente evento: una colisión con la nave nodriza.

Sistema de Agentes

Agente híbrido

El sistema multiagente en el que se basa Spatial Minery Simulation, nuestro proyecto, es un sistema compuesto por los agentes nave nodriza y robot recolector, citados anteriormente. Estos agentes interactúan entre sí de diferentes formas, generando las relaciones nave-recolector y recolector-recolector. La primera marcando la entrega y gestión de los recursos que los recolectores y mostrando al usuario el resultado de esta relación mediante una impresión en pantalla. Básicamente el robot recolector capta un recurso del entorno y lo aporta a la nave nodriza para que esta lo sume a sus resultados y emita salidas audibles y visuales.

La segunda relación, recolector-recolector, es la base del sistema multiagente. Entre los recolectores se establece la habilidad de detectar, producir y seguir migas, residuos radioactivos que marcan el posible camino hacia recursos mineros que llevar a la nave

nodriza. Debido a su carácter basado en reglas de comportamiento, su reacción a través de estados y su aplicación del algoritmo determinista de búsqueda en grafos, se considera un agente híbrido, pues aunque siendo reactivo, aplica resoluciones en las que no lleva a cabo una decisión sino una orden.

Ambiente con incertidumbre

El entorno en el que se mueven los agentes es un ambiente en incertidumbre respecto a su meta más que a su desconocimiento de la figura del mismo. Al aplicar un algoritmo de búsqueda se provee al robot recolector de una figura básica del entorno recorrible; sin embargo, esto no le anula la necesidad de explorar el entorno libremente, llegando a los bordes del mismo si es oportuno.

Otro de los rasgos que lo caracterizan de inoportuno es que aún conociendo el entorno que le rodea, los recursos a los que tiene que acudir, recoger y traer no los conoce. No tiene ubicación de su objetivo y por lo tanto tiene que buscar exhaustivamente.

Comunicación mediante entorno

La comunicación utilizada entre los agentes inteligentes en este sistema está basado en el entorno, como bien se hizo en su momento en las primeras versiones de la idea. Este comportamiento expresa la relación recolector-recolector citada anteriormente y que permite el intercambio de información entre los robots recolectores a la hora de localizar menas con minerales.

La formalización de esta comunicación se produce tal que: al iniciar la simulación, los recolectores inician su movimiento aleatorio y continúan en ese estado hasta localizar una mena. En ese momento, cambia de estado para marcar su vuelta a la nave nodriza, generando migas hasta alcanzarla. Estas migas producen un cambio en las decisiones de los recolectores, haciendo que sigan el rastro hasta parar de encontrarlas o localizar un mineral una vez entren en contacto. Así, los agentes de la simulación intercambian información sin interactuar directamente entre ellos.

Desarrollo de la simulación

Recopilación de información

1.1. Aprendizaje Unity3D en la web oficial.

1.1.1. Creación de escenas.

La enseñanza guiada durante el curso en otras asignaturas y la gran cantidad de tutoriales, guías y material multimedia dispuesto tanto oficialmente como por otros usuarios nos permitió comprender rápidamente la construcción de las escenas y las interacciones entre los objetos.

1.1.2. Scripting.

Debido al framework de desarrollo, MonoDevelop, y las bases de Unity en C# los integrantes del grupo tuvimos que profundizar en un nuevo lenguaje prácticamente desconocido. El punto positivo de esto es que, acompañando a las posteriores clases en otra asignatura, pudimos dar grandes saltos en el desarrollo mediante una enseñanza medianamente guiada.

1.2. Familiarización con realidad virtual.

La gran parte de la fase de aprendizaje se basó en comprender los artefactos de la Realidad Virtual y su posible aplicación al proyecto. Guiados por las clases de Interfaces Inteligentes, pudimos construir una buena base de desarrollo en Realidad Virtual con las Google Cardboard.

1.3. Aprendizaje SketchUP en la web oficial.

1.3.1. Modelado de figuras.

A través de la herramienta SketchUp se aprendió a desarrollar estructuras. Sin embargo, tras invertir tiempo en la obtención de estos conocimientos, nos dimos cuenta de que el objetivo del software no era tan compatible como el de otras herramientas, lo que nos incitó a no aplicarlo en el proyecto, sino extender nuestros conocimientos de modelado mediante Blender.

Modelado

2.1. Creación y animación del personaje del usuario.

El personaje del usuario es una astronauta perteneciente a un proyecto llamado “Robot Lab”, publicado por el equipo de Unity en su plataforma Asset Store. Incluía una animación de estado de reposo, pero se decidió retirarla debido a su intrusión con la inmersión de la realidad virtual al mover esta la cabeza del personaje.

2.2. Creación y animación del robot recolector.

El robot recolector se extrajo del mismo proyecto que el personaje principal. Venía con una animación de las ruedas que fue incluída en la simulación.

2.3. Creación de las migas para marcar el trayecto.

Las migas son una esfera simple sobre la que se ha incluido una textura plana.

2.4. Creación del mineral a buscar.

El mineral a buscar fue creado en Blender enteramente.

2.5. Creación de la nave nodriza.

La nave nodriza se descargó como asset externo a través de la Asset Store. El nombre de la publicación es “Sci-Fi Scout Drone”.

2.6. Creación del dron vigilante.

El dron vigilante se obtuvo en la Asset Store. El nombre de la publicación es “Space Droid”.

2.7. Creación del modelo de superficie utilizado en la escena.

La creación del escenario comenzó con la elevación de las montañas de los extremos en una plataforma de 250x250. Para ello se utilizaron distintos pinceles de volumen con opacidades y tamaños variables para que no hubiese dos montículos iguales. Posteriormente, se elevaron las colinas centrales que servirían de obstáculos para los recolectores.

Con el mapa ya maquetado, se procedió a la nivelación de las elevaciones y su posterior lijado para mejorar la inmersión del usuario. Teniendo el modelo del mapa perfecto, se pasó al pintado del mismo. Tras analizar texturas de todo tipo de rocas, se utilizaron 3 diferentes, una de aspecto rocoso llena con capas para las elevaciones, una imitación de ceniza volcánica para el suelo, y una ligeramente granulada para suavizar la transición entre ambas.

Teniendo en cuenta que la simulación transcurre en un planeta extraterrestre, se decidió utilizar una textura para el cielo con características completamente diferentes a las de la Tierra. Una atmósfera rojiza, con un satélite muy cercano y su estrella acercándose al horizonte.

Montaje

3.1. Montaje de la escena inicial.

La colocación de los elementos en la escena no tardó mucho en realizarse tras la implementación de estos por separado y su posterior fusión en un proyecto.

3.2. Estudio de estados y medidas de decisión de los droides.

En el caso de los robots recolectores, debatimos mucho tiempo sobre las medidas para su implementación. Finalmente se optó por la implementación de agentes reactivos mediante estados de ejecución, lo que simplificó el proyecto.

Scripting

4.1. Script de movimiento del personaje del usuario y de robots recolectores.

El script de movimiento del personaje es un script bastante simple que lo traslada a través de espacio en los ejes X y Z según se pulsen las flechas de dirección o conjuntos de teclas/botones similares.

El de los robots recolectores por otra parte no responde a la pulsación de ninguna tecla, simplemente se les van asignando direcciones aleatorias dependiendo de si chocan con obstáculos o cuánto tiempo llevan desde el último cambio de dirección.

5.1. Script de cálculo de dirección bot-nave nodriza.

Tras el conocimiento de las funciones lookAt() y moveForward() proporcionadas por Unity3D, esto parecía una tarea sencilla, pero tras la implementación del algoritmo A* para evitar el choque en un bucle, todo este cálculo fue cambiado.

5.2. Script de recogida de recurso.

En el momento en el que un robot recolector entra en contacto con una mena, la cantidad de recurso en la misma disminuye, y el robot pasa a estado 3, debiendo volver mediante el camino determinado con el algoritmo A* hasta la nave para depositar el recurso encontrado, tras lo que volverá a por más. Cuando la cantidad de recurso en una mena alcanza 0, la mena desaparece.

5.3. Script de recogida de las migas.

De acuerdo a la implementación final, las migas ya no son recogidas, ya que no son parte del mundo y partes del recurso, sino marcadores gestionados por los recolectores para ayudarse los unos a otros.

5.4. Script generación y colocación de migas.

Tras encontrar un recurso por primera vez, el robot recolector que haya realizado el hallazgo regresará hacia la nave nodriza, dejando atrás migas que señalan a las migas anteriores, permitiendo la navegación entre migas desde la nave hasta los recursos.

Las migas son eliminadas mediante la llegada a 0 de su coste, una variable que aumenta con cada trayecto recurso-nave y disminuye en el trayecto inverso.

6.1. Script de choque

Se desestimó la implementación de un script de choque y se optó por utilizar colliders y rigid bodies en todos los elementos de la simulación.

6.2. Script de reconocimiento de voz

Para el reconocimiento de voz en dispositivo Android se hizo uso de un paquete de pago de internet, dada la dificultad que el uso de esta característica presentaba en Unity en el marco del desarrollo de aplicaciones móviles. Con el uso de este ‘plugin’ se creó un plugin capaz de escuchar al usuario y cotejar sus palabras con una serie de strings predefinidos que desencadenan una serie de acciones efectuadas por un dron centinela que sobrevuela la zona siguiendo las instrucciones del operador

Documentación

7.1. Documentación de los agentes.

Por cuestiones de tiempo, la documentación de los agentes quedó como una definición escueta de su funcionamiento, de por sí entendible tras su uso.

7.2. Documentación de los scripts

Los scripts utilizados en el proyecto fueron completamente documentados.

Problemas durante el desarrollo

En un principio, el proyecto no presentaba mayores dificultades que los plazos. Sin embargo, durante la fase de desarrollo, más específicamente en el modelado, surgieron problemas en cuanto al apartado visual del proyecto. Unity 3D no permite la animación directa sobre humanoides y no dispone de documentación que indique cómo crear clips para los modelos musculares. La búsqueda de una solución, añadida a la carga lectiva que supusieron las clases al principio del proyecto, ralentizó el desarrollo del proyecto levemente en comparación a los problemas siguientes.

Tras la formación del equipo, acercándose a la finalización de noviembre, dos de los cuatro miembros redujeron su comunicación a niveles mínimos, lo que ocasionó un desarrollo prácticamente congelado del proyecto. La fase de aprendizaje previa al conflicto de horarios fue un punto de apoyo a la hora de remontar en lo que respecta a fechas y tras la primera semana de enero pudimos contar con el equipo al completo para trabajar al unísono.

Por si fuera poco, la plataforma Unity, base del proyecto, canceló la suscripción de colaboración al equipo de desarrollo, lo que nos impidió compartir fácilmente los avances entre nosotros en épocas ajetreadas. La finalización del proyecto se logró gracias a las continuas reuniones de todos los miembros del grupo para poner puesta en común de lo desarrollado y avanzar en el desarrollo de la simulación.

Respecto a la funcionalidad de Realidad Virtual, intentamos implementar características específicas de unas gafas virtuales, las Google Cardboard, pero la gestión de sus elementos particulares supuso problemas de compatibilidad con el proyecto y tuvimos que optar por la opción básica y funcional ofrecida por Unity.

El último de los problemas fue el reconocimiento de comandos por voz. Debido al uso de librerías específicas para Windows, tuvimos que remodelar todo el reconocimiento de voz para hacerlo efectivo. Para ello tuvimos que comprar el asset de reconocimiento de voz propio de android para el entorno Unity, lo que nos supuso un golpe imprevisto en el desarrollo.

Conclusiones

A pesar de los conflictos presentados a lo largo del desarrollo del proyecto, se ha logrado una representación básica del lenguaje implícito en la información del entorno, utilizado por los agentes de manera inteligente a la hora de cumplir su cometido, la recolección de recursos dispuesta para los agentes con la incertidumbre de su localización.

El estado actual del proyecto es el de prototipo, pues es funcional, ilustrativo y en su medida su cometido es cumplido, más puede ser mejorado si el código es pulido y la modificación de la aplicación del rastreo de migas y menas de minerales, aunque su mejora

supondría una necesidad mucho mayor de la esperada para un aparato electrónico portable, una de las áreas objetivo que pretendemos abarcar.

Bien cabe decir que este proyecto prototipo aún puede avanzar en cuanto al apartado visual e interactivo, pero su raíz no puede avanzar mucho más allá si se quiere seguir la línea de Steel.