PROYECTO DE GRADO

PLATAFORMA PARA LA SIMULACIÓN DE AGENTES INTELIGENTES

ESTEBAN BETANCUR CORTISSOZ

DIEGO ALEJANDRO GÓMEZ URI BE

Proyecto de grado para optar al

título de Ingeniero de Sistemas

Asesor: Andrés Felipe Cano Cadavid

Medellín

Universdad EAFIT

Departamento de Ingeniería de Sistemas

2011

INTRODUCCIÓN

Este trabajo tiene como objetivo describir los conceptos teóricos, técnicos y prácticos utilizados en el proyecto de grado: plataforma para la simulación de agentes inteligentes.

La motivación para el proyecto fue crear una aplicación que facilitara el aprendizaje y la práctica sencilla de la programación de computadores. Dicha aplicación es un juego que mediante órdenes simples permite al usuario manipular un personaje que se desenvuelve en un entorno gráfico. El uso de órdenes es lo que le da al usuario una visión inicial de lo que implica programar. Las órdenes deben tener una disposición tanto semántica como sintáctica, que es similar a lo que ocurre en todos los lenguajes de programación existentes. La semántica y la sintaxis son las nociones más importantes y básicas que se tratan de presentar en este juego.

El proceso que se llevó a cabo para el desarrollo de este trabajo fue por etapas. Inicialmente, se consultó y analizó la teoría implícita en este tipo de problemas. Los conceptos sobre agentes, juegos, ambientes, reglas, inteligencia artificial brindaron una perspectiva clara de la solución. En la parte de la implementación de la solución se definió la metodología RUP que permite hacer un trabajo secuencial, y aborda el problema por partes lo que facilita su comprensión.

Técnicamente hablando, se pensó en construir una plataforma para la comunicación de los jugadores mediante una red local. La ventaja de una plataforma así es que provee la posibilidad de ejecutar el juego en varios computadores simultáneamente y que interactúen por medio de la red, lo que hace llamativa a la aplicación y brinda mayor flexibilidad a los jugadores.

El hecho de que sea un juego y no otro tipo de aplicación es algo significativo. Un juego de competencia representa un reto para quien lo está jugando, la competencia lo hace interesante ya que es una comparación de habilidades donde gana quien haga las cosas mejor.

Según los sicólogos, se ha demostrado que la utilización del juego en el aprendizaje facilita la comprensión de temas determinados. Para el tema de la programación, un juego

basado en reglas tiene una estructura perfecta para evidenciar ciertos conceptos básicos por parte del jugador.

La investigación teórica fue la etapa inicial y también muy importante para definir los elementos del juego, su comportamiento, su disposición y la interacción con el usuario. Teniendo en cuenta que existen varios tipos de juegos, de implementaciones, de lenguajes de programación, esta etapa brindó una perspectiva de la aplicación (antes de implementar) más simple de la que se tenía antes de comenzar porque al entender que un personaje puede o no ser un agente, ó que un obstáculo dentro del tablero de juego es una parte del ambiente se puede elaborar el diseño de la solución más simplemente. Esto tiene alta relevancia en la parte técnica ya que se aprende a distinguir los objetos que se definen en el lenguaje y los métodos que estos realizarán.

Además de una argumentación académica del objetivo de nuestro proyecto, es relevante explicar que la curiosidad fue parte fundamental. El análisis de temas y conceptos desconocidos, y su implementación utilizando los conocimientos adquiridos en el uso de lenguajes de programación fueron motivaciones adicionales en este trabajo.

La bibliografía se extrajo de fuentes recomendadas por profesionales en los temas afines a este proyecto.

OBJETIVOS

Objetivos generales

- Crear un juego que permita el aprendizaje de conceptos básicos de programación.
- Desarrollar los conceptos relacionados con juegos e inteligencia artificial.
- Brindar una plataforma para la interacción de agentes inteligentes.
- Aplicar un mecanismo para la comunicación entre los usuarios de la plataforma.

Objetivos específicos

- Utilizar Java como lenguaje de programación orientado a objetos para la implementación de la solución.
- Aplicar la herramienta RMI de Java para la comunicación e interacción de agentes inteligentes en un entorno de red local.
- Aplicar los conceptos de juegos e inteligencia para definir los lineamientos de la interacción de los agentes.
- Definir claramente los elementos que participan en el juego.

MARCO TEÓRICO

Proyectos Afines

Scratch: Es un software gratuito que ofrece una plataforma orientada al aprendizaje de programación y ciertos conceptos matemáticos, desde una interfaz gráfica agradable donde se pueden arrastrar elementos, añadir personajes y programar sus "comportamientos". El personaje principal de este juego es un gato, al cual se le aplican sonidos, colores, se modifica su apariencia y lo más importante es que el usuario tiene la posibilidad, basado en sentencias de programación sencillas, de definir cómo actuará éste en un escenario audiovisual definido también por el usuario.

"A Scratch lo desarrolla el Lifelong Kindergarten Group en el Laboratorio de Medios de MIT, con apoyo finanaciero de la National Science Foundation, Microsoft, la Fundación Intel, Nokia, lomega y el consorcio de investigación del Laboratorio de Medios de MIT."

http://info.scratch.mit.edu/es/About_Scratch.

Último acceso: 20 de Noviembre de 2011 a las 10:12 pm

Robot Emil: Es una aplicación que permite la práctica de conceptos de programación informática de forma sencilla. Tiene como objetivo el público infantil. Utiliza un Robot que se desenvuelve en una sala 3D, mediante las órdenes que el usuario decide darle. Las órdenes son: step, turn, put, paint, discolor, pick-out.

"… is nice free educational application for children, which allows programming virtual walking robot activities in the room.

It's ideal for first experience with computer programming, and evolutes algorithmic thinking by the means of game."

http://www.emil.input.sk/info_en.htm

Último acceso: 20 de Noviembre de 2011 a las 10:15 pm

xKarel: Este Robot sirve como ayuda para el aprendizaje de programación estructurada. Provee varias instrucciones, en un lenguaje fácil de entender, para que el jugador pueda determinar qué hara el Robot cuando encuentre un obstáculo, un beeper ó llegue a un límite. Los obstáculos y los beepers son parte del escenario donde se desarrolla el juego.

"xKarel is a game to learn structured programming. You manipulate with the robot Karel using simple programing language. Robot knows any elemetary commands. You learn robot more commands. Any new command is small structured program."

http://xkarel.sourceforge.net/eng/

Último acceso: 20 de Noviembre de 2011 a las 10:32 pm

Los proyectos analizados en los párrafos anteriores tienen como característica principal la utilización de un juego para proveer al usuario de un entorno donde pueda desarrollar conocimientos básicos e intermedios sobre programación.

Esta aplicación toma como base ciertas características y funcionalidades que se evidencian en trabajos similares como los mencionados.

En los juegos donde se intenta brindar un mecanismo para que el usuario manipule, según sus instrucciones, a un personaje que ejecuta acciones (caminar, recoger, voltear, saltar) en su ambiente

CONCEPTOS TEÓRICOS

Los conceptos que se presentan a continuación son importantes para el desarrollo de este proyecto ya que al estudiarlos se comprenderá el alcance y la definición de objetivos generales y específicos más claramente.

Lo que se intenta exponer son distintas definiciones referentes a los temas involucrados en este trabajo como lo son: inteligencia artificial, agentes, juegos, sistemas basados en reglas, ambiente. Y luego, a partir de estas teorías, se presentará una explicación adecuada a nuestro contexto.

Agente

“Un agente es cualquier cosa capaz de percibir su medioambiente con la ayuda de sensores y actuar en ese medio utilizando actuadores.”

“Un agente humano tiene ojos, oídos y otros órganos sensoriales además de manos, piernas, boca y otras partes del cuerpo para actuar. Un agente robot recibe pulsaciones del teclado, archivos de información y paquetes vía red a modo de entradas sensoriales y actúa sobre el medio con mensajes en el monitor, escribiendo ficheros y enviando paquetes por la red. Se trabajará con la hipótesis general de que cada agente puede percibir sus propias acciones (pero no siempre sus efectos).

El término percepción se utiliza en este contexto para indicar que el agente puede recibir entradas en cualquier instante. La secuencia de percepciones de un agente refleja el historial completo de lo que el agente ha recibido. En general, un agente tomará una decisión en un momento dado dependiendo de la secuencia completa de percepciones hasta ese instante. Si se puede especificar qué decisión tomará un agente para cada uno de las posibles secuencias de percepciones, entonces se habrá explicado más o menos todo lo que se puede decir de un agente. En términos matemáticos se puede decir que el

comportamiento del agente viene dado por la función del agente que proyecto una percepción dada en una acción.

La función que describe el comportamiento de un agente se puede presentar en forma de tabla: en la mayoría de los casos esta tabla sería muy grande (infinita a menos que se limite el tamaño de la secuencia de percepciones que se quiera considerar). Dado un agente, con el que se quiera experimentar, se puede, en principio, construir esta tabla teniendo en cuenta todas las secuencias de percepción y determinando qué acción lleva a cabo el agente en respuesta."

Nombre del Libro: Inteligencia Artificial un enfoque moderno. Autor: Stuart Rusell , Meter Norving. Editorial: Prentice Hall.

Agentes que resuelven problemas

"Los agentes inteligentes deben actuar de manera que el entorno experimente una serie de estados tales que permita obtener un máximo en la medida de rendimiento. En términos generales, es difícil lograr la plena traducción de esta especificación en un satisfactorio diseño de un agente. El logro de este objetivo se simplifica cuando el agente tiene una meta y se esfuerza por lograrla. La formulación de metas tomando la situación de un momento dado, es el primer paso para la solución de problemas."

Nombre del Libro: Inteligencia Artificial un enfoque moderno. Autor: Stuart Rusell , Meter Norving. Editorial: Prentice Hall.

Juegos

"En el caso de los juegos de tablero, los programas de juegos se han situado en el mismo nivel que los humanos y, en algunas ocasiones, han conseguido sobrepasarlos. Los éxitos de estos programas, y la naturaleza de sus procesos de razonamiento, tienen mucho que ver con nuestra comprensión de la naturaleza de la inteligencia."

Nombre del libro: Inteligencia Artificial. Cita de: Alan Turing 1950. Autor: Daniel Crevier. Editorial: Acento Editorial, 1996.

Inteligencia Artificial

"La inteligencia artificial estudia cómo lograr que las maquinas realicen tareas que, por el momento, son realizadas mejor por los seres humanos. Esta definición es, por supuesto,

bastante efímera ya que hace referencia al estado actual de la informática. Además falla al no incluir algunas áreas que potencialmente tienen un gran impacto, tales como aquellos problemas que no pueden ser resueltos adecuadamente resueltos ni por maquinas ni por los hombres. Sin embargo, proporciona un buen comienzo para aclarar en qué consiste la IA evitando los aspectos filosóficos que dominan en los intentos de definir los términos artificial o inteligencia."

Nombre del libro: Inteligencia Artificial, Segunda edición. Autor: Elaine Rich, Kevin Knaight. Editorial: Mac Graw Hill.

"Creo que dentro de unos cincuenta años será posible programar los ordenadores […] de forma que puedan desarrollar el juego de la imitación con total calidad que un interrogador medio solo tendrá un setenta por ciento de posibilidades de conseguir una identificación correcta tras cinco minutos de interrogatorio."

Nombre del libro: Inteligencia Artificial. Cita de: Alan Turing 1950. Autor: Daniel Crevier. Editorial: Acento Editorial, 1996.

"La interesante tarea de lograr que las computadoras piensen… máquinas con mentes, en su amplio sentido literal". (Haugeland, 1985)

"La automatización de actividades que vinculamos con procesos de pensamiento humano, actividades tales como toma de decisiones, resolución de problemas, aprendizaje…"(Bellman, 1978)

"El arte de crear máquinas con capacidad de realizar funciones que realizadas por personas requieren inteligencia". (Kurzweil, 1990)

"El estudio de cómo lograr que las computadoras realicen tareas que, por el momento, los humanos hacen mejor". (Rich y Knight, 1991)

"El estudio de las facultades mentales mediante el uso de modelos computacionales". (Charniak y McDermott, 1985)

"El estudio de los cálculos que permiten percibir, razonar y actuar". (Winston, 1992)

“Un campo de estudio que se enfoca en la explicación y emulación de la conducta inteligente en función de procesos computacionales”. (Schalkoff, 1990)

“La rama de la computación que se ocupa de la automatización de la conducta inteligente”. (Luger y Stubblefied, 1993)

Al analizar las definiciones anteriores se puede concluir que la inteligencia artificial, en términos generales, es el estudio de conceptos y procesos del pensamiento lógico humano apoyado en los sistemas computacionales para la simulación e implementación de los resultados obtenidos.

Sistemas basados en reglas

“Los sistemas de reglas encadenadas hacia atrás, de los cuales el PROLOG es un ejemplo resultan muy eficaces para la resolución de problemas dirigidos al objetivo. Por ejemplo un sistema interrogador probablemente utilizara un encadenamiento hacia atrás para razonar acerca de las respuestas a las preguntas del usuario.”

Nombre del libro: Inteligencia Artificial, Segunda edición. Autor: Elaine Rich, Kevin Knaight. Editorial: Mac Graw Hill.

“Prolog es un lenguaje de programación que está orientado a la especificación de relaciones para responder consultas. En ese sentido Prolog es similar a un sistema de base de datos, aunque en el contexto de la inteligencia artificial se prefiere hablar de bases de conocimiento, enfatizando la complejidad estructural de los datos y de las deducciones que se pueden obtener de ellos.”

http://www.dcc.uchile.cl/~abassi/IA/Prolog.html

Último acceso: 24 de Noviembre de 2011 a las 12:10 pm

“Los sistemas basados en reglas encadenadas hacia delante, en lugar de dirigirse por objetivos, se dirige por la información que se va incorporando. Por ejemplo supóngase que sentimos calor cerca de nuestra mano. Seguramente se tenderá a retirar la mano de ahí. Mientras que esto se puede considerar como un comportamiento dirigido a un objetivo, se modela de forma más natural por medio de sitios de modelamiento de actos, característicos de los sistemas de reglas encadenadas hacia delante.”

Nombre del libro: Inteligencia Artificial, Segunda edición. Autor: Elaine Rich, Kevin Knaight. Editorial: Mac Graw Hill.

“Podemos utilizar combinación del razonamiento hacia delante y hacia atrás a veces, cuando aspectos de un problema se asemeja más fácilmente utilizando el encadenamiento hacia adelante, mientras que con otros se solucionan de un modo más sencillo utilizando el encadenamiento hacia atrás. Considérese por ejemplo un programa de diagnósticos médicos basados en el encadenamiento hacia delante. Este puede aceptar, aproximadamente, una veintena de hechos acerca de la condición del paciente, entonces trabajara hacia delante con dichos hechos para intentar deducir la naturaleza y la causa de la enfermedad.”

“Saber si es posible utilizar las mismas reglas tanto para el razonamiento hacia delante, como el razonamiento hacia atrás, también depende de la propia forma de las reglas.”

Nombre del libro: Inteligencia Artificial, Segunda edición. Autor: Elaine Rich, Kevin Knaight. Editorial: Mac Graw Hill.

Propiedades de los entornos de trabajo

“El rango de los entornos de trabajo en los que se utilizan técnicas de inteligencia artificial es obviamente muy grande. Sin embargo, se puede identificar un pequeño número de dimensiones en las que categorizar estos entornos. Estas dimensiones determinan, hasta cierto punto, el diseño más adecuado para el agente y la utilización de cada una de las familias principales de técnicas en la implementación del agente. Primero se enumeran las dimensiones, y después se analizan varios entornos de trabajo para ilustrar estas ideas. Las definiciones dadas son informales; capítulos posteriores proporcionen definiciones más precisas y ejemplos de cada tipo de entorno.

Totalmente observable vs parcialmente observable

Si los sensores del agente le proporcionan acceso al estado completo del medio en cada momento, entonces se dice que el entorno de trabajo es totalmente observable. Un entorno de trabajo es, efectivamente, totalmente observable si los sensores detectan todos los aspectos que son relevantes en la toma de decisiones; la relevancia, en cada momento, depende de las medidas de rendimiento. Entornos totalmente observables son convenientes ya que el agente no necesita mantener ningún estado interno para saber

qué sucede en el mundo. Un entorno puede ser parcialmente observable debido al ruido y a la existencia de sensores poco exactos o porque los sensores no reciben información de parte del sistema, por ejemplo, un agente aspiradora con sólo un sensor de suciedad local no puede saber si hay suciedad en la otra cuadrícula, y un taxi automatizado no puede saber qué están pensando otros conductores.

Determinista vs estocástico

Si el siguiente estado del medio está totalmente determinado por el estado actual y la acción ejecuta por el agente, entonces se dice que el entorno es determinista; de otra forma es estocástico. En principio, un agente no se tiene que preocupar de la incertidumbre en un medio totalmente observable y determinista. Sin embargo, si el medio es parcialmente observable entonces puede parecer estocástico. Esto es particularmente cierto si se trata de un medio complejo, haciendo difícil el mantener constancia de todos los aspectos observados. Así, a menudo es mejor pensar en entornos deterministas o estocásticos desde el punto de vista del agente.

Agente individual vs multiagente

La distinción entre el entorno de un agente individual y el de un sistema multiagente puede parecer suficientemente simple. Por ejemplo, un agente resolviendo un crucigrama por sí mismo está claramente en un entorno de agente individual, mientras que un agente que juega al ajedrez está en un entorno con dos agentes. Sin embargo hay algunas diferencias sutiles. Primero, se ha descrito que una entidad puede percibirse como un agente, pero no se ha explicado qué entidades se deben considerar agentes. ¿Tiene el agente A (por ejemplo el agente taxista) que tratar un objeto B (otro vehículo) como un agente, o puede tratarse meramente como un objeto con un comportamiento estocástico, como las olas de la playa o las hojas que mueve el viento? La distinción clave está en identificar si el comportamiento de B está mejor descrito por la maximización de una medida de rendimiento cuyo valor depende del comportamiento de A. Por ejemplo, en el ajedrez, la entidad oponente B intenta maximizar su medida de rendimiento, la cual, según las reglas, minimiza la medida de rendimiento del agente A. Por tanto, el ajedrez es un entorno multiagente competitivo.

El problema de perseguir y evadir

“The chasing/evading problem consists of two parts. The first part involves the decision to initiate a chase or to evade. The second part involves effecting the chase or evasion that is, getting your predator to the prey, or having the prey get as far from the predator as possible without getting caught. In a sense, one could argue that the chasing/evading problem contains a third element: obstacle avoidance. Having to avoid obstacles while chasing or evading definitely complicates matters, making the algorithms more difficult to program.”

“…the simplest chase algorithm involves correcting the predator's coordinates based on the prey's coordinates so as to reduce the distance between their positions. This is a very common method for implementing basic chasing and evading. (In this method, evading is virtually the opposite of chasing, whereby instead of trying to decrease the distance between the predator and prey coordinates, you try to increase it.)”

Algoritmo básico para persecución

“

```
if (predatorX > preyX)
predatorX--;
else if (predatorX < preyX)
predatorX++;
if (predatorY > preyY)
predatorY--;
else if (predatorY < preyY)
predatorY++;
```

In this example, the prey is located at coordinates *preyX* and *preyY*, while the predator is located at coordinates *predatorX* and *predatorY*. During each cycle through the game loop the predator's coordinates are checked against the prey's. If the predator's x-coordinate is greater than the prey's x-coordinate, the predator's xcoordinate is decremented, moving it closer to the prey's x-position. Conversely, if the predator's x-coordinate is less than the prey's, the predator's x-coordinate is incremented. Similar logic applies to the predator's ycoordinate based on the prey's y-coordinate. The end result is that the predator will move closer and closer to the prey each cycle through the game loop.”

Algoritmo básico para evasion

“Using this same methodology, we can implement evading by simply reversing the logic.

```
if (preyX > predatorX)
preyX++;
else if (preyX < predatorX)
preyX--?>;
if (preyY > predatorY)
preyY++;
else if (preyY < predatorY)
preyY--;”
```

Nombre del libro: AI for game developers. Autores: David M. Bourg, Glenn Seemann. Editorial: O’reilly.

CONCEPTOS TÉCNICOS

Lenguaje de programación Java

Aquí, más que una descripción ó reseña de Java, lo que se intenta es dar una justificación para el uso de esta tecnología basándonos en la experiencia obtenida con este lenguaje.

Java es un lenguaje de programación orientado a objetos. Representa uno de los lenguajes multiplataforma más completos y versátiles. El hecho de que un código Java pueda ser ejecutado en diferentes sistemas operativos es lo que lo hace más llamativo, además de su gran cantidad de herramientas adicionales. Actualmente, su uso es amplío a nivel mundial por programadores informáticos, lo que lo convierte en un factor beneficioso ya que existe abundante información en Internet como ejemplos de código, manuales generales y específicos, foros de discusión; y en la literatura convencional también se encuentra muy buen material que apoya el aprendizaje de este lenguaje.

Como valor agregado, aunque muchos lenguajes son de este tipo, se puede resaltar su distribución gratuita, lo que brinda facilidad para adquirir las herramientas necesarias para desarrollar código Java. Todas las aplicaciones requeridas para operar este lenguaje se pueden descargar sin ningún problema de la página Web de la empresa Oracle.

RMI (Remote Method Invocation)

“RMI applications often comprise two separate programs, a server and a client. A typical server program creates some remote objects, makes references to these objects accessible, and waits for clients to invoke methods on these objects. A typical client program obtains a remote reference to one or more remote objects on a server and then invokes methods on them. RMI provides the mechanism by which the server and the client communicate and pass information back and forth. Such an application is sometimes referred to as a *distributed object application*.

Distributed object applications need to do the following:

Locate remote objects. Applications can use various mechanisms to obtain references to remote objects. For example, an application can register its remote objects with RMI's simple naming facility, the RMI registry. Alternatively, an application can pass and return remote object references as part of other remote invocations.

Communicate with remote objects. Details of communication between remote objects are handled by RMI. To the programmer, remote communication looks similar to regular Java method invocations.

Load class definitions for objects that are passed around. Because RMI enables objects to be passed back and forth, it provides mechanisms for loading an object's class definitions as well as for transmitting an object's data."

http://docs.oracle.com/javase/tutorial/rmi/overview.html

Último acceso 23 de Noviembre de 2011 a las 7:30 am

Para este trabajo se utiliza un mecanismo que provee Java que permite la comunicación entre máquinas virtuales (Java Virtual Machines) distribuidas remotamente llamado RMI (Remote Method Invocation). Dicha comunicación se realiza mediante la invocación a métodos remotos como si fuera de manera local.

Como se indica en la definición anterior, RMI plantea dos tipos de aplicaciones: cliente y servidor. Cada una con sus respectivas clases, interfaces, métodos y objetos. Cabe resaltar que una de esas interfaces debe ser definida para el objeto remoto, llamada típicamente la interfaz remota. Por medio de ésta el cliente puede invocar al objeto remoto, ya que en ella se definen los métodos que se expondrán para ser utilizados por el cliente. Se pueden encontrar otros métodos definidos en el objeto remoto.

Este tipo de comunicación permite abstraer al programador de los detalles técnicos necesarios para establecer conexiones entre aplicaciones remotas en red, aunque igualmente opera de forma local.

Como el modelo utilizado para la elaboración de este juego fue cliente servidor, se estimó que el manejo de RMI podría facilitar su desarrollo, porque se integra simplemente al lenguaje y brinda herramientas adecuadas para las aplicaciones distribuidas.

DEFINICIÓN DEL JUEGO

Esta aplicación está basada en el clásico juego de persecución: policías y ladrones. El objetivo principal del juego es que el personaje que interpreta a un policía intente atrapar el ladrón, y el ladrón trate de evadir al policía evitando su captura.

Los jugadores tendrán la posibilidad de elegir cuál personaje quieren utilizar y también podrán definir instrucciones, basados en un seudo lenguaje de programación definido para la aplicación. Dichas instrucciones serán interpretadas por cada personaje (policía ó ladrón) para tomar decisiones a medida que se desarrolla el juego.

Los personajes tendrán la capacidad de analizar su entorno y teniendo como base las órdenes programadas por los usuarios cambiar su comportamiento y tomar decisiones según la situación en que se encuentre. La situación analizada por los agentes se define por la posición propia en el ambiente, el tiempo trascurrido, las opciones de movimiento, los obstáculos que impiden ciertas funciones, posiciones de otros agentes, entre otras. Toda esta información brinda la opción al agente de decidir qué tareas básicas y programadas realizar para lograr su objetivo.

Existen tres tipos de características que poseen los agentes. Estas características son: visibilidad, velocidad y sobrepaso.

La visibilidad está definida por la posibilidad que tiene un agente de “ver” las casillas siguientes a su posición actual. Dependiendo de la orientación (norte, sur, este u oeste) que tenga el agente en una determinada situación en el tablero.

La velocidad se refiere a la capacidad que tiene un agente de recorrer las casillas del tablero. Esta puede disminuir ó normalizarse dependiendo de algunos elementos del entorno.

El sobrepaso tiene que ver con la capacidad de un agente de estar en la misma casilla de otro agente, teniendo cuenta las características de cada uno de ellos.

A continuación se expondrán los elementos y sus objetivos dentro del juego, con lo cual se pretende que el lector comprenda más claramente las funciones y restricciones de los agentes.

Elementos del juego

Agentes activos

Policía: su función principal es perseguir.

Ladrón: su función principal es evadir.

Agentes pasivos

Pared: impide el sobrepaso y la visibilidad totalmente, no disminuye la velocidad de los personajes.

Malla: no impide la visibilidad ni la velocidad e impide totalmente el sobrepaso.

Lodo: disminuye la velocidad parcialmente. No afecta la visibilidad y permite el sobrepaso.

Celda vacía: es una celda de libre movimiento y visibilidad sin afectar la velocidad de los personajes.

Tablero: ambiente donde se desarrolla el juego.

Reglas de juego

Un agente activo no puede sobrepasar a otro, es decir, dos agentes activos no pueden estar en la misma celda simultáneamente.

Un agente activo no impide la visibilidad a otro agente activo, es decir, el hecho de que un agente esté en frente de otro y en la misma dirección, no impedirá que ambos puedan “ver” el estado de las celdas posteriores a su oponente ó compañero.

Los agentes activos no pueden moverse sobre el tablero diagonalmente, sólo lo hacen horizontal ó verticalmente.

Los agentes activos podrán moverse hacia el norte, sur, este, oeste, nor-este, nor-oeste, sur-este y sur-oeste.

Los agentes activos podrán incrementar o disminuir la velocidad hasta cierto límite de cuadros por segundo. El número de cuadros lo define el usuario.

Existe un rango de visión que es el límite de celdas en las que un agente activo puede percatarse de la presencia del otro, en campo abierto sin obstáculos.

Un agente activo sólo podrá conocer lo que hay dentro del rango de visión. Es decir, el participante solo ve lo que esté en hasta número de celdas definido anteriormente.

Al participante que pase por el lodo, se le disminuirá la velocidad hasta en un cuadro por segundo.

El participante que esté en frente de un muro no podrá ver nada más allá del muro, en esa dirección.

La cerca limitará la visión del participante hasta por 2 celdas, es decir más allá de la celda el participante no verá si no dos cuadros más.

El agente perseguidor (policía) habrá capturado al ladrón cuando no existan celdas entre ellos.

Existe un límite de tiempo en el que se desarrolla el juego.

El agente evasor (ladrón) habrá evadido satisfactoriamente al perseguidor si el policía no logra capturarlo en un límite de tiempo.

POSIBLES MEJORAS

A la funcionalidad del juego

Un agente perseguidor podrá reclutar a otro del mismo tipo si y solo si el agente reclutado se encuentra dentro del rango de visión del participante que inicio dicha acción. Además, el agente policía solo podrá hacer uso de esta opción si en su rango de visión se encuentra al menos un participante ladrón.

A la interfaz gráfica

En la implementación inicial de la interfaz gráfica se manejó el API de Java 2D. La mejora más importante que se puede realizar en esta parte es la aplicación del API de Java 3D. Que permite la construcción de gráficos en tres dimensiones, muchos más potentes y estéticamente llamativos que los de dos dimensiones. Esto favorecerá la experiencia de juego del usuario.

“The Java 3D API enables the creation of three-dimensional graphics applications and Internet-based 3D applets. It provides high-level constructs for creating and manipulation 3D geometry and building the structures used in rendering that geometry. With this software, you can efficiently define and render very large virtual worlds.”

http://www.oracle.com/technetwork/java/javase/tech/index-jsp-138252.html

Último acceso el día 19 de Noviembre de 2011 a las 2:56 pm

CONCLUSIONES

Las siguientes conclusiones son presentadas teniendo en cuenta los objetivos específicos. Aquí se expresan los resultados obtenidos en todo el proceso de desarrollo del proyecto y se trata de dar una explicación concreta de cómo se lograron. Cada uno de los objetivos específicos refleja tareas que se realizaron en las diferentes etapas del trabajo y funciones ejecutadas en la aplicación.

El uso de juegos para el aprendizaje de los fundamentos básicos de programación es de gran ayuda para los usuarios con conocimientos elementales.

RMI brinda al programador la posibilidad de abstraer algunos los procesos detallados de comunicación entre computadores al lograr que las implementaciones de los métodos en el servidor remoto se hagan como si fuera una implementación local. Al ser parte del estándar de Java, RMI se acopla fácilmente al código (funcionalidades de las reglas del negocio) y no necesita mayores cambios para su aplicación.

Al tener cierta experiencia en el tema de programación utilizando Java y RMI la implementación de la solución se realizó con base a las prácticas anteriores donde ya se había manejado los procesos de comunicación remota, sin embargo, varios conceptos nuevos fueron consultados e implementados para el desarrollo del proyecto.

La interpretación de las funciones de un agente, y del papel tan importante que cumplen los demás elementos con los que interactúa permitieron que el desarrollo de este juego aplicará ciertos conceptos de la inteligencia artificial que se suman como valor agregado a la investigación, y aportan un panorama muy amplio para las posibles mejoras. Cuando se tiene la capacidad de reconocer que un personaje (dentro del juego) se comporta como

un agente (policías y ladrones), y que el entorno (tablero) debe intercambiar información (desde la parte técnica) con este para colaborar en la funcionalidad total del juego, se puede decir que se tiene una buena comprensión y se puede realizar un modelo para implementar más claro. Eso fue lo que sucedió en este trabajo. La teoría analizada sirvió para plantear una solución adecuada con los medios elegidos.

El ambiente debe actualizar su estado para conocer posibles cambios. En otras palabras, el ambiente debe conocer en qué situación se encuentra en todo momento. Esto es importante porque la información que recolecta debe ser transmitida a los agentes que interactúan con él para que los agentes puedan tomar decisiones según sus criterios y sobre todo según las órdenes establecidas por el usuario.

Un entorno donde conviven varios agentes es complicado para implementar, ya que existen múltiples variables que deben ser tenidas en cuenta para garantizar su correcta funcionalidad. Es decir, el hecho de que un agente tenga que evaluar condiciones para realizar tareas, mantenerse informado de su entorno, y además competir (en este juego de persecución) con los demás agentes, hace que el diseño de la solución deba ser cuidadoso, detallado y sencillo de explicar antes de comenzar con el desarrollo en el lenguaje de programación.

El diseño de la interfaz gráfica es una parte relevante para esta tesis, aunque no sea algo complejo. La determinación de utilizar un tablero y gráficos simples fue tomada teniendo como referencia los modelos de juegos actuales referentes al tema tratado en este proyecto. Sin desconocer que el diseño gráfico sencillo es más adecuado para este tipo de juegos donde se impone la funcionalidad y el objetivo principal que es el aprendizaje, a otros factores que son tenidos en cuenta en juegos comerciales que despliegan gráficos potentes.

Los personajes que participan en el juego son llamados agentes ya que según el marco teórico, un agente en un ambiente competitivo como el de este juego, debe tratar de maximizar su rendimiento, implicando a su vez que los otros agentes minimicen el suyo. Un agente policía al tratar de capturar a un agente ladrón, deberá reducirle las posibilidades de escape, representando una pérdida de opciones para que el ladrón evite ser atrapado. En caso contrario, el ladrón trata de evadir al policía el mayor tiempo posible, y si lo logra impedirá que el policía cumple su objetivo.