UNIVERSIDAD
DE GRANADA

E.T.S. DE INGENIERÍAS INFORMÁTICA y DE TELECOMUNICACIÓN

PRACTICA 1

AGENTES REACTIVOS

Inteligencia Artificial

Luis Orts Ferrer
77695861K

Objetivos

En esta primera práctica implementaremos un agente reactivo dentro del mundo desarrollado para está asignatura : Los mundo de Belkan. Aplicaremos diferentes reglas y condiciones para que nuestro agente consiga obtener la mayor información posible del mapa en el que se encuentra.

Comportamiento del Agente

Este agente realizará un comportamiento puramente reactivo, en este tipo de comportamientos el agente percibirá el ambiente que contiene el mapa y a partir de la información captada, actuará de una manera u otra. En nuestro caso el agente avanzará en una dirección, hasta encontrar un obstáculo, en ese caso, el agente realizará un giro hacia la izquierda o derecha para evitar dicho obstáculo. También se realizan giros si se lleva un cierto número de pasos en la misma dirección. Además de estos movimientos, el agente podrá escoger diferentes rutas, si dentro de su campo de visión puede divisar un objeto o casilla que necesite el agente en ese momento.

Además de este comportamiento, el agente tiene la posibilidad de recoger un máximo de dos objetos del mapa para usarlos en diferentes zonas del mapa, además se ha añadido el comportamiento para evitar un objeto que ya tiene el agente. Los objetos se mantendrán hasta que el agente caiga por un precipicio o se choque con un lobo, en estos casos el agente se reiniciará en otro lugar,perderá todos los objetos y perderá la orientación, para evitar esta situación se ha realizado varios comportamientos, si el agente encuentra un precipicio justo delante de él o un lobo, el agente girará en una dirección aleatoria y opuesta al precipicio o al lobo. Además de estos aspectos, a lo largo de la memoria se describirán con más detalle el comportamiento del agente.

Tener en cuenta que el agente tendrá prioridad para encontrar casillas ‘G’, al encontrar una, esa prioridad se eliminará, para ello marcaremos rutas para encontrar dichas casillas.

Variables de estado

- Fila, colum, brújula: estas se usarán para conocer la posición y la orientación del agente en el mapa correspondiente. Tendrán un valor inicial 99 en las filas y 99 en las columnas, estos valores valdrán -1 en el nivel 1, en la brújula tendrá el valor indicado por el sensor en el nivel 0 y 1 pero en los niveles 2,3 y 4 tendrá el valor por defecto 0, el cual indica el norte. Cada paso realizado por el agente actualizará estas

variables y al alcanzar el la casilla ‘G’, fila y colum tomarán los valores adecuados en el mapa.

- ultAccion: esta variable se usa para almacenar la última acción que se ha realizado y la cual el agente tendrá en cuenta para las próximas decisiones.

- girar_der: variable booleana, la cual se inicializa a falso y el valor se calculará de manera aleatoria, si está activa se girará a la derecha y si está desactivada se girará a la izquierda a la hora de realizar un giro.

- hacemos_giro: una variable de tipo int que contiene el número máximo de avances que puede realizar el agente seguidos, se usará para cuando el agente avanza sin ninguna ruta establecida. Cuando se alcanza ese número el agente girará de manera aleatoria en una dirección y reiniciando de está manera la variable. Se inicializará a 61 puesto que al realizar varías pruebas, este valor nos genera un mayor porcentaje de información sobre el mapa.

- bien_situado: variable booleana su valor será verdadero cuando el jugador pasa por una casilla ‘G’ en el caso de niveles superiores a 1, en el nivel 0 siempre será verdadero, al tener esta casilla verdadera, se podrá almacenar la información captada por el agente en mapaResultado.

- tiene_bikini: variable booleana la cual será verdadera cuando el jugador se posa en la casilla que contiene un objeto de tipo bikini.

- tiene_zapatillas: variable booleana la cual será verdadera cuando el jugador se posa en la casilla que contiene un objeto de tipo bikini.

- destino: booleano el cual indica si el agente sigue una ruta fijada.

- posicion_des: en esta variable se guarda la posición del vector sensores.terreno en la que se encuentra el objetivo. Con esta variable se creará la ruta para ir al destino.

- Giro: variable de tipo acción para girar hacia la izquierda o hacia la derecha, se aplicará si para alcanzar al objetivo necesitamos girar

- num_pasos_al_objetivo: el número de avances que necesitamos para alcanzar el destino.

- cargado: variable entera, la cual indica el número de recargas de batería que lleva el agente
- sensor: variable de tipo sensores.

Método think

Este método es el principal a la hora de elaborar el comportamiento del agente y el cual consta de 2 fases en una primera fase, se observa cómo ha cambiado el mundo en función de la última acción que se realizó y se actualiza la información que tenemos del mundo y una segunda fase, se determina cuál es la siguiente acción a realizar.

En nuestro caso lo primero que implementaremos es una condición la cual se cumple cuando estamos en el nivel 1 o 0 y no estamos bien situados, nos dará la orientación del agente sin tener que pasar por una casilla 'G'.
Posteriormente se revisará si se ha reiniciado la p MIRAR ESTO

Lo siguiente será actualizar la posición del agente en el mapa, para ello se usará la variable ultAccion que según el valor que tenga, se modificarán el valor de las variables fila, colum o brújula. Se ha usado como plantilla la implementación proporcionada por el tutorial, pero en esta implementación añadimos una condición para comprobar si el agente ha colisionado, usamos el sensor de colisión y de esta manera evitamos que realice la acción de avanzar pero sin moverse el agente.

A continuación incluimos otro condicional el cual se cumplirá si estamos en el nivel 0 la variable booleana bien_situado estará a true y se modificará el valor de fil y colum por los sensores de posición de la filas y de las columnas.

Si no estamos en el nivel 0 y la condición previa no se cumple, pasamos directamente a otra condición en la cual modifica la variable booleana bien_situado a true y se modificará el valor de fil y colum por los sensores de posición de la filas y de las columnas. Está condición será verdadera si el agente se coloca sobre una casilla 'G' y no esté la variable bien_situado a true.

Una vez configurado la posición y orientación del agente pasaremos a incluir la información perteneciente del cono de visión del agente a la matriz mapaResultado donde solamente se realizará esta acción si el agente está bien situado. Para introducir la información de manera correcta debemos tener en cuenta la orientación actual del agente ya que dependiendo de ésta los valores de cada casilla de visión varía su fila y su columna. En este caso por cada valor que puede tomar la variable brújula, indicamos por cada uno de los sensores que detecta el agente, se añaden a mapaResultado en su posición correspondiente.

Ya implementado la manera de almacenar la información correcta sobre el mapa que recorre el agente, se continuará con la toma de decisiones en función de lo percibido por los sensores del agente, se usarán los condicionales if y esle if.

La primera prioridad de este agente es estar bien situado el máximo de tiempo posible para poder captar la máxima información del mapa en el que está situado.
En el caso de que no se encuentre bien situado, realizaremos un bucle for donde se comprobará si en alguno de los sensores.terreno que el agente divisa se encuentra alguna casilla 'G', en el caso de encontrar alguna, se calculará una ruta con el método Ruta(se explicará su funcionamiento posteriormente).

En el caso de estar bien situado, pero no tener ni bikini ni zapatillas, se realizará el mismo método explicado previamente para crear una ruta hacia uno de esos objetos, en el caso de tener alguno de estos objetos, se marcará una ruta hacia el objeto que le falta al agente. La posesión de estos objetos será una de las principales propiedades del comportamiento del agente, puesto que teniendo estos objetos se reducirá el consumo de batería en algunas zonas del mapa donde el consumo de está es bastante alto y por consiguiente podremos recorrer el mapa durante muchas más iteraciones.

En el aspecto de batería, cuando tengamos los objetos, estemos bien situados y tengamos una batería con un valor menor de 1500, se indicará una ruta si divisamos una casilla de recarga de batería, el cual se quedará en esa casilla hasta que la batería tenga el valor 1500, está implementación tenía un inconveniente, ya que al salir de esta casilla el valor de la batería decrementa al instante y en algunas ocasiones el agente volvió a detectar esa casilla cada vez que recargaba la batería y por tanto el agente volvía una y otra vez a esta casilla y tendríamos una partida infinita, en la que el agente nunca se quedaba sin batería pero tampoco terminaba de explorar por completo el mapa. Para solucionar está situación, implementando una condición la cual indica que si la batería es mayor o igual que 1500 y estamos justo encima de la casilla de recarga se incrementará la variable cargado, y para finalizar la mejora de este comportamiento se le añade una condición más en la condición de carga, la cual indica que solamente puede ir a recargar batería si la variable cargado es menor que 4.

Al implementar las rutas hacía casillas detectadas por los sensores, se indicará a continuación las reacciones del agente al toparse con los diferentes terrenos y entidades del mapa:

- Si justo delante del agente hay casillas las cuales no requieren un uso mayor de batería, no suponen un reinicio o simplemente permiten el avance del agente, se comprobará si tenemos una ruta fijada, si es así, se realizará el camino hacia el destino, en caso contrario realizaremos el movimiento ordinario del agente, avanzar hacia delante un cierto número de pasos y al alcanzar ese número se girará de manera aleatoria y se volverá a ir hacia delante.

- Si justo delante del agente hay un bikini, se comprobará si el agente tiene ese objeto, en caso negativo, el agente pasará sobre esa casilla y se pondrá a true la variable tiene_bikini. En el caso de que el agente ya disponga del objeto, se evitará para no pasar sobre él.

- Si el agente tiene delante unas zapatillas, el agente realizará el mismo comportamiento que realizó al encontrarse un bikini.

- Si justo delante el agente encuentra un bosque, se comprobará si el agente tiene zapatillas, en caso afirmativo, se comprobará si tiene un destino o simplemente realizará el movimiento ordinario del agente, en el caso de no tener zapatillas, se comprobará si en el campo de visión que tiene el agente sobre el bosque, hay alguna casilla que no consuma un extra de batería, en caso negativo el agente evitaría el bosque.

- Si justo delante el agente encuentra agua, se realizará el mismo comportamiento que si encuentra delante un bosque, pero en este caso se comprobará si el agente tiene el objeto bikini.

- En el caso de que empecemos o reiniciemos justo en un bosque o en el agua, en primer lugar se comprobará si en el campo de visión se divisa terreno arenoso o pedregoso, en caso afirmativo se marcará una ruta hacia ese lugar y el agente se dirigirá hacia ese destino, en caso contrario se realizará el movimiento ordinario del agente hasta que salga o hasta que divise el terreno arenoso o pedregoso.

- En el caso de que el agente se tope con un muro o un precipicio, esté lo evitará girando hacia la izquierda o hacia la derecha.
- Si el agente tiene justo delante un lobo, este lo intentará evitar cambiando la dirección actual y en el caso de tener algún destino, esté se eliminará.Si finalmente el agente se encuentra en la misma casilla que el lobo, el agente se reiniciará y perderá, los objetos, y la orientación, por lo que se pondrán las variables tiene_zapatillas, tiene_bikini y bien_situado a falso.

Por último en el método think se actualiza el valor de ultAccion y se devuelve la acción elegida.

RutaG:

Este método creará una ruta para el destino fijado, dependiendo de la posición donde se encuentre el objetivo, añadirá 1, 2 o 3 pasos, utilizando el condicional if y else if. En el caso de no estar alineado con el agente, se usará la variable Giro para una vez llegado a la misma altura, girar hacia donde está el destino.