Universitat de les Illes Balears

Escola Politècnica Superior

Curs 2021-2022

Inteligencia Artificial

Bitxos

Profesor:
Ramon Mas Sansó

Alumnos:
Pau Bonet Alcover
Joan Martorell Ferriol
Maurici Ruiz Plaza

5/11/2021

Introducción

El objetivo de esta memoria es el de explicar la estrategia que sigue nuestro agente así como la funcionalidad de cada uno de los métodos implementados.

Estrategia

Nuestra estrategia como agente será la de movernos de manera aleatoria por la zona de combate esquivando las paredes y en cada ejecución revisar los recursos que visualizamos. Si es un recurso enemigo, disparamos para destruirlo y si es un recurso nuestro o un escudo, entonces vamos a por él. Priorizamos los recursos que están más cerca.

En el caso de que nos encontremos con un agente enemigo, le seguiremos y le dispararemos. Si el agente enemigo nos dispara antes y detectamos el disparo, entonces nuestro agente activará un escudo (si tiene).

Para las situaciones complejas donde llevamos tiempo sin ver recursos o si estamos a poca vida por culpa de un enfrentamiento, nuestro agente hará uso del hiperespacio y activará un escudo.

Funcionalidad

A continuación, explicamos la funcionalidad de nuestro agente. Para ello describiremos los métodos que se han implementado en la clase.

public void movRandom (): Método que genera un movimiento aleatorio utilizando los métodos endevant() y gira(). Utilizando una variable “repetir” ejecutamos un giro una vez cada 20 ejecuciones. El giro se realiza a la derecha o a la izquierda según un número aleatorio.

private boolean detectaPared (int dist): Método que retorna true en caso de detectar una pared a una distancia menor a la que le pasamos por parámetro. Si la detecta, antes de retornar realiza un giro aleatorio para esquivarla.

private int detectaAnyRecurs (): Método que detecta el recurso o escudo más cercano. Si lo detecta se para, lo mira y retorna el tipo de recurso (1 si es un recurso enemigo, 0 si es un recurso o un escudo y -1 si no ve ningún recurso).

private void proteccioLlançament (): Método que detecta un disparo enemigo. Si el agente tiene escudos y el disparo enemigo está a una distancia inferior a 30, entonces activa un escudo.

private void dispararEnemic (): Método que comprueba si el agente está viendo un enemigo y aun tiene disparos para efectuar. En caso afirmativo mira al objetivo y dispara una vez cada 30 ejecuciones.

private void salvacio (): Método que comprueba las fuerzas del agente y si hay enemigos. Si ha recibido 3 o más disparos y las fuerzas son inferiores a 5000 entonces comprueba si tiene hiperespacios. Si tiene lo utiliza y activa un escudo. Si no tiene, entonces el agente activa escudo únicamente.

private void novaUbicacio (): Método que comprueba cuánto tiempo lleva el agente sin ver ningún recurso o escudo. Al cumplirse el tiempo utiliza un hiperespacio para cambiar de localización.

public void avaluaComportament (): Actualizamos nuestro estado. Si estamos realizando una pausa para disparar, entonces tenemos un enemigo en el visor, así que lo miramos. En caso contrario comprobamos si estamos en colisión. Si estamos colisionando ponemos el valor de la variable "state" a 0 y si no lo estamos comprobamos el entorno llamando al método detectaAnyRecurs(). Según el valor devuelto realizaremos las siguientes acciones:

- state = -1 (no se observa ningún recurso)
  Aumentamos nuestra variable que cuenta el tiempo que llevamos sin ver recursos y llamamos a los métodos:
  - detectarPared(14)
  - movRandom()
  - proteccioLlançament()
  - salvacio()
  - disparaEnemic()
  - novaUbicacio()

- state = 0 (se observa un recurso o un escudo)
  Ponemos a cero nuestra variable que cuenta el tiempo que llevamos sin ver recursos y llamamos al método detectarPared(14). Si estamos en colisión el agente se moverá hacia atrás y girará durante 5 ejecuciones. Si no está en colisión avanzará hacia el recurso o escudo.

- state = 1 (se observa un recurso enemigo)
  Ponemos a cero nuestra variable que cuenta el tiempo que llevamos sin ver recursos y llamamos al método dispararEnemic(). Si no tenemos una pared a una distancia inferior a 40 entonces disparamos a ese enemigo.