Inteligencia Artificial

Memoria: El Robot Trufero
Agente Reactivo

Lothar Soto Palma
Universidad de Granada
14 de abril de 2015

Índice

Índice de figuras

Idea Inicial

La idea inicial del problema fue hacer que nuestro agente fuera capaz de reaccionar dentro de un entorno en primer lugar desconocido, de manera que a medida que fuera explorando dicho entorno este fuera capaz de realizar decisiones de tipo reactivas, es decir, se basa en el conocimiento de un área limitada del entorno a su alrededor para la toma de decisiones. Para llevar a cabo esto lo primero es darle memoria a nuestro agente, dotarlo de una matriz lo suficientemente grande como para albergar un "mapa"del entorno, debido a que cada mapa es distinto y no es posible preveer la forma y los obstaculos del mismo, se hace necesaria la idea de introducir un camino que el agente deba seguir de manera que todo el mapa sea explorado ya que no podemos limitarnos a una zona del mapa.

Por lo que el agente tomará decisiones con respecto a su alrededor y la antiguedad de cada una de las casillas que le rodean, olfateará y recogerá en caso de que la cota de recogida lo permita y por último el giro vendrá dado por la posición y la orientación en la que se encuentre el propio agente, todo esto con el fin de intentar reducir al mínimo el número de pasos "perdidos".

Comportamiento

El comportamiento del agente se basa en tres fases:

1. **Exploración**: Debido a que los primeros pasos del agente no son beneficiosos en lo que a la recolección y olfateo se refiere intentamos ahorrarnos el olfateo en los 100 primeros pasos (dependiendo), de manera que comenzará a olfatear a partir de entonces cuando las trufas estén más crecidas, podemos diferenciar dos casos:

- **Crecimiento Lento**: Si el crecimiento es lento, exploramos el mapa en los 100 primeros pasos ahorrandonos al menos otros 100 pasos que se realizarían al olfatear en cada una de las posiciones, debido a que las trufas no crecen tan rápido como para pudrirse en lo que el agente da 100 pasos.
- **Crecimiento Rápido**: Finalmente se probó una implementación que permitida diferenciar el crecimiento modificando su condicion de recogida pero no resultó satisfactoria (ver Modificación en recogida).

2. **Vuelta al inicio**: El agente como va a ir a la posición más antigua de las que le rodean (posiciones horizontales o verticales a él) va a acabar volviendo al inicio y como ha estado explorando y dejando crecer las trufas, una vez tiene explorado gran parte de el mapa comienza la recolección.
3. **Recolección**: El agente irá olfateando y recogiendo una vez se ha explorado el mapa ademas el algoritmo seguido impide que este vuelva a chocarse con paredes por lo que también ahorraremos pasos con eso, por último la recolección se realiza conforme a una cota, es decir, si al olfatear hay un número de trufas mayor al mínimo de recolección recogerá dicha posición.

Diseño de la estructura del agente

Analizaremos el agente y su estructura interna, es decir analizaremos agente.h y sus variables y funciones definidas:

- **mapa[19][19]**: Matriz que se usa para ir guardando el mapa, vamos a definir una leyenda de valores para poder interpretar el mapa:
  - -1 : Valor que se usa para indicar que en esa posición hay una pared o obstaculo.
  - 0 : Valor que indica que no se ha explorado la posición.
  - $[1, 2000]$: Valor que indica el paso en el que el agente exploró la posición.

  Entonces como podemos observar en la figura 1, el mapa al inicializarse lo hace con todas las posiciónes a 0 y en ese caso la primera vez que avance la casilla en la que se encontraba marcará 1. En la figura 2 vemos como el agente ha explorado el mapa determinando la localización de

las paredes y se mueve por el dirigiendose a la posición más antigua o que hace mas turnos que no visita.

Figura 1: Estado inicial del Mapa.

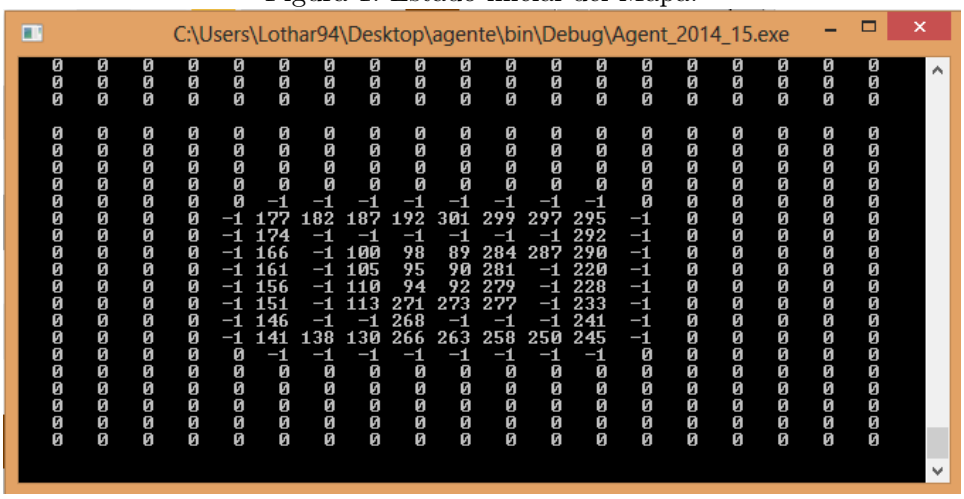

Figura 2: Estado avanzado del Mapa.

- **pos_x, pos_y**: Indican la posición en la que se encuentra el agente, de forma inicial pos_x = 9 = pos_y para que se encuentre aproximadamente en el centro de la matriz, y a partir de ese momento incrementar o disminuir en función de la dirección del agente.
- **paso**: Guarda el paso (turno) en el que se encuentra el agente para poder ir cambiando de acción, además esta variable se va guardando en cada posición de la matriz que no sea una pared para encontrar la posición más antigua con la función obtenerDireccion().

- **director**: Nos indica la dirección en la que se encuentra el agente, hace uso de los valores 0,1,2,3 y por ejemplo si comienza mirando en dirección norte la sucesión es norte, oeste, sur, este respectivamente.
- **recogida**: Indica el número de trufas que se ha recogido hasta el momento, puede ser de utilidad pero finalmente se obtuvieron mejores resultados sin usarlo.
- **ultimo_paso**: Contiene la acción anterior, gracias a ella es que el agente es capaz de saber cuando recoger o oler entre otras.
- **obtenerDireccion()**: Esta es una típica función de minimo entre cuatro valores, los valores relacionados con las posiciones que se encuentran adjacentes al agente, haciendo uso de la variable paso, además también introduce una mejora que es si el agente tiene una posición inexplorada adjacente en la dirección en la que se encuentra escoge dicha dirección.
- **Mostrarmat()**: Función simple que nos permite mostrar el estado de la matriz.

Selección de acciones

La selección de acciones del agente implementado es sencilla, se basa en:

- Avanzar en la dirección que indique la función obtenerDireccion() en caso de que en la posición que este justo delante de la dirección en que te encuentres sea 0 seguimos avanzando en esa dirección no cambia.
- Cuando el agente se choque con un obstaculo gira a la derecha o izquierda dependiendo de lo que le cueste menos, si girar a la derecha le permite dar un solo giro para obtener la dirección deseada gira a la derecha si no por defecto siempre gira a la izquierda.
- A partir del paso 100, comenzamos el olfateo siempre que no nos choquemos y anteriormente se avanzó, esto nos permite comprobar si la posición en la que nos encontramos tiene 10 o más trufas.
- Si tiene 10 o más trufas y en el paso anterior se olfateo, entonces se recogen.
- Además ya no volvera a chocarse, puesto que la función obtenerDireccion() lo impide ya que el valor -1 se considera como inaccesible por lo que no volverá a darte una dirección en la que se encuentre un obstaculo.

Resultados y otros algoritmos

Vamos a analizar los resultados obtenidos finalmente con los obtenidos con algoritmos con ligeras modificaciones.

Resultados finales

Los resultados finales obtenidos son los representados en las siguientes dos tablas:

Mapas	agent.map	mapa1.map	mapa2.map	mapa3.map
Total(10 runs)	13744	15230	14696	15364
Media(10 runs)	1374.0	1523.0	1469.6	1536.4

Mapas	agent_rap.map	mapa1_rap.map	mapa2_rap.map	mapa3_rap.map
Total(10 runs)	29036	31326	30952	30686
Media(10 runs)	2903.6	3132.6	3095.2	3068.6

Modificación en recogida

Se intento mejorar los resultados cambiando la condición de recogida, lo que se intentó fue ir almacenando cada 200 pasos la diferencia del total de trufas recogidas menos las que se recogieron hace 200 pasos, y si esa diferencia es menor o igual que un valor establecido (se uso el valor 150 debido a que era la diferencia mayor que se solia dar en los mapas de crecimiento rapido y no se alcanzaba en los lentos) la acotación de la condición de recogida era reducida ya que eso quiere indicar que hemos recogido muchas y por tanto las trufas del mapa pueden haber crecido mucho, en caso contrario se establecia la cota a la que se tiene por defecto los resultados obtenidos fueron:

Mapas	agent.map	mapa1.map	mapa2.map	mapa3.map
Total(10 runs)	13744	15230	14696	15364
Media(10 runs)	1374.0	1523.0	1469.6	1536.4

Mapas	agent_rap.map	mapa1_rap.map	mapa2_rap.map	mapa3_rap.map
Total(10 runs)	29034	31278	30942	30740
Media(10 runs)	2903.4	3127.8	3094.2	3074.0

Como se puede ver deja a todos los mapas de crecimiento lento igual pero los de crecimiento rápido se ven afectados aunque reducen su recogida, recoge menos en todos los casos a excepción del último mapa3_rap.map que recoge un poco más, por esto determine quitar la condición de recogida y recoger solo cuando nos encontraramos con 10 trufas o más.

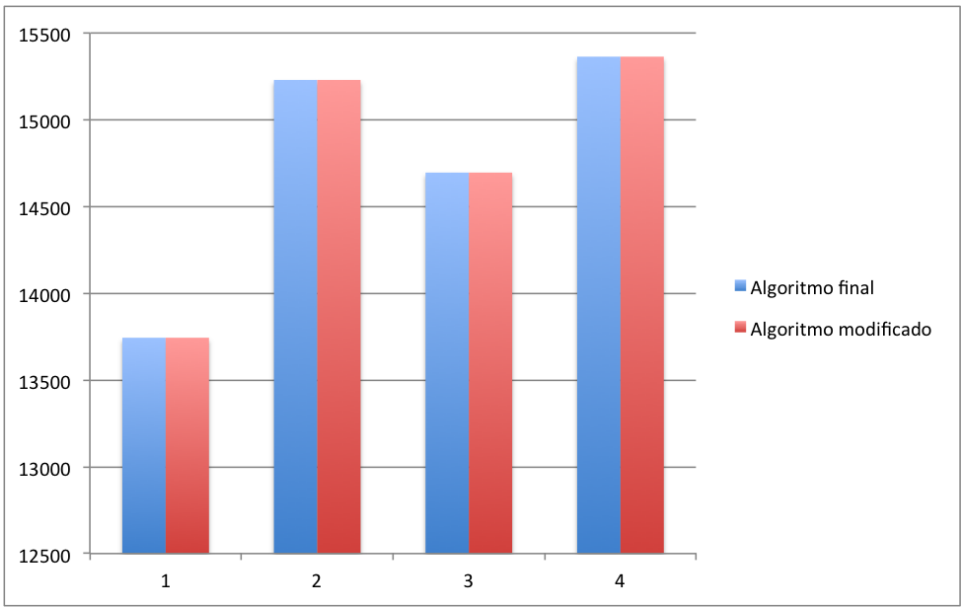

Figura 3: Gráfica mapa crecimiento lento.

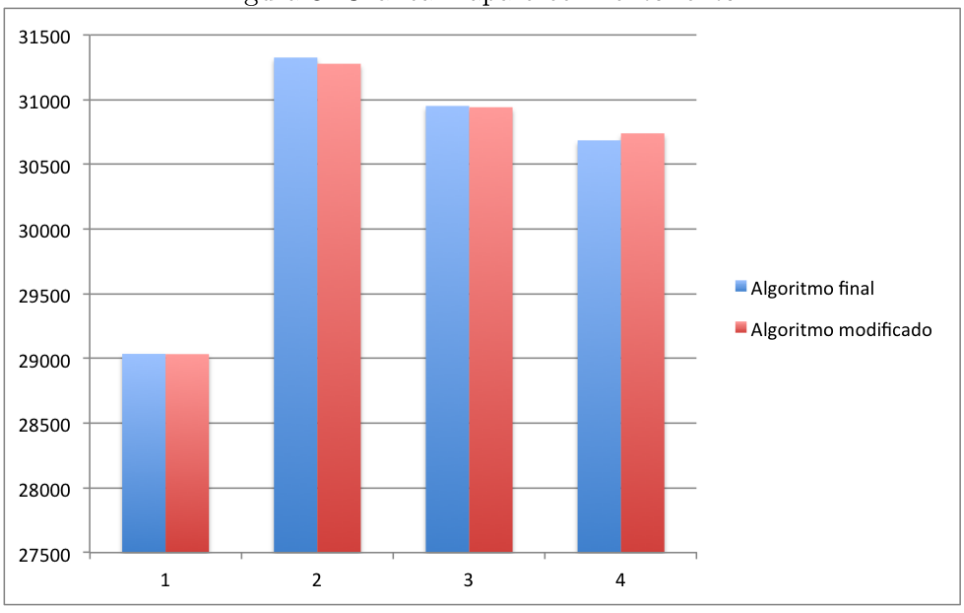

Figura 4: Gráfica mapa crecimiento rápido.