Algoritmo para desarrollar un autómata que reconozca ambientes desconocidos

Carlos Gerardo Euresty Uribe
Instituto Tecnológico de Celaya
gerardo.euresty@itcelaya.edu.mx

Francisco Gutiérrez Vera
Instituto Tecnológico de Celaya
francisco.gutierrez@itcelaya.edu.mx

Resumen

Hacer un autómata que descubra un ambiente desconocido por medio de un algoritmo heurístico, el objetivo es que el programa logre el reconocimiento de un mapa de dos dimensiones por medio del autómata que utilice distancias para el reconocimiento del área en dónde el autómata se mueve.

Palabras Clave: Algoritmo, Heurística, Inteligencia artificial.

Abstract

Do an automaton that discovers an environment not known by means of a heuristic algorithm, the aim is that the program achieves the recognition of a map of two dimensions by means of the automaton that uses distances for the recognition of the area where the automaton moves.

Keywords: *Algorithm, Heuristics, artificial intelligence.*

Introducción

En la actualidad la mayoría de los modelos de arquitectura de inteligencia artificial proponen nuevas técnicas, para hacer que un autómata se convierta en un ser consciente, a través de métodos diversos como los algoritmos genéticos, las redes neuronales o los algoritmos basados en la entropía, sin embargo para crear un ser consciente es necesario iniciar desde un concepto más simple de conocimiento, a partir del cual antes de saber cómo va a operar un autómata debe ser consciente de su entorno y de las cosas que existen a su alrededor, estas nuevas formas de manejar el comportamiento de un robot han permitido crear sistemas hasta cierto punto autónomos como el explorador Curiosity de Marte ya que debe mantener la prioridad en los objetivos y tareas de exploración en el planeta, evitar obstáculos y peligros que de otro modo tardaría demasiado al enviar información a la Tierra y esperar instrucciones, es decir, las decisiones parciales o totales son responsabilidad de él. El presente trabajo tiene como objetivo crear un modelo de inteligencia artificial basado en nuevas tendencias para generar un razonamiento similar al pensamiento del ser humano que permitan un reconocimiento de ambientes haciendo posible descubrir el ambiente de trabajo antes de proponer cómo el autómata pueda tomar decisiones y realizar acciones en consecuencia.

Uno de los objetivos clásicos de la inteligencia artificial ha sido la simulación del razonamiento humano con el fin de proporcionar a un autómata de dicha funcionalidad, comenzando con una fase de percepción seguida por otra de comprensión. La simulación del mecanismo artificial de “pensar” requiere de una fase previa de análisis en la que obtenemos un modelo de cómo realizamos la interpretación al resolver una situación desde el punto de vista de un ser racional.

Usualmente los métodos heurísticos determinan buenas soluciones con tiempos razonables de ejecución, una conceptualización heurística tomada del pensamiento humano debería de funcionar, porque las personas encuentran soluciones en periodos

relativamente cortos de tiempo y casi siempre con resultados aceptables, debido a que esos raciocinios provienen de la experiencia para solucionar un problema. Preguntarnos como pensamos una solución es una pregunta sin respuesta, pero trabajando a la inversa, es decir, al tratar de descifrar como obtuvimos las respuestas, se pueden encontrar reglas no escritas que nos permiten simular el funcionamiento del cerebro y que representan mapeos intrínsecamente correctos y completos para el desempeño de un autómata.

A pesar de los avances en todas las ciencias, la aproximación a la estructura del cerebro aún son básicas, cómo un cerebro recuerda, cómo piensa y toma decisiones, cómo olvida y se retroalimenta cuando dormimos, es importante saber que somos buenos para ciertas actividades, pero no sabemos exactamente como tomamos decisiones y comprobamos las reacciones que nos hacen seres inteligentes.

Existen proyectos que están siendo desarrollados a través de métodos heurísticos y bio-inspirados proponiendo que el primero que logre desentrañar el funcionamiento biológico o abstracto del cerebro tendrá una herramienta de aprendizaje casi ilimitada.

En el área de la mecánica cuántica se están desarrollando máquinas con inteligencia artificial que funcionen de manera similar a la inteligencia biológica, ya que existen ciertos paralelismos entre una máquina cuántica y el funcionamiento de la mente humana, el cerebro se caracteriza por la reorganización de la neuronas basado en estímulos, lo que indica que el aprendizaje biológico es distinto al aprendizaje de la máquina clásica.

El análisis de antivirus o malware por medio de la heurística ayuda a detectar archivos potencialmente dañinos por medio de un comportamiento basado en reglas para diagnosticar nuevos virus que mutan su código, que son nuevos o desconocidos en base a la detección proactiva de firmas reactivas del virus, de ahí su importancia para la detección de archivos maliciosos.

Problemática

Crear un autómata que tenga un comportamiento autónomo, (según el diccionario: máquina automática programable capaz de realizar determinadas operaciones de manera autónoma y sustituir a los seres humanos en algunas tareas, en especial las pesadas, repetitivas o peligrosas; puede estar dotada de sensores, que le permiten adaptarse a nuevas situaciones). Así el problema que surge es que el autómata sepa cómo es su ambiente para su desplazamiento, la perspectiva inicial es colocar al autómata dentro de un espacio cerrado completamente desconocido, en dónde lo que único que sabe el autómata son las dimensiones del área y la posición en dónde se encuentra, el objetivo del robot es iniciar un proceso de búsqueda con movimientos en el eje horizontal y vertical, que le permitan reconocer la mayor parte posible del terreno antes de que pueda realizar otras funciones sobre el mismo.

El área a investigar por el autómata sería un laberinto cerrado de dos dimensiones, en donde este no contiene otros elementos aparte de espacios vacíos y muros que dividen el laberinto en segmentos más pequeños, para facilitar el proceso de búsqueda inicial. Otro problema consiste en determinar cómo el autómata deberá ver su ambiente como por ejemplo a través de visión artificial u otros métodos para lograr detectar cuales son las características de su entorno.

Metodología

Antecedentes del autómata.

Se pretende que el autómata no utilice técnicas de búsqueda de solución como Hill *climbing* o A*, sino que se basara en un algoritmo heurístico que simule el comportamiento de un niño.

El proceso de elaboración del autómata fue incremental debido a que el modelo original está basado en una algoritmo heurístico y por lo tanto es necesario comprobar que tan buen oes el prototipo para solucionar el problema y a partir de ahí corregir e

incrementar las reglas del algoritmo tratándose de acercarse cada vez más a una solución óptima o eficiente, por lo tanto hubo algunos inconvenientes que obligaron a revisar, modificar y alterar la secuencia del ejecución del algoritmo y la cantidad de estados utilizados, por ejemplo cuando la primera versión del algoritmo funcionó bien en un laberinto convencional se procedió a probarlo en un laberinto más elaborado, en dónde si al mapa se le aplicaba la regla de la mano derecha sobre una de las paredes del laberinto, se podía recorrer todas las área dentro de él y en algún momento regresar al punto de origen, al notar que el programa perdía zonas y otras nunca las detectaba se modificó el comportamiento del autómata hasta que logrará reconocer la mayoría de las áreas en el laberinto.

Otro ejemplo de modificación al algoritmo fue la eliminación de una de las memorias, originalmente existía una memoria de corto plazo y otra de largo plazo, para simular como aprende el cerebro humano, sin embargo la implementación de estas memorias traía más problemas que beneficios al detectarse que el algoritmo utilizaba casi siempre la memoria de corto plazo, por lo que dejó únicamente esta.

Funcionamiento del autómata.

Se trató de dotar al autómata de un comportamiento ayudado por una coalición de procesos similares a los del cerebro humano, en donde básicamente existen sensores que le dan nueva información, esta información a su vez se almacena en su memoria, se mueve a nuevas posiciones y finalmente el sentimiento o la parte afectiva se maneja como estados en los que el autómata puede estar y realizar ciertas funciones de acuerdo al estado actual.

Estas definiciones forman coaliciones que le permiten al autómata cumplir con su propósito. A partir de estos estados de transición se tomaron hechos heurísticos basados en el comportamiento de un ser humano, al explorar un ambiente desconocido definidos en los puntos siguientes:

1. Si existe incertidumbre (no sabemos hacia dónde movernos), cualquier camino es bueno.

2. Encontrar nuevos lugares nos motiva a recordarlos y continuar explorándolos en su momento.

3. La memoria espacial es muy útil para reconocer los lugares que ya hemos visitado o agotado para descartarlos.

4. Los caminos desconocidos deben registrarse en la memoria para su posterior investigación.

Por lo tanto para el proceso de reconocimiento del autómata podemos establecer tres estados de comportamiento, inicialmente el autómata no sabe nada de su ambiente pero tiene que decidir qué camino tomar para comenzar su trabajo de búsqueda, el estado inicial del autómata sería *confundido*. En este estado simplemente se obtiene una alternativa al azar viable (que no es un camino bloqueado) para hacer el rastreo, las direcciones en las que puede caminar el autómata se definieron hacia arriba del mapa como *norte*, hacia la derecha *este*, hacia abajo *sur* y hacia la izquierda como *oeste*.

A partir del estado anterior el autómata cambia a un nuevo estado de *caminando* que le permite detectar la distancia a la que se encuentra de objeto sólido por ejemplo una pared, por medio de un sensor de distancia ubicado al frente del autómata y posteriormente recorrer ese camino hasta “topar” con pared, cada vez que el autómata camina el mapa se actualiza, en este estado se utilizan dos sensores colocados a la izquierda y derecha del autómata para registrar su ambiente alrededor y determinar si hay rutas nuevas lo suficientemente interesantes para guardarse en su memoria.

El tercer estado denominado *recordar* le permite al autómata recuperar la última dirección registrada en su memoria para regresar al estado *caminando* obteniendo la última posición y dirección que se debe investigar, en caso de que no existan datos en la memoria del autómata regresaría al estado *confundido*, para continuar navegando en el mapa. En la figura 1 se muestra un diagrama de estados del autómata en dónde el estado final se determinaría cuando el porcentaje de ambiente encontrado por el autómata igualará o superará el 90% del mapa descubierto.

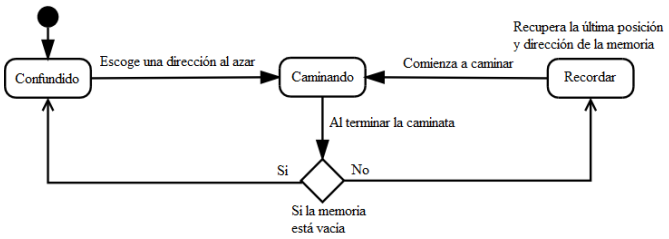

Figura 1. Diagrama de estados del autómata.

El algoritmo de búsqueda consiste en un método que permite al autómata llevar a cabo su trabajo de búsqueda, en la figura 2 se muestra el pseudocódigo.

Existen tres bloques de decisiones en donde cada bloque representa un conjunto de instrucciones para cada posible estado del autómata. En el estado caminando el robot sabe a qué distancia se encuentra una pared y por la tanto conoce el número de pasos que puede avanzar antes de chocar, por lo tanto en la primera decisión si la variable numeroDePasos es diferente de cero, el autómata camina un paso hacia adelante con el método caminaUnpaso(), este método al final también reduce el contenido de la variable numeroDePasos en uno. A continuación se llama a los métodos activaSensorIzquierdo() y activaSensorDerecho() para determinar las distancias de las paredes en su nueva posición, el último método actualizaMapa() actualiza los valores del mapa para mostrarlos en la pantalla.

En caso de que la variable numeroDePasos sea igual a cero el autómata verifica para su siguiente estado *recordar* la posibilidad de usar una dirección de su memoria, para continuar su proceso de búsqueda. Si su memoria no está vacía, el robot recuperará las coordenadas *x* e *y* además de la dirección en la que puede seguir su rastreo comprobando cuantos pasos puede dar ahora con el método activaSensorDelantero() y asignándole el número de pasos a la variable numeroDePasos.

Si el autómata no tiene información en su memoria se encontrará en la opción SINO de la segunda decisión anidada y por lo tanto en el estado *confundido*, de ser así el autómata determinará una posible dirección al azar por dónde pueda seguir caminando

sin obstáculos, de nuevo activando su sensor delantero y determinando el número de pasos que puede avanzar en esa dirección.

```
MÉTODO DE BÚSQUEDA
      INICIO
          SI numeroDePasos <> 0                // Estado caminando
              caminaUnPaso()
              activaSensorIzquierdo()
              activaSensorDerecho()
              actualizaMapa()
          SINO
                 SI memoriaDelRobot NO vacia    // Estado recordar
                 recuperaCoordenadasXY()
                  recuperaDireccion()
                 numeroDePasos ← activaSensorDelantero()
              SINO
                 escogeCaminoAzar()           // Estado confundido
                 numeroDePasos ← activaSensorDelantero()
      FIN
```

Figura 2. Pseudocódigo del método de búsqueda.

Si el autómata encuentra nuevas direcciones para registrar en su memoria una forma de discriminar si vale la pena guardar la dirección es verificar ese candidato contra la dirección de la última referencia almacenada, en la figura 3 se muestra un ejemplo de cómo ignora el autómata ciertas direcciones escaneadas con los sensores laterales debido a que puede caminar de la posición original a la posición actual del autómata y regresar al punto de origen formando una rectángulo sin encontrar un objeto que se interponga en el camino, es este caso la flecha verde es la última dirección registrada y la flecha roja es la dirección candidata para verificar si es descartable o no. Si es posible ir de la dirección almacenada y regresar al punto de origen entonces ese camino no vale la pena registrarlo en la memoria. El autómata está representado en la figura como un cuadro rojo y las flechas negras indican la dirección en la que está avanzando el robot, es decir, hacia el norte.

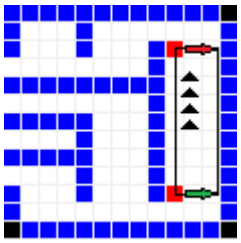

Figura 3. Eliminación de caminos no interesantes.

Implementación.

El software se desarrolló en C# para crear rápidamente una interfaz de prueba que permita registrar dos factores importantes para los resultados de eficiencia, el tiempo que tarda el robot en completar el mapa y el porcentaje de áreas encontradas.

En el software se utilizan dos mapas, el primer mapa llamado MapScanning es una matriz de ceros y unos que nunca se muestra en la pantalla de la aplicación, en donde cero (0) representa un espacio vacío y uno (1) un obstáculo, es decir un objeto sólido, este concepto se usa para simular el funcionamiento de los sensores de distancia, se está suponiendo que los sensores soportan la detección de cualquier distancia en el mapa, esto en la práctica no es posible debido a que los sensores tiene un alcance máximo, pero esto se ajustará cuando se llegue a la implementación del modelo en hardware.

El segundo mapa llamado MapUnknow es una matriz completamente llena con un número que representa cuadros negros cuando el mapa se dibuja en pantalla para mostrar las zonas descubiertas y desconocidas en él.

En la figura 4 se muestra un ejemplo de la pantalla de la aplicación para mostrar como el autómata (cuadro rojo) encuentra nuevas áreas y va dibujando su mapa actualizando las áreas conocidas y mostrando en las áreas oscuras lo que le falta por explorar. En la pantalla también se muestra información sobre el autómata como sus coordenadas, la dirección en la que está caminando y su estado e información del mapa mostrando el número de bloques, el porcentaje del mapa descubierto y el tiempo de ejecución.

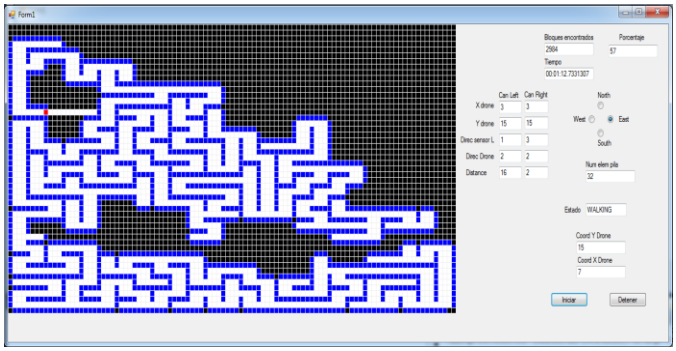

Figura 4. Algoritmo de reconocimiento espacial (ARE).

Resultados

Para las pruebas de eficiencia del autómata, el software se ejecito en una computadora con un procesador Intel Core i5, con una velocidad de 3.3.Ghz y con 4 GB de memoria, se eligieron cuatro tamaños de laberintos de prueba: suponiendo que el autómata mide 30 unidades^2, se creó un laberinto de 90x90 unidades^2 metros, el segundo tipo de laberinto tiene un tamaño de 180x180 unidades^2, el tercero es de 270x270 unidades^2 y el último laberinto corresponde a 360x360 unidedes^2. Se crearon 10 laberintos de cada tipo para tomar el tiempo que el autómata necesitaba para alcanzar al menos el 99% del área reconocida, la parte gráfica no se tomó en cuenta para obtener los tiempos reportados, se crearon 10 laberintos diferentes de cada categoría para poder obtener tiempos promedio y desviación estándar. En la tabla 1 se muestra un concentrado de los tiempos para cada ejecución del autómata.

Tabla 1. Resultados de la ejecución del autómata.

Identificación de laberintos	Tamaño del laberinto			
	30x30 u^2	60x60 u^2	90x90 u^2	120x120 u^2
Mapa 1	9.687617	47.361683	1:38.28017	2:56.43630
Mapa 2	9.974996	43.960877	1:39.18497	2:59.58751
Mapa 3	11.282419	44.662878	1:38.56097	2:56.15530
Mapa 4	11.044819	43.929677	1:37.96817	2:58.27711
Mapa 5	11.700020	44.616078	1:40.02737	2:58.33108
Mapa 6	10.998019	43.617676	1:38.65457	2:55.71870
Mapa 7	9.921617	45.301868	1:39.06017	2:58.32391
Mapa 8	10.389618	44.460078	1:40.58897	2:56.42070
Mapa 9	11.495617	45.598880	1:36.64216	2:54.48630
Mapa 10	10.342818	45.302479	1:40.81516	2:57.85591
Tiempo promedio	10.683756	44.881217	1:38.97827	2:57.15928
Desviación estándar	0.71162422	1.08865254	1.26326469	1.55123197

En la tabla 1 los tiempos están definidos en minutos, segundos y centésimas de segundo con un intervalo de tiempo entre cada ciclo de operación del autómata de 40 milisegundos, aunque no es una referencia de tiempo real en la que el autómata trabajaría, sólo se está usando para determinar cuál es el orden del algoritmo.

Una vez obtenidos los tiempos promedio de las pruebas, se utilizó el software CurveExpert para usar un método de regresión lineal que nos diera una fórmula de comportamiento del algoritmo, en la figura 5 se muestra la gráfica de la ecuación obtenida. En la figura se observa que el tiempo de ejecución aumenta de manera considerable cuando el tamaño del área a reconocer es más grande y el comportamiento del algoritmo es lineal.

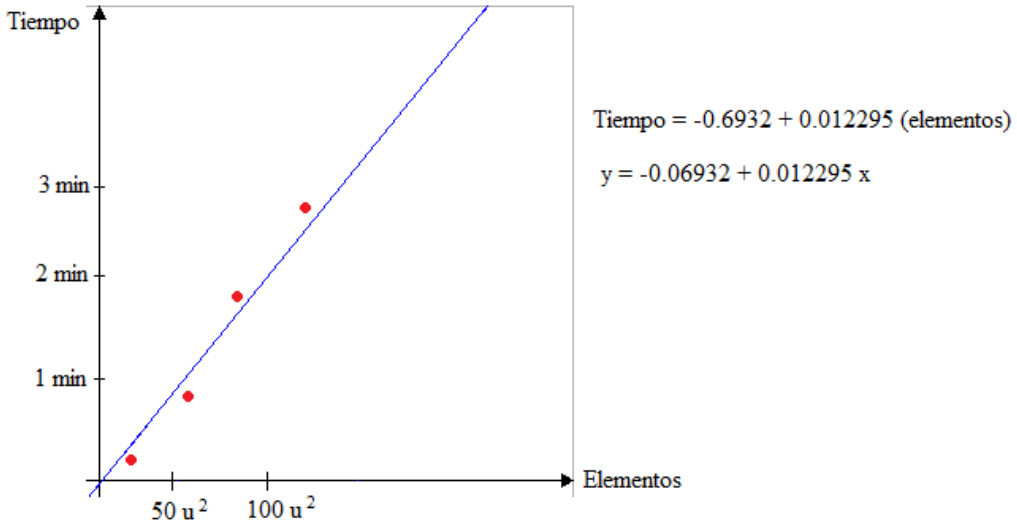

Figura 5. Gráfica de comportamiento del algoritmo.

Discusión

Se logró hacer un algoritmo que encuentra la mayoría de las áreas en el mapa en un periodo de tiempo, cuyo proceso de obtención de resultados debe mejorarse substancialmente. Este prototipo tiene muchos puntos de oportunidad, por ejemplo. Se puede mejorar la eficiencia del algoritmo creando una memoria que almacene en un orden diferente las direcciones que va registrando, es decir, guardando las nuevas direcciones más cercanas al autómata al inicio de la memoria y las más lejanas en posiciones finales de la memoria. Esto evitará que el autómata deje zonas cercanas sin revisar y avance sin muchos retrocesos en el laberinto.

Aunque no se pudo obtener un algoritmo que superara los tiempos que obtiene Hill climbing o A* es interesante poder generar otras formas de trabajar, es decir, muchas veces la única forma de saber si una hipótesis es cierta es experimentando, se aprende más de una falla que de muchos aciertos.

Si se pretendiera implementar este modelo en un ambiente real de tres dimensiones obviamente será necesario mejorar la heurística para adaptarse a los nuevos objetivo de búsqueda y aumentar el nivel de complejidad en el mapa a fin de hacerlo factible para nuevos proyectos de desarrollo, así como hacer una revisión sobre los tipos de sensores que existen para el reconocimiento del ambiente por medio de la visión del autómata.

Se pretende crear un autómata con hardware y sensores para probar la teoría en la realidad lo que permitirá mejorar el software en base a las respuestas del robot. La finalidad al implementar este algoritmo es crear un dron autónomo que tenga funciones definidas como por ejemplo entregar paquetes o ser un guardián para controlar la vigilancia de un almacén o una zona industrial.

Bibliografía

[1] ALEJANDRO MADRUGA GONZÁLEZ, *Inteligencia Artificial, el futuro del hombre*, Futuro del libro.

[2] MARÍA ISABEL ALFONSO GALIPIENSO, MIGUEL ANGEL CAZORÍA QUEVEDO, *Inteligencia Artificial, modelos técnicas y áreas de aplicación*, Paraninfo.

[3] DANIEL BORRAJO, *Inteligencia Artificial, métodos y técnicas*, Centro de Estudios Ramón Areces, 1993.