Visión por Computadora

Nestor D. Castillo
nestor.castillo@ucuenca.edu.ec
Universidad de Cuenca

***Abstract*—Se utilizara la herramienta MATLAB con el fin de poder encontrar el camino mas óptimo utilizando búsquedas heurísticas, el mapa a analizarse contendrá objetos y obstáculos, a los cuales se les aplicara segmentación y operaciones morfológicas para poder realizar un análisis de nodos y aplicar una heurísticas como lo es la distancia euclidiana en conjunto a pesos de cada nodo.**

***Index Terms*—Algoritmo A*, Pathfinding, Búsquedas Erísticas, Visión por Computador**

I. Introducción

El propósito del siguiente proyecto proponer la solución de un problema industrial asociado a la planeación de trayectorias en bodegas, almacenes y/o pisos de planta. La planeación de trayectorias usando inteligencia artificial y visión por computador puede usarse, entre otras aplicaciones, para navegación de vehículos autónomos terrestres (UGV, por sus siglas en inglés) [1], minimización de tiempos de recorrido en rutas dentro de una planta, identificación de ubicaciones específicas, como por ejemplo de contaminantes [2], etc.

La Inteligencia Artificial (IA) o Artificial Intelligence (AI) en inglés, es una rama de la informática que estudia procedimientos automatizados para lograr que un autómata logre ser o parecer inteligente [3]. Éstos, tratan de imitar diversas áreas del comportamiento humano, con el fin de acercarse o llegar a superar a una persona común y corriente, por ejemplo, un experto en cierta materia, en la toma de decisiones o diversas tareas que un sistema computacional puede realizar mejor o más rápido. Existe un área de la inteligencia artificial llamada Pathfinding, la cual responde a la problemática de ser capaz de moverse de un punto a otro, esquivando, de forma inteligente y natural, obstáculos que se pueden presentar [4].

La búsqueda es una de las técnicas más utilizadas para resolver los problemas de pathfinding. De los distintos tipos de algoritmos de búsqueda, los algoritmos búsqueda heurística completa se encuentran ampliamente difundidos. Sin dudas, el algoritmo A* es el algoritmo de búsqueda heurística más popular [5].

Dentro de la inteligencia artificial se pueden establecer varias analogías entre la visión humana y por computador, ya que ambas tendrán un elemento sensor (el ojo y la cámara) y un procesador de la información (el cerebro y el computador) aunque nunca perdiendo de vista que no se trata de imitar o replicar el sentido de la vista en el hombre. A través de los ojos recibimos información de la posición de los objetos del entorno, determinamos el camino libre de obstáculos, distinguimos lo que se mueve de lo estático, calculamos trayectorias, reconocemos a personas, detectamos la presencia de posibles peligros, analizamos el color de las imágenes, etc. Por esta razón debido a la importancia del sentido de la vista humana era lógico que, con la aparición de los primeros ordenadores, una de las primeras aplicaciones en las que se investigara fuera la visión artificial: el análisis de imágenes a través de computadores para obtener una descripción de los objetos físicos que son captados por la cámara [6].

A. Contexto del proyecto

El proyecto se desarrolla bajo la idea de tener diferentes Imágenes que visualizan un mapa, en donde utilizando las técnicas de procesamiento de imágenes estudiadas en el capitulo de Visión artificial del presente curso se busca poder trazar en la imagen un camino que trace el recorrido entre un punto de inicio hacia un punto final teniendo en cuenta que durante el proceso podemos encontrarnos con diferentes obstáculos, Las imágenes con las que se va a trabajar se observan en la Figura 1;

Fig. 1. Imágenes a trabajar

II. Descripción metodológica

A. Localización de Patrones a través de la correlación cruzada

Para iniciar con el desarrollo del proyecto, se necesita encontrar las coordenadas que definen el inicio y el final del camino, cada una de las Figuras con las que se trabajarán

tienen definido y descrito la palabra Inicio y Meta, hace falta un procedimiento para obtener las coordenadas respectivas, para éste proceso se plantea utilizar la herramienta de correlación cruzada; En el código de inicio otorgado por el docente, se tiene un parte que sustrae los índices de posición de cada figura circular que se encuentre en la imagen, como ejemplo se tomará la figura 'layout0.jpg' que se observa en la Figura 2 donde se tiene dos círculos y cada uno etiquetados con palabras claves que representan el inicio y el final.

Fig. 2. Imágen de ejemplo

Para poder localizar los puntos de inicio y de meta, se plantea recortar las etiquetas mencionadas en la Figura 2, y poder obtener los índices de posición de cada etiqueta y a partir, de estas posiciones se busca encontrar los índices de posición de cada circulo con lo que se tenga menor distancia, como se observa en la Figura 5.

Fig. 3. Etiqueta de Inicio

Fig. 4. Etiqueta de Final

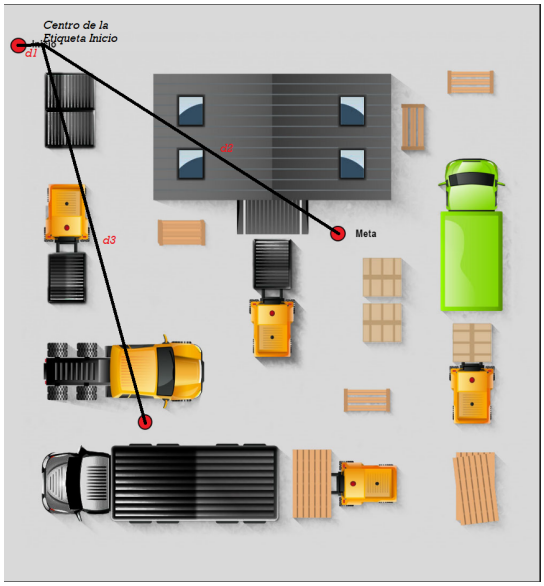

Fig. 5. Ejemplo de Distancias de las circunferencias con respecto al centro de la etiqueta

B. Tratamiento de Segmentación y Operaciones de Morfología

Para la imagen en escala de grises se establece una binarización de acuerdo a un umbral que se tomara justamente en valle que se encuentre en el histograma de de nivel de grises, al ser de un canal, observamos que la representación del histograma se verifica en la figura 6, en donde si verificamos en el nivel de 200 se encuentra un valle, a este valor lo definiremos como umbral, el cual realizando un barrido por toda la imagen, todos niveles por encima de este valor lo definiremos con un 1 (blanco) y por debajo del mismo lo definiremos como 0 (negro). [7]

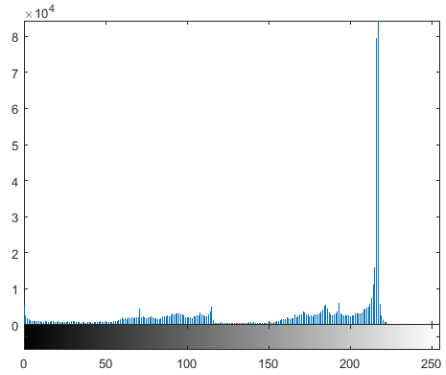

Fig. 6. Histograma de escala de grises de la imagen.

Con esto ya binarizariamos la imagen, una visión previa seria la que se describe en la figura 7, en donde se binariza la imagen con lo expuesto anteriormente.

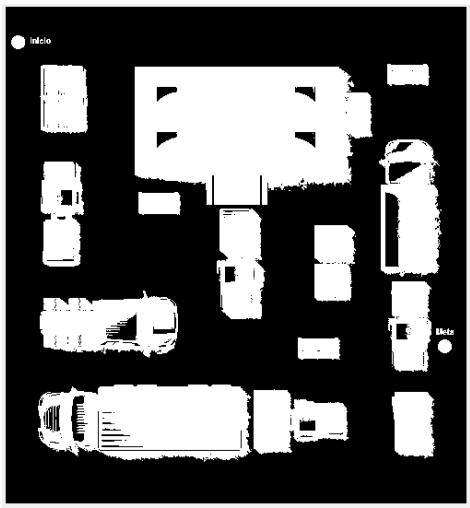

Fig. 7. Imagen Binarizada

Ahora lo que procedemos a realizar es simplicar lo mas que podamos a la imagen para poder tratarla, aqui lo que procedemos a hacer es a obtener los bordes de los objetos existentes, se puede utilizar las distintas ventanas para la detección perimetral (roberts, canny, sobel, prewitt). Para el presente trabajo se utilizara la ventana de *canny*, una previsualización se aprecia en la figura 8 en donde obtiene las orillas de la imagen. Matlab nos ayudara a encontrar las orillas de una manera sencilla con el metodo *edge*.

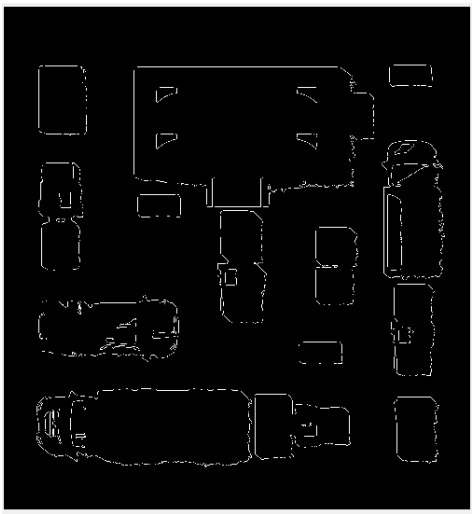

Fig. 8. Detección de Orillas con Canny.

De acuerdo con [8], la morfolización se puede dilatar, erosionar, aperturar y realizar un cierre. Con la ayuda de la plataforma Matlab esta manipulación de la imagen se puede realizar mas facilmente. Por ejemplo, para el presente proyecto se utilizara la morfolización llamada **dilatación con propiedades rectangulares**, esto con el motivo de cerrar bordes que formen cuñas y que de acuerdo a una parametrización de los pesos $g(n)$ se minimice el riesgo a que el algoritmo se encuentre en un bucle. Además se utilizara un rellenado de huecos, en este caso rellenara los bordes encontrados anteriormente. Asi lo podemos apreciar en la figura 9, donde si se observa, se realiza un rellenado de los bordes por completo y se expande rectangularmente todos los objetos identificados con nivel de gris 1. [7]

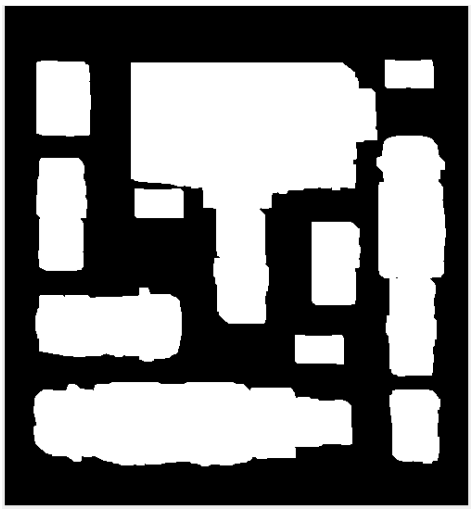

Fig. 9. Morfolización de la Imagen

C. Uso del Algoritmo A Estrella

De acuerdo con [1] el objetivo de este algoritmo es encontrar la ruta mas óptima en mapas cuadriculados, desde el nodo de origen (x_{ini}, y_{ini}) hasta un nodo meta (x_{meta}, y_{meta}). Combina la búsqueda de menor coste con la búsqueda heurística pura, permitiendo soluciones eficientes. Así definiendo las funciones $g(n)$ el peso del nodo, y $h(n)$ la heurística del sistema que puede ser la distancia euclidiana o la distancia de manhattan, tendríamos la composición a analizar de:

$$f(n) = g(n) + h(n) \tag{1}$$

Asi la maleabilidad de las funciones nos permite comprender que:

- Si $h(n) = 0$, el algoritmo A* se reduce al algoritmo de Dijkstra $f(n) = g(n)$ el cual encontrara el camino de menor coste.
- Si $h(n)$ es menor que el costo del traslado del nodo n hacia la meta, A* expandirá los nodos ocancionando una convergencia lenta.
- Si $h(n)$ es varias veces mas grande que el coste del nodo n a la meta, A* se interpretara que se seguirá el camino mas corto pero con una convergencia mas rápida.
- Si $h(n)$ es exactamente igual al costo del nodo n a la meta, el algoritmo tomara la mejor ruta y no se expandirá a otro otro lado.
- Si $h(n)$ es mas grande que $g(n)$ entonces $f(n)$ $h(n)$ y A* se reduce a ser un algoritmo de Búsqueda Primero el Mejor.

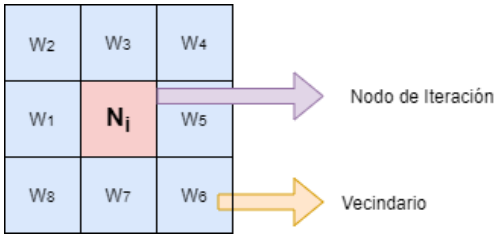

Fig. 10. Visión del Nodo a Iterar

Lo que se realiza a continuación es describir el problema para modelizar el algoritmo de A* de acuerdo a nuestros requerimientos, conforme nos muestra la rubrica [9] nuestra intención es encontrar la ruta mas óptima desde un nodo de inicio hasta el final, esquivando los obstáculos. Por lo que se propone generar un nodo que itere de acuerdo con a las características de peso y heurísticas de sus nodos vecinos, eligiendo la ruta de menor coste posible, tal como se mira en la figura 10. Asi se propone que se genere un objeto denominado ”Nodo Viajero”, el cual se moverá y trazara el camino mas óptimo.

1) Generación de Objeto: Se genera un objeto llamado ”Nodo Viajero” el cuál contendrá los atributos esenciales para la iteración como las coordenadas en la imagen, las coordenadas de meta, los parámetros de las funciones f,g y h, así como los niveles de grises de la imagen morfolizada (de 0 y 1). También se definirá variables locales del nodo iterativo como lo son las listas abiertas y cerrada. En la tabla I se encuentra una descripción breve acerca de los atributos y los métodos que se utilizaran del nodo a iterar.

TABLE I
DIAGRAMA DE CLASE

Clase: Nodo Viajero
xn; yn xmeta; ymeta f(n); g(n), h(n) Lista Abierta Lista Cerrada
analisisVecinos() heuristica() append()

2) Heurística: Para la heuristica se escogió la distancia euclidiana, la cual se describe con el teorema de pitagóricas ($h = \sqrt{x^2 + y^2}$) para su obtención.Para ello se genera un método llamado *heuristica()* como se vio en la tabla I, en las siguientes lineas de código se define el método mencionado:

```
function self = heuristica(self,xi,yi,i)
    y = norm(self.ymeta-yi);
    x = norm(self.xmeta-xi);
    self.h(i) = sqrt(y^2+x^2)*10;
end
```

3) Pesos de los nodos: Los pesos de los nodos se dan de acuerdo a la posición del nodo iterativo n_i de acuerdo con el nodo meta n_meta, dado que tiene que detectar objetos y buscar la mejor ruta, los pesos comunes es darle a la figura 10 los siguientes valores:

$$W = [\ 10\ 14\ 10\ 14\ 10\ 14\ 10\ 14\ ] \tag{2}$$

este contexto es idealizado, dado que las orillas de los objetos tiene irregularidades, se propuso colocar pesos de acuerdo a la posición obteniendo una estructura de pesos por posición, asi cuando el nodo se encuentre en la esquina superior izquierda se utilizara los siguientes pesos:

$$W = [\ 10\ 32\ 30\ 32\ 10\ 14\ 10\ 14\ ]$$

Cuando se encuentre en la esquina inferior izquierda:

$$W = [\ 30\ 32\ 30\ 32\ 10\ 14\ 10\ 14\ ]$$

Cuando se encuentre en la esquina inferior derecha

$$W = [\ 10\ 14\ 10\ 14\ 10\ 32\ 30\ 32\ ]$$

Para los demás casos se utilizo los pesos de nombrados al principio.

III. ANÁLISIS DE RESULTADOS

A continuación se realiza el análisis de resultados obtenidos para cada una de las figuras layout0.jpg, layout2.jpg, layout3.jpg, layout4.jpg y layout5.jpg, de esta manera se verificará la correcta operación del código implementado.

- **Layout 0**

  En la figura 11 se tiene a imagen del Layout0, en la cual se puede apreciar que se tienen únicamente dos circunferencias rojas correspondientes al inicio y a la meta del camino a buscar, esta dos circunferencias se encuentran separadas por un obstáculo.

Fig. 11. Layout 0

La operación del código para este layout se puede apreciar en la figura 12, como se puede visualizar se ha

encontrado correctamente un camino que comienza en el Inicio y terminar en la Meta, este camino rodeando el obstáculo que los separaba.

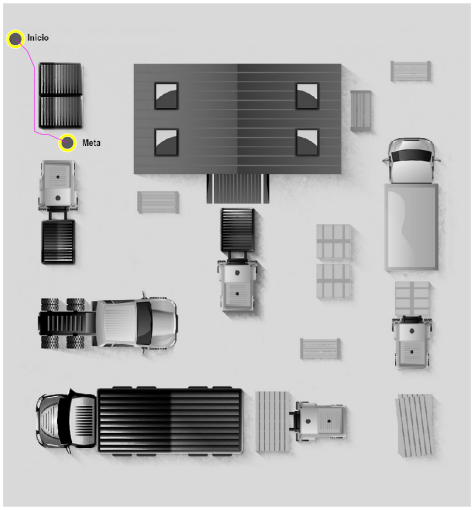

Fig. 12. Operación del Código con el Layout 0

- **Layout 1**

En el layout 1 mostrado en la figura 14, a diferencia del caso anterior no se tiene ninguna circunferencia roja que indique el inicio y el final del camino a buscar.

Fig. 13. Layout 1

En vista de no tener inicio y fin en el layout, el código valida que no se tiene ningún punto de partida para empezar a realizar el algoritmo de búsqueda A* (ver figura 14).

```
No existe ningun punto de partida
>>
```

Fig. 14. Operación del Código con el Layout 1

- **Layout 2**

Para el caso del Layout 2 (ver figura 15) se tiene dos circunferencias para inicio y final del camino pero a diferencia del caso del layout 0, estas se encuentra separas por múltiples obstáculos.

Fig. 15. Layout 2

En la operación del código para este layout de la figura 17 se visualiza que a pesar de los múltiples objetos en el inicio y el final, el programa encuentra el camino correcto esquivando los obstáculos.

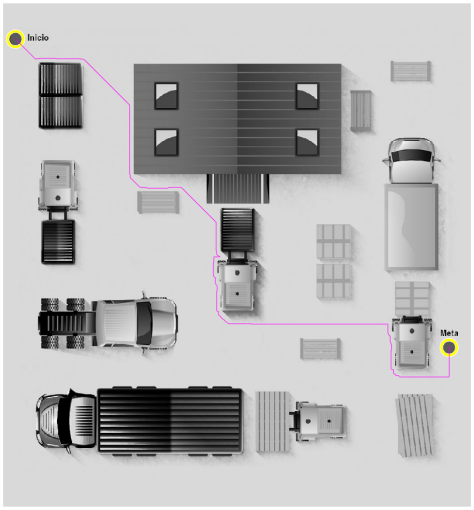

Fig. 16. Operación del Código con el Layout 2

- **Layout 3**

En este caso se tiene tres circunferencias rojas (ver figura 17), de las cuales únicamente dos pertenecen al inicio y final del camino, otro parámetro a observar en este layout, es que la meta se encuentra ubicada entre dos obstáculos.

Fig. 17. Layout 3

En la figura 18 se puede observar que el programa ignora la tercera circunferencia que no pertenece ni al inicio ni al final, y encuentra de manera correcta la trayectoria ya que la misma rodea el ultimo obstáculo que se tiene y encuentra la meta.

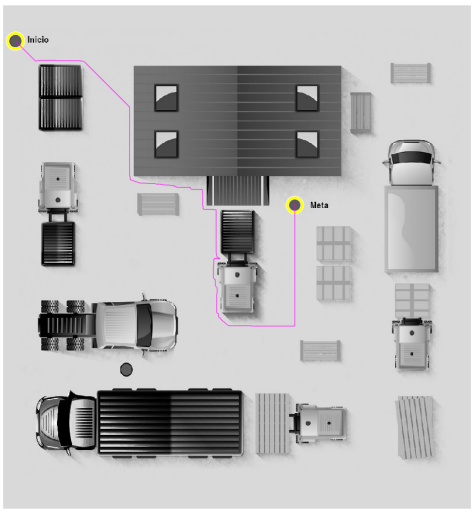

Fig. 18. Operación del Código con el Layout 3

- **Layout 4**

  El Layout 4 es muy parecido al Layout3, igualmente se tiene tres circunferencias donde una de ellas no pertenece a los indicadores de inicio y final del camino a buscar, la diferencia radica en que el Layout 4 tiene un ultimo obstáculo justo bloqueando el camino encontrado por el Layout 3 (ver figura 19).

Fig. 19. Layout 4

Para el layout 4, también se ha encontrado correctamente el camino ahora rodeando el obstáculo extra que se ha impuesto, de igual manera que el caso anterior el código valida que la tercera circunferencia (que no pertenece al Inicio ni al final) sea ignorada, esto lo podemos observar en la figura 20.

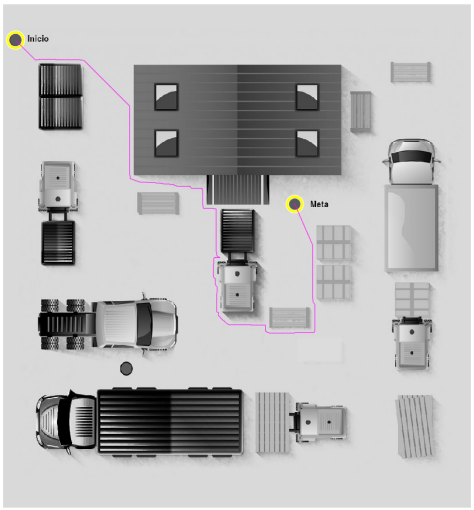

Fig. 20. Operación del Código con el Layout 4

- **Layout 5**

  Como último experimento se realizó la búsqueda del camino para el layout 5, donde se tiene tres circunferencias, la circunferencia correspondiente a la meta se encuentra la borde la imagen justo detrás de un obstáculo (ver figura 21).

Fig. 21. Layout 5

En la figura 22 se tiene el camino encontrado por el programa, validado de esta manera la correcta operación del código para cada uno de los Layout's dados, ya que como se puede observar en la figura ignora la circunferencia extra y llega a la meta esquivando los obstáculos existentes.

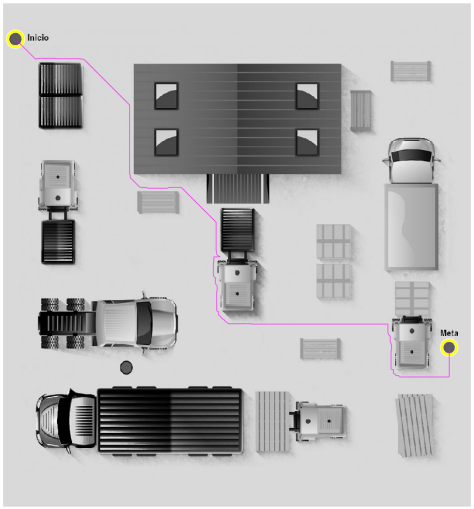

Fig. 22. Operación del Código con el Layout 5

Como punto importante, cabe mencionar que se realizó una dilatación a las imágenes binarizadas de los Layout's, esto a manera de evitar cometer errores en la búsqueda del camino ya que algunos de los obstáculos tiene formas irregulares donde existen secciones con aberturas en la cuales el camino podría verse estancado. En la figuras 23 y 24 se puede observar el camino encontrado en la imágenes binarizadas y dilatadas de los Layout 3 y 4 correspondientemente, como se puede observar este camino se encuentra justo en el borde de los obstáculos, si observamos las figura 18 y 20 notamos que el camino no toca los los bordes de los objetos si no que existe un margen de separación entre ellos, esto es producto de la dilatación realizada.

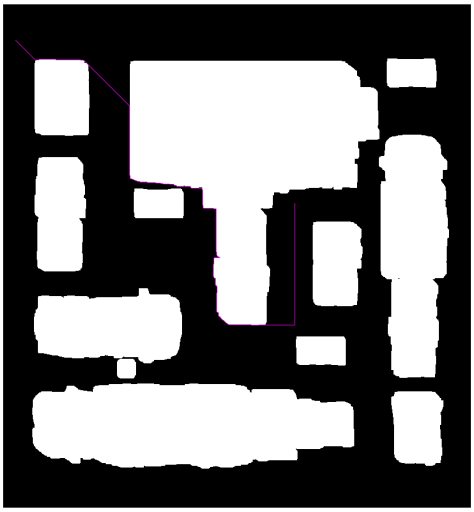

Fig. 23. Layout 3 Binarizada

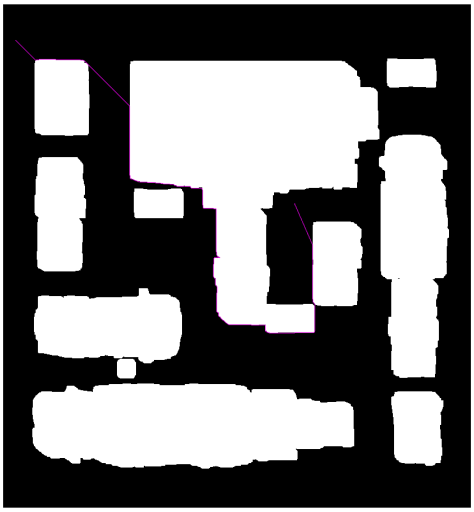

Fig. 24. Layout 4 Binarizada

IV. Conclusiones

- Con el uso de la correlación cruzada, podemos hacer uso de conocimientos adquiridos en la materia, el propósito es buscar puntos entre dos variables y la relación que tienen entre si, en el contexto de las imágenes, nos arroja como resultado el parecido que existe entre las imágenes lo cual nos ayuda a emplear esta herramienta en la detección de objetos dentro de imágenes, una de estas aplicaciones podría ser enfocada al famoso juego "Donde esta Wally?", o en la detección de caminos de un mapa, o en los famosos m
- Se observo que la segmentación simplifica la complejidad de la imagen a sus características mas básicas, donde la obtención de la binarizacion y el reconocimiento de orillas nos proporciona recursos para aplicar algoritmos

de búsquedas heurísticas. Además la aplicación de operaciones morfológicas a imágenes ayudan a configurar una imagen binarizada, para un mejor desempeño de una búsqueda heurística, dado que pueden existir irregularidades de continuidad en las orillas.

- Se observo que la connotación de los pesos y la definición de la heurística serán los parámetros que configuraran el algoritmo A Estrella, es pues de esta forma que los pesos serán únicos de acuerdo a la topología de la imagen, y que estos pueden ser configurados de acuerdo a la posición de un nodo iterativo con respecto a la heurística utilizada. Para futuros trabajos seria interesante aplicar una heurística mediante direcciones y magnitudes de fasores en acuerdo con la serie de Fourier.

REFERENCES

[1] D. A. Calle and C. D. Sosoranga, "Desarrollo De Un Vehículo Autónomo Terrestre Para Navegación Óptima En Ambientes Cerrados," p. 130, 2017.

[2] N. Yungaicela-Naula, L. E. Garza-Castañon, Y. Zhang, and L. I. Minchala-Avila, "Uav-based air pollutant source localization using combined metaheuristic and probabilistic methods," *Applied Sciences*, vol. 9, no. 18, p. 3712, 2019.

[3] D. Harabor, "Fast pathfinding via symmetry breaking," *Aigamedev Com*, 2012.

[4] D. A. C. Guzmán, "Análisis de algoritmos pathfinding," Ph.D. dissertation, PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO, 2013.

[5] B. Stout, "Smart moves: Intelligent pathfinding," *Game developer magazine*, vol. 10, pp. 28–35, 1996.

[6] A. d. l. Escalera Hueso, "Visión por computador: Fundamentos y métodos," 2001.

[7] R. Gonzales and R. Woods, *Digital Image Processing*, 2013, vol. 53, no. 9.

[8] Mundo Tecnológico, "OPERACIONES MORFOLÓGICAS en MATLAB -DILATACIÓN y EROSIÓN -CURSO PROCESAMIENTO DE IMÁGENES (PARTE 7)," 07 2020. [Online]. Available: https://www.youtube.com/watch?v=tQYYWjZyBwk

[9] I. Minchala, "PROYECTO DE VISIÓN POR COMPUTADOR (REDES NEURONALES)," no. 3, p. 6, 2021.