**Navegación Autónoma para un Velero Inteligente - Sensailor**

**Marcelo Fajardo-Pruna1, Daniela Sánchez-Orozco 1, Karen Torres-Medina 1, Luis López-Estrada2, Christian Tutivén1, Yolanda Vidal3**

1 Ingeniería Mecatrónica, Facultad de Ingeniería Mecánica y Ciencias de la Producción, Escuela Superior Politécnica del Litoral, ESPOL, Guayaquil, Ecuador. Email: [mrfajard@espol.edu.ec](mailto:mrfajard@espol.edu.ec), [dansoroz@espol.edu.ec](mailto:dansoroz@espol.edu.ec), [kttorres@espol.edu.ec](mailto:kttorres@espol.edu.ec), [cjtutive@espol.edu.ec](mailto:cjtutive@espol.edu.ec)

2 Ingeniería en Procesos de Fabricación, Escuela Técnica Superior de Ingenieros Industriales, Universidad Politécnica de Madrid, Madrid, España. Email: [luis.lopez.estrada@alumnos.upm.es](mailto:luis.lopez.estrada@alumnos.upm.es)

3Control, Modelització, Identificació i Aplicacions (CoDAlab), Departament de Matemàtiques, Escola d’Enginyeria de Barcelona Est (EEBE), Universidad Politécnica de Catalunya, Barcelona, España. Email: [yolanda.vidal@upc.edu](mailto:yolanda.vidal@upc.edu)

**Resumen**

Este trabajo discute una arquitectura necesaria para los sistemas de mapeo, localización y evasión de obstáculos requeridos en los sistemas de navegación autónoma de veleros. Se presenta la solución desarrollada para un prototipo de velero construido para la validación experimental de la arquitectura propuesta. Se presentan y analizan las técnicas utilizadas en veleros, incluidos los sensores, actuadores y leyes de control necesarias. También se presenta la arquitectura necesaria para mapear el entorno del dispositivo, así como para localizar obstáculos que podrían presentarse durante su funcionamiento. Del mismo modo se establece la planificación de trayectorias a través de puntos de ruta predeterminados proporcionados por una estación base. Finalmente, se presentan resultados simulados para validar los sistemas propuestos.

**Palabras clave:** vehículos autónomos, velero autónomo; planificación de trayectorias; robótica.

**Abstract**

This work discusses a necessary architecture for mapping, localization and obstacle avoidance systems required in sailboat autonomous navigation systems. The solution developed for a sailboat prototype built for the experimental validation of the proposed architecture is presented. Techniques used on sailboats, including sensors, actuators, and control laws, are presented, and discussed. The necessary architecture to map the device's environment is also presented, as well as to locate obstacles that could appear during its operation. In the same way, trajectory planning is established through predetermined waypoints provided by a base station. Finally, simulated results are presented to validate the proposed systems.

**Keywords:** self-manned vehicles, autonomous sailboat, path planning, robotics

# Introducción

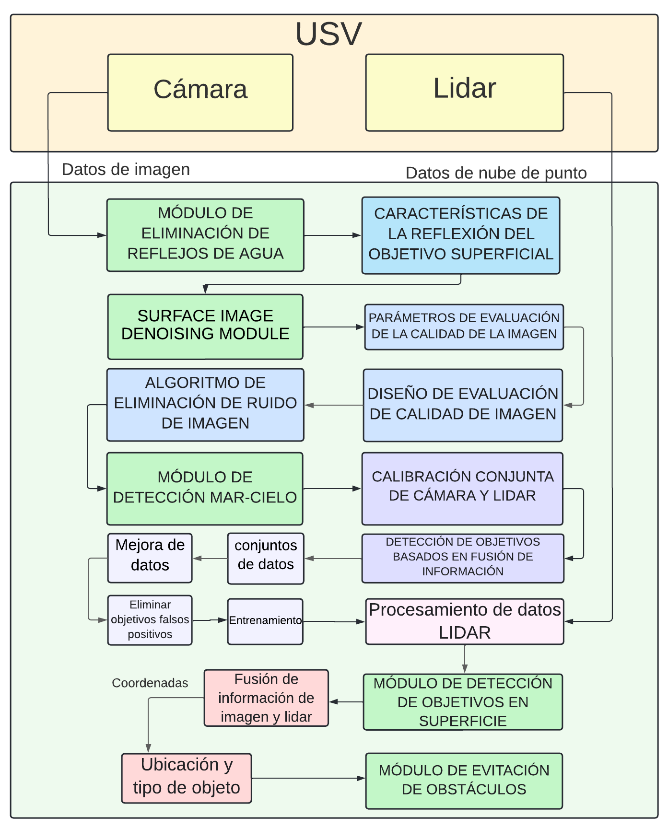
La implementación de vehículos autónomos es una de las áreas de investigación con mayor desarrollo, especialmente para aplicaciones de exploración científica [1]. El desarrollo de la navegación autónoma ha permitido desarrollar dispositivos capaces de estabilizarse y orientarse dependiendo de una variedad de parámetros externos como el viento, el agua, los obstáculos, entre otros. [2]. Esto genera la necesidad de tener sistemas de control que permitan estabilizar el sistema ante los diversos factores que pueden desestabilizar el mismo, como la fuerza, velocidad y dirección del viento [3].

Se han desarrollado modelos matemáticos para compensar el efecto del viento y mantener estable la embarcación. Para lo cual se mide la dirección y velocidad del viento a partir de dos referencias: absoluta y relativa a la posición de un velero [4]. Todo esto ha impulsado el desarrollo de veleros autónomos con capacidades de navegación autónoma [5] en los cuales las tareas de planificación y detección de obstáculos son fundamentales dado que permiten al velero realizar su trabajo de una forma inteligente, adaptándose a su entorno para cumplir con las tareas asignadas.

En el presente trabajo se presentan los sistemas de mapeo y localización autónoma y de planificación de trayectorias desarrollados para un velero autónomo (Sensailor).

# Mapeo y localización de obstáculos

En lo que concierne a navegación autónoma la localización y mapeado del vehículo no tripulado es toda una hazaña, por tal motivo se aplican técnicas SLAM en la presente investigación. El velero autónomo se desplazará en un entorno muy cambiante, por ello surge la necesidad de asegurar la integridad del vehículo no tripulado cuando se encuentre navegando, implementando una estimación de su posición y orientación, como también mapeando sus alrededores. Adicionalmente, en cada escenario marítimo existe la presencia de obstáculos y escenarios impredecibles, como prevención ante alguna colisión, se refuerza el uso de multisensores y se recopila información, fusionando los datos obtenidos de una cámara con una nube de puntos de un Lidar.



**Figura 1.** Diagrama de módulos para evasión de obstáculos

Es indispensable que en primera estancia se cree una capa para poder preprocesar el filtrado de las imágenes y el filtro de la nube de puntos LIDAR. Por ello, para mejorar la efectividad de la detección de obstáculos se han estipulado una lista de pasos según la ilustración 1.

## Cámara

Las primeras tres secciones de la ilustración 1 consisten en el tratamiento de datos con la cámara en donde se define el módulo de eliminación de reflejos de agua, el módulo de eliminación de ruido de imagen superficial, y el módulo de detección mar-cielo[6] Respectivamente, en la primera etapa se trata de eliminar el reflejo que se genera en la superficie del agua, ya que usualmente este interfiere al detectar y hacer el seguimiento de los objetos en movimiento que se encuentran en la superficie.

Por consiguiente, se transforma en una imagen a escala de grises la imagen de entrada y se utiliza la función de umbral medio [7] para convertir dicha entrada en una imagen binaria, consecuentemente se implementa el método de cuadrícula de 9 cuadrados para la detección de contorno. Si no existiese en las proximidades aledañas de un píxel una diferencia de 8 píxeles se establece contorno.

|  |  |  |  |
| --- | --- | --- | --- |
| Imagen en blanco y negro de un velero en el mar  Descripción generada automáticamente | Un conjunto de letras negras en un fondo negro  Descripción generada automáticamente con confianza media | Imagen que contiene grande  Descripción generada automáticamente | Imagen en blanco y negro de un barco en el agua  Descripción generada automáticamente |
| Imagen en blanco y negro  Descripción generada automáticamente con confianza baja | Imagen que contiene monitor, viendo, gato, pantalla  Descripción generada automáticamente | Imagen que contiene avión, vuelo, grande, barco  Descripción generada automáticamente | Dibujo en blanco y negro  Descripción generada automáticamente con confianza baja |
| **Figura 5.** Corrección de imagenes por medio de segmentación | | | |

|  |  |
| --- | --- |
|  |  |
| **Figura 2.** Imagen en escala grises convertida en imagen binaria | |

Además, se considera la interferencia de ruido existente en las imágenes que toma el USV pues también afecta la detección de obstáculos, algunas imágenes tendrán mala calidad, de ser así, se realiza un procesamiento de eliminación de ruido, ilustración 3, caso contrario se omite este paso para ahorrar recursos informáticos.

|  |  |  |
| --- | --- | --- |
| a | b | c |
| **Figura 3.** a) Imagen original. b) Imagen con ruido.  c) Imagen con filtro Gaussiano. | | |

Para determinar la calidad de la imagen se emplean las técnicas de las ecuaciones (1) a (3) como el Error cuadrático medio MSE, la Proporción Máxima de Señal a Ruido PSNR y el Índice de similitud estructural SSIM. Mientras que, en el caso de que sea necesario quitar el ruido por una distorsión inaceptable se aplica el algoritmo BM3D.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | |  |
| a | b | | c |
|  | |  | |
| d | | e | |
| **Figura 4**. a) Imagen original. b) Ruido Impulsivo con MSE:10590 y PSNR:18.11. c) Ruido Gaussiano. MSE:619.06 y PSNR:20:21 d) Supresión de ruido Impulsivo con MSE:279.18 y PSNR:23.67 e) Supresión de ruido con filtro Gaussiano. MSE:238.15 y PSNR:24.36 | | | |

|  |  |
| --- | --- |
|  | (1) |
|  | (2) |
|  | (3) |

Se utiliza el algoritmo Watershed, como se observa en la Ilustración 4, para segmentar imágenes complejas detectando contornos y encontrando objetos superpuestos dentro de la imagen. Previamente es imprescindible que se haya disminuido el ruido de la imagen para poder aplicarlo.

Para la detección del mar-cielo existentes varios métodos, entre ellos se incluye el método de Otsu [8] encargándose de calcular el umbral para que en cada segmento sea menor la dispersión, pero al contrastarse con otros segmentos sea muy alta. Por medio de la transformada de Hough se encargada de extraen posibles horizontes y a través de intervalos probabilísticos se elige un único horizonte [9].

|  |  |  |
| --- | --- | --- |
| Un papalote de colores en el agua  Descripción generada automáticamente con confianza media |  | Una captura de un videojuego  Descripción generada automáticamente con confianza media |
| Un barco en el mar  Descripción generada automáticamente | Un cuerpo de agua  Descripción generada automáticamente | Imagen en blanco y negro de un barco en el mar  Descripción generada automáticamente |
| **Figura 6.** Identificación de línea del cielo y mar. | | |

## Fusión Lidar y cámara

Se vuelve indispensable complementar la percepción del entorno del USV utilizando varios tipos de sensores, en este caso se opta por fusionar las imágenes con calidad alta de la cámara con la nube de puntos del Lidar, puesto que con la cámara es posible conseguir alta resolución y con el Lidar datos de profundidad más precisos. Al ser complementarios es fundamental considerar el tipo de datos.

La relación entre las coordenadas del mundo arbitrario y el sistema de coordenadas [10] para los pixeles de las imágenes se estima tomando el centro del lente como el origen del sistema, el eje x y el eje y como paralelos a los lados opuestos de la fase, y por último el eje z representa el eje óptico del lente que está perpendicular al plano de la imagen como se aprecia en la ecuación (4).

|  |  |
| --- | --- |
|  | (4) |

Por un lado, los datos de imagen de la cámara se representan por una nube de celosía tridimensional hecha por el Lidar [11], para lo que se crea una matriz de transformación, ecuación (5), que asigna puntos 3D a puntos 2D.

|  |  |
| --- | --- |
|  | (5) |

A través de YOLOV3 ha sido posible detectar obstáculos siguiendo ciertas normas, se selecciona ciertas áreas dentro de la imagen, etiquetando esas regiones según su posición. El modelo utiliza una red neuronal convolucional que extrae datos de la imagen característica prediciendo la ubicación y la categoría según el modelo de la red neuronal, misma que se contrasta con la etiqueta y se obtiene la función de pérdida evaluando la desviación entre el escenario real y el predicho en base a lo que aprendió la red.

|  |  |
| --- | --- |
| Una persona con una tabla de surf en el mar  Descripción generada automáticamente | Un barco en el mar  Descripción generada automáticamente |
| Una playa con el mar de fondo  Descripción generada automáticamente | Un barco en el mar  Descripción generada automáticamente |
| Un barco en el mar  Descripción generada automáticamente | Un barco en el mar  Descripción generada automáticamente |
| Un barco en el mar  Descripción generada automáticamente | Un barco en el agua  Descripción generada automáticamente |
| Un barco en el agua  Descripción generada automáticamente | Un barco en el agua  Descripción generada automáticamente |
| Un barco en un lago  Descripción generada automáticamente | Un barco en el agua con una montaña al fondo  Descripción generada automáticamente con confianza media |
| Una persona con una tabla de surf en el mar  Descripción generada automáticamente | Un barco en el agua  Descripción generada automáticamente |
| **Figura 7.** Reconocimiento de obstáculos con YoloV3. | |

Continuamente también se procesa la nube de puntos obtenida del Lidar consiguiendo el clúster de nube de puntos que es capaz de detectar el obstáculo, para minimizar la cantidad de cálculos se limita el rango de extracción de muestreo utilizando la distribución de puntos que más nos interesa.

Considerando que las condiciones climáticas generan dispersión y puntos de interferencia en la nube de puntos del Lidar, se utiliza un filtro de eliminación condicional [12] para analizar la vecindad entre cada punto y realizar el cálculo de la distancia euclidiana para cada uno de los puntos [13], es decir, si un punto tiene muy pocos vecinos se lo considera interferencia atípica.

Mediante un KdTree se simplifica el cálculo con los puntos filtrados, se los agrupa euclidianamente, se extraen los puntos característicos y se proyectan los puntos 3D de la nube de puntos dentro de un plano 2D. Con ello, es posible delimitar y etiquetar el obstáculo segmentado.

La fusión de datos de la cámara y el Lidar ocurre al proyectar el clúster de la nube de puntos-objetivo y los cuadros delimitadores de la imagen, calculando cuadros delimitadores en dos dimensiones y uniendo la información de ambos.

|  |
| --- |
|  |
| **Figura 8**. Sistema de coordenadas del obstáculo con respecto al USV |

|  |  |
| --- | --- |
|  | (6) |
|  | (7) |

Nuestro USV tiene la capacidad de detectar obstáculos con su distancia y dirección de ángulo relativo al velero autónomo, mientras que, a través de la brújula electrónica y navegación inercial se reúne la latitud, longitud y ángulo de trayectoria; y con la ayuda de la ecuación se halla la longitud J y latitud W del obstáculo, siendo U el velero autónomo y A el obstáculo, tomando en cuenta que la latitud difiere por cada grado cada 111km.

# Planificación de trayectorias

## Coordenadas de obstáculos

El algoritmo recibe las coordenadas correspondientes a la ubicación de los obstáculos detectados mediante el mapeo del área de trabajo. Esta área corresponde a la superficie del cuerpo de agua sobre el cual se desplaza el velero. Se procede a discretizar la superficie a fin de representarla a través de una matriz, de manera que cada celda represente un metro cuadrado del área de trabajo. Esto garantiza que el cuerpo del velero siempre quepa dentro de una celda, disminuyendo el riesgo de choques en la implementación por invadir celdas no seguras adyacentes a las celdas que conformen el camino planificado. Las coordenadas de los obstáculos son evaluadas para redefinirse en función de las coordenadas de la celda a la cual equivale su ubicación dentro de la matriz representativa.

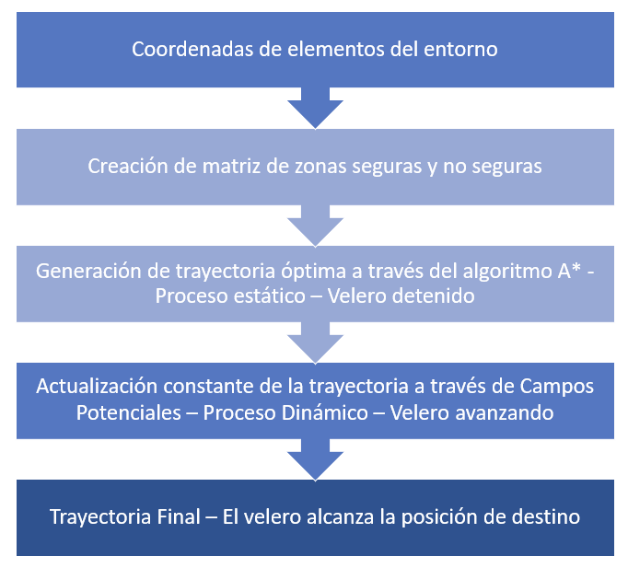
## Zonas seguras y no seguras

Los espacios disponibles dentro de la matriz representan el área a través de la cual puede desplazarse el velero. Estos espacios carentes de obstáculos se denominan zonas seguras. Por otro lado, las zonas no seguras son aquellas que posean un obstáculo o una porción del mismo en el área comprendida dentro de la celda. Los límites de la superficie de desplazamiento también serán denominados zonas no seguras.

Dentro de las condiciones para garantizar un movimiento seguro del velero se deben considerar las normas estipuladas en el Reglamento Internacional para Prevenir Abordajes, también conocido como maniobras COLREG por sus siglas en inglés. Entre estas, se especifica que los cambios de dirección que efectúa el velero deben ser lo suficientemente amplios para ser detectados por otros vehículos, ya sea por métodos visuales o radares. Se debe también establecer una distancia de seguridad entre la trayectoria del velero y la ubicación de otro vehículo reconocido como obstáculo en el área de desplazamiento. Esta distancia depende de factores como el tamaño de los vehículos, las velocidades de estos, las condiciones del entorno, etc. Se debe tener presente además que bajo ciertas condiciones puede requerirse que el velero se detenga o disminuya su velocidad. Estos eventos son responsabilidad del controlador del sistema, y son independientes de la planificación de trayectorias [14].

## Trayectorias

El sistema de generación de trayectorias tiene como finalidad encontrar la ruta que permita al velero trasladarse desde su posición actual hacia la posición objetivo. Estos caminos deben priorizar la seguridad el velero, transitando únicamente a través de las zonas seguras. En segundo lugar, se pretende optimizar las trayectorias, de manera que estas representen el camino seguro más corto. Además, se debe prestar especial atención a la redundancia de movimientos, evitando que el camino presente bucles o recorra un mismo segmento varias veces sin justificación. Para esto, se decidió utilizar una combinación de dos algoritmos de planificación de trayectorias. El algoritmo A\*, el cual permite hallar la ruta óptima con gran eficiencia en un entorno estático, es decir, donde los obstáculos posean posiciones fijas. Debido a la alta probabilidad de cambios en la ubicación de los obstáculos durante el movimiento del velero, se utiliza conjuntamente el algoritmo basado en Campos Potenciales. A pesar de que este en muchos casos no resulta en la ruta más corta, presenta una gran eficiencia al trabajarse de manera dinámica o en tiempo real.

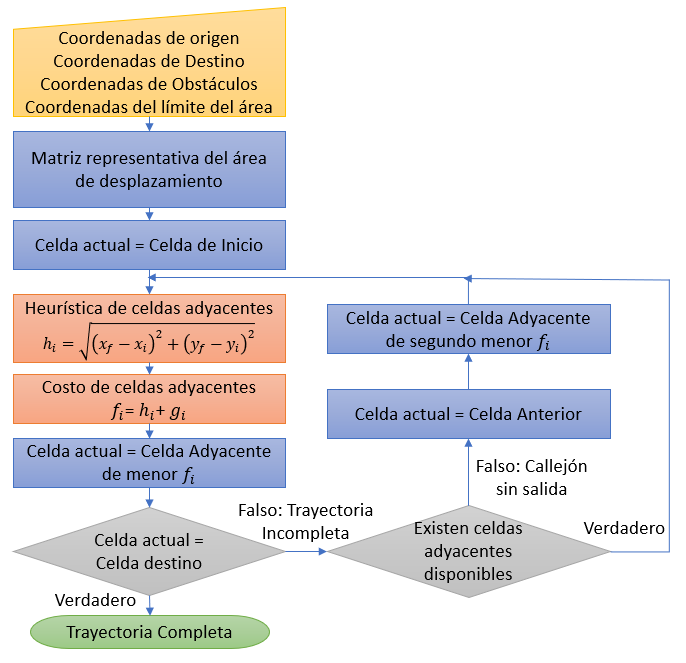


**Figura 9.** Algoritmo General para Planificación de Trayectorias

El algoritmo de búsqueda inteligente A\* evalúa la matriz clasificando sus celdas únicamente como ocupadas o libres. Este pretende hallar el camino óptimo, o el más corto, entre los puntos de inicio y de destino. Para esto, se toma como referencia una función de heurística admisible, la cual corresponde a la respuesta en el mejor escenario posible. En esta investigación, esto corresponde al caso de ausencia de obstáculos en el cual el desplazamiento pueda realizarse mediante una única línea recta. Por lo tanto, la función heurística utilizada en este trabajo es la distancia euclidiana entre el punto evaluado y el punto final.

|  |  |
| --- | --- |
|  | (8) |

Además, cada casilla tiene un valor asociado correspondiente a la cantidad de pasos que deben darse desde la celda de origen hasta esta. Sumando ambos parámetros se determina el costo de cada casilla. El algoritmo tendrá como prioridad escoger la celda de menor costo y dirigirse a esta para actualizar su posición actual. En el caso de que la trayectoria llegue a una celda en la cual todos sus caminos se encuentren bloqueados, el algoritmo retoma la posición anterior , elimina la posibilidad de desplazarse a la celda , y selecciona la celda adyacente con el segundo menor costo . Estas acciones se repiten hasta que el programa halle un camino que logre alcanzar la posición de destino a través de las celdas de menor costo posible. De esta forma se determina que se ha hallado la ruta óptima hasta la posición final [15].



**Figura 10.** Algoritmo A\* - Diagrama de Flujo

Este algoritmo garantiza que se ha hallado una ruta óptima para desplazarse entre dos puntos evadiendo los obstáculos presentes. Sin embargo, la implementación dinámica de este algoritmo no es recomendable. Dado que el sistema requiere realizar una actualización constante del espacio muestreado para reconocer los posibles cambios en la ubicación de los obstáculos, se complementa el sistema de generación de trayectorias con un algoritmo basado en campos potenciales. A diferencia del algoritmo A\*, este permite implementarse en tiempo real, actualizando la trayectoria constantemente según el movimiento de los obstáculos. Este algoritmo se basa en la asignación de un campo potencial artificial a cada una de las celdas que conforman la matriz de representación del espacio de desplazamiento del velero. Esto permite asociar una fuerza de atracción al punto de destino del velero, mientras que los obstáculos presentes generan fuerzas de repulsión. La base del algoritmo de campos potenciales clásico indica que la dirección de desplazamiento del vehículo es establecida por la dirección de la fuerza resultante de la sumatoria de fuerzas de atracción y repulsión mencionadas [16].

Se define un radio que determina la distancia máxima a la cual se puede ubicar un obstáculo respecto al velero para considerar que este puede influir significativamente en la trayectoria. Esto permite ahorrar recursos computacionales cuando los cambios detectados en el entorno no interfieren con el movimiento del velero. Los obstáculos que se encuentren dentro del radio definido generarán fuerzas de repulsión que alejan al vehículo de estos puntos. La magnitud de la fuerza de repulsión efectuada por cada obstáculo se define mediante la siguiente ecuación, donde corresponde a la constante de repulsión, y representa la distancia entre la posición actual del velero y las coordenadas del obstáculo en cuestión .

|  |  |
| --- | --- |
|  | (9) |
|  | (10) |

La fuerza de repulsión total aplicada sobre la posición actual del velero se define como la sumatoria de todas las contribuciones ejercidas por los obstáculos dentro de la distancia .

|  |  |
| --- | --- |
|  | (11) |

La fuerza de atracción generada por el punto de destino del velero es proporcional a la distancia entre ambos puntos, como se muestra en la siguiente expresión donde representa la constante de atracción.

|  |  |
| --- | --- |
|  | (12) |

Adicionalmente, para asegurar que este algoritmo permita obtener una trayectoria final óptima, se utiliza el resultado del algoritmo A\* como camino referencial . Por lo tanto, se toma el punto perteneciente a dicho camino, más próximo a la posición actual del velero, de manera que este también ejerza una fuerza de atracción . Para calcular la magnitud de esta fuerza se utiliza la misma ecuación anterior, reemplazando las coordenadas del punto final por las correspondientes al punto .

La fuerza de atracción total se define como la suma entre ambas fuerzas de atracción mencionadas.

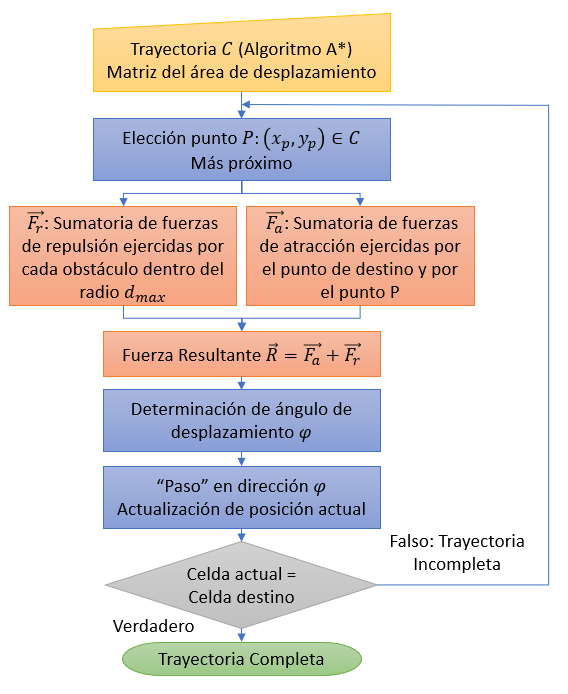
|  |  |
| --- | --- |
|  | (13) |

Finalmente, se obtiene la fuerza resultante experimentada por el velero en su posición actual mediante la sumatoria de fuerzas de atracción y repulsión consideradas.

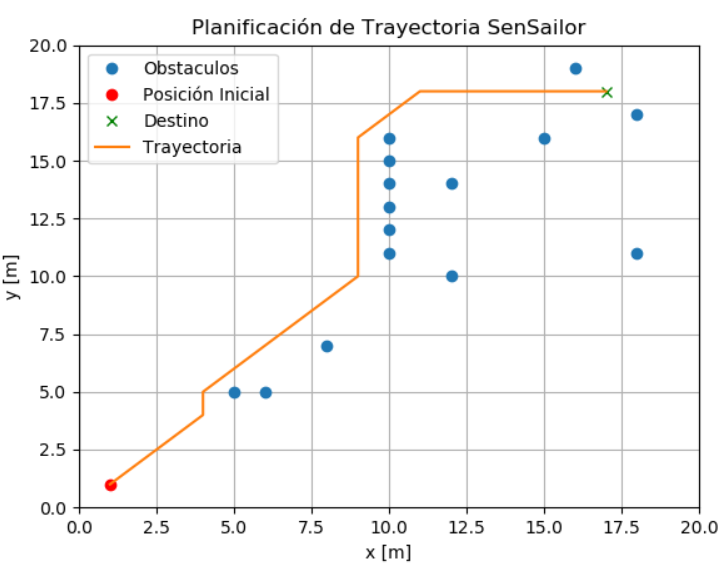
|  |  |
| --- | --- |
|  | (14) |

La dirección de este vector resultante determina el ángulo con el cual se desplazará el velero durante una distancia definida como “paso”. Se actualiza la posición actual del velero, y se verifica si existen cambios dentro del radio . El algoritmo de campos potenciales es repetido, permitiendo variar la dirección del desplazamiento según las actualizaciones de los obstáculos del entorno conforme el velero avanza hasta el punto de destino [17].

De esta forma, se asegura que la trayectoria final seguida por el USV sea muy cercana a la ruta óptima generada por el algoritmo A\*. Además, se garantiza que este algoritmo trabajará eficientemente actualizando la trayectoria en tiempo real según se requiera.



**Figura 11.** Algoritmo de Campos Potenciales – Diagrama de Flujo



**Figura 12.** Ejemplo de Trayectoria Generada

# Conclusiones

En la presente investigación se presenta una solución a la navegación autónoma de un Velero, en donde se consiguió detectar obstáculos segmentando imágenes y etiquetando la fusión de los datos de la nube de puntos obtenida por el Lidar con los de la cámara. Es importante señalar que para que el velero sea capaz de percibir cualquier objeto que se encuentre cerca de este también se ha hecho una corrección a la calidad de la imagen y reducir el ruido. Cabe decir, que al implementar Yolo para etiquetar obstáculos en tiempo real se obtuvo una respuesta bastante rápida de hasta 30 FPS quien utilizó Deep Learning y Redes Neuronales Convolucionales.

Al tener los datos de la ubicación del obstáculo el velero automáticamente traza una ruta de evasión de obstáculos para que este no sufra ningún percance. Par lo cual primero identifica las zonas seguras y no seguras a las que se puede desplazar, posteriormente se ayuda del algoritmo de búsqueda inteligente A\*, haciendo posible que nuestro USV pueda tomar decisiones eficientemente hacia qué lugar desplazarse y de ser necesario incluso retroceder. Adicionalmente, para que la navegación sea en tiempo real se complementó el algoritmo de campos potenciales.

Se utilizaron técnicas SLAM, sin embargo, para el mapeo y localización del vehículo no tripulado se asumió que el GPS obtenía correctas sin error, por lo que en futuras investigaciones se podría agregar un filtro Kalman para tratar la ubicación del USV.

De igual forma, para incrementar las posibilidades de detecciones inmediatas de obstáculos se puede adicionar más clases que incluyan nuevos sets de datos con otros obstáculos para entrenar redes.

# Referencias

[1] H. S. Stenersen, “Construction and Control of an Autonomous Sail Boat,” *IFAC-PapersOnLine*, vol. 49, no. 23, pp. 524–531, Jan. 2016, doi: 10.1016/J.IFACOL.2016.10.489.

[2] H. Chen, Z. Wu, R. Zheng, and S. Zhang, “Design of Autonomous Obstacle Avoidance Unmanned Boat System for Wetland Monitoring,” *J Phys Conf Ser*, vol. 1486, no. 7, p. 072033, Apr. 2020, doi: 10.1088/1742-6596/1486/7/072033.

[3] A. Moreno-Ortiz, D. Sanchez-Orozco, L. Lopez-Estrada, C. Tutiven, Y. Vidal, and M. Fajardo-Pruna, “Modelling of an Intelligent Control Strategy for an Autonomous Sailboat - SenSailor,” in *2022 5th International Conference on Advanced Systems and Emergent Technologies (IC\_ASET)*, Mar. 2022, pp. 34–38. doi: 10.1109/ic\_aset53395.2022.9765928.

[4] C. Viel, U. Vautier, J. Wan, and L. Jaulin, “Position keeping control of an autonomous sailboat,” *IFAC-PapersOnLine*, vol. 51, no. 29, pp. 14–19, Jan. 2018, doi: 10.1016/J.IFACOL.2018.09.462.

[5] J. Wirz, M. Tranzatto, A. Liniger, M. Colombino, H. Hesse, and S. Grammatico, “AEOLUS, the ETH Autonomous Model Sailboat,” *Robotic Sailing 2015*, pp. 103–112, 2016, doi: 10.1007/978-3-319-23335-2\_8.

[6] W. Zhang, F. Jiang, C. F. Yang, Z. P. Wang, and T. J. Zhao, “Research on Unmanned Surface Vehicles Environment Perception Based on the Fusion of Vision and Lidar,” *IEEE Access*, vol. 9, pp. 63107–63121, 2021, doi: 10.1109/ACCESS.2021.3057863.

[7] H. Kim *et al.*, “Vision-Based Real-Time Obstacle Segmentation Algorithm for Autonomous Surface Vehicle,” *IEEE Access*, vol. 7, pp. 179420–179428, 2019, doi: 10.1109/ACCESS.2019.2959312.

[8] M. Kristan, V. S. Kenk, S. Kovačič, and J. Perš, “Fast Image-Based Obstacle Detection from Unmanned Surface Vehicles,” *IEEE Trans Cybern*, vol. 46, no. 3, pp. 641–654, 2016, doi: 10.1109/TCYB.2015.2412251.

[9] D. Liang and Y. Liang, “Horizon detection from electro-optical sensors under maritime environment,” *IEEE Trans Instrum Meas*, vol. 69, no. 1, pp. 45–53, 2020, doi: 10.1109/TIM.2019.2893008.

[10] J. Ren, J. Zhang, and Y. Cui, “Autonomous obstacle avoidance algorithm for unmanned surface vehicles based on an improved velocity obstacle method,” *ISPRS Int J Geoinf*, vol. 10, no. 9, 2021, doi: 10.3390/ijgi10090618.

[11] P. Wu *et al.*, “Autonomous obstacle avoidance of an unmanned surface vehicle based on cooperative manoeuvring,” *Industrial Robot*, vol. 44, no. 1, pp. 64–74, 2017, doi: 10.1108/IR-04-2016-0127.

[12] J. Muhovic, R. Mandeljc, B. Bovcon, M. Kristan, and J. Pers, “Obstacle Tracking for Unmanned Surface Vessels Using 3-D Point Cloud,” *IEEE Journal of Oceanic Engineering*, vol. 45, no. 3, pp. 786–798, 2020, doi: 10.1109/JOE.2019.2909507.

[13] R. Polvara, S. Sharma, J. Wan, A. Manning, and R. Sutton, “Obstacle Avoidance Approaches for Autonomous Navigation of Unmanned Surface Vehicles,” *Journal of Navigation*, vol. 71, no. 1, pp. 241–256, 2018, doi: 10.1017/S0373463317000753.

[14] M. R. Benjamin and J. A. Curcio, “COLREGS-based navigation of autonomous marine vehicles,” *2004 IEEE/OES Autonomous Underwater Vehicles*, pp. 32–39, 2004, doi: 10.1109/AUV.2004.1431190.

[15] C. Ju, Q. Luo, and X. Yan, “Path Planning Using Artificial Potential Field Method and A-star Fusion Algorithm,” *2020 Global Reliability and Prognostics and Health Management, PHM-Shanghai 2020*, Oct. 2020, doi: 10.1109/PHM-SHANGHAI49105.2020.9280929.

[16] L. Deng, H. Chen, H. Liu, H. Zhang, and Y. Zhao, “Overview of UAV path planning algorithms,” *2021 IEEE International Conference on Electronic Technology, Communication and Information, ICETCI 2021*, pp. 520–523, 2021, doi: 10.1109/ICETCI53161.2021.9563566.

[17] S. Su, C. Peng, J. Shi, and C. Choi, “Potential Field: Interpretable and Unified Representation for Trajectory Prediction”.