IV Conferencia Científica Internacional
Universidad de las Ciencias Informáticas

Temática: Software libre

Simulador para el bote robótico Krick Felix basado en software-in-the-loop

Simulator for the Krick Felix robotic boat based on software-in-the-loop

Omar Milián Morón[1*], Luis Manuel Milián Perez[1], Yunier Valeriano Medina[1]

[1]Departamento de Control Automático. Universidad Central "Marta Abreu"de Las Villas, Carretera a Camajuaní Km 5 1/2, Santa Clara, Cuba

[*]Autor para correspondencia: omilian@uclv.cu

Resumen
Incorporar al desarrollo de la industria marítima simulaciones de alto nivel con el uso de sistemas cada vez más exactos y lo suficientemente cercanos a la realidad, contribuye a evitar la pérdida de recursos en situaciones que puedan presentarse ante fallos en las embarcaciones marítimas. En el presente trabajo se diseña un simulador para representar el comportamiento del bote robótico *Krick Felix* en entornos marinos tridimensionales en la herramienta Gazebo. El simulador ha sido implementado basado en la estrategia *software-in-the-loop* de la plataforma Ardupilot. Los resultados obtenidos en las simulaciones confirman la validez del comportamiento alcanzado por el simulador en los entornos marinos generados.

Palabras claves: Ardupilot, Gazebo, Krick Felix, Simulador, software-in-the-loop

Abstract
Incorporating high-level simulations into the development of the maritime industry, using systems that are increasingly accurate and close enough to reality, contributes to avoiding the loss of resources in situations that may arise in the event of maritime vessel failures. In this paper, we present a simulator based on software-in-the-loop for simulating the behavior of the Krick Felix robotic boat in three-dimensional marine environments. The proposed simulator has been designed based on the software-in-the-loop strategy implemented by Ardupilot. The experimental simulations confirm the validity of the behavior achieved by the proposed simulator in the generated environments.

Keywords: *Ardupilot, Gazebo, Krick Felix, Simulator, software-in-the-loop*

Introducción

En el desarrollo de la industria marítima se ha vuelto una necesidad la implementación de sistemas de simulación cada vez más robustos y exactos. Las simulaciones de alto nivel tienen la posibilidad de simular entornos lo suficientemente cercanos a la realidad, que resultan útiles para probar las capacidades de las embarcacio-

nes ante distintos cambios del medio. Estas son consideradas un salvoconducto, ya que evitan la pérdida de recursos en situaciones reales ante fallos en las embarcaciones.

Las simulaciones basadas en *software-in-the-loop* (SITL) permiten la simulación de entornos ambientales y condiciones físicas, además de componentes de hardware modelados mediante expresiones matemáticas (Demers et˜al., 2007). Este tipo de simulación no necesita de componentes físicos a diferencia del *hardware-in-the-loop* (HITL). La estrategia *software-in-the-loop* permite vincular un grupo de programas para la experimentación virtual, incluyendo las generalizaciones de modelos matemáticos para simular los elementos reales del sistema (Russo et˜al., 2007). Los SITL ofrecen múltiples ventajas entre ellas resalta la validación de modelos dinámicos y secciones de código para el control del sistema. Además, pueden ser usados en la etapa de prueba para analizar el comportamiento del sistema y así evitar daños y pérdidas de elementos físicos de alto valor. La flexibilidad y los bajos costos computacionales de las simulaciones son otras de las ventajas que los distinguen (Demers et˜al., 2007).

Las aplicaciones de la estrategia SITL en vehículos autónomos son recurrentes (Silano and Iannelli, 2021; Windiatmaja et˜al., 2021). En (Yazdani et˜al., 2020), los autores se apoyan en la plataforma SITL "Flinders"para la comprobación de algoritmos de seguimiento de trayectorias. Simulaciones basadas en la estrategia SITL para el análisis del comportamiento de un vehículo aéreo tipo quadcopter son presentadas en (Nguyen and Nguyen, 2019). Por su parte en (Jackson et˜al., 2016) se desarrolla otra investigación que aborda las potencialidades de los SITL en simulaciones orientadas a analizar el comportamiento de un vehículo áereo multirotor, con inclusión de sensores externos como cámaras o escáneres láser. Aunque múltiples son las investigaciones que abordan la aplicación de los SITL (Yazdani et˜al., 2020; Demers et˜al., 2007; Nguyen and Nguyen, 2019), la implementación de los mismos varía según los software que se utilicen. La estructura básica de los SITL para vehículos autónomos se centra en tres elementos principales: los complementos del SITL, los Sistemas de Control en Tierra *(Ground Control Systems, por su siglas en inglés (GCS))* y los simuladores 3D.

Complementos del SITL: Están compuestos por estructuras de software donde se incluye el código de control del hardware previsto y las herramientas de acceso y simulación de elementos del software. Se incluyen librerías de trabajo como las de la plataforma Ardupilot, así como el sistema operativo y *middleware Robot Operative System* (ROS) para el control de robots (Jackson et˜al., 2016; Sardinha et˜al., 2018; Nguyen and Ha, 2018). Las herramientas utilizadas como complementos SITL pueden por sí mismas crear el SITL, pero su implementación por sí sola carece de objetivo, ya que se necesitan software de visualización para facilitar el entendimiento por parte de los operadores.

Ground Control Systems: Son utilizados para el trazado de la trayectoria de los vehículos. Contienen funcionalidades para la comunicación con los software y hardware de control. Los GCS permiten la ubicación

espacial en dos dimensiones de los objetivos. Varios autores muestran (Nguyen and Ha, 2018; Kim and Lee, 2019) la factibilidad de este tipo de software en la ubicación espacial de los vehículos autónomos. Estos software pueden ser destinados al desarrollo de HITL, entre ellos *QGround Control* y *Mission Planner*. Sin embargo, las potencialidades de los mismos pueden ser aprovechadas por los SITL para generar entornos virtuales semejantes a los reales.

Simuladores 3D: Son software que se utilizan para generar entornos virtuales tridimensionales que incluyen modelos físicos y matemáticos de los vehículos y sensores. Para la creación de un SITL no es obligatorio el uso de los mismos, pero su inclusión permite la simulación de colisiones, la simulación de motores físicos, la recreación de ambientes y la inclusión de perturbaciones y obstáculos. Algunos de los simuladores utilizados en investigaciones relacionadas con la robótica son: Gazebo y *Flight Gear* (Sardinha et˜al., 2018; Kim and Lee, 2019). Gazebo permite la inclusión de diseños virtuales preconcebidos, que logran semejanza a los reales mediante modelos matemáticos programables. Mientras que *Flight Gear* es utilizado en soluciones destinadas a vehículos aéreos.

En general la implementación de simulaciones SITL se ha manifestado de diversas maneras debido a la variedad de software que permiten su desarrollo. Específicamente, las soluciones SITL basadas en la plataforma Ardupilot han sido ampliamente utilizadas debido a que esta plataforma brinda soporte a una variedad de microcontroladores dedicados a la robótica. Motivados por la utilidad de las simulaciones de alto nivel y las potencialidades de la plataforma Ardupilot para brindar soporte al microcontrolador Pixhawk PX4 usado en las investigaciones del Grupo de Automatización, Robótica y Percepción (GARP) de la Universidad Central "Martha Abreu"de Las Villas (UCLV), en este trabajo se presenta un simulador basado en la estrategia SITL para el bote robótico *Krick Felix*. Este vehículo marino constituye una aproximación a escala de un buque del puerto de Hamburgo. La embarcación cuenta a bordo con una variedad de sensores (telemetría, GPS, barómetro, etc.) y elementos de acción final (motor de paso, motor por modulación PWM, etc.) conectados al controlador del barco Pixhawk Px4 (Hernández-Morales et˜al., 2020). Vale señalar, que la realización de pruebas experimentales en entornos marinos con el bote robótico *Krick Felix* somete a los elementos de hardware a daños de distinta índole. Por lo que, el desarrollo de un ambiente de simulación donde se puedan realizar maniobras contribuirá a detectar y reducir afectaciones en el desempeño de la embarcación. Tomando en consideración lo antes expuesto, esta investigación tiene el objetivo de diseñar un simulador basado en *software-in-the-loop* para representar el comportamiento del bote robótico *Krick Felix* en entornos marinos tridimensionales.

El presente trabajo se organiza como sigue. En la próxima sección se presenta el diseño del simulador para el bote robótico *Krick Felix*, lo cual constituye la principal contribución de este trabajo. En la sección Resultados y Discusión se muestra un análisis de los resultados alcanzados en las simulaciones realizadas para confirmar

la validez del simulador desarrollado. Las conclusiones y trabajo futuro son presentados a continuación de la discusión de los resultados.

Materiales y métodos

El simulador propuesto parte de generar en el software Gazebo un entorno marino virtual que incluya al vehículo *Krick Felix*. En este ambiente virtual se incorporan las variables físicas que definen el entorno de operación del bote *Krick Felix*, así como el modelo tridimensional del barco con parámetros como masa, inercia, relación de longitudes, entre otros. Una vez construido este ambiente visual, el próximo paso consiste en conectar el modelo del *Krick Felix* con el autopiloto disponible de la plataforma Ardupilot. Para ello se utiliza el *complemento Ardupilot-Gazebo*.

El simulador se sustenta en la interconexión de tres programas principales: SITL de Ardupilot, el GCS QGroundControl y el programa de simulaciones y visualización tridimensional Gazebo. Las pruebas presentadas en la sección de Resultados y Discusión para comprobar la validez del simulador desarrollado han sido ejecutadas en el sistema operativo Ubuntu 18.04LTS.

Ardupilot: Constituye uno de los grandes proyectos en el mundo de la robótica y el control de vehículos no tripulados. La plataforma Ardupilot, entiéndase como el software resultante del conocido proyecto, contiene autopilotos dedicados a múltiples vehículos *(rover, plane, copter)*. APMrover2 versión 3.4.2 es el autopiloto utilizado para realizar el SITL. Esta versión es una de las más estables del software desde su aparición. Ardupilot cuenta con herramientas que permiten la simulación del firmware en el entorno virtual.

QGroundControl: El software de control de misión es utilizado en su versión de salida 4.0.8. Este software brinda las facilidades para establecer el trazado de trayectorias y seleccionar los distintos modos de trabajo utilizados entre los puntos de las trayectorias. Suele ser útil también porque permite orientarle al software de control las señales específicas de determinados sensores o actuadores de los vehículos.

Gazebo: Gazebo 9.12.0 es la versión de este software que se utiliza para el desarrollo del simulador. El software cuenta con múltiples herramientas dedicadas a la inserción de modelos, componentes que representan las fuerzas y perturbaciones marinas, así como la manipulación del entorno y sus cámaras para la visualización de la simulación. Gazebo implementa una estructura basada en mundos, modelos y complementos. Los mundos son archivos generados a través de extensiones `.world` programados a través del formato XML *(Extensible Markup Language)* y contienen las especificaciones del terreno, así como los distintos modelos del entorno. Los modelos insertados están programados a través del formato XML dispuestos en archivos de extensión `.sdf` y cuentan con las características propias de los objetos que representan. Los complementos constituyen

herramientas aditivas insertadas dentro de los modelos o mundos que permiten agregar una serie de algoritmos matemáticos para definir la interacción entre los objetos del mundo.

Entorno de desarrollo

El entorno del simulador es desarrollado en el software Gazebo. El mundo utilizado para el desarrollo del simulador dedicado al bote *Krick Felix* cuenta con una superficie que representa y se comporta como el entorno marino acostumbrado para este tipo de vehículos. Para ello se aprovecha el mundo desarrollado por (Mainwaring, 2020), el cual cuenta con un entorno marino con presencia de comportamientos y variables características como el viento y las olas. Las olas presentes en la superficie marina están basadas en espectros de energía (Milián et˜al., 2019). Estas ondas son sustentadas por el modelo matemático que representa al espectro de Pierson-Moskowitz. De esta forma resulta posible el uso de datos estadísticos derivados de la oceanografía física. La simulación se realiza atendiendo a estos criterios, contando con la suficiente exactitud para poder representar los distintos estados y la visualización de diferentes efectos visuales.

El siguiente elemento imprescindible en el simulador lo constituye el modelo matemático y tridimensional del bote *Krick Felix*. Este modelo cuenta con un apartado matemático que representa las características físicas del bote. Algunas de las características de este modelo matemático de 3 GDL son de sencilla inserción en el código de Gazebo. Sin embargo, otras interacciones como las existentes entre las superficies del bote y el mar necesitan de otras herramientas para ser insertados dentro de Gazebo. Dos de los complementos que cobran gran importancia son los llamados *Buoyancy* y *LiftandDrag*. Estos complementos permiten establecer la flotabilidad y las fuerzas de empuje y arrastre generados por el barco al interactuar con el medio marino. Para el trabajo con estas fuerzas se definen parámetros como: la densidad del fluido, el centro volumétrico y el volumen del objeto, todos dentro del código correspondiente al modelo del bote. La Figura 1 muestra la apariencia real y simulada en Gazebo del bote *Krick Felix*.

(a) Bote real (b) Bote simulado

Figura 1: Bote robótico *Krick Felix*.

Creación del mundo de trabajo

El mundo de trabajo tridimensional que se ha desarrollado integra dos elementos centrales: el entorno marino virtual y el modelo matemático del bote. El entorno marino virtual incluye parámetros del mundo como la representación visual del cielo, la superficie marina y la terrestre. En el software Gazebo el mundo es generado a partir del archivo con extensión `.world`. El área terrestre es un factor importante en el entorno donde debe operar el barco. El terreno se recrea a partir del área real de pruebas del barco. Gazebo permite generar terrenos a partir de mapas de altura. El código correspondiente al área del terreno debe contar además, con una malla de colisión que detenga el avance del bote al interactuar con las regiones terrestres. El último elemento del mundo marino virtual necesario es la superficie marina. Para ello se aprovecha el código presentado por (Mainwaring, 2020), a través de la inclusión del mundo `ocean-waves.world`.

Una vez desarrollado el mundo marino virtual se hace necesario incluir el modelo del bote *Krick Felix*. El modelo creado dentro de los directorios del simulador contiene dos archivos principales. El primero es un archivo con extensión `.config`, que especifica las características del modelo (`.sdf`) y los datos de los realizadores. El segundo es el archivo con extensión `.sdf`, el cual contiene todas las características dinámicas y de programación destinada a definir el bote *Krick Felix* de manera virtual. Algunas de las características en la estructura del código del modelo son las etiquetas para describir la forma principal del casco del bote, las mallas de colisión, los parámetros relacionados a la masa y los vectores de inercia.

Al cuerpo principal del modelo se le adhieren nuevos objetos tipo `<link>`, a través de uniones `<joint>` que permitan darle libertad de movimiento a elementos como la propela y el timón. La propela del bote cuenta con tres hélices unidas a un eje central, cada una con sus respectivas mallas de colisión. Esta propela utiliza el complemento `LiftAndDragplugin` para generar las fuerzas de arrastre y empuje que producen las aspas al rotar e interactuar con el agua. El timón del barco, de forma similar a la propela, está unido al cuerpo principal del bote y utiliza los mismos complementos para garantizar el cambio de dirección del bote cuando se mueve el timón en distintos sentidos.

Los sensores con los que cuenta el modelo son la Unidad de Medición Inercial (IMU por sus siglas en inglés) y el sensor LIDAR. El sensor LIDAR permite detectar objetos en las inmediaciones cercanas al bote. Este se encuentra en la proa del barco en dirección frontal. A dicho sensor se le debe especificar características de gran importancia como la frecuencia de muestreo y el ángulo de barrido del sensor. También, se cuenta con la intención de colocar una cámara para el análisis de imágenes de misiones en futuros proyectos. La IMU es vital para el modelo del bote. Este sensor realiza mediciones necesarias para conocer la posición y estabilidad del bote respecto a un sistema de referencia. Se vuelve imprescindible a la hora de la comunicación con el SITL de Ardupilot, ya que retroalimenta al autopiloto de Ardupilot y le permite conocer al operador y al sistema el

estado del modelo simulado.

Comunicación entre el SITL de Ardupilot y Gazebo

Para simular el modelo tridimensional del bote se establece la conexión Ardupilot-Gazebo a través del complemento `ardupilot-gazebo-plugin`. Una vez incorporado el complemento en Gazebo y activada la instancia de Ardupilot, automáticamente se habilitan dos puertos de comunicación (entrada-salida). Los puertos son enumerados por defecto con los valores 9002 para la entrada y 9003 para la salida. Sin embargo, si más instancias de Ardupilot se ejecutan, más puertos son habilitados por $9002 + 10 * n$ y $9003 + 10 * n$, para entradas y salidas respectivamente, donde n corresponde al número de instancias, comenzando desde cero. Esta configuración de nuevos puertos debe ser especificada para evitar conflictos en la conexión de nuevos robots. Cambiando la arquitectura del complemento se puede agregar múltiples robots a la conexión (Sardinha et˜al., 2018). La comunicación entre Gazebo y Ardupilot, a través del complemento antes mencionado, permite que exista una relación entre los parámetros físicos que poseen los modelos tridimensionales y los algoritmos matemáticos destinados a controlarlos. La Figura 2 muestra el esquema de interconexión entre los componentes del simulador.

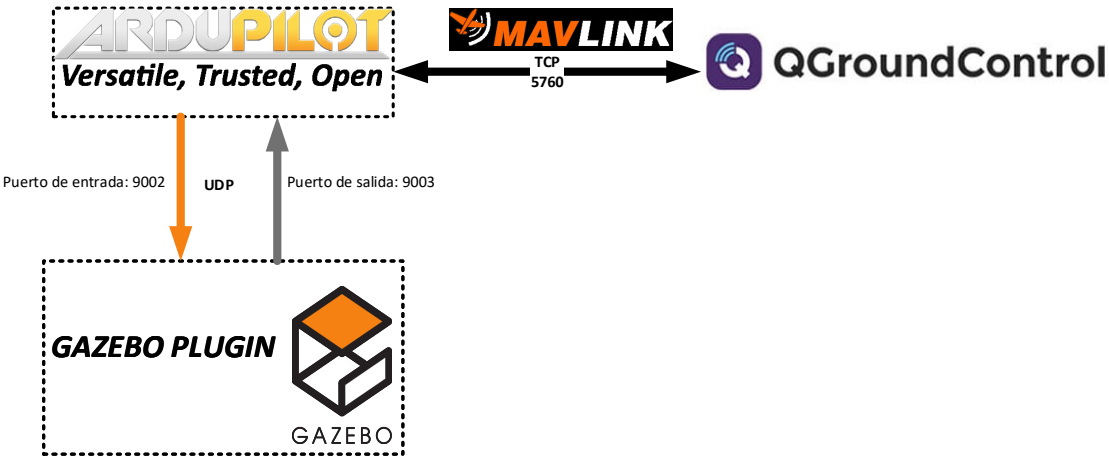


Figura 2: Interconexión entre los componentes del simulador.

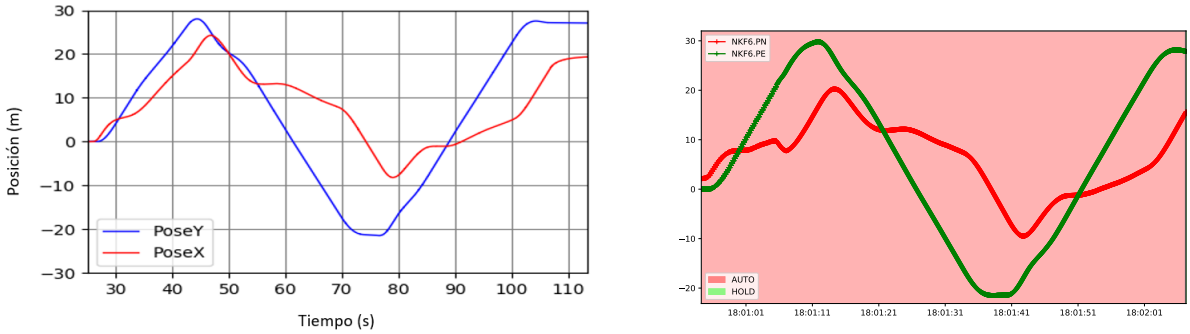
Resultados y discusión

Para comprobar el correcto funcionamiento del simulador se desarrollada una prueba con el objetivo de comprobar el correcto funcionamiento de los distintos software que intervienen en el simulador, a la par de observar la fiabilidad y correspondencia del mismo con el entorno creado. Para el desarrollo de la prueba se ejecutó con-

juntamente el SITL de Ardupilot, el entorno creado en Gazebo y el GCS QGroundControl. El GCS es utilizado para trazar la trayectoria a seguir por el bote modelado y especificar los modos de funcionamiento presentes en el firmware de Ardupilot que guiarán los comportamientos del bote virtual. La Figura 3 muestra la trayectoria trazada en el GCS y la seguida por el bote mediante el algoritmo de guiado.

Figura 3: Visualización de la trayectoria deseada y realizada por el bote.

También fueron analizados los comportamientos de algunas variables de mayor importancia para el control del vehículo, entre ellas la posición en X e Y. Estas variables son vitales para los algoritmos de control del vehículo, por lo que debe existir una correspondencia entre los movimientos generados en Gazebo y los interpretados por el SITL de Ardupilot. La Figura 4 muestra la correspondencia entre los comportamientos de las variables de posición en el plano horizontal del vehículo durante el desarrollo de la misión planteada.



(a) Posición generada en Gazebo

(b) Posición interpretada en Gazebo

Figura 4: Comportamiento de los movimientos en el plano horizontal.

Al bote virtual se le ha incluido un sensor LIDAR y una cámara virtual para su uso futuro en algoritmos de detección de obstáculos. Estos algoritmos se encuentran en desarrollo por el GARP y podrán ser probados en el simulador. En la Figura 5 se puede observar la detección de un obstáculo por parte del sensor LIDAR, así como la vista frontal percibida por la cámara del bote.

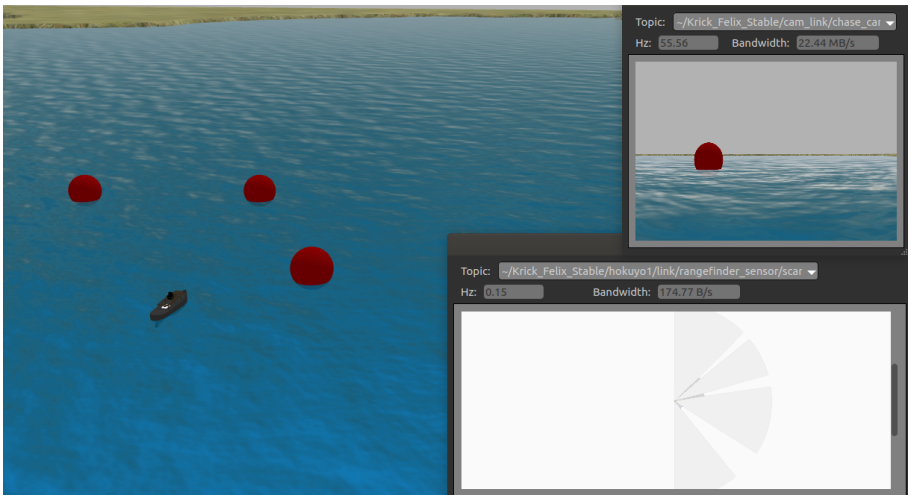


Figura 5: Detección de obstáculos en el simulador.

Conclusiones

En el presente trabajo se diseña un simulador para representar el comportamiento del bote robótico *Krick Felix* en entornos marinos tridimensionales con la herramienta Gazebo. El simulador ha sido implementado basado en la estrategia *software-in-the-loop* de la plataforma Ardupilot, lo que permite generar entornos virtuales de control y evaluar el desempeño de distintos tipos de algoritmos. Los resultados alcanzados en las simulaciones confirman la validez del comportamiento alcanzado por el simulador en los entornos marinos generados. El simulador desarrollado contribuye a disminuir el riesgo de errores durante la navegación, así como a asegurar un mejor desempeño de los algoritmos implementados en el vehículo. La inclusión de un sensor LIDAR y una cámara virtual permite el uso del simulador en futuras investigaciones para la detección y evasión de obstáculos.

Referencias

Demers, S., Gopalakrishnan, P., and Kant, L. (2007). A generic solution to software-in-the-loop. In *IEEE Military Communications Conference*, Florida, Estados Unidos. IEEE.

Hernández-Morales, L., Valeriano-Medina, Y., Hernández-Santana, L., and Mesa-Suarez, E. (2020). Nonlinear guidance law algorithm applied to a small unmanned surface vehicle. *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment*, 234(3):623–633.

Jackson, T., Ellingson, G., and McLain, T. (2016). ROSflight: A lightweight, inexpensive mav research and development tool. In *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 758–762, Arlington, Estados Unidos. IEEE.

Kim, H. and Lee, D. (2019). Integrated simulation environment for heterogeneous unmanned vehicles using ros and pixhawk. *Academia Journal of Scientific Research*, 3(3):1–20.

Mainwaring, R. (2020). Srmainwaring. Página web: www.github.com/srmainwaring/asv-wave-sim/, último acceso: Mayo-2020.

Milián, O., Valeriano, Y., and Oria, H. J. (2019). DEPTHWAVE: aplicación para la simulación gráfica del comportamiento del mar. *Revista Cubana de Ciencias Informáticas*, 13:105 – 115.

Nguyen, K. and Ha, C. (2018). Development of hardware-in-the-loop simulation based on gazebo and pixhawk for unmanned aerial vehicles. *International Journal of Aeronautical and Space Sciences*, 19:238–249.

Nguyen, K. and Nguyen, T. (2019). Vision-based software-in-the-loop-simulation for unmanned aerial vehicles using gazebo and px4 open source. In *2019 International Conference on System Science and Engineering (ICSSE)*, Dong Hoi, Viet Nam. IEEE.

Russo, R., Terzo, M., and Timpone, F. (2007). Software-in-the-loop development and validation of a cornering brake control logic. *Vehicle System Dynamics*, 45(2):149–163.

Sardinha, H., Dragone, M., and Vargas, P. (2018). Closing the gap in swarm robotics simulations: An extended ardupilot/gazebo plugin. Reporte técnico ID: 53669810, ArXiv.

Silano, G. and Iannelli, L. (2021). Mat-fly: An educational platform for simulating unmanned aerial vehicles aimed to detect and track moving objects. *IEEE Access*, 9:39333–39343.

Windiatmaja, J. H., Tjahaja, J. C., Amin, K. A., and Sari, R. F. (2021). Implementation of socket priority module for unmanned aerial vehicle network using FlyNetSimulator. *IOP Conference Series: Materials Science and Engineering*, 1077(1):012021.

Yazdani, A., Sammut, K., Yakimenko, O., and Lammas, A. (2020). A survey of underwater docking guidance systems. *Robotics an Autonomous Systems (2019)*, 124:12–29.