Revista Iberoamericana de Automática e Informática Industrial

Revista Iberoamericana de Automática e Informática Industrial 22 (2025) 208-218

www.revista-riai.org

Software de seguimiento de *way-points* para vehículos teleoperados subacuáticos.

Luciano Alonso Renteria[a], Jose Joaquin Sainz[a,*], Elías Revestido Herrero[a], Jose Ramon Llata[a], Carlos Torre-Ferrero[a], Javier Llamas [b].

[a] *Departamento de tecnología electrónica, ingeniería de sistemas y automática, Universidad de Cantabria, Av. de los Castros, s/n, 39005 Santander, España.*
[b] *Casco Antiguo Comercial S.L. C/ de Senda Galiana D15, 2821, Coslada, Madrid, España.*

To cite this article: Alonso, L., Sainz, J.J., Revestido, E., Llata, J.R., Torre, C., Llamas, J. 2025. Way-point tracking software for underwater teleoperated vehicles. Revista Iberoamericana de Automática e Informática Industrial 22, 208-218. https://doi.org/10.4995/riai.2025.21948

Resumen

En este trabajo se presenta una herramienta de software desarrollada para realizar el seguimiento de trayectorias mediante *way-points* de forma autónoma por un "Vehículo submarino teleoperado" (ROV). Se ha realizado la integración mediante una interfaz gráfica del *software* de control y del software desarrollado para el seguimiento de la trayectoria facilitando al operador el establecimiento de la misma en un entorno conocido y el inicio de seguimiento autónomo de forma sencilla. Además de las simulaciones por computador, se han realizado ensayos en la bahía de Santander donde se ha verificado el *software* desarrollado. Es posible el empleo de esta investigación con fines industriales y docentes para la formación de ingenieros relacionados con el control y el manejo de este tipo de vehículos y de aquellos usuarios que desean desarrollar software con lenguajes de programación como Python.

Palabras clave: Adquisición de datos de sensores remotos, Educación continua de control en la industria, Laboratorio virtual y remoto, Posicionamiento dinámico, Vehículo submarino autónomo.

Way-point tracking software for remotely operated vehicles

Abstract

This work presents a software tool developed to autonomously track trajectories using *Way-points* by a "*Remotely Operated Vehicle*" (ROV). The integration of the control software and the software developed for the trajectory tracking has been carried out by means of a graphical interface, making it easier for the operator to establish the trajectory in a known environment and to start autonomous tracking in a simple way. In addition to the computer simulations, tests have been carried out in Santander Harbour where the developed software has been verified. It is possible to use this research for industrial and educational purposes for the training of engineers related to the control and operation of this type of vehicle and for those users who wish to develop software with programming languages such as Python.

Keywords: Autonomous underwater vehicles, Continuing control education in industry, Dynamic positioning, Remote sensor data acquisition, Virtual and remote labs.

1. Introducción

La evolución de las capacidades de los vehículos submarinos destinados a la supervisión de estructuras marinas ha experimentado un auge importante constituyendo un tema de investigación muy activo de crecimiento exponencial en las últimas décadas (Schjlberg and Utne, 2015), (Fay *et al*., 2019), (Sainz Gutiérrez *et al*., 2022). Estas tareas de supervisión implican operaciones en ocasiones de alto riesgo, como la exploración de zonas submarinas, la realización de trayectorias de prospección que recojan datos para elaborar mapas batimétricos y las inspecciones de las citadas estructuras, que requieren de métodos complejos de control del movimiento, así como de métodos de localización con diferentes sistemas de medida. Los recientes avances en las prestaciones de los "Vehículos Submarinos no Tripulados" (UUV) les han

*Autor para correspondencia: sainzjj@unican.es

otorgado un papel importante en las citadas operaciones, relegando a los ROV y a los sumergibles tripulados a un segundo plano. Sin embargo, las aplicaciones de intervención, en las que el sistema manipula el entorno, siguen siendo realizadas principalmente por un ROV debido a la complejidad de la tarea. En la operación remota de bucle abierto, el usuario debe realizar maniobras con cierta exactitud mientras se enfrenta a corrientes, olas y los efectos del cable umbilical sobre la dinámica del ROV. Además, para algunas clases específicas de ROV (I o II) (Christ and Wernli, 2014), (Capocci et al., 2017), las restricciones cinemáticas debidas a la subactuación pueden ser un obstáculo adicional para el rendimiento de la teleoperación. Por lo tanto, un esquema de control aplicado a un ROV de forma total o semiautónoma es probablemente la mejor solución para estas aplicaciones submarinas.

En lo que se refiere al control de movimientos del ROV se pueden encontrar en la literatura un conjunto de publicaciones amplio donde se incluyen desde controladores clásicos PID combinados con otras técnicas hasta controladores avanzados de diferentes tipos como se puede ver en las publicaciones de Álvarez et al.(2009), Dong and Duan (2023), Chen et al.(2020), Luo et al.(2024) y Hosseinnajad et al. (2023) por citar algunas de las contribuciones más representativas sin profundizar en este aspecto de la literatura al no ser objeto principal de este trabajo. Por otra parte, el desarrollo de herramientas de *software* que incluyen los citados controladores amplía significativamente las capacidades de los vehículos, al realizar de manera autónoma el seguimiento de la trayectoria establecida por los diferentes *way-points*, dejando de depender del manejo a distancia por parte de un operador humano o proporcionado soporte al mismo. Con estas nuevas capacidades los vehículos pueden trabajar en diferentes actividades de monitorización estructuras submarinas, parques eólicos *offshore* o en el mantenimiento de embarcaciones entre otras muchas actividades. Con la automatización de las operaciones de traslado y utilización de herramientas gracias a los ROV hasta los puntos de interés, se puede disminuir notablemente el coste de las operaciones al reducir el personal necesario para la realización de las mismas y programar la supervisión de determinadas operaciones y elementos subacuáticos. En el estado del conocimiento se puede encontrar trabajos relacionados con ROVs y el desarrollo de *software* para su control (Martínez et al., 2013), (Aguirre-Castro et al., 2019), (Gupta and Maurya, 2021), haciendo que amplíen sus capacidades.

En relación a los criterios para establecer una clasificación entre entornos de experimentación, tanto para aplicaciones docentes como de investigación, se encuentran dos aspectos fundamentales: uno es la forma de acceder a los recursos de experimentación y otro es la naturaleza de los recursos disponibles. El primero es el llamado acceso local, es decir, la experimentación in situ¸ con el propio recurso. Otra forma es el acceso remoto, lo que se conoce habitualmente como experimentación a través de Internet, donde no es necesaria la presencia física en la zona donde se opera (Andújar and Mateo, 2010), (Velasco *et al*., 2012), (Velasco *et al*., 2018). El *software* desarrollado en este trabajo corresponde a la primera vía (in situ) con el propio recurso, en este caso un ROV para la experimentación en aguas abiertas de la Bahía de Santander. Por otro lado, se encuentran en la literatura lo que se conoce como laboratorios virtuales (Andújar and Mateo, 2010), (Martínez *et al*., 2017)(Andújar *et al*., 2010)(Martínez *et al*., 2017) en los que la experimentación se basa en un modelo matemático simulado, aspecto que también se incluye en este trabajo mediante un modelo de maniobra de un ROV para poder establecer una comparativa entre las pruebas en entorno real y el virtual.

En este trabajo se presenta un software de seguimiento de trayectorias mediante *way-point* para un ROV abierto con fines docentes y de investigación. Para ello se ha realizado una interfaz gráfica que integra diferentes aplicaciones *software* por medio del lenguaje de programación Python. Se propone una aplicación docente donde se incluyen simulaciones y ensayos en la bahía de Santander. Por otra parte, el software desarrollado en este trabajo, así como los controladores incluidos en el mismo sirve de punto de partida para llevar a cabo posteriormente comparativas con controladores más avanzados y en base a estos llegar a realizar una monitorización evolutiva de estructuras subacuáticas, objetivo del proyecto citado en los agradecimientos de este artículo.

2. Modelo dinámico del ROV

El modelo de maniobra no lineal, del ROV objeto de estudio de este trabajo (ver Apéndice A para más detalles) puede expresarse de la siguiente forma (Fossen T. I., 2002), (Fossen T. I., 2011), (von Benzon, et al., 2022):

$$M\,\dot{\nu} + C(\nu)\nu + D(\nu)\nu + g(\eta) = \tau \quad (1)$$

$$\dot{\eta} = J(\eta)\nu \quad (2)$$

Donde $\eta = [x, y, z, \varphi, \theta, \psi]^T$es el vector de posición y ángulos de Euler, $\nu = [u, v, w, p, q, r]^T$ son las velocidades lineales y las velocidades angulares, $\tau = [X, Y, Z, K, M, N]^{\mathrm{T}}$ son las fuerzas y momentos. *M* es la matriz del cuerpo rígido y la masa añadida, *C(ν)ν* es el término de Coriolis, *g(η)* es la matriz de restauración, *J(η)* es la matriz de rotación y *D(ν)ν* representa las fuerzas de amortiguamiento hidrodinámico que son una combinación de amortiguamiento lineal y no lineal. La relación cinemática entre la velocidad *ν* en el sistema de coordenadas fijas al cuerpo y la posición *η* en el sistema de coordenadas que se encuentra en la Tierra en "*North East Down*" (NED) (Figura 1) viene dada por:

$$J(\eta) = \begin{bmatrix} J_1(\eta) & 0_{3x3} \\ 0_{3x3} & J_2(\eta) \end{bmatrix} \quad (3)$$

$$J_1(\eta) = \begin{bmatrix} J_{1A} & J_{1B} & J_{1C} \\ J_{1D} & J_{1E} & J_{1F} \\ J_{1G} & J_{1H} & J_{1I} \end{bmatrix} \quad (4)$$

Donde:

$$\begin{aligned} J_{1A} &= \cos(\psi)\cos(\theta) \\ J_{1B} &= -\sin(\psi)\cos(\phi)+\cos(\psi)\sin(\theta)\sin(\phi) \\ J_{1C} &= \sin(\psi)\sin(\phi)+\cos(\psi)\cos(\theta)\sin(\phi) \\ J_{1D} &= \sin(\psi)\cos(\theta) \\ J_{1E} &= \cos(\psi)\cos(\phi)+\sin(\phi)\sin(\theta)\sin(\psi) \\ J_{1F} &= -\cos(\psi)\sin(\phi)+\sin(\theta)\sin(\psi)\cos(\phi) \\ J_{1G} &= -\sin(\theta) \end{aligned} \quad (5)$$

$J_{1H} = \cos(\theta)\sin(\phi)$
$J_{1I} = \cos(\theta)\cos(\phi)$

$$J_2(\eta) = \begin{bmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi)/\cos(\theta) & \cos(\phi)/\cos(\theta) \end{bmatrix} \quad (6)$$

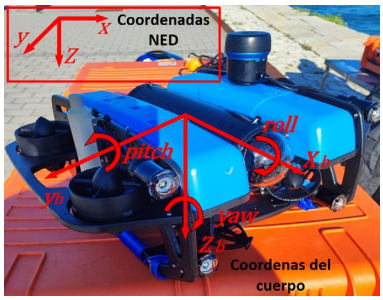


Figura 1 : Sistemas de coordenadas NED y coordenadas del cuerpo.

El resto de las matrices para el modelo (1) son las siguientes: Las matrices de cuerpo rígido y la masa añadida, $M = M_{RB} + M_A$.

$$M_{RB} = \begin{bmatrix} mI_{3X3} & 0_{3x3} \\ 0_{3x3} & I_C \end{bmatrix} \quad (7)$$

Donde m es la masa del vehículo, $I_C = \text{diag}\{I_x, I_y, I_z\}$ es el momento de inercia. Al realizar los desplazamientos sumergido en un fluido el vehículo desplaza parte del fluido circundante. Con el objeto de modelizar este fenómeno se añade una masa ficticia denominada masa añadida. La matriz de masa añadida es M_A.

$$M_A = -diag\{X_{\dot{u}}, Y_{\dot{v}}, Z_{\dot{w}}, K_{\dot{p}}, M_{\dot{q}}, N_{\dot{r}}\} \quad (8)$$

La matriz de Coriolis $C(v) = C_{RB}(v) + C_A(v)$. La matriz de Coriolis representa las fuerzas que se generan debido a la rotación del cuerpo respecto al marco de referencia NED. Los valores de los parámetros de las matrices $C_{RB}(v)$ correspondiente a las fuerzas de Coriolis generadas por el sólido rígido y $C_A(v)$ que son las fuerzas de Coriolis generadas por la masa añadida para representar fenómenos hidrodinámicos presentes en el movimiento, pueden consultarse en von Benzon et al. (2022).

$$C_{RB}(v) = \begin{bmatrix} 0 & 0 & 0 & 0 & mw & -mv \\ 0 & 0 & 0 & -mw & 0 & mu \\ 0 & 0 & 0 & mv & -mu & 0 \\ 0 & mw & -mv & 0 & -I_z r & -I_y q \\ -mw & 0 & mu & I_z r & 0 & I_x p \\ mv & -mu & 0 & I_y q & -I_x p & 0 \end{bmatrix} \quad (9)$$

$$C_A(v) = \begin{bmatrix} 0 & 0 & 0 & 0 & -Z_{\dot{W}} w & Y_{\dot{V}} v \\ 0 & 0 & 0 & Z_{\dot{W}} w & 0 & -X_{\dot{u}} u \\ 0 & 0 & 0 & Y_{\dot{v}} v & X_{\dot{u}} u & 0 \\ 0 & -Z_{\dot{W}} w & Y_{\dot{v}} v & 0 & -N_{\dot{r}} r & M_{\dot{q}} q \\ Z_{\dot{W}} w & 0 & -X_{\dot{u}} u & N_{\dot{r}} r & 0 & -K_{\dot{p}} p \\ mv & -mu & 0 & -M_{\dot{q}} q & K_{\dot{p}} p & 0 \end{bmatrix} \quad (10)$$

Las matrices de amortiguación $D(v) = D_l + D_{nl}(v)$.

$$D_l = -diag\{X_u, Y_v, Z_w, K_p, M_q, N_r\} \quad (11)$$

$$D_{nl}(v) = -diag\{X_{u|u|}|u|, Y_{v|v|}|v|, Z_{w|w|}|w|, K_{p|p|}|p|, M_{q|q|}|q|, N_{r|r|}|r|\} \quad (12)$$

Donde se incluyen los términos de amortiguamiento lineal D_l y los no lineales $D_{nl}(v)$. Las matrices de amortiguamiento se incluyen para representar matemáticamente los fenómenos de amortiguamiento que, en general y tal como se indica en Fossen T. I. (2011), están causados por el amortiguamiento potencial, la fricción, las fuerzas de elevación y la amortiguación debida a los vórtices.

El vector de las fuerzas de restauración es *g(η):*

$$g(\eta) = \begin{bmatrix} (W-B)sin(\theta) \\ -(W-B)cos(\theta)sin(\phi) \\ -(W-B)cos(\theta)cos(\phi) \\ y_b B\, cos(\theta)\, cos(\phi) - z_b B cos(\theta) sin(\phi) \\ -z_b B\, sin(\theta) - x_b B\, cos(\theta)\, cos(\phi) \\ x_b B\, cos(\theta)\, sin(\phi) + y_b B\, sin(\theta) \end{bmatrix} \quad (13)$$

Donde W $= mg$ y B $= \rho g \nabla$ son las fuerzas de gravedad y flotabilidad, respectivamente. A la actuación de estas fuerzas sobre el vehículo submarino se las denomina fuerzas de restauración (ρ la densidad del agua y ∇el volumen del fluido desplazado por el vehículo), y (x_b, y_b, z_b) son las coordenadas del centro de flotabilidad expresadas en el sistema de coordenadas de cuerpo del vehículo, ver Figura 1.

En esta sección, se ha expuesto el modelo matemático del ROV, constituyendo las nociones del comportamiento dinámico del vehículo que el estudiante ha de adquirir previo al estudio e implementación de controladores. En las simulaciones se ha empleado un *software* de simulación denominado *Software In The Loop* (SITL) que permite verificar las tareas de control antes de su implementación sobre el vehículo real. Además, es especialmente útil para para simular de forma aproximada el movimiento del ROV y que los alumnos se familiaricen con el manejo del sistema. Para más información acerca del modelo empleado por SITL consultar ArduSub. (2024).

3. *Software*

En esta sección, se presenta el *software* de seguimiento de *way-points* que integra un conjunto de aplicaciones *software* necesario para la utilización del ROV de este trabajo (ver Apéndice 1), estas últimas son de código abierto y fácilmente accesible en Internet. Los componentes principales que integra son los siguientes:

- ArduSub: es el *firmware* que realiza la tarea de piloto automático, y se ejecuta sobre una tarjeta compatible instalada en el ROV (ArduSub, 2024). Realiza el control del ROV y ejecuta los comandos enviados por el operador desde el computador de superficie mediante el protocolo Mavlink (MAVLink , 2024).
- *Software* de Acompañamiento: se encarga de las comunicaciones vía ethernet entre la estación de superficie y

el piloto automático. Se ejecuta en un computador tipo *Single Board Computer* (SBC) *instalado en el ROV.*

- QGroundControl: es la interfaz de usuario que se ejecuta en el computador de superficie. Permite la configuración del ROV, enviar comandos, y recibir y visualizar información de este (QGroundControl-user-guide, 2024).

Además, con el *software* ArduSub se incluye el *software* SITL que, como ya se ha comentado, permite verificar las tareas de control antes de su implementación sobre el vehículo real.

Todos estos componentes de *software* son independientes unos de otros, aunque trabajan conjuntamente, por lo que su manejo desde el computador de superficie es algo engorroso. Para facilitar su utilización se ha implementado una interfaz gráfica que permite, desde una única aplicación, iniciar y cerrar todos los programas, así como iniciar y detener una tarea de seguimiento de trayectorias mediante *way-points*.

Tanto la interfaz gráfica, como el algoritmo de seguimiento de la trayectoria se han desarrollado en lenguaje Python, utilizando los siguientes paquetes:

- PyQt5: paquete para el desarrollo de interfaces gráficas en Python, utilizando la librería Qt5 de C++.
- Pymavlink: paquete para la utilización del protocolo Mavlink en lenguaje Python.
- PyKML: paquete para la utilización en Python del lenguaje KML, creado por Google para la codificación de datos geográficos.
- NumPy: paquete para la realización eficiente de cálculos matemáticos en Python.
- Matplotlib: paquete para la visualización gráfica de resultados en Python.

La interfaz gráfica desarrollada puede observarse en la Figura 2 así como el Algoritmo 1 de seguimiento de seguimiento de *way-points.*

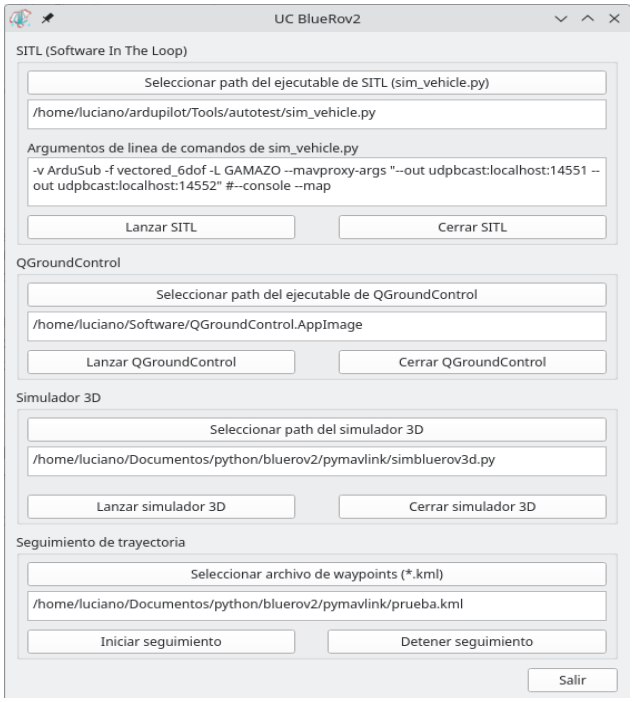


Figura 2. Interfaz gráfica desarrollada.

Antes de ejecutar el algoritmo de seguimiento es necesario establecer los *way-points,* para lo cual se utiliza QGroundControl, que permite definirlos y almacenarlos en un archivo en formato kml. Además, permite visualizar en tiempo real la trayectoria del ROV sobre un mapa, así como la trayectoria ideal entre *way-points.*

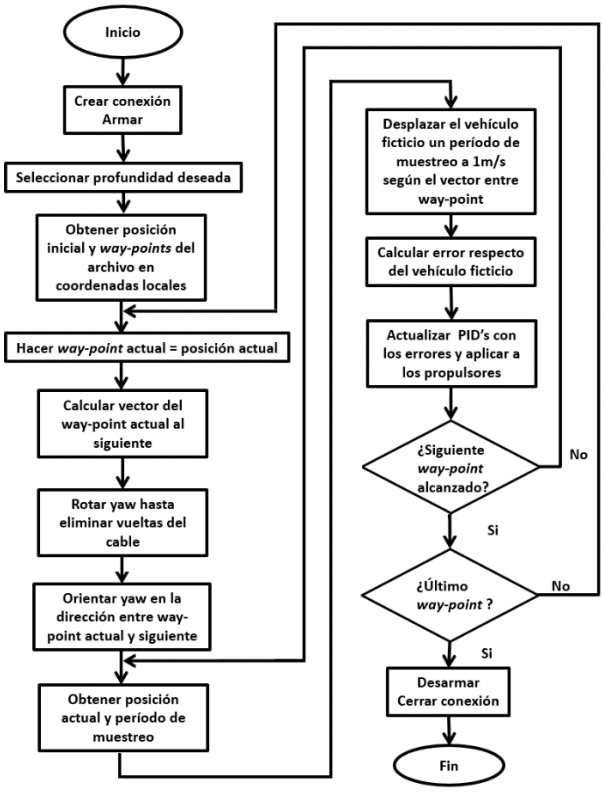
El algoritmo consiste en hacer que el submarino siga a un vehículo ficticio que se mueve de un *way-point* al siguiente en línea recta, a una velocidad constante de 1 m/s. Se podría emplear otro tipo de trayectorias en base a *splines*, ver Mora *et al*. (2023).

Se tiene que controlar cuatro grados de libertad: el *yaw* (rotación respecto del eje z, con referencia al norte), y las coordenadas x (longitudinal), y (transversal), z (vertical) ya que los otros dos grados restantes se autoestabilizan. Para ello utilizamos cuatro reguladores PID. Se han establecido los mismos valores en los parámetros de los cuatro reguladores PID ($K_p = 0{,}5$, $K_i = 0{,}01$, $K_d = 0{,}01$), sintonizados de forma manual utilizando el simulador SITL, que proporcionan resultados satisfactorios en las posteriores pruebas en la bahía de Santander, no siendo necesario otro ajuste de los valores.

Pasos para hacer un seguimiento de *way-points*:

1- En QGroundControl crear los *way-points* y salvarlos en un archivo ".plan" (formato json), o ".kml" (formato kml).
2- Lanzar la interfaz creada en Python.
3- En la interfaz seleccionar el archivo creado con los *way-points.*
4- En la interfaz lanzar el algoritmo de seguimiento (se puede detener en cualquier momento desde la propia interfaz).

Los principales pasos que ejecuta el algoritmo de seguimiento pueden observarse en Algoritmo 1.



Algoritmo 1. Algoritmo de seguimiento way-points.

4. Resultados

En esta sección se detallan las pruebas llevadas a cabo con el *software* descrito en el apartado anterior. Dichas pruebas han consistido en primer lugar en pruebas de simulación en el que se ha verificado el funcionamiento del *software* y se han ajustado los controladores. En segundo lugar, se han realizado pruebas reales en la bahía de Santander en las que se ha validado en un entorno real que el *software* propuesto funciona adecuadamente.

4.1 Resultados de pruebas en simulación

Se han realizado diversas simulaciones de operaciones para verificar que el *software* funciona correctamente previamente a las pruebas reales en la bahía de Santander.

La Figura 3 muestra una pantalla del programa QGroundControl al final de una ejecución del algoritmo descrito, ubicada en el muelle de Gamazo de Santander, utilizando el simulador SITL.

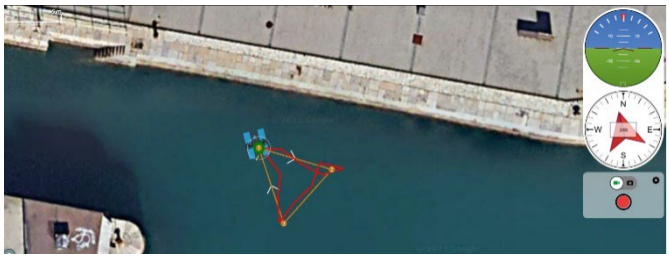


Figura 3: Pantalla del programa QGroundControl en una simulación.

Los resultados de la simulación son adecuados a los requerimientos de posición establecidos. Como puede observarse en la Figura 4 y Figura 5 el ROV es posicionado en *x* e *y* siguiendo la referencia, entrando dentro del radio de aceptación de cada uno de los *way-points* y por tanto alcanzados los *way-points* establecidos de forma correcta. Una vez alcanzado el *way-point* correspondiente el ROV comienza la corrección de las posibles vueltas del cable y se realiza la orientación en *yaw* para posteriormente realizar el desplazamiento hacia el nuevo *way-point.* Esto se manifiesta aparentemente como un desfase temporal entre la referencia y la trayectoria que no supone un mal seguimiento de las referencias.

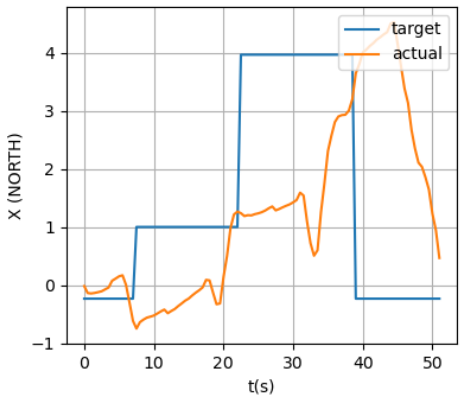


Figura 4: Resultados en x(m) de una simulación.

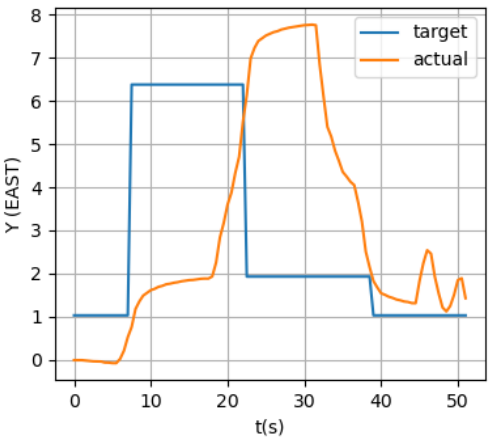


Figura 5: Resultados en y(m) de una simulación.

Hay que destacar que, aunque puede observarse que el ROV en ocasiones sobrepasa los *way-points* no supone un inconveniente debido a que las simulaciones se están empleando únicamente para configurar de manera preliminar el sistema y entrenar a los estudiantes, debido a que el modelo empleado en las simulaciones no representa de forma exacta la dinámica real del ROV. Como podrá observarse posteriormente en los resultados de las pruebas realizadas en la bahía de Santander no se sobrepasan los *way-points* establecidos por encima del radio de aceptación.

Por otra parte, en la Figura 6 y Figura 7 se observa como los movimientos realizados por el vehículo se hacen manteniendo la posición en *roll* y *pitch* manteniendo la estabilidad del ROV. Además, posiciona siguiendo la referencia en *yaw* con un error despreciable Figura 8, en la que se puede observar cómo entorno a los 10 segundos el ROV se posiciona en *yaw* para deshacer las vueltas que tiene el cable. A partir de la gráfica se puede inferir que ha deshecho una única vuelta y posteriormente retoma el posicionamiento hasta alcanzar la posición deseada entorno a los 100º. En este caso, se podría haber posicionado directamente el ROV de aproximadamente 70 º a 100º ya que solo había una vuelta en el cable que posteriormente se ha vuelto a sumar. Sin embargo, en caso de que el cable tuviera más de una vuelta es necesario deshacer por defecto las vueltas, siendo este procedimiento indispensable para salvaguardar la integridad del cable.

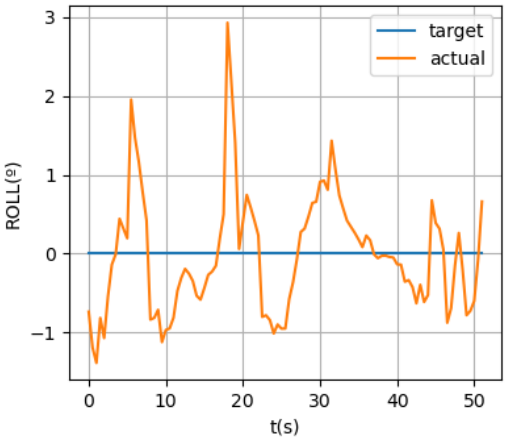


Figura 6: Resultados en roll de una simulación.

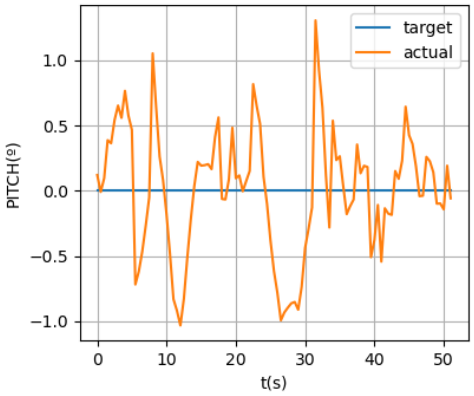


Figura 7: Resultados en pitch de una simulación.

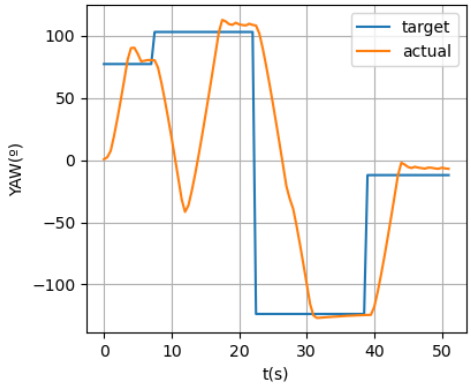


Figura 8: Resultados en yaw de una simulación.

En la Figura 9 y Figura 10 se muestra el periodo de la señal "*Pulse Width Modulation*" (PWM) con la que se comanda a los servos que forman parte de los motores de cada uno de los propulsores ver Apéndice A. El valor de PWM(microsegundos) está comprendido entre 1100 y 1900. Para más información sobre la relación entre la señal PWM y el empuje que proporciona cada propulsor consultar t200-thruster, (2024).

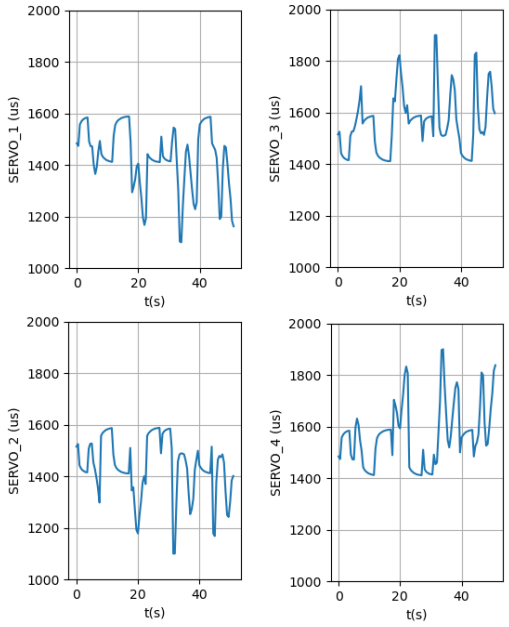


Figura 9 : Resultados de los motores 1 a 4 en una simulación.

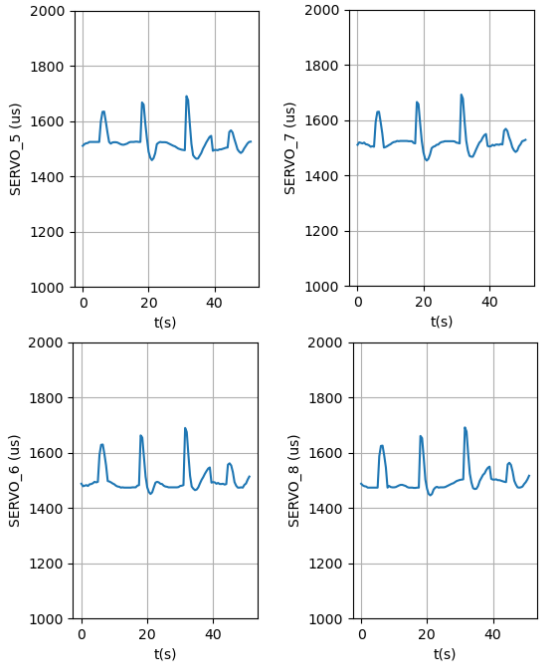


Figura 10: Resultados de los motores 5 a 8 en una simulación.

Por último, tal como puede observarse en la Figura 9 y Figura 10 hay que destacar que los actuadores tienen una respuesta aceptable ya que no presentan excesivas oscilaciones ni saturación persistente apreciable, lo cual es adecuado para alargar la vida útil de los motores.

4.2 Resultados de pruebas en la bahía de Santander

Tras realizar diversas simulaciones del *software* de seguimiento de *way-points* se ha procedido a realizar pruebas reales en la bahía de Santander, para ello se ha interconectado los equipos que componen el sistema (Figura 11).

Figura 11: Esquema de conexiones de los equipos.

Los elementos del sistema son los siguientes:

1- Maleta de control
2- Cable umbilical
3- SBL
4- ROV
5- DVL
6- Profundímetro
7- Mando de control
8- GPS superficie
9- Antena con receptores acústicos

Una vez que se ha montado y puesto en marcha el sistema se ha realizado el seguimiento de los *way-points* por parte del ROV ejecutando el *software* expuesto anteriormente. Como puede observarse en la Figura 12 el vehículo sigue la línea entre *way-point* de forma autónoma a una profundidad constante de 1m a pesar de las olas y las corrientes presentes en el entorno real de operación.

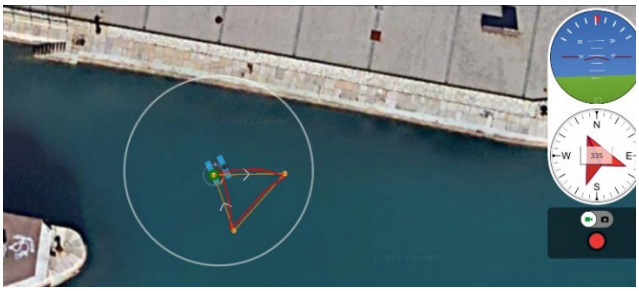


Figura 12: Pantalla del programa QGroundControl en una prueba en la bahía de Santander.

En la Figura 13 se observa como el vehículo se posiciona en *x* siguiendo la referencia sin excesivos sobreimpulsos. Por otra parte, el desfase temporal entre el *target* y la posición se debe a que una vez que ha llegado al *way-point* fijado revisa cuantas vueltas tiene el cable (ver Algoritmo 1) y se orienta en *yaw* para ir al siguiente punto.

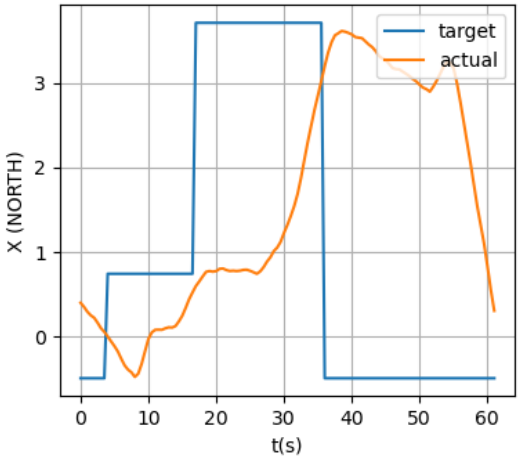


Figura 13: Resultados en x(m) de la prueba en una bahía de Santander.

De igual forma, en la Figura 14 se observa como el vehículo se posiciona en *y* de forma adecuada siguiendo la referencia sin grandes oscilaciones y con poco error respecto a la referencia.

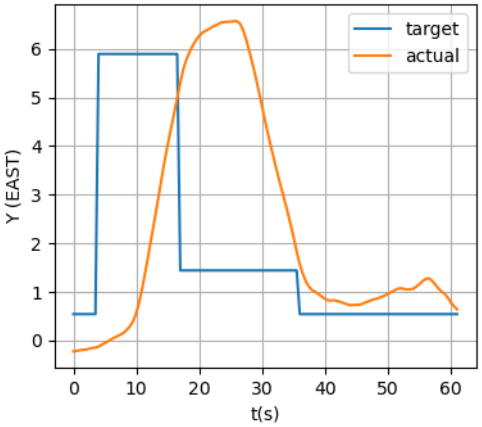


Figura 14: Resultados en y(m) de una prueba en una bahía de Santander.

Por otra parte, el vehículo sigue manteniendo la posición correctamente en *roll* y *pitch* de forma que realiza los movimientos de forma plana tal como se muestra en las Figura 15 y Figura 16.

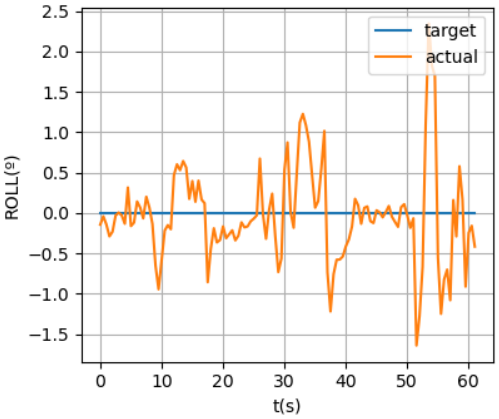


Figura 15: Resultados en roll de la prueba en una bahía de Santander.

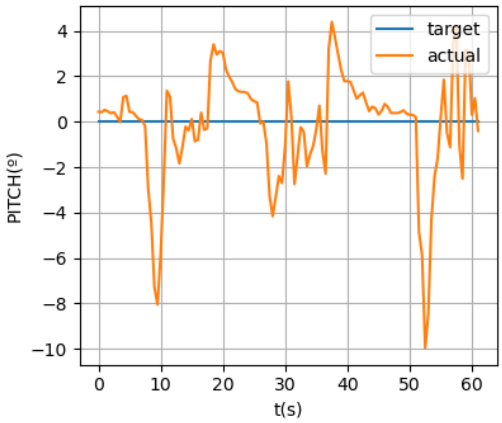


Figura 16: Resultados en pitch de la prueba en una bahía de Santander.

Además, en la Figura 17 se puede observar como el vehículo se posiciona en *yaw* siguiendo la referencia establecida por el *software*. En el Algoritmo 1 se ha

programado de 0º a 180º y de 0º a -180º por esta razón hay un salto en la Figura 17.

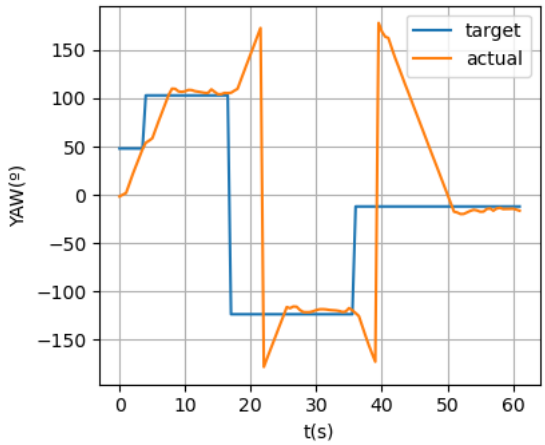


Figura 17: Resultados en yaw de la prueba en una bahía de Santander.

Cabe destacar que tal como puede observarse en la Figura 18 y Figura 19 no hay saturación apreciable en los actuadores y la respuesta de su actuación es aceptable ya que no hay oscilaciones o saturaciones que pudieran dañar los motores o reducir su vida útil a largo plazo.

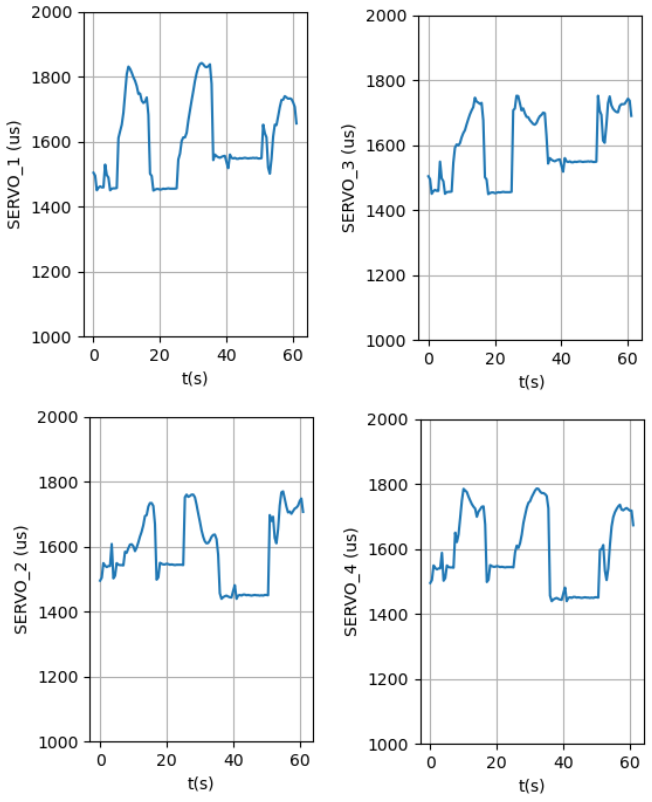


Figura 18: Resultados de los motores 1 a 4 en la prueba en una bahía de Santander.

Por último, no se observan desviaciones significativas entre los resultados obtenidos en simulación y los obtenidos en las pruebas en la bahía de Santander. Siendo los resultados comentados equiparables en simulación y en pruebas reales.

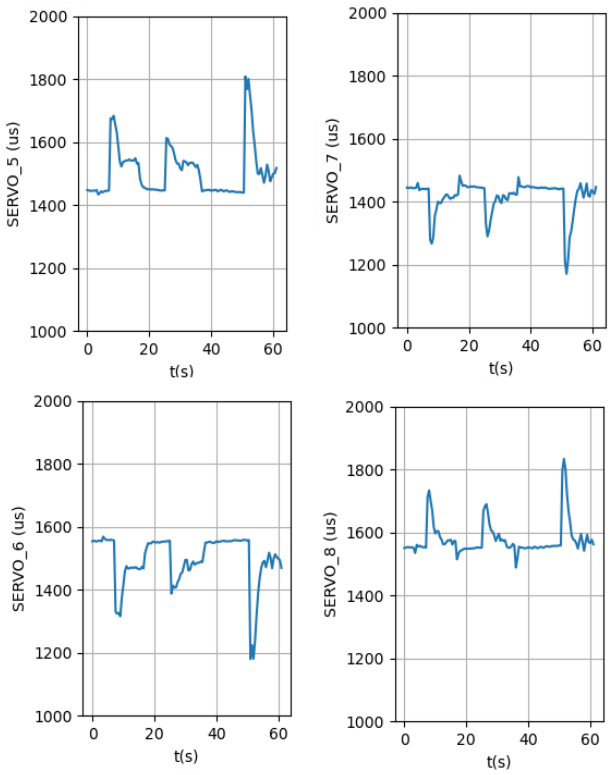


Figura 19: Resultados de los motores 5 a 8 en la prueba en una bahía de Santander.

5. Aplicación docente

En la actualidad, en la práctica totalidad de los vehículos acuáticos de cualquier clase, se encuentra como uno de los elementos fundamentales para su correcto funcionamiento el software relacionado con la gestión de los datos provenientes de los sensores y el *software* de control. Es de especial relevancia en estos vehículos el *software* relacionado con el procesado de datos de diferentes sensores, a partir del cual se puede determinar la posición. Actualmente, están tomando especial relevancia en la industria naval los avances en la automatización mediante *hardware* y *software* específico de ciertas operaciones de navegación y control. Sin embargo, para realizar una operación de forma autónoma por parte de un vehículo es necesaria la integración de diferentes programas de software. Por tanto, es imprescindible que el ingeniero involucrado en este campo tenga conocimientos tanto del modelo dinámico del vehículo como de lenguajes de programación, que le permitan desarrollar este tipo de herramientas de *software*.

Con el trabajo presentado en este documento, es posible adquirir conocimientos relacionados con el modelo dinámico del ROV. También es posible adquirir conocimientos en el ámbito del desarrollo de *software*, especialmente en el desarrollo de diferentes programas y su integración en una única interfaz gráfica que facilite el manejo de los mismos por parte del operario. Es necesario contar previamente con conocimientos matemáticos que permitan realizar operaciones con matrices y conocimiento de modelado matemático. Además, se requieren nociones básicas de programación

especialmente en Python y conocer someramente como manejar la interfaz gráfica del vehículo, en este caso Qgroundcontrol.

En este trabajo, se propone a modo de ejemplo la realización de dos prácticas docentes en las que los alumnos pueden entrar en contacto con el manejo de los ROVs y con el *software* desarrollado que se ha comentado en este documento, así como adquirir los conocimientos indicados en las citadas prácticas.

Se prevé la implantación de estas prácticas en futuros cursos de la asignatura Automatización en los planes de estudio de la Escuela Técnica superior de Náutica de Santander por los beneficios docentes indicados en este artículo.

4.1 Práctica 1: Introducción a la operación con ROVs.

Tiempo Estimado: 3 horas.

Objetivos de la práctica: El objetivo de esta práctica es que el alumno se familiarice con los diferentes elementos y sensores que componen el equipo necesario para realizar operaciones con el ROV siendo manejado por un operador en modo manual.

Además, se persigue que el alumno sea capaz de realizar maniobras sencillas con el vehículo para adquirir conocimiento empírico del entorno y del manejo del vehículo.

Conocimientos previos: El alumno debe conocer las especificaciones operativas de los diferentes elementos del sistema y de los sensores instalados en el vehículo. Por otra parte, es necesario conocer previamente a la realización de la práctica el esquema de conexiones de los elementos, así como las diferentes comprobaciones a realizar y advertencias de seguridad que se deben tener en cuenta para poder operar de forma segura.

Descripción de la práctica: En primer lugar, se ha de comprobar que todos los elementos que componen el sistema están en perfectas condiciones y que ninguno de ellos presenta daños. Posteriormente se debe guardar cada uno de los elementos en las maletas de transporte diseñadas para la protección de los equipos durante los desplazamientos. Una vez establecida el área de operación por parte del instructor, se debe encontrar una zona en la que realizar el despliegue de los equipos y en el que se conecten los diferentes elementos del sistema.

Tras esto y antes de comenzar a realizar ninguna operación, el alumno debe revisar y comprobar que los equipos no presentan daños debidos al transporte y tendrá que realizar las comprobaciones siguiendo una lista preestablecida. Una vez que todo este verificado siguiendo las indicaciones, se procederá al encendido del sistema. A continuación, el ROV será introducido en el mar de la bahía de Santander y darán comienzo las maniobras.

Las maniobras a realizar en esta práctica se centrarán en el manejo básico del ROV. Estas maniobras consistirán en el avance en línea recta, la inmersión a escasa profundidad, el manejo de la inclinación de la cámara y el giro alrededor del eje *z* (ángulo *yaw*). Simultáneamente a estas operaciones, el alumno realizara el seguimiento por pantalla de los diferentes valores de los sensores entre los que destacan el profundímetro, el sonar y el GPS.

4.2 Práctica 2: Planificación y realización de seguimiento de *way-points* de forma automática en la bahía de Santander.

Tiempo Estimado: 3 horas.

Objetivos de la práctica: El objetivo de esta práctica es la adquisición por parte del alumno de los conocimientos necesarios para crear planes de operación en el *software* específico y la puesta en marcha de los mismos en un entorno real de operación como es la bahía de Santander.

Conocimientos previos: Para la correcta realización de esta práctica se han de tener conocimientos de los diferentes elementos que componen el sistema y de su correcto despliegue y conexión. Se han de tener nociones básicas de manejo de ROVs para entender el comportamiento en el agua del mismo y poder establecer los *way-points* de los planes de operación de forma adecuada. Por último, será necesario conocimientos básicos sobre el lugar en el que se quieran desarrollar las operaciones. También conocimientos de controladores y de dinámica de los ROV.

Descripción de la práctica: En primer lugar, el alumno establecerá en el Qgroundcontrol los *way-points* del plan de operaciones a realizar por el ROV en la salida de la bahía de Santander.

A continuación, una vez que se encuentre en la zona de despliegue tendrán que conectar los diferentes elementos del sistema, realizar las comprobaciones y la puesta en marcha del sistema. Tras esto, se procederá a lanzar al agua el ROV y comenzar el plan de seguimiento de *way-points* automático previamente creado. Durante la realización del seguimiento de los *way-points* de forma automática el alumno supervisara que todo se realice según lo previsto y que los datos proporcionados por el Qgroundcontrol estén dentro de lo esperado. Por último, una vez realizado el plan se guardarán los datos de la telemetría para su posterior estudio si se estima conveniente, siendo este el último paso antes de la desconexión y almacenamiento de los equipos en sus respectivas maletas de transporte.

6. Conclusiones

En este trabajo se ha desarrollado un *software* para el seguimiento autónomo de trayectorias mediante *way-point* de un ROV. El *software* desarrollado presenta una interfaz gráfica que integra los diferentes programas que se van a emplear facilitando el manejo por parte del operario. El *software* es capaz de importar la trayectoria establecida en el QGroundControl y de posibilitar su seguimiento mediante los *way-points* fijados. Se han realizado simulaciones por computador y diversas pruebas en la bahía de Santander para verificar el correcto funcionamiento del *software*, especialmente en el seguimiento de la trayectoria en un entorno real con las perturbaciones inherentes al mismo. Por todo lo anteriormente comentado se puede concluir que, además de las ventajas operativas en el manejo de ROVs, este trabajo es especialmente adecuado para la formación de ingenieros y operadores de ROV que tengan interés en el desarrollo de herramientas de *software* destinadas al movimiento autónomo de estos vehículos y al procesamiento de datos de sus sensores.

Agradecimientos

Este proyecto ha sido parcialmente apoyado a través del proyecto TED2021-132158B-I00 Monitorización Evolutiva con Vehículos Submarinos No Tripulados para el Mantenimiento del Fondo y Anclajes de Parques Eólicos Marinos MICIU/AEI /10.13039/501100011033 y por la Unión Europea Next GenerationEU/PRTR Figura 20 y a través del proyecto Control de Vehículos Submarinos No Tripulados para la Supervisión de Estructuras para Obras Marítimas Fondeadas. Controladores Avanzados e Inteligentes y Supervisión 3D financiados por el Ministerio de Universidades, Igualdad, Cultura y Deporte del Gobierno de Cantabria.

Figura 20: MICIU/AEI /10.13039/501100011033 y por la Unión Europea Next GenerationEU/PRTR.

A Casco Antiguo Comercial S.L por su colaboración y apoyo en el desarrollo de este trabajo.

Referencias

Aguirre-Castro, O. A., Inzunza-González, E., García-Guerrero, E. E., Tlelo-Cuautle, E., López-Bonilla, O. R., Olguín-Tiznado, J. E., & Cárdenas-Valdez, J. R. (2019). Design and Construction of an ROV for Underwater Exploration. Sensors (Basel, Switzerland), 19, 5387.

Álvarez, C., Saltaren, R., Aracil, R., & García, C. (2009). Concepción, Desarrollo y Avances en el Control de Navegación de Robots Submarinos Paralelos: El robot Remo-I. Revista Iberoamericana de Automática e Informática Industrial, 6, 92-100.

Andújar, J. M., & Mateo, T. J. (2010). Diseño de Laboratorios Virtuales y/o Remotos. Un Caso Práctico. Revista Iberoamericana de Automática e Informática Industrial, 7, 64-72.

ArduSub. (2024). Recuperado el febrero de 2024, de Overview.: https://www.ardusub.com/

BlueRobotics. (2024). Recuperado el julio de 2024, de https://bluerobotics.com/store/rov/bluerov2/

Capocci, R., Dooly, G., Omerdić, E., Coleman, J., Newe, T., & Toal, D. (2017). Inspection-class Remotely Operated Vehicles-a Review. Journal of Marine Science and Engineering, 5, 13.

Chen, W., Wei, Q., & Zhang, Y. (2020). Research on anti-interference of ROV based on particle swarm optimization fuzzy PID. (pp. 342-347). IEEE.

Dong, J., & Duan, X. (2023). A Robust Control via a Fuzzy System with PID for the ROV. Sensors (Basel, Switzerland), 23, 821.

Fay, D., Stanton, N., & Roberts, A. P. (2019). Exploring Ecological Interface Design for Future ROV Capabilities in Maritime Command and Control., (pp. 264-73). Cham. Retrieved from http://dx.doi.org/10.1007/978-3-319-93885-1_{2}{4}

Fossen, T. I. (2002). Marine Control Systems: Guidance, Navigation and Control of Ships, Rigs and Underwater Vehicles. Marine Cybernetics.

Fossen, T. I. (2011). Handbook of Marine Craft Hydrodynamics and Motion Control. John Wiley and Sons, Ltd. Obtenido de https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119994138.fmatter

Gupta, M., & Maurya, P. (2021). Hardware in the Loop Simulator for a Coral Reef Monitoring Robot (C-Bot). (pp. 1-9). MTS.

Hosseinnajad, A., Mohajer, N., & Nahavandi, S. (2023). Novel barrier Lyapunov function-based backstepping fault tolerant control system for an ROV with thruster constraints. Ocean Engineering, 285, 115312.

Luo, G., Gao, S., Jiang, Z., Luo, C., Zhang, W., & Wang, H. (2024). ROV trajectory tracking control based on disturbance observer and combinatorial reaching law of sliding mode. Ocean Engineering, 304, 117744.

Martínez, A., Rodriguez, Y., Hernández, L., Guerra, C., Lemus, J., & Sahli, H. (2013). Diseño de AUV.Arquitectura de hardware y software. Revista Iberoamericana de Automática e Informática Industrial, 10, 333-343.

Martínez, J., Padilla, A., Rodríguez, E., Jiménez, A., & Orozco, H. (2017). Diseño de Herramientas Didácticas Enfocadas al Aprendizaje de Sistemas de Control Utilizando Instrumentación Virtual. Revista Iberoamericana de Automática e Informática Industrial, 14, 424-433.

MAVLink . (2024). Recuperado el febrero de 2024, de MAVLink: https://mavlink.io/en/

Mora, J. P., Samper, J., & Rodriguez, C. F. (2023). Estudio de la optimización bayesiana para reducir el consumo energético de un robot paralelo durante tareas pick and place. Revista Iberoamericana de Automática e Informática Industrial, 20, 1-12.

QGroundControl-user-guide. (2024). Recuperado el febrero de 2024, de QGroundControl: https://docs.qgroundcontrol.com/master/en/qgc-user-guide/index.html

Sainz Gutiérrez, J. J., Revestido Herrero, E., Llata García, J. R., & Velasco González, F. J. (2022). Aplicación de un Unscented Kalman filter para el filtrado de las señales en un vehículo subacuático teleoperado. XLIII Jornadas de Automática: libro de actas, (págs. 31-37). doi:https://doi.org/10.17979/spudc.9788497498418.0031

Schjlberg, I., & Utne, I. B. (2015). Towards autonomy in ROV operations. IFAC-PapersOnLine, 48, 183-188. doi:https://doi.org/10.1016/j.ifacol.2015.06.030

t200-thruster. (2024). Recuperado el febrero de 2024, de bluerobotics: https://bluerobotics.com/store/thrusters/t100-t200-thrusters/t200-thruster-r2-rp/

Velasco, F. J., Revestido, E., Moyano, E., & López, E. (2012). Remote Laboratory for Marine Vehicles Experimentation. Computer Applications in Engineering Education, 20, 728-740.

Velasco, F. J., Vega, L. M., Revestido Herrero, E., & Lastra, F. J. (2018). Vessel stability experiences with a remote laboratory for training based on IMO STCW training code. Computer Applications in Engineering Education, 26, 1435-1444.

von Benzon, M., Sørensen, F. F., Uth, E., Jouffroy, J., Liniger, J., & Pedersen, S. (2022). An Open-Source Benchmark Simulator: Control of a BlueROV2 Underwater Robot. Journal of Marine Science and Engineering, 10, 1898.

Apéndice A.

El vehículo objeto de este trabajo es un ROV denominado BlueROV2 (Figura 21) de la compañía BlueRobotics (BlueRobotics, 2024) cuyas dimensiones principales se observan en la Tabla 1.

Tabla 1: Medidas del ROV (cm)

Dimensión	cm
Largo (dirección de x)	45,71
Ancho (dirección y)	57,5
Alto (dirección z)	25,39

Figura 21. Zoom del ROV realizando pruebas en la bahía de Santander.

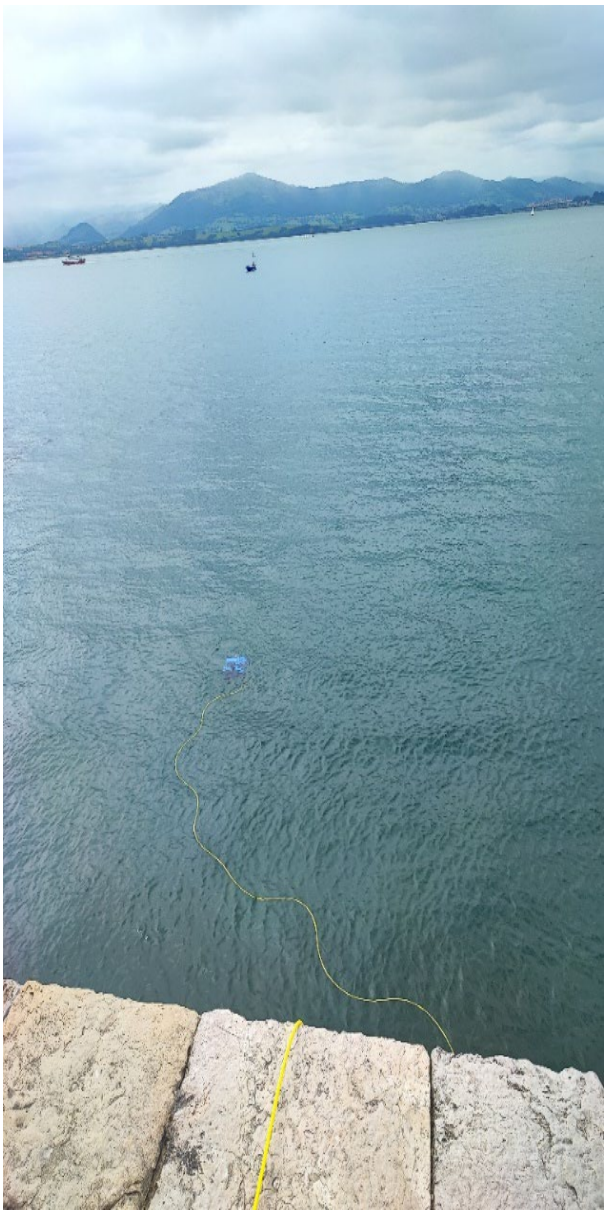


Figura 22. ROV realizando pruebas en la bahía de Santander.

La disposición de los propulsores se encuentra en la Figura 23.

Figura 23. Disposición de los propulsores en el ROV