Universidad Carlos III de Madrid

Escuela Politécnica Superior

Proyecto Fin de Carrera

Sistema de telemetría y control de un barco autónomo

Autor:
Fernando J. Pereda

Director:
Jose María Girón Sierra
Universidad Complutense de Madrid

Tutor:
Jesús García Herrero
Universidad Carlos III de Madrid

Julio de 2010

Índice general

Índice de figuras

Índice de tablas

Resumen

Los vehículos autónomos tienen más relevancia cada día. Son especialmente interesantes en misiones que presentan peligro para las personas. Tareas sistemáticas como el reconocimiento y escaneo de áreas son idóneas para este tipo de vehículos. Distintos tipos de vehículos autónomos se utilizan según las tareas a las que se destinen. Vehículos aéreos (UAV), vehículos submarinos, o vehículos marinos de superficie (AMSV).

Las aplicaciones para barcos autónomos son muchas, ejemplos de actualidad incluyen el desminado, la limpieza de vertidos, la exploración de catástrofes, la búsqueda de cajas negras, ...

Este proyecto desarrolla una versión funcional de un barco autónomo y el sistema de telemetría para monitorizarlo.

Capítulo 1

Introducción y motivación

Desde tiempos inmemoriales, la navegación ha sido una actividad importante en el desarrollo de las sociedades. Siglos de continuos esfuerzos han sido necesarios para pasar de simples balsas flotantes a las complejas embarcaciones actuales.

Quizá por ser la navegación un arte tan antiguo, la automatización y el control han sido introducidos principalmente en tiempos recientes y solo parcialmente. El control sigue recayendo en el ser humano y se pone aún más de manifiesto cuando se deben hacer maniobras marítimas.

Si se tiene en cuenta el control con realimentación sobre embarcaciones marítimas [Fossen02] enumera un gran número de ejemplos de sistemas disponibles comercialmente, como pilotos automáticos para barcos y vehículos submarinos para el control de giro y mantenimiento del rumbo. O incluso sistemas de control de torpedos.

Desde el punto de vista de los problemas de control, los entornos marítimos ofrecen una gran variedad de escenarios interesantes:

- Los problemas de *seakeeping* han concentrado algunos de los esfuerzos recientes como los de [Lloyd98]. Por ejemplo, para mejorar la estabilidad de los barcos y el rendimiento global mediante el uso de aletas, T-foils y de flaps [Esteban00, Esteban02, Giron-Sierra01, Giron-Sierra02, Haywood95, Ryle98].

- La realización de maniobras es, quizá, el campo más variado. Pueden referirse a una sola embarcación o a varias en un amplio rango de escenarios. Por ejemplo, las maniobras conjuntas entre dos o más embarcaciones para evitar que choquen, teniendo en cuenta ligaduras y los objetivos a cumplir, componen también un sistema muy complejo [Johansen03].

El despliegue de dispositivos como redes, conjuntos de boyas o barreras, los cuales pueden ser empleados para delimitar o confinar un área determinada de mar, por ejemplo, un vertido de petróleo, también constituyen otro interesante escenario. En él, diferentes embarcaciones deben cooperar para transportar y desplegar el dispositivo de un modo correcto.

La necesidad del control cooperativo en operaciones marítimas ha sido reconocida recientemente por varios autores e instituciones [Soetanto03, Stillwell00] son referencias ilustrativas. Una forma de tratar con estos problemas es mirarlos desde el lado de la robótica, donde la cooperación entre agentes dinámicos está atrayendo el interés investigador desde hace años.

Otro de los escenarios interesantes es el rastreo de minas. Las minas submarinas son, generalmente, difíciles de detectar y desactivar. Por ejemplo, el Mar Báltico alberga alrededor de cien mil de estas minas [Szarejko99]. Dentro del mundo del desminado marítimo existen distintos métodos para desactivar minas, desde métodos mecánicos hasta métodos más sofisticados basados en las huellas electromagnéticas de los barcos. Este último método se llama desminado por influencia [CMWA01]. El método consiste en arrastrar un *pez* submarino con un cable largo. Utilizando tanto el *pez* como el cable se imita la huella electromagnética de barcos mayores. Al percibir esta huella, algunas minas modernas pueden ser engañadas. Utilizando un barco autónomo se pueden aliviar riesgos para las personas.

El problema del desminado por influencia presenta un enlace entre los problemas de control marítimo y aquellos de rastreo de áreas por medio de robots [Choset01, Choset03, Jung09]. El problema puede ser abordado por vehículos marinos autónomos [1]. [Bertram08] tiene una recopilación extensa sobre este tipo de vehículos.

Los vehículos marinos autónomos presentan algunas ventajas. La primera y más importante es que la dinámica del sistema es estable, no es necesario un control de estabilidad. Los modelos físicos son relativamente simples al poder reducir el sistema a un cuerpo de 3 grados de libertad. Tampoco necesitan trabajar con muchos actuadores de forma que se puede hacer un controlador simple con relativa facilidad.

El proyecto toma la idea de un barco autónomo y se plantea dos grandes objetivos.

Construir un sistema de telemetría y control remoto para un barco Debe diseñarse y construirse el software tanto de a bordo como de tierra para poder monitorizar el barco. Además, debe poder controlarse el barco de manera remota utilizando un joystick. Será necesario diseñar e implementar el protocolo de comunicación así como el driver de un enlace de radio. El enlace de radio sirve para comunicar el barco con la estación de tierra.

Sistema de control autónomo El sistema debe permitir que el barco tome sus propias decisiones de cara a misiones y situaciones simples. El barco debe ser capaz de seguir un rumbo, mantener una cierta velocidad y ser capaz de ir a un punto dado. Todos estos comportamientos y sistemas deben poderse activar, desactivar, monitorizar y configurar desde el sistema de telemetría y control remoto (objetivo 1).

La memoria del proyecto describe el proceso de construcción de los sistemas de telemetría y control del barco. Desde el comienzo hasta los experimentos. Esta memoria se ha dividido en los siguientes capítulos:

Descripción del barco y del hardware En este capítulo se describe la plataforma experimental. Se da un pequeño repaso sobre el hardware y sobre las capacidades y características mecánicas del barco. Realmente es, después de los objetivos, la fuente de información más importante en cuanto al análisis de la solución. La plataforma experimental es la que es y el proyecto debe adaptarse a ella.

Diseño software El software del proyecto se define aquí. No se presenta el diseño con mucho nivel de detalle pues el software es muy complejo y no aporta mucho a la

[1] AMSV – Autonomous Marine Surface Vehicles

propia memoria del proyecto. El capítulo hace una pequeña introducción al sistema y protocolo de comunicaciones que es el enlace entre las dos secciones del mismo: el software de a bordo y el software de la estación de tierra.

Dinámica del sistema y control El barco que se va a controlar se rige por unas leyes físicas. En este capítulo se presenta un modelo simplificado del barco que sirve para diseñar y desarrollar los distintos controladores. Se hace una pequeña presentación del control en lazo cerrado y una introducción rápida a los controladores que se van a usar. Una vez hecho esto, el capítulo describe en detalle el funcionamiento de los controladores más básicos y luego los más complejos.

Experimentos y análisis de resultados Durante el desarrollo del proyecto, se han hecho muchos experimentos. Los experimentos siempre han tenido un objetivo claro y unas conclusiones. Este capítulo presenta los experimentos en estricto orden cronológico. Presenta los objetivos y los resultados y conclusiones de cada uno de ellos. Las descripciones de los experimentos vienen acompañadas de datos y gráficas reales obtenidos durante los mismos.

Conclusiones y trabajos futuros En este capítulo se da una vista rápida sobre qué ha supuesto el proyecto. Se presentan las conclusiones técnicas y personales sobre el mismo. En la última parte del capítulo se dan ideas sobre por dónde debe avanzar el trabajo utilizando la plataforma experimental creada.

Gestión de proyecto El proyecto ha estado sujeto a plazos y costes. Este capítulo presenta una planificación así como un presupuesto del proyecto.

El proceso de construcción ha seguido una metodología iterativa en base a prototipos. La memoria, sin embargo, presenta el conjunto del proyecto. No se describe el diseño e implementación paso a paso, se da el diseño final y se comenta como tal. Sin embargo, esto no es así en el capítulo *Experimentos y análisis de resultados*. Este capítulo está escrito de forma estrictamente temporal. Los resultados se van presentando a medida que se van conociendo. De esta forma, se intenta dar mucha importancia al proceso experimental y al aprovechamiento del tiempo de experimento para apoyar el desarrollo del proyecto.

Los objetivos del proyecto plantean diferentes problemas a resolver. Desde el punto de vista del sistema de la telemetría debe darse solución a los siguientes problemas. La solución adoptada para cada uno de ellos se detalla a lo largo de la memoria. Muchas veces la solución al problema ha seguido un ciclo de vida propio con distintas versiones de la misma. La memoria no recoge este proceso, simplemente presenta la solución final.

- Definición del protocolo de comunicaciones. Es necesario definir el protocolo de alto nivel. Cómo se comunicarán datos los distintos sistemas y los formatos de mensaje de bajo nivel. Detallado en la primera parte del capítulo 3.

- Formato de los datos en los ficheros de registro. El formato en el que se guarden los datos es muy importante ya que determina la capacidad de interactuar con otros sistemas y programas. Detallado a lo largo de todo el capítulo 3.

- Diseño del software multihilo de forma que sea capaz de procesar un gran número de datos. El diseño es crucial para conseguir un sistema de telemetría eficiente. El diseño se aborda en la sección 3.3.

En lo relativo al control autónomo del barco, también se ha tenido que dar solución a algunos problemas.

- Control en lazo cerrado. Ha sido necesario aprender el control en lazo cerrado, sus principios y su aplicación en un sistema real. Una introducción al control en lazo cerrado y a los controladores que se han utilizado puede encontrarse en la sección 4.3.1.
- Planteamiento de un modelo físico y desarrollo de un simulador que permita prototipar controladores y misiones avanzadas.
- Implementación de un controlador Proportional Integral Derivative (PID) y el ajuste del mismo. Parte del problema en el control autónomo del barco es ajustar los controladores una vez han sido implementados. Esto también incluye el filtrado de señales (ver sección 4.2). El proceso de ajuste puede verse en los distintos experimentos descritos en el capítulo 5.
- Interactuar (activar, desactivar y configurar) los distintos subsistemas del barco desde la estación de tierra. Esto es necesario para que los experimentos sean más útiles y se puedan sacar los datos necesarios en cada caso.

El proyecto se ha realizado en la facultad de Ciencias Físicas de la Universidad Complutense de Madrid (http://www.ucm.es) bajo la dirección del Prof. José María Girón Sierra. Profesor titular del Departamento de Arquitectura de Computadores y Automática (http://www.ucm.es/info/dacya.

Capítulo 2

Descripción del barco y del hardware

Para el desarrollo del proyecto se ha utilizado una plataforma experimental consistente en un barco de radio control tradicional y un hardware de navegación diseñado ex profeso para el mismo. El desarrollo de esta plataforma no es parte del presente proyecto. No obstante, se describe con cierto detalle debido a que el principal objetivo del proyecto es hacer un sistema de telemetría y el sistema de control para esta plataforma.

El barco mide unos 80 cm de largo y, completamente equipado pesa unos 3 Kg. La figura 2.1 muestra el barco sin montar.

Figura 2.1: Casco del barco de radio control.

El hardware consiste en una placa con un microcontrolador, distintos sensores y equipamiento para controlar el barco y obtener datos de los experimentos:

- Brújula para conocer la orientación del barco. Utiliza la interfaz Two Wire Interface (TWI).
- Sensores de presión y tubos pitot para medir la velocidad del barco con respecto

al agua donde se mueve. Estos sensores utilizan un Conversor Analógico Digital (ADC).

- Receptor y antena Global Positioning System (GPS) para posicionar globalmente al barco. Se utiliza una interfaz serie y el protocolo Trimble Standard Interface Protocol (TSIP) para comunicarse con el receptor.
- Una radio para comunicarse con la estación de tierra.
- Una ranura para tarjetas Tarjeta Secure Digital (SD) para guardar toda la información de los experimentos y poder ser analizada. El componente utiliza una interfaz Serial Peripheral Interface Bus (SPI).

La figura 2.2 muestra dos copias de la electrónica utilizada en estos barcos. Se pueden identificar los distintos componentes: la radio azul, el receptor GPS está conectado a la antena rectangular y el microcontrolador es la placa roja central.

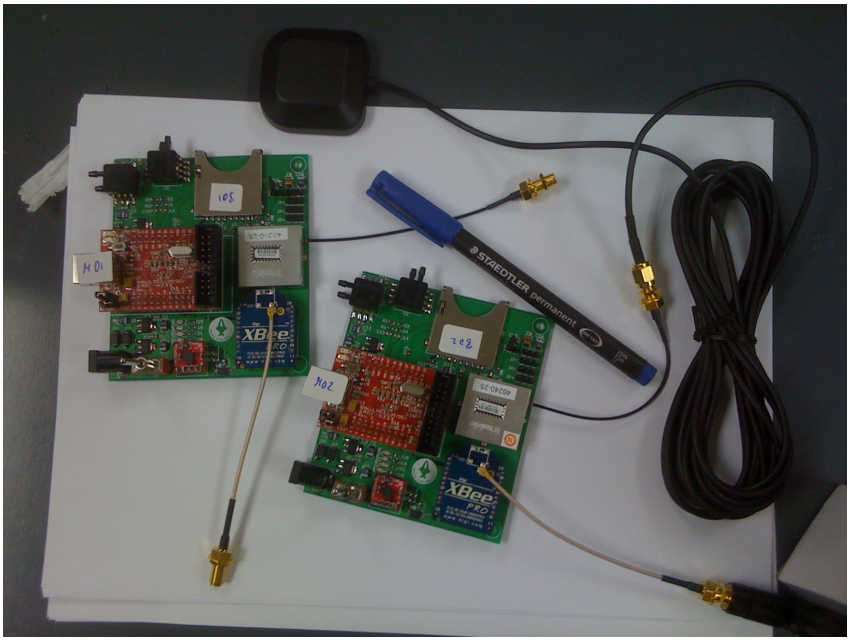


Figura 2.2: Dos copias de la electrónica.

La tabla 2.1 detalla los componentes utilizados por la electrónica del barco.

Componente	Fabricante	Modelo
Microprocesador	Atmel	AT91SAM7S256
Magnetómetro	Honeywell	HMC5843
GPS	Trimble	Lassen IQ DGPS
Presión	Freescale	MPXV5004
Radio	Digi	XBee v1.x

Cuadro 2.1: Componentes de la electrónica del barco.

Existe un enlace de radio que conecta el barco con la estación de tierra. El hardware de radio ocuparía los niveles enlace, red y transporte en un modelo de capas tipo TCP/IP.

Este hardware utiliza el protocolo IEEE 802.15.4 para las capas físicas y de control de acceso al medio. Para las capas superiores, utiliza un protocolo propietario, XBee. La capa de transporte garantiza una entrega fiable y confirmada de todos los paquetes de datos.

Como actuadores, el barco cuenta con un motor de continua de tipo fuera borda y un servo para moverlo. Para controlar tanto el motor como el servo se utilizan señales Modulación por Ancho de Pulsos (PWM). La figura 2.3 resume todos los componentes e interfaces utilizadas por el hardware de a bordo.

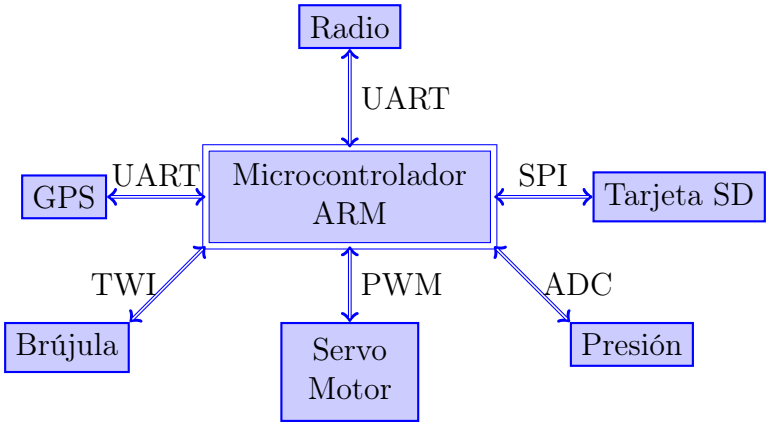


Figura 2.3: Resumen de componentes e interfaces de la electrónica de navegación.

Para el acceso a los distintos sensores y componentes de la electrónica, es necesario implementar los distintos protocolos de acceso e interfaz. En algunos casos, el fabricante (Atmel) ofrece librerías para algunos protocolos; en otros, es necesario implementarlo desde cero. Generalmente el microprocesador ofrece el acceso a las distintas líneas de datos y control mediante variables *especiales* en el lenguaje C. A continuación se describe el trabajo realizado para el acceso a los distintos componentes:

Brújula Se utiliza el protocolo TWI. El fabricante ofrece una librería que permite leer datos utilizando este protocolo. Leer los datos de la brújula pasa por leer seis bytes distintos (tres enteros de 16 bit) y unirlos dos a dos.

Sensores de presión Estos sensores son analógicos, de modo que es necesario utilizar un conversor analógico digital (conocidos como ADC). El microprocesador utilizado cuenta con conversores de este tipo y los ofrece como valores enteros de 16 bit (aunque el sensor solo ofrece 10 bit de precisión).

GPS El receptor se conecta por un puerto serie. El microprocesador ofrece dos formas de trabajar con este puerto, de forma síncrona o asíncrona (Direct Memory Access (DMA)). En este proyecto se ha utilizado DMA para comunicarse con el receptor GPS. La interfaz DMA ofrecida por el microprocesador es muy simple. Permite asignar dos *buffer* de datos, uno para lectura y uno para escritura. Ofrece también un contador para saber cuántos bytes quedan en el *buffer*. Además, para permitir crear secciones críticas y de acceso secuencial a los *buffer* de datos, se ofrecen dos señales para inhibir y activar el DMA del microprocesador. Además, el receptor

utilizado (fabricado por Trimble) utiliza un protocolo propio de comunicaciones sobre el puerto serie llamado TSIP que ha sido necesario implementar.

Radio Se utilizan las radios XBee de Digi. Estas radios permiten la comunicación utilizando un puerto serie y un protocolo de comunicación descrito en la documentación del fabricante. Del mismo modo que con el GPS, se ha utilizado DMA para gestionar la transmisión de datos a través de este puerto.

Tarjeta SD Para comunicarse con este tipo de tarjetas es necesario implementar un protocolo llamado SD. Para conectarse eléctricamente a la tarjeta, hay dos formas de hacerlo, utilizando el método SD[1] o utilizando SPI. El método SPI es estándar y muy extendido de modo que es el utilizado en este proyecto. En ambos métodos de conexión el protocolo de comunicación se llama igual, SD. No obstante, existen serias diferencias según el método de conexión eléctrica. [2]

Servo y motor Ambos se controlan con señales PWM. Estas se generan utilizando líneas de propósito general e implementando de forma nativa la modulación PWM.

Se utilizan tres baterías para alimentar los distintos sistemas del barco. Por un lado el hardware de navegación utiliza una batería, el motor utiliza otra y el variador y el servo una adicional. La razón de utilizar tantas baterías es que se pueda cambiar la batería del motor sin necesidad de tener que apagar la electrónica durante los experimentos. Esta configuración permite aislar los sistemas todo lo posible.

Se identifican dos grandes sistemas dentro del barco: el sistema de navegación y el de potencia. El sistema de navegación está compuesto por el hardware y su batería. El sistema de potencia comprende el servo, el motor (y su variador) y las baterías para alimentarlos. Interesa tener estos sistemas físicamente separados de modo que se montan en dos cajas estancas independientes. La figura 2.4 resume el despliegue de los sistemas dentro del barco.

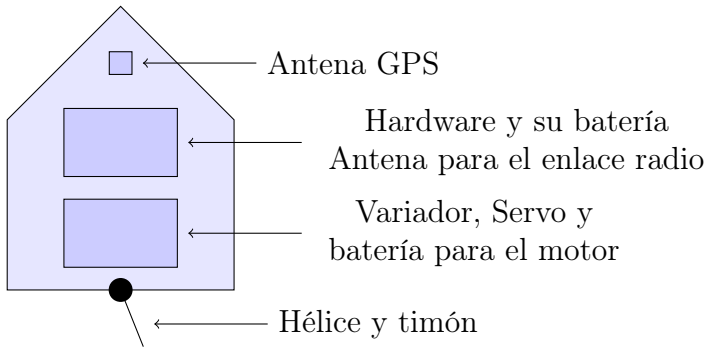


Figura 2.4: Despliegue de los sistemas del barco.

Para controlar el barco, es necesario utilizar el enlace de radio. En el lado de la estación de tierra es necesario utilizar un receptor (figura 2.5) así como un joystick para enviar las órdenes de control remoto (figura 2.6).

[1] Cuidado, SD significa muchas cosas: el nombre común de las tarjetas, el protocolo de acceso y el método eléctrico de conexión.

[2] Esto significa que, efectivamente, hay dos protocolos distintos de acceso a la tarjeta y ambos tienen el mismo nombre aunque no son iguales, solo parecidos.

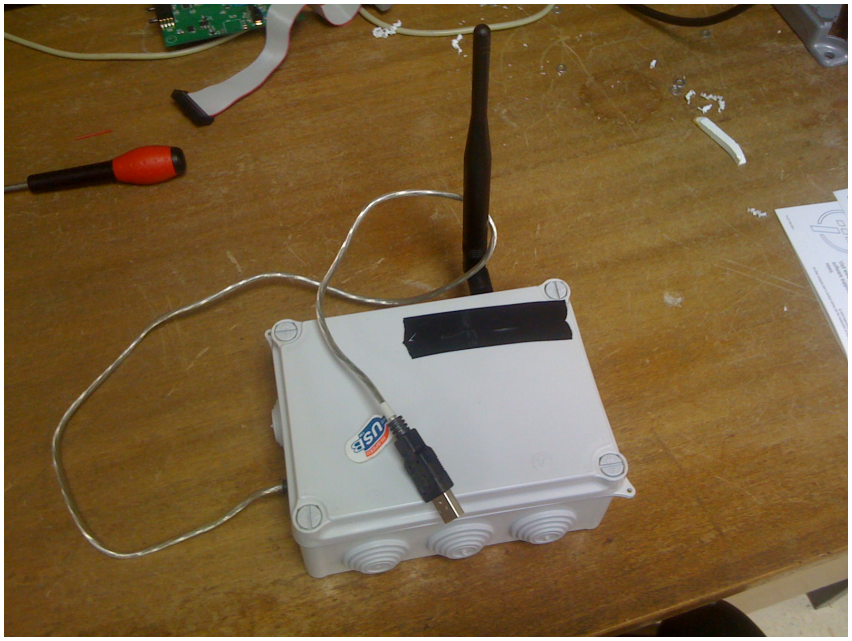


Figura 2.5: Enlace radio en la estación de tierra.

Figura 2.6: Joystick de la estación de tierra.

Capítulo 3

Diseño Software

El presente capítulo describe los componentes software del proyecto. Existen dos grandes sistemas claramente diferenciados: la estación de tierra y el sistema de a bordo. Ambos sistemas utilizan un enlace de radio.

Ambos sistemas se han diseñado siguiendo distintos objetivos. La estación de tierra ha sido diseñada para poder ser reutilizada en distintos vehículos autónomos. Sin embargo, el software de a bordo en el barco ha sido escrito y diseñado específicamente para ese hardware.

Los sistemas utilizan un enlace radio para comunicarse. El único componente compartido entre ambos sistemas es el driver del enlace radio. Se ha decidido utilizar exactamente el mismo driver para evitar discrepancias entre los dos sistemas.

El presente capítulo describe los tres componentes por separado. Empieza describiendo la comunicación entre los dos sistemas así como algunos detalles del driver de la radio. Luego describe los sistemas de a bordo y de tierra.

3.1. Comunicaciones

Existe una necesidad de intercambiar datos entre el barco y la estación de tierra. El barco genera muchos datos (de parte de los sensores, de los cálculos de control, ...) y la estación debe estar lista para recibirlos. Por otro lado, la estación genera órdenes esporádicas que el barco debe atender. Estas órdenes tienen que ver con la configuración o activación de distintos sistemas o parámetros.

El objetivo de este subsistema es permitir la comunicación de datos entre los distintos elementos de los experimentos o misiones. Debe proporcionar la posibilidad de comunicar datos entre los barcos y la estación de tierra o entre distintos barcos.

El barco debe enviar los datos de los sensores y su estado a la estación de tierra varias veces por segundo. Ambos sistemas van a trabajar con un gran volumen de datos, de modo que el diseño debe permitir una implementación eficiente.

Los datos que el barco envía a la estación de tierra se pueden dividir en paquetes. Cada uno de los paquetes agrupa conceptualmente los datos en el sistema.

- Brújula: 3 valores de 16 bit cada uno indicando la magnitud del campo magnético en cada uno de los tres ejes del barco.

- Sensores de presión: 2 valores de 16 bit indicando la velocidad longitudinal y lateral con respecto al fluido.
- Estado de los actuadores: 2 bytes con el valor del timón y del motor.
- Datos del GPS:
  - Posición: 3 float. Latitud y longitud en grados y altitud en metros.
  - Velocidad: 3 float. Valores de velocidad en las direcciones norte, este y arriba.
  - Hora: La hora GPS. Un float y un valor de 16 bit.
- Estado del barco. Información sobre qué subsistemas del barco están activos y los parámetros del control:
  - Estado de los subsistemas de control: rumbo, velocidad e "ir a un punto".
  - Valores de consigna de los subsistemas de control.

Estos paquetes se generan de forma asíncrona en el barco, esto es, los datos de nivel de aplicación que se transmiten en cada paquete de nivel de transporte son de longitud variable. De modo que el protocolo de nivel de aplicación debe poder identificar qué paquetes se están enviando en cada momento.

El hardware no puede transportar paquetes de más de 100 bytes. Debido a este límite de datos por transmisión, es importante que el formato de los datos de nivel de aplicación sea lo más denso posible, es decir, que no tenga apenas *overhead*. Por esto, y dado que la pila de protocolos garantiza la entrega fiable de los datos, se ha decidido no utilizar ningún código corrector de errores en los datos de aplicación.

El protocolo de nivel de transporte, así como la comunicación con el dispositivo de radio ha sido definido en el capítulo 2 de modo que esta sección solo se centra en los datos de aplicación.

Los datos consisten en una cabecera seguida de la carga útil de la transmisión. La cabecera consta de un byte indicando qué paquetes de datos se incluyen en la transmisión. El resto del paquete se destina a carga útil (los datos de los paquetes).

El mismo protocolo se utiliza en el otro sentido de la comunicación, exceptuando el hecho de que se usan dos bytes de cabecera para duplicar el número de paquetes que pueden identificarse, ya que se espera que la estación envíe más tipos de paquetes y más cortos. Los paquetes de datos que envía la estación de tierra al barco son:

- Conmutar el envío de información no vital a la estación de tierra.
- Conmutar el log de datos a la tarjeta SD.
- Actuación: valores de las señales del timón y el motor.
- Marcar un cambio de experimento en la tarjeta.
- Parámetros del control de rumbo:
  - Activarlo / desactivarlo

    - Parámetros del control (consigna y constantes del controlador).

- Parámetros del control de velocidad. Mismos parámetros que para el control de rumbo.

- Control 'ir a un punto': coordenadas del punto.

En ambos sentidos los datos se envían en el formato nativo de la plataforma ya que ambas plataformas (ARM7 y Linux/x86) los comparten. Los únicos tipos de datos que se envían son: enteros con y sin signo de distintas longitudes y números en coma flotante de simple precisión (*float*). [1] La razón más importante detrás de esta decisión es la eficiencia. Convertir los datos nativos a un sistema de representación intermedio para volverlos a convertir al *mismo* sistema nativo sería muy costoso. No obstante, esta decisión obliga a que la estación de tierra siempre sea ejecutada en una máquina que comparta los tipos nativos del sistema del barco.

3.2. Software de a bordo

El software de a bordo debe enviar los datos de los sensores así como el estado de los distintos subsistemas a la estación de tierra. Además, debe permitir la navegación automática del barco en base a distintos controladores. Esta última parte se cubre en el capítulo 4 por plantear problemas distintos del diseño software propiamente dicho.

El software debe funcionar en un ARM7TDMI (a partir de ahora, ARM7), ya que este es el microprocesador que tiene el hardware. Ha sido implementado en el lenguaje C99 utilizando el compilador GNU Compiler Collection (GCC). El código se ejecutará directamente en el microcontrolador, no existe un sistema operativo que gestione procesos; el software es responsable de gestionar el reloj y lanzar las tareas necesarias con la frecuencia deseada.

Además de GCC, para programar el microcontrolador se utiliza una interfaz JTAG. El software utilizado para interactuar con el programador es OpenOCD. Todo el software de a bordo se ha desarrollado utilizando herramientas abiertas (*OpenSource*) y gratuitas.

En el software a bordo del barco existen dos partes claramente diferenciadas:

- El código de bajo nivel: inicializa el reloj y los periféricos y ofrece una interfaz para interactuar con los distintos sensores y componentes del hardware.

- El código de alto nivel: trata los datos de los sensores, interactúa con la radio, graba datos en la tarjeta y ejecuta las acciones de control.

Esta sección se va a centrar en el código de alto nivel. Por razones de completitud se describe someramente la parte de bajo nivel.

Distintos drivers han tenido que ser desarrollados o incorporados al software de bajo nivel. Cada interfaz requiere de un protocolo para comunicarse con ella de modo que ha habido que desarrollar los siguientes drivers:

[1] Ambas plataformas usan una representación en complemento a dos para los números enteros y el estándar IEEE 754 de simple precisión para los *float*.

- Protocolos de interfaz: TWI, SPI.
- Modulación PWM para interactuar con los actuadores.
- Protocolo TSIP para interactuar con el receptor GPS.
- Driver para controlar la parte física de la tarjeta SD.
- Driver para interactuar con la radio.

Además se ha implementado una versión básica de FAT32 para grabar los datos en la tarjeta SD. Se utiliza FAT32 por lo simple que es de implementar y por lo fácil que es acceder a ese sistema de ficheros desde cualquier sistema operativo moderno.

El software de bajo nivel ofrece la forma de definir tareas que se ejecutarán con una cierta frecuencia. El software de alto nivel tiene las siguientes responsabilidades:

- Leer datos de los sensores: Con una frecuencia de 10Hz se leen los datos de la brújula y los sensores de presión. Constantemente se monitoriza el receptor de GPS aunque suele dar las medidas cada segundo. Cada segundo se informa del estado de los subsistemas a la estación de tierra.
- Monitorizar el enlace radio y obedecer órdenes remotas: Constantemente se monitoriza el enlace de radio y se ejecutan las ordenes remotas.
- Control: Las acciones de control se ejecutan cada 10 Hz.
- Enviar datos de telemetría por la radio: Se envían los datos que hayan cambiado desde el último envío 5 veces por segundo.
- Guardar datos de telemetría: 10 veces por segundo se graban los datos que hayan cambiado. Cada 2 segundos se escribe una firma especial de sincronización en la tarjeta.

Desde el punto de vista de la telemetría, el software de a bordo no tiene mucho que hacer. Simplemente se encarga de leer los datos de los sensores y enviarlos por la radio cada cierto tiempo. Adicionalmente, guarda estos datos en la tarjeta.

Desde el punto de vista del sistema de control, el software de a bordo no es tan simple. Por esta razón se dedica un capítulo entero al diseño de esta parte del sistema. El capítulo 4 se centra en la descripción de la dinámica del barco y del diseño de los algoritmos de control.

Los datos de telemetría se guardan en un fichero en una tarjeta SD que lleva el barco durante experimentos y misiones. Dado que potencialmente se pueden generar muchos datos, el formato de los mismos debe ser muy denso. Es por esto que se ha utilizado una modificación del formato descrito en el comienzo de este mismo capítulo.

Para poder estudiar cómodamente los datos guardados en la tarjeta, es necesario poder indicar un *cambio de experimento*. Así como en enlace de radio proporciona una comunicación fiable y confirmada, la tarjeta no. Es posible que un fallo de hardware corrompa los datos de la tarjeta. Dada la naturaleza del formato, un fallo en un bit de los mapas que describen qué datos hay en ese *paquete* echaría a perder todos los datos a partir de ese bit erróneo.

Para solucionar el segundo problema (el de la fragilidad del formato), se ha añadido una *firma* de sincronización. Esta *firma* consiste en un valor de 64 bits cuyo primer byte está prohibido como mapa de bits. En caso de existir un error en un mapa de bits, solo se perderán los datos hasta la siguiente *firma* de sincronización. El software escribe una de estas *firmas* cada dos segundos. El software de extracción de datos, al encontrar una de estas *firmas*, simplemente la ignora.

El primer problema (el del cambio de experimento) es fácil solucionarlo añadiendo una *firma* de cambio de experimento. Esta *firma* consiste en un valor de 64 bits cuyo primer byte está prohibido como mapa de bits. El software de extracción de datos, al encontrar una de estas *firmas*, separa los datos en un fichero nuevo.

3.3. Estación de tierra

El software de la estación de tierra debe permitir mostrar los datos recibidos por la radio de los distintos barcos. Además, debe permitir enviar distintas órdenes al barco para interactuar con sus subsistemas.

El software debe funcionar en un ordenador relativamente moderno. Es importante que funcione tanto en Linux como en Mac OS X ya que son los sistemas operativos utilizados por los usuarios de la estación.

Se ha elegido el lenguaje de programación C++ por contar con numerosas librerías y estar ampliamente extendido en la comunidad científica. Para la programación de las interfaces de usuario se utilizará la librería Gtkmm. Para los gráficos 2D Cairo.

El código utiliza características y clases introducidas en Technical Report 1 (TR1) de modo que la versión de GCC que se use debe soportar este estándar [2].

Al igual que en el software de a bordo, todas las herramientas utilizadas en el desarrollo del software de la estación de tierra son abiertas y gratuitas.

La estación de tierra ha sido diseñada para funcionar de forma paralela. Un hilo se encarga de monitorizar la radio, otro de leer los valores del joystick y otro gestiona la interfaz gráfica.

La estación de tierra debe cumplir las siguientes funcionalidades básicas:

1) Mostrar los datos de telemetría de forma textual (valores numéricos) y algunos de forma gráfica.

2) Grabar los datos de telemetría en ficheros.

3) Permitir activar, desactivar y configurar los sistemas del barco.

4) Controlar manualmente el barco mediante un joystick.

5) Permitir seguir y controlar varios barcos a la vez.

El sistema se concibe como un gran *Productor-Consumidor* donde los productores y los consumidores corren todos en hilos distintos. Los productores de información o datos serían el gestor de la radio y el joystick. Los consumidores serían los distintos elementos para mostrar los datos de telemetría (por ejemplo un gráfico en tiempo real).

[2]Cualquier versión a partir de la 4.1.0 debería servir.

Se ha utilizado el patrón de diseño *Observer* [GoF94]. Los productores de datos (a partir de ahora *observables*) ofrecen una función para iniciar su ejecución. Los consumidores (a partir de ahora *observers*) se registran en los *observables* y reciben las actualizaciones correspondientes. Los *observers* deben garantizar su correcta ejecución en un entorno paralelo, por ello se ofrecen primitivas como *mutex* y variables de condición.

El patrón de diseño *Observer* permite que un objeto mantenga una lista de objetos interesados en él. En el caso de ocurrir un evento en el objeto observado, este notifica a todos los objetos interesados. Es idóneo para el caso de la estación de tierra y representación de datos ya que los visualizadores de datos se pueden suscribir a los productores de información (generalmente el driver de la radio). De este modo se desacoplan los productores de los consumidores y sistema mantiene una alta cohesión y un bajo acoplamiento.

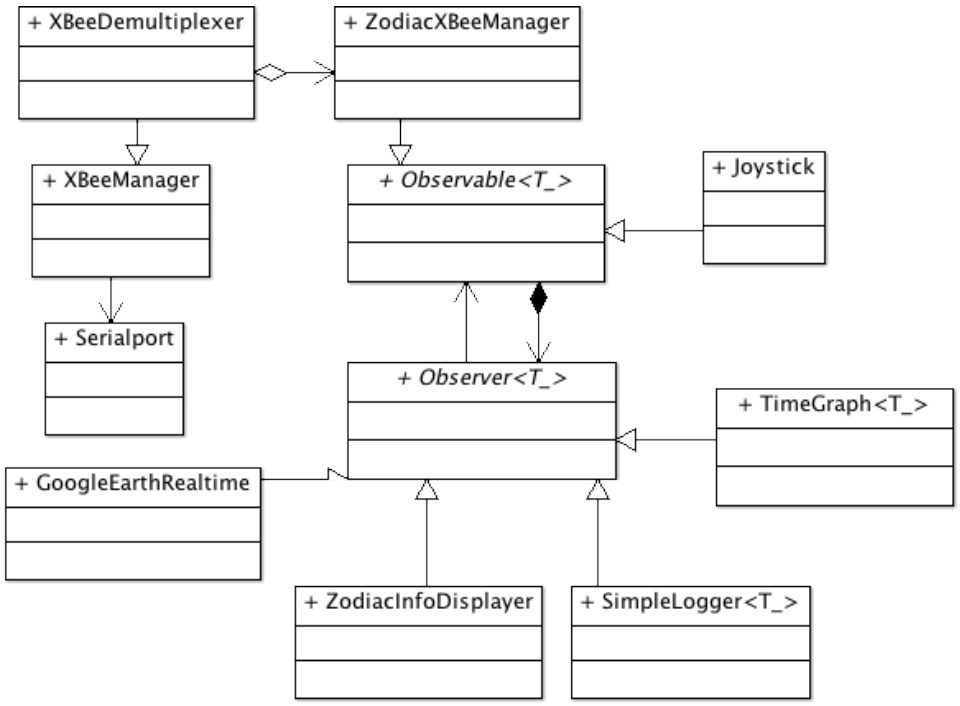


Figura 3.1: Diagrama de clases de la estación de tierra.

La figura 3.1 muestra una simplificación del diseño de la estación de tierra. Por comodidad se han omitido las clases relacionadas con la interfaz gráfica ya que no aportan mucho valor al diseño y, sin embargo, lo hacen más difícil de entender. Las clases del diagrama se describen a continuación:

Observable y Observer Proporcionan las clases que implementan el patrón *Observer*. Los productores de información pueden notificar a los consumidores que hay actualizaciones en los datos.

SerialPort Clase envoltura que permite manejar un puerto serie.

XBeeManager Clase que permite interactuar con la radio de la estación de tierra. Proporciona una interfaz de alto nivel al driver de la radio.

XBeeManagerDemultiplexer Esta clase permite generar distintos *Observables* para poder distinguir entre diferentes vehículos. Por ejemplo para combinar dos barcos en un experimento.

ZodiacXBeeManager Clase que permite decodificar los datos que envía el barco.

Joystick Productor de información que lee los datos de un joystick. Un consumidor puede conectarse a este productor para enviar órdenes remotas al barco.

SimpleLogger Consumidor de información que guarda los datos que recibe en ficheros de texto. Ver la sección 3.3.1 para más información.

TimeGraph Consumidor de información que muestra una gráfica respecto al tiempo. Ver sección 3.3.2 para más información.

GoogleEarthRealtime Clase que permite visualizar el camino seguido por un vehículo en tiempo real utilizando Google Earth.

ZodiacInfoDisplayer Esta clase muestra el estado de los distintos subsistemas del barco así como los valores de consigna de los sistemas de control.

3.3.1. *Logs* de datos

Una de las capacidades principales (ver página 17) de la estación de tierra es poder guardar los datos recibidos en ficheros para posterior inspección y estudio. En la estación de tierra, todos los datos de telemetría se encapsulan en clases puras de datos. La solución idiomática de este problema en C++ es hacer que las clases implementen:

```
std::ostream & operator<< (std::ostream & o, const ZodiacStateData & data)
```

Así, la plantilla de clase `SimpleLogger` permite crear un *observer* de datos que grabe los datos que recibe en un fichero de cualquier clase que implemente esa función.

El formato de los ficheros de registro de datos es el siguiente:

```
Hora Dato_1 Dato_2 Dato_3 .... Dato_N
```

La *Hora* se representa como dos números, el primero la fecha en formato Unix [3], y el segundo el microsegundo dentro de la fecha Unix. Se hace esto para evitar ambigüedad de datos en el mismo segundo.

La parte de *Datos* tiene los datos numéricos en texto plano. Grabar los datos en texto plano tiene un inconveniente, y es que es más lento que escribir los datos directamente en binario. Sin embargo, tiene la ventaja de que los datos los puede leer un humano directamente y, es muy fácil utilizar otros paquetes software para trabajar con esos datos. Por ejemplo, es trivial importarlos en una hoja de cálculos o en Matlab. Por ejemplo, el listado 1 muestra cómo cargar un fichero hipotético *prueba.log* y obtener una gráfica de uno de los valores utilizando Matlab.

Poder cargar fácilmente los datos desde Matlab es muy importante para poder diseñar filtros y examinar las señales de telemetría del barco cómodamente. Ha sido uno de los aspectos clave de la estación de tierra; poder prototipar filtros y algoritmos en Matlab y probarlos con datos reales para validarlos.

[3] Hora Unix: Segundos desde Epoch, el 1 de Enero de 1970.

Fragmento 1 Carga de un fichero de registro de datos en Matlab.

```
load prueba.log
tiempo_prueba = prueba(:,1) + prueba(:,2) / 1000000;
datos_prueba = prueba(:,3);
plot(tiempo_prueba, datos_prueba);
```

3.3.2. Visualización de datos en tiempo real

Una de las razones de tener la estación de tierra es poder controlar remotamente al barco. Es decir, utilizando un joystick Universal Serial Bus (USB) tradicional, utilizar el barco como si fuera un barco de radio control normal. Esto es útil para posicionar el barco antes de los experimentos y para traerlo de vuelta una vez finalizan. Otra de las razones de existir la estación de tierra, como se ha visto en la sección anterior (sección 3.3.1), es poder guardar los datos de telemetría que se generan en el barco y hacerlo de forma que se puedan utilizar estos datos con el mayor número de aplicaciones posibles.

La estación de tierra también debe permitir ver los datos de telemetría en tiempo real. Existen distintas formas de visualizar los datos dependiendo de la naturaleza de estos. Por ejemplo, los datos GPS se pueden visualizar de forma textual o utilizando un mapa. La figura 3.2 muestra una forma de representar estos datos de forma textual.

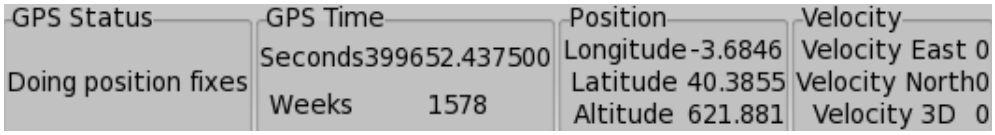


Figura 3.2: La estación de tierra mostrando datos GPS usando `GPSTextVisualizer`.

En temas de control, es muy común representar la señal que se quiere controlar frente a la consigna de control. De modo que también deben representarse gráficas en tiempo real de algunos de los valores de telemetría. Esto puede ser interesante para evaluar, en el campo de experimentos, los parámetros de los controladores y reajustarlos sobre la marcha en caso de que fuera necesario. La figura 3.3 muestra esta capacidad con unas señales sintéticas.

Cuando se trabaja con vehículos, a veces es útil visualizar las trayectorias utilizando Google Earth (o software similar). La forma más simple de visualizar los datos del receptor GPS en Google Earth es hacerlo de forma *offline*. Una vez se vuelve al laboratorio, se utilizan los ficheros de registro para generar un fichero Keyhole Markup Language (KML) [4]. La otra opción es generar un fichero KML periódicamente a medida que se reciben datos del GPS durante el experimento o misión. Utilizando la característica *Network Link* de Google Earth se puede hacer que el software recargue el fichero cada cierto tiempo. Esto tiene el efecto de ver, en tiempo real, qué está haciendo el barco.

[4]La especificación del lenguaje KML puede encontrarse en http://www.opengeospatial.org/standards/kml/.

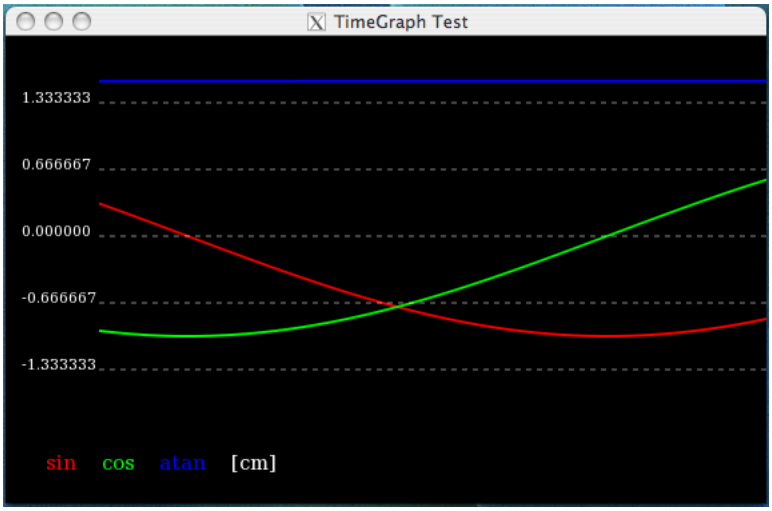


Figura 3.3: TimeGraph mostrando la gráfica en tiempo real de tres funciones trigonométricas.

3.3.3. Sistema de telemetría y control remoto

El diseño presentado en este capítulo comprende el núcleo de la estación de tierra. El diseño permite construir distintos sistemas de telemetría y control dependiendo de las necesidades de cada experimento. De hecho, el mismo sistema de telemetría se está utilizando para monitorizar aviones autónomos además de barcos. El código central es el mismo, lo único que cambia es la forma de utilizar los distintos componentes.

Para monitorizar un barco, durante los primeros experimentos, se utilizó la siguiente combinación de componentes del núcleo del sistema de telemetría:

- Para visualizar las señales de los distintos sensores, se utilizan 4 instancias de `TimeGraph` para visualizar:
  - La señal en bruto de los 3 ejes de la brújula.
  - La señal en bruto del sensor de presión.
  - El rumbo del barco frente a la consigna de control de rumbo.
  - La velocidad local longitudinal frente a la consigna de control de velocidad.
- Una instancia de `GPSTextVisualizer` para monitorizar el estado del receptor GPS.
- Una instancia de `ZodiacInfoDisplayer` para monitorizar el estado de los subsistemas de a bordo.
- Varias instancias de `SimpleLogger<T>` para guardar un registro de los datos que se reciben.
- Una instancia de `GoogleEarthRealtime` para monitorizar, sobre un mapa, la posición del barco.

Además de monitorizar el barco, la estación de tierra debe poder controlarlo, ya sea utilizando un joystick o para activar, desactivar o configurar subsistemas. Para esto es

necesario hacer una interfaz que permita interactuar con estos subsistemas. De modo que una instancia de la clase `Joystick` se utiliza para hacer el control remoto. La figura 3.4 muestra una captura del controlador remoto. El color de la ventana del controlador identifica el barco que se está controlando actualmente para evitar confusiones durante los experimentos.

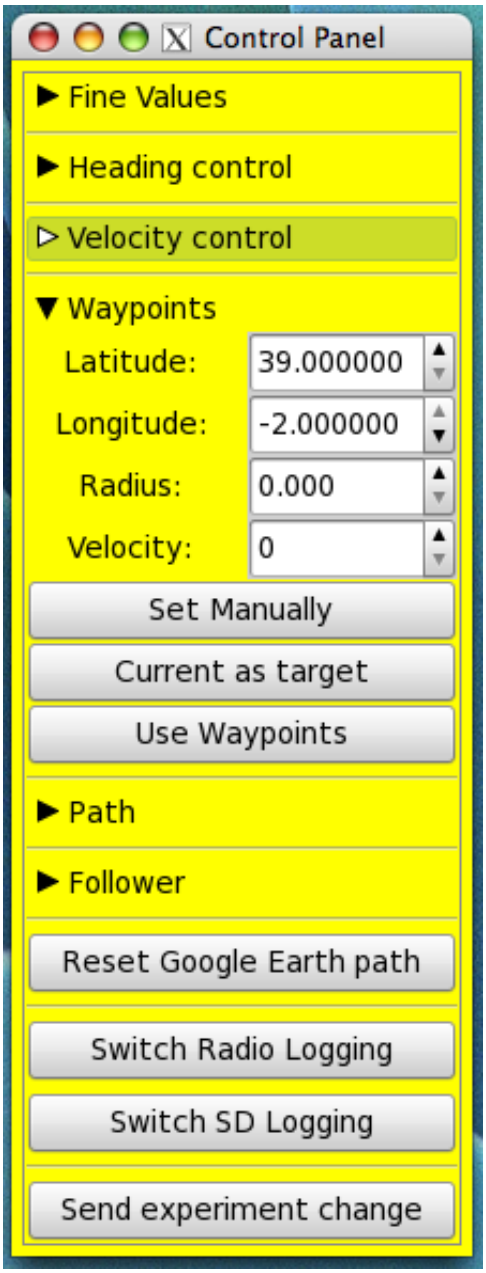


Figura 3.4: Controlador remoto de un barco.

3.3.4. Módulo de análisis de experimentos

Otra de las cosas interesantes de la estación de tierra es la posibilidad de analizar los experimentos una vez han pasado. Generalmente, es más que suficiente utilizar progra-

mas como Matlab o Google Earth. Sin embargo, a veces es interesante reproducir en el laboratorio el experimento en tiempo real.

Para esto debemos permitir la construcción de las clases de datos del sistema a partir de los ficheros de registro generados durante el experimento. Dado que estos ficheros se generan en texto plano, es fácil leer los datos. Para esto, todas las clases de datos implementan una función estática que permite que se construyan desde un `std::istream` [5]. Por ejemplo, la clase `ZodiacStateData` implementa:

```
ZodiacStateData ZodiacStateData::unserialize(std::istream & i);
```

Dada la flexibilidad del diseño del núcleo del software de la estación de tierra, es posible construir un *observable* que lea los ficheros de log y genere notificaciones en tiempo real de lo que va leyendo. Utilizando los *templates* de C++ la clase se puede implementar de forma genérica. El listado 2 muestra la implementación del método principal de esta clase.

Como se puede ver, la función utiliza los distintos métodos `unserialize` para construir objetos de datos en el sistema y emite las notificaciones respetando los tiempos en los que llegaron los distintos datos.

Dado que la clase `DataFromLog<T>` es un productor de información, se le pueden conectar los mismos visualizadores (gráficas, visualizadores textuales, Google Earth, ...) que a la estación de tierra durante un experimento. Como cada entrada en los ficheros de registro tiene una marca de tiempo, la clase `DataFromLog<T>` puede utilizar la diferencia de tiempos entre una entrada y la siguiente para generar los datos en *pseudo tiempo real*. En la práctica, es como volver a vivir el experimento en la estación de tierra.

[5] `std::istream` es el tipo de datos que usa C++ para referirse a flujos de datos de entrada.

Fragmento 2 Clase `DataFromLog<T>` para reproducir experimentos en tiempo real.

```
void run()
{
    unsigned long osec, ousec;
    _i >> osec;
    _i >> ousec;

    for (;;)
    {
        {
            Lock l(_m);
            if (_finish)
                break;
        }

        Observable<T_>::notify(T_::unserialize(_i));

        unsigned long sec, usec;
        _i >> sec;
        _i >> usec;

        if (_i.eof())
            break;

        unsigned long s((sec - osec) * 1000000 + (usec - ousec));
        osec = sec;
        ousec = usec;
        usleep(s);
    }
}
```

Capítulo 4

Dinámica del sistema y control

Este capítulo está dedicado al segundo objetivo del proyecto, al diseño e implementación del control automático del barco.

Para poder diseñar e implementar los algoritmos de control, es necesario saber qué se va a controlar. Es por esto que la primera sección del capítulo se dedica a la dinámica del barco. Se presenta un modelo físico simplificado.

Antes de sumergirse en el mundo del control, es necesario pararse a observar las señales que proporcionan los sensores. Es importante trabajar con señales sin ruido. Cuanto más limpias son las señales, más fiable será el control. De modo que una de las secciones trata sobre el filtrado de las señales.

Por último, el capítulo da una introducción al control en lazo cerrado y a los controladores PID. Tras introducir las bases del control y el controlador que se va a usar, se describen los controladores utilizados en el proyecto.

4.1. Dinámica del barco

Esta sección presenta un modelo físico del barco. El modelo permite saber qué se va a controlar y permite tener una idea de cómo se debe controlar el sistema.

A lo largo del presente capítulo se utiliza la siguiente convención:

$$\dot{x} = \frac{dx}{dt}$$

La figura 4.1 muestra un esbozo del modelo físico del barco. Dado que el barco debe moverse en el mundo, es interesante ajustar los ejes del modelo a los ejes del mundo para facilitar el cálculo de rumbos. El modelo se plantea en *ejes mundo* por esta razón. El ángulo θ es el rumbo que lleva el barco con respecto al norte. F es la fuerza que ejerce el motor, se asume que esta fuerza tiene la misma dirección del barco (θ). El ángulo η representa la posición del timón con respecto a la posición neutral. Si $u = \dot{x}$, $v = \dot{y}$ y $\omega = \dot{\theta}$, las ecuaciones (4.1), (4.2) y (4.3) constituyen el modelo físico del barco.

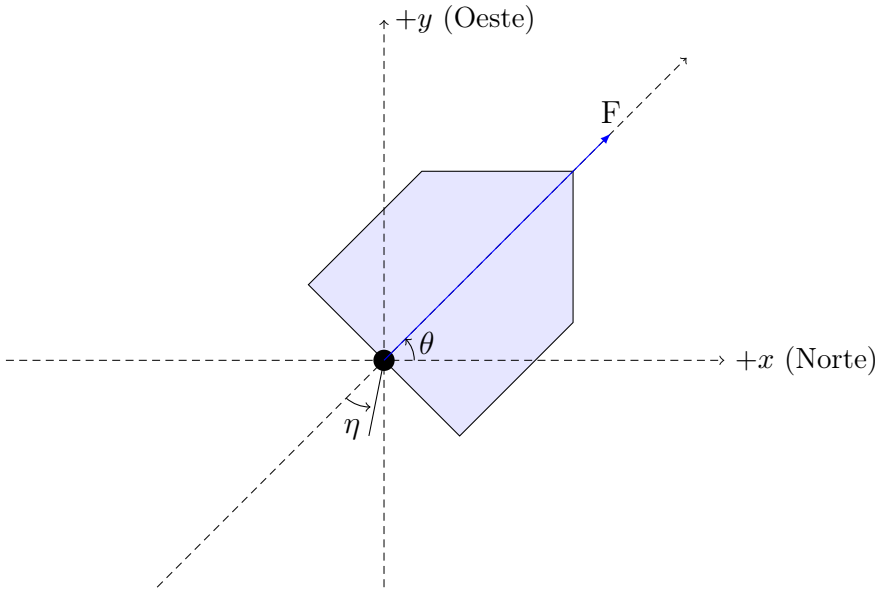


Figura 4.1: Modelo del barco.

$$\dot{u} = (F \cdot \cos\eta \cdot \cos\theta - \mu_t \cdot u) \cdot \frac{1}{m} \tag{4.1}$$

$$\dot{v} = (F \cdot \cos\eta \cdot \sin\theta - \mu_t \cdot v) \cdot \frac{1}{m} \tag{4.2}$$

$$\dot{\omega} = (F \cdot d_{cm} \cdot \sin\eta - \mu_r \cdot \omega) \cdot \frac{1}{I} \tag{4.3}$$

μ_t y μ_r son los coeficientes lineales de resistencia tanto al avance como al giro respectivamente. m es la masa del barco, I su momento de inercia y d_{cm} la distancia del motor al centro de masas del barco.

Teniendo un modelo del barco, es posible realizar prototipos iniciales del control. Una vez se refinan los prototipos, se prueba el control en experimentos reales con el barco.

Para el control automático del barco se utilizan controladores PID. Se ha dedicado una sección completa al diseño de estos controladores. Dicha sección (4.3) contiene una pequeña introducción al control en lazo cerrado y a los controladores PID.

4.2. Filtrado de señales

> *In theory, there is no difference between theory and practice. But, in practice, there is.* – Jan L. A. van de Snepscheut / Yogi Berra

A la hora de trabajar con hardware y elementos reales hay que tener cuidado con los ruidos. Generalmente los sensores son *ruidosos*. Por tanto, la concepción del estado del sistema que tiene el controlador es *ruidosa*. Por ejemplo, la figura 4.2 muestra el rumbo del barco registrado por la brújula durante uno de los primeros experimentos.

Cuando se utilizan controladores PID es importante trabajar con señales *limpias*. El ruido puede generar problemas en la parte D de los controladores; ya que el efecto derivativo de las mismas lo amplifica.

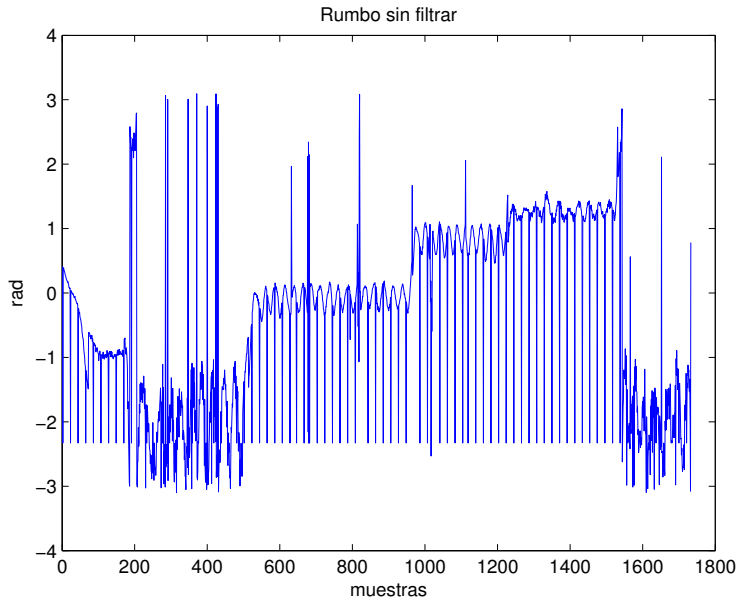


Figura 4.2: Datos reales del rumbo del barco sin filtro.

Lo primero que debe hacerse con las señales es quitar los valores atípicos que se van claramente del rango razonable de valores del sensor. Una vez hecho esto, es posible seguir filtrando la señal por métodos más sofisticados.

El ruido de los sensores suele encontrarse en componentes de alta frecuencia. No obstante, no deben usarse filtros paso bajos ideales ya que esto reduciría la efectividad del controlador D. Sin embargo, se puede utilizar un filtro de mediana (no-lineal) que mejora la eficiencia del controlador D [Li06].

Un filtro de mediana es relativamente fácil de implementar. Para el caso del rumbo se ha decidido utilizar un filtro de 6 posiciones que apenas introducirá retardo. El resultado del filtrado puede verse en la figura 4.3.

4.3. Control

Una vez descritos los sistemas de telemetría del barco, estando probados y funcionando, se ha cumplido el primero de los dos objetivos más importantes del proyecto. El segundo objetivo trata de hacer el barco autónomo. Se quiere poder dar órdenes de alto nivel al barco como "ve a este punto" o "sigue a este otro barco" y que las cumpla.

Existen muchas formas de hacer este control autónomo. Se ha decidido utilizar controladores PID para este proyecto. Estos controladores son muy fáciles de implementar y entender así como de depurar. Son los mejores controladores cuando no existe ningún conocimiento sobre el sistema que se va a controlar [Bennet93, p. 48]. Aunque no es el caso, ya que este mismo capítulo ha presentado un modelo físico del barco, su simplicidad los hace propicios para el proyecto.

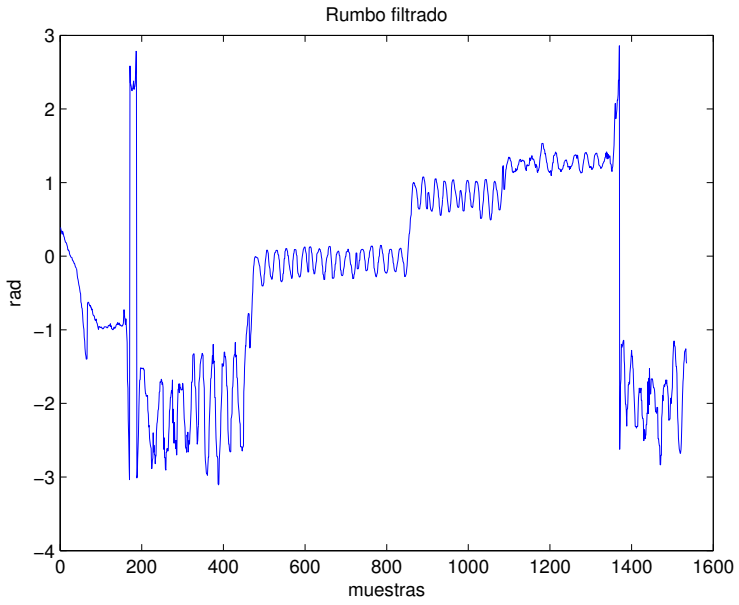


Figura 4.3: Datos del rumbo una vez filtrados.

4.3.1. Control de lazo cerrado y controladores PID

El control de sistemas es una rama interdisciplinar de ingeniería, matemáticas y física. Controlar un sistema es conseguir que de una respuesta deseada. Esta respuesta deseada se llama *consigna* o valor de referencia. Se supone que el sistema o planta tiene una cierta capacidad de actuación. Las acciones sobre los actuadores se denominan *señales de control*.

Si el sistema fuera ideal y el modelo perfectamente conocido, con el simple hecho de hacer las actuaciones que dictara el modelo, la planta estaría controlada. Esto es lo que se conoce como *control en lazo abierto*. Sin embargo, la realidad es muy distinta. Los modelos nunca son lo suficientemente detallados ni precisos y el mundo real está lleno de ruidos y situaciones inesperadas. La solución a esto es monitorizar periódicamente la planta y modificar las actuaciones en función del estado de la misma.

Para monitorizar la planta se utilizan los sensores descritos en el capítulo 2. Estos sensores no son perfectos, de ahí que en la sección 4.2 se diseñara e introdujera un filtro. La señal de estos sensores es la concepción que tiene el sistema sobre su propio estado.

Se dice que el control es *en lazo cerrado* cuando se utilizan sensores para realimentar al controlador con el estado del sistema. La figura 4.4 muestra el esquema tradicional de un controlador en lazo cerrado. En la figura ε es el error del sistema, la diferencia entre el valor de consigna y el estado del sistema. u es la señal de control o actuación del controlador sobre la planta.

Los controladores PID se basan en la combinación de tres actuaciones:

P proporcional. Relativa al error *actual*.

I integral. Relativa a los errores acumulados o *pasados*.

D derivativo. Proporciona una predicción lineal del error *futuro*.

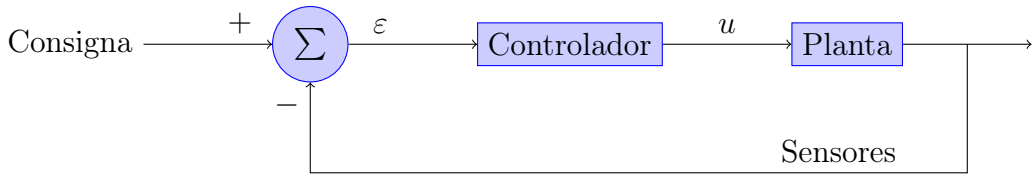


Figura 4.4: Controlador en lazo cerrado.

La función del controlador PID es:

$$u(t) = K_p \cdot \varepsilon(t) + K_i \cdot \int_0^t \varepsilon(\tau) \cdot d\tau + K_d \cdot \frac{d}{dt}\varepsilon(t)$$

Donde se ven las tres distintas actuaciones perfectamente diferenciadas. Este algoritmo tiene solo tres parámetros: K_p, K_i y K_d. Estos parámetros se denominan *ganancias* y son la única configuración del controlador. Dependiendo del tipo de controlador que se requiera y del sistema o planta que se va a controlar, es posible que alguna de las ganancias K_i o K_d sea 0. En este caso se denominan controladores PD y PI respectivamente.

La elección de los valores de las ganancias determina el comportamiento del controlador. Valores incorrectos pueden dar resultado a inestabilidad en el sistema. Para ajustar los parámetros es interesante conocer cómo funcionan estos controladores. Variaciones en las ganancias dan resultados más o menos predecible sobre el comportamiento del controlador:

Ganancia proporcional, K_p Cuanto mayor es el valor, más agresivo es el controlador. Menos tiempo tardará en alcanzarse el valor de referencia. Sin embargo, un valor demasiado alto puede llevar el sistema a la inestabilidad o a grandes oscilaciones.

Ganancia integral, K_i A mayor ganancia integral, antes se eliminarán los errores en el estado estacionario. Sin embargo, un valor muy alto tiene como consecuencia una gran sobreoscilación inicial (conocida como *overshoot*). Existen sistemas donde esta oscilación inicial es inaceptable.

Ganancia derivativa, K_d Grandes valores disminuyen el *overshoot* pero hacen al controlador menos agresivo y, por tanto, más lento. La ganancia derivativa es muy sensible a ruidos en la señal que se quiere controlar. Estos ruidos y errores pueden llevar al sistema a la inestabilidad.

Existen distintos métodos para ajustar un controlador PID. En el ámbito de este proyecto el ajuste será manual y online. Es decir, se observará el comportamiento del sistema con los distintos parámetros y se utilizará conocimiento experto para ajustarlos.

A veces, el controlador querrá hacer acciones demasiado agresivas, incluso, fuera de los límites físicos del actuador. En estos casos, es necesario limitar la actuación del controlador. Es decir *saturar* la salida del controlador.

En otras ocasiones, como se verá más adelante en el diseño del control de velocidad, interesa que la salida del controlador sea suave en lugar de brusca. Una técnica típica es forzar que la salida del controlador sea una *rampa*.

Cada controlador es distinto y estas modificaciones no son estándar para todos los controladores. Cada caso requerirá unas modificaciones distintas y particulares. Estas

modificaciones surgirán debido a la experiencia tanto en la fase de diseño como durante los primeros experimentos.

4.3.2. Control de rumbo

Una de las operaciones más comunes en un barco es mantener un cierto rumbo. El control de rumbo hace exactamente eso. Dado un rumbo de consigna el controlador debe mantener dicho rumbo. Se ha decidido utilizar un controlador PD ($K_i = 0$) sobre la señal de rumbo. El actuador que se debe utilizar para modificar el rumbo del barco es el servo del timón.

Utilizando la brújula (es decir, un magnetómetro de 3 ejes) se puede obtener una medida de θ que llamaremos $\hat{\theta}$ [1]. De la brújula obtenemos el valor del campo magnético en cada uno de los tres ejes del barco: mx, my y mz. De modo que:

$$\hat{\theta} = \arctan\left({}^{my}/_{mx}\right)$$

Los lenguajes de programación proporcionan una función, `atan2(y,x)`, que tiene en cuenta los signos de y y de x para dar el ángulo en el cuadrante correcto. De modo que se utilizará esta función en la implementación del control. El diagrama del controlador se resume en la figura 4.5.

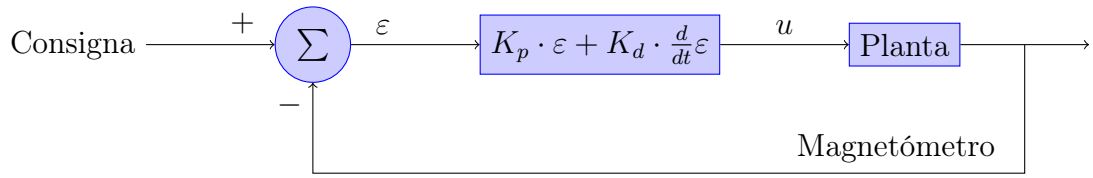


Figura 4.5: Controlador en lazo cerrado para el control de rumbo.

Es importante hacerse una idea del orden de magnitud de las ganancias del controlador. Para ello observamos los rangos de ε y los rangos de entrada del servo. El rango de ε es simple, pues tanto $\hat{\theta}$ como los valores de consigna se miden en radianes:

$$\hat{\theta} \in [-\pi, \pi] \implies \varepsilon \in [-2 \cdot \pi, 2 \cdot \pi]$$

Los valores de entrada del servo u_{servo} deben ser consultados en la documentación del hardware. En este caso el servo es controlado por una señal PWM que se genera a partir de los valores de 1 byte:

$$u_{servo} \in [0, 255]$$

Sin embargo, para evitar problemas durante lo experimentos reales, se va a limitar la actuación del controlador. Se introduce una saturación de u_{servo} para que solo tome valores entre 80 y 150.

Las señales con las que trabaja el controlador ya se han presentado en una sección anterior; de todos modos, a modo de referencia, la figura 4.3 muestra una señal real tras ser filtrada.

[1] Es decir, $\hat{\theta} = \theta \pm \varepsilon$.

En sucesivas secciones se definen controladores complementarios al control de rumbo así como otros controladores que se basan en el correcto funcionamiento de este.

4.3.3. Control de velocidad

El control de velocidad tiene como objetivo mantener una cierta velocidad con respecto al fluido. El barco tiene dos sensores capaces de medir la velocidad del barco. Por un lado cuenta con el GPS que nos da la velocidad en $^{m}/_{s}$ en las direcciones Norte, Este y Arriba. Sin embargo, a bajas velocidades el GPS no es muy preciso. Para medir la velocidad de avance del barco se ha instalado un sensor de presión y un tubo de pitot. La presión medida en el tubo está relacionada con la velocidad relativa del barco con respecto al fluido en el que se mueve:

$$P = \frac{1}{2} \cdot \rho \cdot v^2 \implies v = \sqrt{\frac{2 \cdot P}{\rho}}$$

De forma similar a como se hizo en la sección 4.3.2 se define la señal que se decide controlar. Se va a controlar directamente $\hat{P}$.

El actuador que permite cambiar la velocidad del barco es el motor. Para controlar la velocidad del motor se utiliza el variador. El variador ofrece 255 señales distintas de velocidad.

Existe una problemática adicional asociada al control de velocidad cuando se trabaja con la señal de error. Cuando el motor alcanza la velocidad deseada $\varepsilon = 0$ de modo que $u(t) = 0$. Al enviar 0 al motor, este se para con la consiguiente acción del controlador para contrarrestarlo. Al estar constantemente apagando y encendiendo el motor, es muy fácil agotar la batería muy pronto y no conseguir un control decente.

Para solucionar este problema es necesario diseñar el controlador de tal forma que $\varepsilon \to 0$. Si llamamos F a la señal que le enviamos al variador del motor, entonces lo que el controlador modificará será la variación de esa señal, es decir ΔF. La figura 4.6 muestra este controlador.

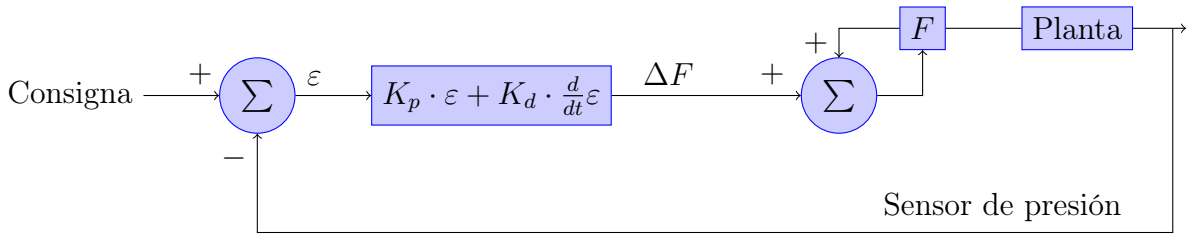
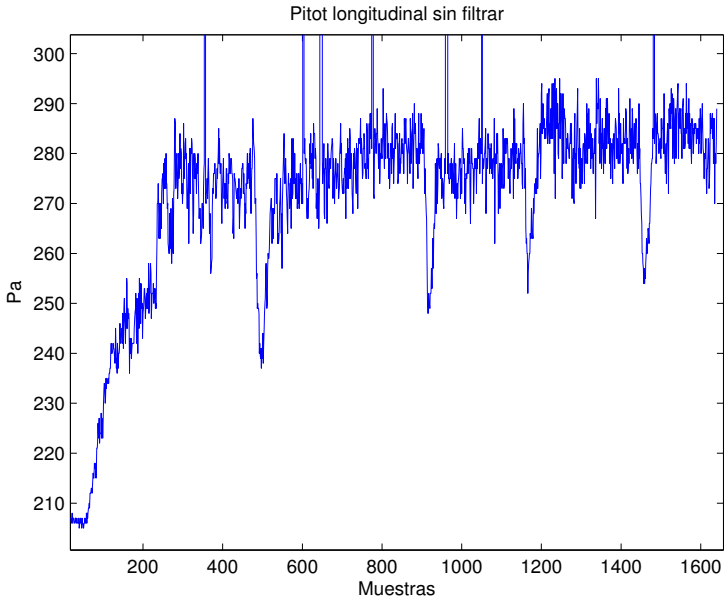


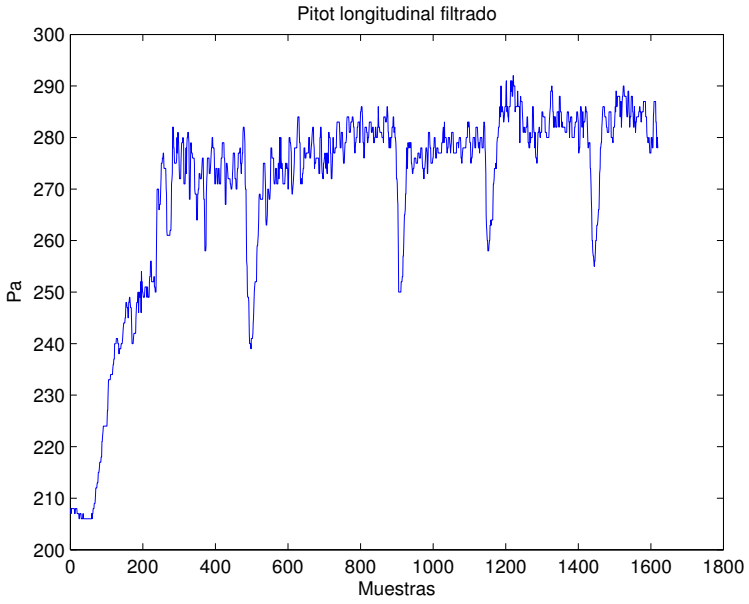
Figura 4.6: Controlador en lazo cerrado para el control de velocidad.

Es importante saber la forma que tienen las señales con las que va a trabajar el controlador. Las figuras 4.7a y 4.7b muestran la misma señal antes y después de aplicar el filtro.

Una de las modificaciones que se ha incluido en este controlador es la aceleración progresiva del barco. Si el controlador, tal cual está planteado, recibe una entrada escalón. Por ejemplo, si $t_{x-1} = 0$ y $t_x = 300$ entonces el controlador dará una salida muy grande. Esta acción se correspondería con un acelerón muy fuerte al motor que, además, no dará



(a) Sin filtrar



(b) Filtrada

Figura 4.7: Señales reales del sensor de presión.

el resultado esperado. Para evitar esto se introduce una saturación en ΔF de forma que la señal de motor (F) crezca solo linealmente y no lo haga de golpe. La figura 4.8 muestra este efecto.

Figura 4.8: Aceleración lineal frente a una entrada escalón.

4.3.4. Control *Ir a un punto*

El control *ir a un punto* consiste en hacer que el barco vaya a un punto dado que, a partir de ahora, llamaremos *waypoint*. La mejor forma que tiene el barco de posicionarse en el mundo es utilizando el receptor GPS. Este componente da al barco su posición utilizando un sistema de referencia geodésico. En concreto, el sistema de referencia es el WGS84. La posición del barco en el mundo se puede definir por su latitud (ϕ), su longitud (λ) y su altitud (h).

Dada una posición arbitraria del barco $B = (\phi_{Barco}, \lambda_{Barco}, h_{Barco})$ y la del *waypoint* $W = (\phi_w, \lambda_w, h_w)$. La tarea del controlador es encontrar un rumbo δ que permita al barco llegar a W.

Para calcular δ es necesario convertir tanto B como W a un sistema de referencia en el que se puedan calcular ángulos fácilmente. Una vez calculado δ simplemente habrá que utilizar el control de rumbo (ver sección 4.3.2) para conseguir que el barco llegue a W.

En navegación, la forma tradicional de hacer esto es convertir ambas posiciones a un sistema de referencia local y plano. Los más utilizados son los sistemas East-North-Up (ENU) y North-East-Down (NED). Mientras que en la navegación aérea lo normal es usar NED, en navegación marítima se suele utilizar ENU. La conversión entre estos sistemas es trivial, de modo que tampoco habría ningún problema en usar uno u otro.

Para poder convertir las posiciones del sistema de referencia WGS84 al sistema ENU, es necesario convertirlas antes a un sistema de referencia que toma la tierra como una esfera llamado Earth Centered Earth Fixed (ECEF). La figura 4.9 muestra la relación entre todos estos sistemas.

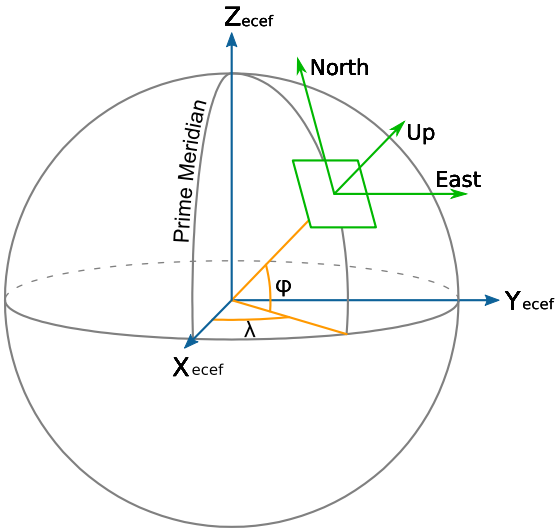


Figura 4.9: Sistemas de referencia.

Por razones de completitud, las ecuaciones de conversión entre los distintos sistemas se incluyen aquí. Las ecuaciones (4.4), (4.5) y (4.6) hacen la primera parte del trabajo. Dada una posición en WGS84 (ϕ, λ, h) se obtiene la misma posición en el sistema de referencia ECEF (X, Y, Z).

$$X = \left(\frac{a}{\chi} + h\right) \cdot \cos\phi \cdot \cos\lambda \tag{4.4}$$

$$Y = \left(\frac{a}{\chi} + h\right) \cdot \cos\phi \cdot \sin\lambda \tag{4.5}$$

$$Z = \left(\frac{a \cdot (1 - e^2)}{\chi} + h\right) \cdot \sin\phi \tag{4.6}$$

Donde

$$\chi = \sqrt{1 - e^2 \cdot \sin^2\phi}$$

Los parámetros a y e están definidos en el estándar WGS84. $a = 6378137.0m$ es el semi-eje mayor mientras que $e^2 = 6.69437999014 \cdot 10^{-3}$ es el cuadrado de la primera excentricidad.

El sistema de referencia ENU tiene una peculiaridad, y es que uno decide donde pone el origen del sistema. En este caso, haremos que el origen sea la posición del barco y el otro punto, el *waypoint*. Al calcular las coordenadas del plano ENU estamos, efectivamente, calculando la descomposición en coordenadas mundo del vector que une el barco con el *waypoint*. Usando la ecuación (4.7) se obtienen las componentes (E, N, U) entre los puntos $[(X_p, Y_p, Z_p), (\phi_p, \lambda_p, h_p)]$ y (X_r, Y_r, Z_r). [2]

[2]Es importante ver que las ecuaciones necesitan la posición en el sistema de referencia geodésico del punto origen además de su posición en el sistema ECEF.

$$\begin{bmatrix} E \\ N \\ U \end{bmatrix} = \begin{bmatrix} -\sin\lambda_r & \cos\lambda_r & 0 \\ -\sin\phi_r \cdot \cos\lambda_r & -\sin\phi_r \cdot \sin\lambda_r & \cos\phi_r \\ \cos\phi_r \cdot \cos\lambda_r & \cos\phi_r \cdot \sin\lambda_r & \sin\phi_r \end{bmatrix} \cdot \begin{bmatrix} X_p - X_r \\ Y_p - Y_r \\ Z_p - Z_r \end{bmatrix} \tag{4.7}$$

Ahora es necesario encontrar el rumbo que hay que seguir para ir de B a W. La figura 4.10 ilustra la situación.

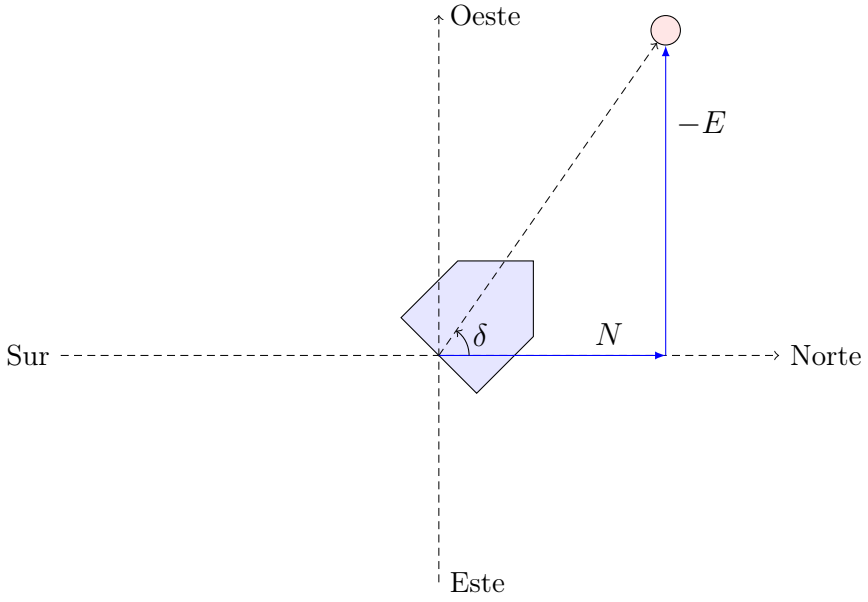


Figura 4.10: Cálculo del rumbo en el sistema de coordenadas ENU.

Se asume que la componente U es nula ya que tanto el *waypoint* como el barco están a la misma altura. El ángulo δ es el rumbo que debe seguir el barco:

$$\delta = \arctan\left(\frac{-E}{N}\right)$$

Dado que todos los sistemas involucrados en este controlador tienen un cierto error, es importante ver que el barco nunca llegará con exactitud al punto. Es necesario establecer algún criterio para decidir si se ha llegado al punto. En este caso se establece la condición de que el barco se acerque lo suficiente al *waypoint*; que esté a menos de R_w metros. Se establece como criterio de llegada:

$$\sqrt{E^2 + N^2} < R_w$$

Resumiendo, las consignas del controlador vienen dadas por dos puntos en formato WGS84 mientras que la salida es la derrota que debe mantener el barco. Este rumbo debe evaluarse periódicamente ya que el barco nunca se mueve en una línea recta perfecta. Evaluando el rumbo periódicamente tiene el efecto de compensar la deriva del barco por efecto de corrients

4.4. Cooperación – *seguir a otro*

Utilizando varias copias del barco, es posible plantear escenarios de cooperación de robots. La cooperación de robots es un campo muy amplio en el que hay muchos problemas abiertos. Este proyecto plantea un escenario simple de cooperación en el que un barco seguirá a otro. El escenario planteado es el típico en un convoy, un barco hace de líder y otro barco le sigue.

Una de las grandes cuestiones del mundo de la cooperación de robots es la comunicación entre los distintos agentes. Existen distintos métodos y protocolos estándar para comunicar distintos agentes. En este caso, se ha decidido utilizar la infraestructura de comunicaciones existente así como los protocolos que rigen la comunicación entre el barco y la estación de tierra. Implementar otros sistemas habría distraído el objetivo del proyecto que es el desarrollo completo de los sistemas de control y telemetría de un barco autónomo.

El escenario de cooperación puede realizarse utilizando los controladores desarrollados en la sección 4.3. En concreto, una vez que el control *ir a un punto* funciona, es posible desarrollar el seguidor. Dado que los barcos, según el protocolo de comunicaciones descrito en el capítulo 3, pueden comunicarse información. Se hace que cada segundo se intercambien las siguientes variables de estado:

- Posición GPS. Latitud, longitud y altitud.
- Rumbo actual. En radianes con respecto al norte.
- Velocidad local longitudinal [3].

Con esta información y con el control descrito en la sección 4.3.4 es posible hacer un pequeño controlador que haga que un barco B_1 siga a otro B_2. La forma de hacer esto es fijando en B_1 la posición de B_2 como consigna del control *ir a un punto*. Si esta consigna se actualiza periódicamente, por ejemplo, cada vez que se recibe la información de B_2, el barco le seguirá.

Habiendo planteado el escenario de cooperación así como la solución en base a los controladores presentados en la sección 4.3, el sistema puede clasificarse utilizando una taxonomía para sistemas multirobot cooperativos [Farinelli04].

1) Dimensiones de coordinación:

   a) Nivel de cooperación.

      En esta dimensión es necesario evaluar la habilidad que tiene el sistema para cooperar, dada una tarea específica. Se distinguen sistemas cooperativos y no cooperativos.

      El sistema se considera cooperativo dado que los distintos barcos se intercambian información para resolver su misión.

   b) Nivel de conocimiento.

[3] Es decir, velocidad con respecto al agua

Este nivel tiene como objetivo diferenciar entre sistemas enterados y no enterados. Llamaremos sistema enterado a aquel en el que los elementos poseen conocimiento sobre el resto.

El sistema se considera no enterado ya que no tiene ningún conocimiento a priori de qué está siguiendo. Simplemente sigue unos *waypoints*.

c) Nivel de coordinación.

Según la confianza que se le de a la comunicación (también llamada protocolo de coordinación) con el resto de robots, los sistemas serán fuertemente o débilmente coordinados.

Nuestro sistema da confianza total a la comunicación, de modo que se considera fuertemente coordinado.

d) Nivel de organización.

Durante las misiones, el escenario de cooperación puede tener un líder fijo, conmutar la responsabilidad de líder entre los distintos robots, o no tener ningún líder. Según este criterio, los sistemas serán fuertemente centralizados, débilmente centralizados o distribuidos, respectivamente.

El sistema es considerado fuertemente centralizado debido a que el rol de líder lo ejerce el barco o robot que está siendo seguido mientras que el otro se limita a seguirle. En un momento dado, el sistema permitiría hacer que el barco que originalmente hacía de líder siga al otro. Sin embargo, esto se consideraría una nueva misión.

2) Dimensiones del sistema:

a) Comunicación.

Se definen dos tipos de comunicaciones, directa o indirecta, según si se usa un dispositivo dedicado a la comunicación o si se utilizan otros métodos.

El sistema tiene un enlace de radio de modo que se considera de comunicación directa.

b) Composición del equipo.

Este nivel trata de la homogeneidad del conjunto de robots.

El sistema se plantea de manera homogénea. Sin embargo, nada impediría montar el hardware en otro vehículo.

c) Arquitectura del sistema.

Las arquitecturas se dividen en deliberativas, que permiten a los miembros del equipo el arreglarse con los cambios del entorno proveyendo una estrategia para reorganizar los comportamientos del equipo, y en reactivas, donde cada robot se arregla con los cambios en el entorno siguiendo una aproximación individual para reorganizar sus tareas para alcanzar una determinada meta.

El sistema el reactivo debido a que el barco seguidor hace su propio control y toma sus decisiones sin necesidad de confirmación por parte del barco seguido.

d) Tamaño del equipo.

El sistema está compuesto por dos barcos. Sin embargo, el sistema es fácilmente extensible a más barcos encadenando las tareas de líder y seguidor; B_1 sería

el líder y no seguiría a nadie mientras que B_i sería el líder de B_{i+1}. De este modo se podría crear una larga cadena de barcos. No obstante, este escenario introduciría demasiada complejidad de modo que, en principio, se mantiene el equipo compuesto por dos barcos.

Habiendo planteado el escenario y clasificado el sistema en el mundo de la cooperación de robots, es necesario validarlo con experimentos.

Capítulo 5

Experimentos y análisis de resultados

Este capítulo describe paso a paso la parte experimental del proyecto. Los experimentos han sido una fase importante de cada a la validación de los distintos prototipos y mejoras. Así como en otro tipo de desarrollos existen fases de pruebas de integración o de sistema, en un desarrollo como el que ocupa este proyecto las pruebas más fiables son los propios experimentos.

Los experimentos permiten identificar fallos y debilidades así como ver problemas que solo aparecen en el plano real y rara vez en el teórico.

Esta sección no presenta la solución final; describe los experimentos, datos y conclusiones tal cual se fueron obteniendo en el desarrollo del proyecto.

Para la realización de los experimentos, es necesario un campo experimental adecuado. La localización y facilidad de acceso al campo experimental es crucial a la hora de sacar buenos datos de los experimentos. Es importante que el campo experimental esté cerca del laboratorio así como que sea de libre disposición. Esto permite poder hacer muchos más experimentos y familiarizarse con la propia acción experimental. A la hora de buscar campo experimental para este proyecto el objetivo era que no necesitara organizar a más de dos o tres personas y que los experimentos pudieran plantearse con un día de antelación y sin necesidad de pedir permisos o pagar tasas. Esto garantizaba poder hacer tantos experimentos como fuera necesario.

Cerca del laboratorio se encuentra el parque Tierno Galván que cuenta con una serie de estanques como puede verse en la figura 5.1. En concreto, el que está situado más al suroeste es el que tiene mejores dimensiones.

El campo experimental tiene la siguiente posición GPS $(40.385304, -3.683907)$. Y el área útil es de unos 70 metros de largo por 100 de ancho. La figura 5.2 muestra el campo experimental en más detalle.

El estanque escogido tiene, además, la ventaja de que cuenta con un chorro a modo de fuente. El chorro, cuando está activo, genera una corriente radial que añade un grado importante de realismo a los experimentos.

5.1. Primeras pruebas de campo

Las primeras pruebas de campo tienen como objetivo validar los sistemas de telemetría. Se relega la responsabilidad del control remoto a una emisora y receptoras tradicionales de radio control. Esto es, el enlace de radio con la estación de tierra no se utiliza

Figura 5.1: Estanques del parque Tierno Galván.

Figura 5.2: Campo experimental. Posición $(40.385304, -3.683907)$.

para controlar el barco, se utiliza para para monitorizarlo.

Este tipo de experimentos es necesario para confiar en los sistemas de navegación ya que estos utilizarán los datos que se van a monitorizar. Fue necesario hacer varios experimentos de este tipo, casi todos fuera del agua, en el propio laboratorio o en las cercanías de la facultad.

Los primeros experimentos en sirven para comprobar todos los sistemas. En uno de ellos se introduce el barco en el estanque experimental y se comprueban los ruidos creados por el motor y los servos en una situación real. En este experimento el barco tenía dos tubos de pitot conectados a sendos sensores de presión, uno longitudinal y uno transversal. El objetivo era intentar medir tanto la velocidad transversal como la longitudinal.

Se extraen las siguientes conclusiones de ese experimento:

- El motor apenas introduce ruidos en las señales. Las perturbaciones creadas son despreciables.

- El servo introduce perturbaciones en las señales de los magnetómetros. Al mover el servo, se genera un pico de corriente que genera un campo magnético que *engaña* a la brújula. Se decide cambiar la localización de los cables y la alimentación del servo para minimizar estas perturbaciones.

- El sensor de presión transversal no lee lo que debe. Se decide prescindir de él. El barco, debido a su perfil, casi no se desplaza lateralmente. Para poder medir esa velocidad, los tubos de pitot deberían colocarse de forma muy precisa. Ese tipo de montajes y construcciones no está al alcance de las herramientas y materiales del laboratorio. Al ser ínfima la velocidad que mediría, no hay problemas en prescindir de él.

5.2. Experimentos iniciales – calibración

Es necesario calibrar dos sensores, la brújula y el sensor de presión.

La brújula tiene un cierto sesgo distinto en cada uno de los 3 ejes, de modo que es necesario estimarlo y restárselo a la señal leída para poder confiar en las medidas de rumbo. Para estimar el sesgo se dan varias vueltas completas en los distintos ejes del barco. Las señales están desfasadas $^{\pi}/_{2}$ dos a dos. Se buscan los máximos y los mínimos locales de las distintas señales y se calcula la media. Al restar este valor a las señales estas deberían centrarse en 0. La tabla 5.1 muestra los resultados. El fabricante de la brújula documenta una ganancia de $^{1}/_{1.2}$ si se quieren medir miligauss; sin embargo, dado que solo se medirá el rumbo, se prescinde de la ganancia.

Eje	Sesgo
X	39.3
Y	16.7
Z	26.5

Cuadro 5.1: Tabla de calibración de la brújula.

Para estimar el sesgo del sensor de presión se asume que, en reposo, la lectura debe ser 0. De modo que la medida a la que se estabilice el sensor cuando el barco está en reposo será el sesgo. La tabla 5.2 muestra el resultado.

Eje	Sesgo
X	201

Cuadro 5.2: Tabla de calibración del sensor de presión.

El fabricante del sensor ofrece la fórmula para obtener la velocidad en $^{m}/_{s}$ a partir de la señal del sensor (s):

$$P = \frac{1}{2} \cdot \rho \cdot v^2 \implies v = \sqrt{\frac{3.22265 \cdot (sesgo - s)}{500}}$$

Una vez calibrados los sensores ya se pueden tomar medidas cuantitativas sobre el estado del barco. En este momento es posible introducir los controladores, tomar datos, medirlos y analizarlos.

5.3. Experimentos de control – rumbo simple

El controlador más básico es el de rumbo. Este controlador intenta mantener el rumbo del barco a una consigna dada. Utiliza el servo del timón como actuador. El controlador ha sido descrito en más detalle en la sección 4.3.2. Al pasar del plano teórico al plano real, es necesario hacer algunas modificaciones al controlador. Lo primero es que todos los ángulos con los que trabaje el controlador estén en el rango $[-\pi, \pi]$, para esto es necesario utilizar la función `atan2` en lugar de `atan` y, además, utilizar una rutina como la descrita a continuación cada vez que se haga una operación con un ángulo.

```
float normalize_angle(float h)
{
    while (h >= M_PI)
        h -= 2 * M_PI;
    while (h < -M_PI)
        h += 2 * M_PI;
    return h;
}
```

La segunda modificación con respecto al controlador ideal es introducir una saturación en la salida del controlador para evitar giros muy bruscos que podrían llevar el sistema a la inestabilidad. En este caso, de las 255 posiciones de timón que ofrece el servo, se le permite al controlador utilizar las 90 centrales. En la implementación, la saturación no es más que una rutina como la siguiente.

```
float saturate(float t, float min, float max)
{
    if (t < min)
```

```
        t = min;
    if (t > max)
        t = max;
    return t;
}
```

Una vez que se han introducido los límites físicos en el controlador, es el momento de iniciar los experimentos.

Para realizar los experimentos, la estación de tierra ofrece la posibilidad de configurar e interactuar con los distintos subsistemas del barco. Para el control de rumbo, la figura 5.3. La estación de tierra permite cambiar las constantes del control, enviar variaciones de consigna o ajustar una consigna en concreto.

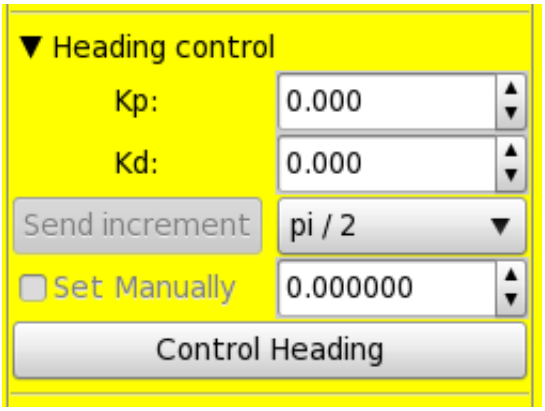


Figura 5.3: Modulo para interactuar con el sistema de control de rumbo del barco.

El objetivo de estos experimentos es obtener unas constantes K_p y K_d que hagan que el barco se comporte de forma razonable. Para ajustar las constantes se presenta un escenario en el que el controlador, con una señal de motor constante, recibe una entrada escalón en rumbo, se le deja estabilizar, recibe otra entrada escalón, se le deja estabilizar, ... Se ha utilizado un cambio de consigna de $^{\pi}/_{2}$ como entrada escalón.

Usando este procedimiento el barco debería describir un cuadrado o un rectángulo al recibir 4 cambios de consigna sucesivos. Una vez que el sistema sea estable y funcione aceptablemente bien, se harán cuadrados y rectángulos para luego estudiarlos sobre un mapa.

El ajuste de las constantes y la interpretación que se hizo del sistema se presentan en la tabla 5.3. [1]

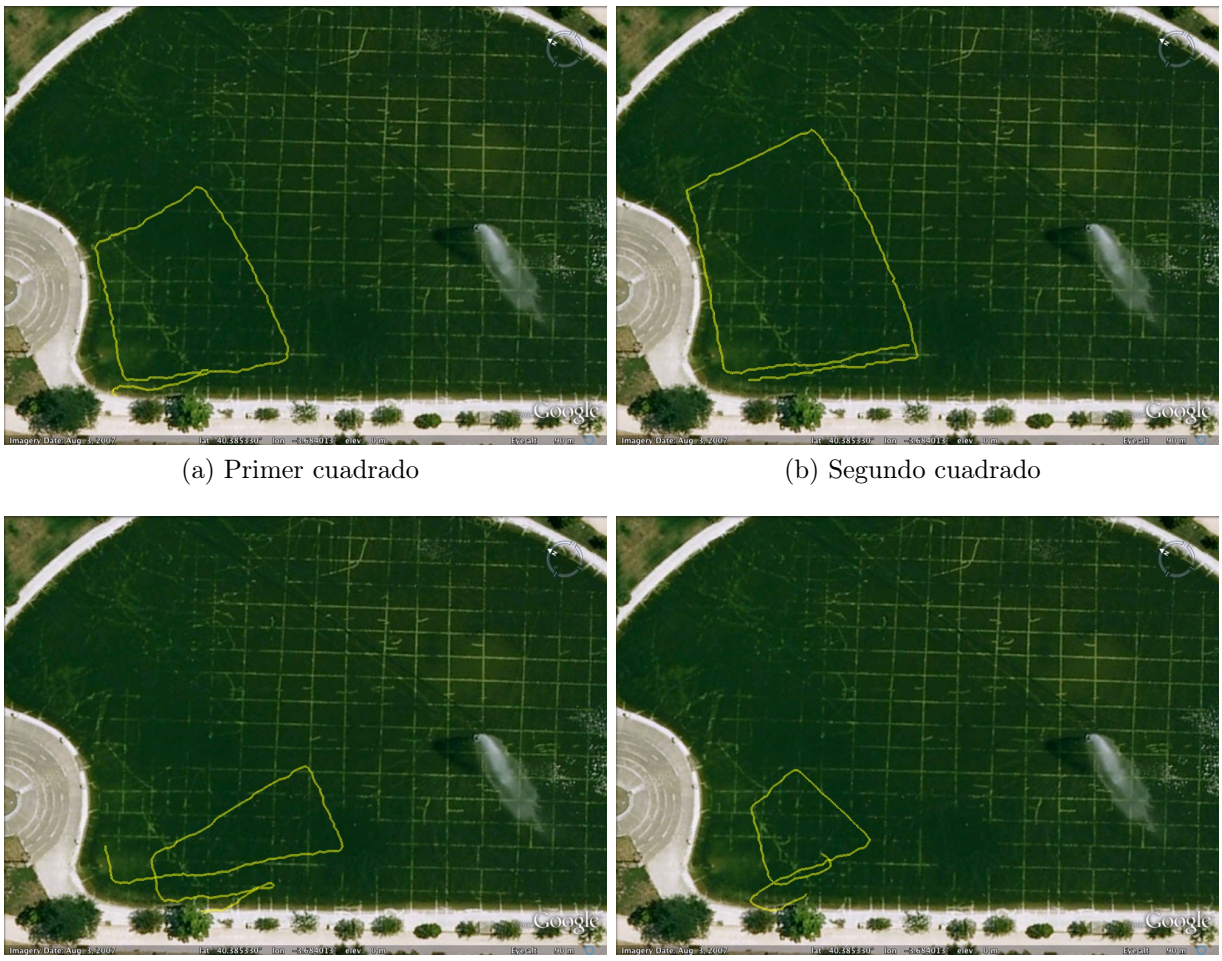
Las imágenes 5.4a, 5.4b, 5.4c y 5.4d muestran la trayectoria que siguió el barco en cada uno de los cuadrados que se hicieron.

La figura 5.5 muestra la respuesta del control de rumbo frente a varias entradas escalón (en este caso, incrementos de $^{\pi}/_{2}$). En esa misma figura pueden verse claramente los cambios de consigna y como el controlador no sobreoscila. No obstante, y como era de esperar, el rumbo no se estabiliza nunca a la consigna; sin embargo, esto no es un problema ya que las oscilaciones son estables (no crecen) y parecen centradas.

[1] En algún momento se dice que el controlador era *nervioso*, esto quiere decir que el timón se movía de un lado a otro aunque el rumbo final era relativamente bueno.

K_p	K_d	Observación
10	1	El sistema es muy lento
20	1	El sistema es muy lento
200	10	Demasiada sobreoscilación (*overshoot*). Aumentar K_d.
200	50	Demasiado *overshoot*. Aumentar K_d.
200	100	Demasiado agresivo, reducir K_p.
150	100	Rumbo suave pero controlador *nervioso*, reducir ambas.
100	50	Nervioso, reducir ambas.
100	100	Nervioso, $K_d = 0$.
100	0	Nervioso, aumentar K_d suavemente.
100	20	Nervioso, pero considerablemente mejor.
100	30	Agresivo, reducir K_p.
80	30	Agresivo, pero mucho mejor.
60	30	Bastante mejor.
80	40	Aún sigue siendo nervioso, pero el controlador es suave. Se decide parar.

Cuadro 5.3: Ajuste de las constantes de control de rumbo.



(a) Primer cuadrado

(b) Segundo cuadrado

(c) Primer rectángulo

(d) Tercer cuadrado

Figura 5.4: Trayectorias medidas con el receptor GPS.

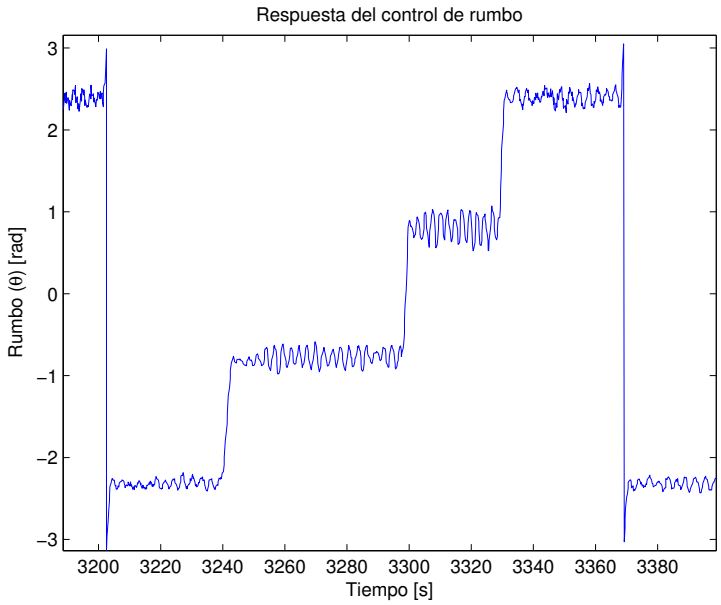


Figura 5.5: Rumbo del barco cuando se le fuerza a hacer cuadrados.

Tras estos experimentos se obtuvieron unas señales muy ruidosas que debían ser filtradas. Se ha dedicado una sección entera a la descripción y justificación del filtro utilizado. La sección 4.2, además, muestra las señales antes y después del filtrado.

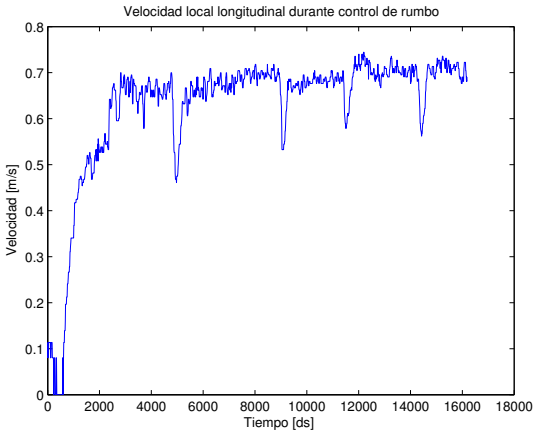
Al aplicar un filtro a las señales, el efecto del controlador cambia. En los siguientes experimentos se hace un reajuste de las constantes para dar más peso al controlador D ahora que las señales están filtradas. Esto hará que el control sobreoscile menos y sea más fino.

La figura 4.7b (página 32) muestra la señal del sensor de presión durante uno de los cuadrados. Se reconocen perfectamente los cambios de consigna del control de rumbo porque hay una caída de velocidad local longitudinal. Esto es porque cuando el barco gira, la popa se desalinea del rumbo de modo que la velocidad de avance en la dirección de la popa es mucho menor.

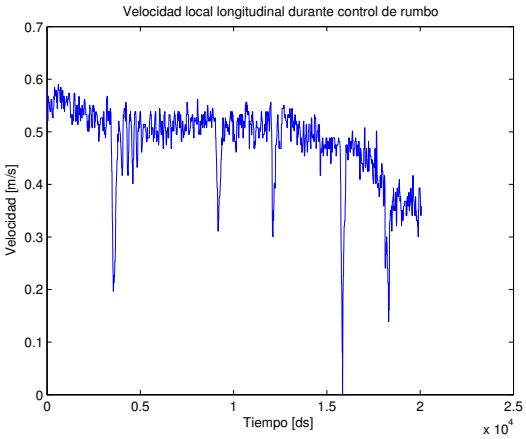
Las figuras 5.6a, 5.6b, 5.6c y 5.6d muestran la señal del pitot filtrada durante los experimentos descritos anteriormente. En las figuras 5.6c y 5.6d es posible ver que la velocidad cae a 0 en algunos casos. Esto se corresponde a un cambio de consigna de rumbo de $\pm\pi$. Al dar un giro completo, en algún momento la velocidad de avance debe ser 0 y esto es, efectivamente, lo que se está viendo en ambas figuras.

Sin embargo, esto no es lo que está ocurriendo en la caída a 0 en la figura 5.6b, ya que en este experimento no existen giros de $\pm\pi$. En este caso lo que ha podido pasar es que durante un giro el tubo de pitot se ha alineado con la dirección de la corriente del agua. Dado que el sensor de presión mide la velocidad con respecto al fluido, este caso daría una medida de 0. Se asume que eso es lo que ha pasado durante ese experimento.

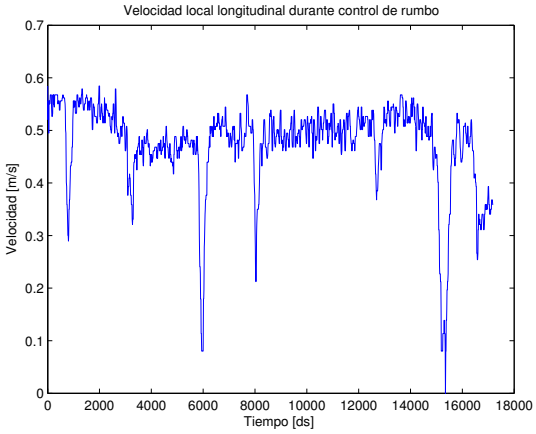
Los experimentos han servido para ajustar y validar el control de rumbo del barco. Este control es básico en cualquier control de trayectoria ya que todos se basarán en él. Los experimentos han servido para validar la robustez del control en situaciones normales experimentales. La frecuencia de las oscilaciones es de 1 Hz. Es posible reducir la amplitud



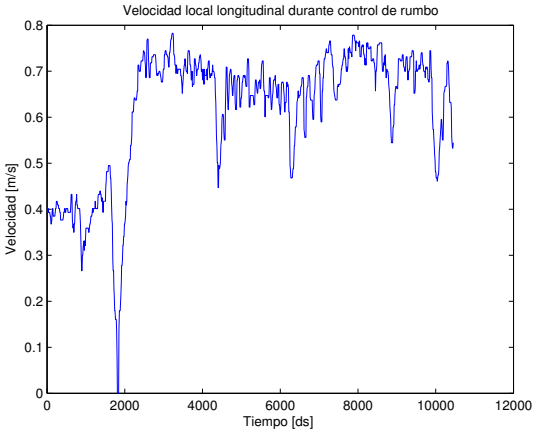
(a) Primer cuadrado



(b) Segundo cuadrado



(c) Primer rectángulo



(d) Tercer cuadrado

Figura 5.6: Señales del pitot durante los experimentos.

de las mismas reajustando las constantes del controlador, sin embargo, lo normal es que esto nos diera un controlador más lento. Se decide mantener el controlador así.

Una vez el control de rumbo ha sido validado, se debe proceder con los siguientes controladores. El siguiente controlador en ser diseñado e implementado fue el control de velocidad.

5.4. Experimentos de control – velocidad

El controlador de velocidad permitirá al barco llevar una cierta velocidad de avance estable. Este controlador ha sido descrito en la sección 4.3.2. El módulo de la estación de tierra que controla este sistema es muy parecido al del control de rumbo. Puede verse en la figura 5.7.

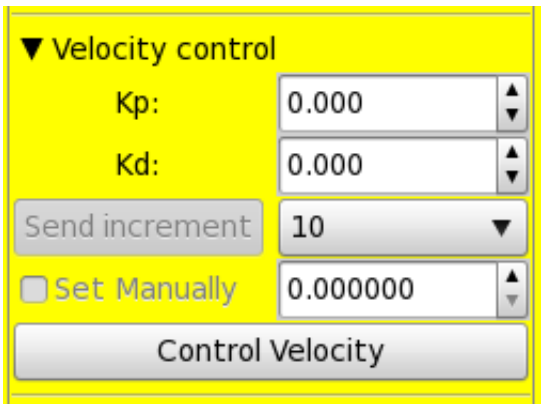


Figura 5.7: Modulo para interactuar con el sistema de control de velocidad del barco.

Una vez más, al igual que se hizo con el control de rumbo, hay que hacer algunas modificaciones al pasar del control teórico a la implementación en un sistema real. En este caso se ha añadido una saturación en la señal de motor. El motor es controlado por un variador de radio control que ofrece 255 valores de potencia. Se limita la actuación del controlador de forma que las salidas siempre estén en el rango $[0, 100]$.

Igual que en el caso del control de rumbo, es necesario ajustar las constantes K_p y K_d del control de velocidad. Para facilitar este ajuste, durante el diseño del control se introdujo una salida rampa. La tabla 5.4 contiene el proceso de ajuste.

K_p	K_d	Observación
1	1	El sistema es muy lento. Aumentar K_p y K_d.
5	5	Se decide mantener las constantes.

Cuadro 5.4: Ajuste de las constantes de control de rumbo.

Las expectativas sobre el controlador son algo menores que las que se tienen sobre el control de rumbo. Mientras que el barco mantenga un cierto rumbo es clave para conseguir que siga ciertas trayectorias o vaya a un cierto punto, el control de velocidad no es tan crítico. El control de velocidad sirve para dos cosas: mantener una velocidad más o menos estable durante un experimento o una misión y ayudar a que el control de

rumbo sea más rápido. El control de velocidad mejora el control de rumbo porque, como se pudo ver en las figuras 5.6, durante los giros la velocidad longitudinal cae; al caer la velocidad, el control reacciona y acelera el motor. Esta respuesta provoca que el sistema tenga más par y, por tanto, más aceleración angular, y así alcance antes la consigna de control.

Las primeras pruebas del control de velocidad demostraron que era muy inefectivo sin utilizar control de rumbo. Al no tener un control de rumbo, el barco no va recto y le cuesta mucho alcanzar las consignas de control. Una vez se activan ambos controles, la respuesta del sistema es la esperada. Puede verse el efecto del controlador frente a una entrada escalón en la figura 5.8.

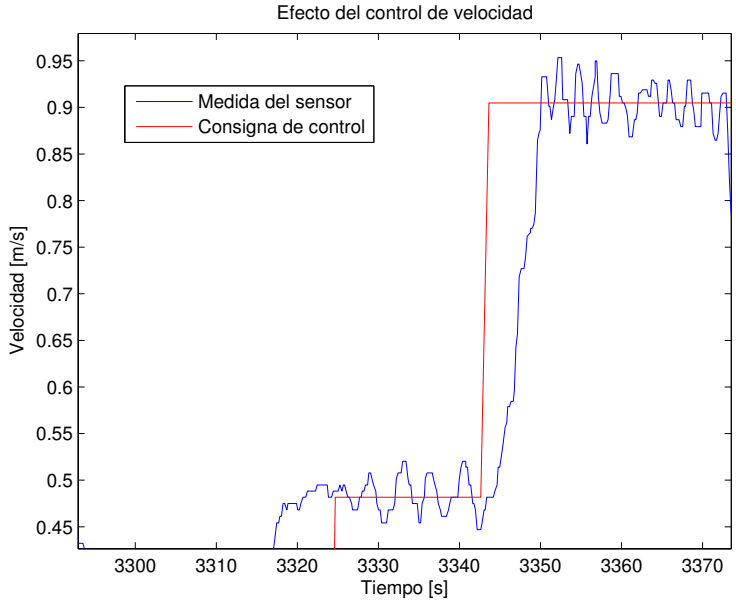
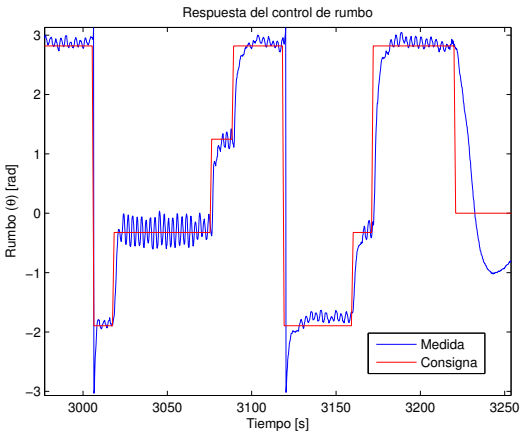


Figura 5.8: Efecto del control sobre la velocidad local longitudinal.

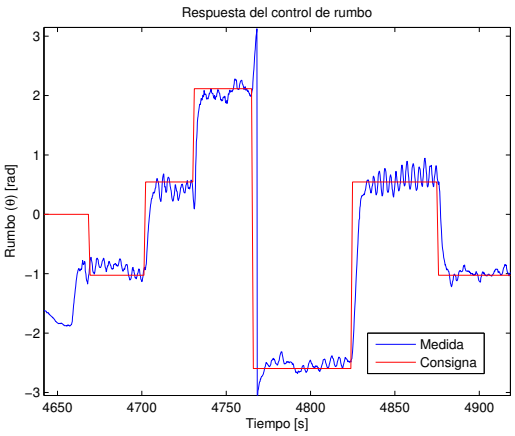
5.5. Experimentos de control – reajuste control de rumbo

Una vez funciona el control de velocidad y se ha añadido el filtrado de las señales, hay que reajustar el control de rumbo. Como se ha comentado en la sección 5.4, el barco acelera durante los giros debido a que el tubo de pitot se desalinea con el rumbo del barco. Al acelerar, el giro cambia y, por tanto, el controlador de rumbo debe ser reajustado. Además, al añadir un filtro de mediana, es posible dar más peso al controlador D. De modo que, tras mirar algunas señales en el campo experimental se decide dejar las constantes de rumbo en $K_p = 20$ y $K_d = 20$. Las figuras 5.9 muestran la respuesta escalón del control de rumbo.

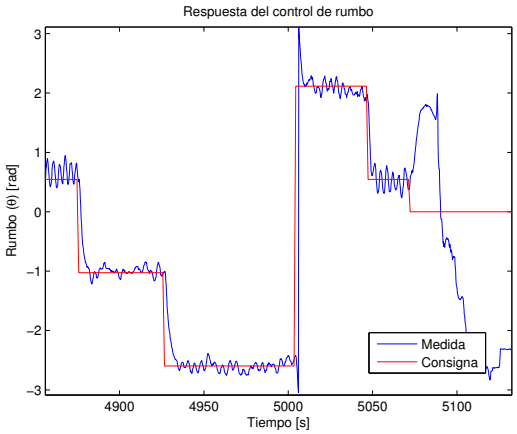
Se observa que el controlador es algo más lento que antes, algo que era de esperar ya que es uno de los efectos de darle más peso al controlador D. Sin embargo, se ha



(a)



(b)



(c)

Figura 5.9: Respuesta del control de rumbo y velocidad frente a distintas entradas escalón en rumbo.

eliminado cualquier *overshoot* que pudiera haber. El control se nota más suave y parece estabilizarse antes.

En las figuras 5.9 se observa que las oscilaciones no son igual de amplias a lo largo del rango $[-\pi, \pi]$. En concreto, parece que en un cierto entorno de 0 las oscilaciones son mucho más amplias a las de otras zonas. Viendo un poco más de cerca los datos (figuras 5.10), se confirma el hecho.

La zona cercana al 0 corresponde al norte magnético, es posible que haya habido algún error de calibración de la brújula. Además de eso, si uno observa la gráfica de la función `atan` (ver figura 5.11) se ve que es muy sensible en la zona del 0 de modo que cualquier error de la propia brújula (independientemente de la calibración) se magnificaría mucho en esa zona.

5.6. Experimentos de control – modelo

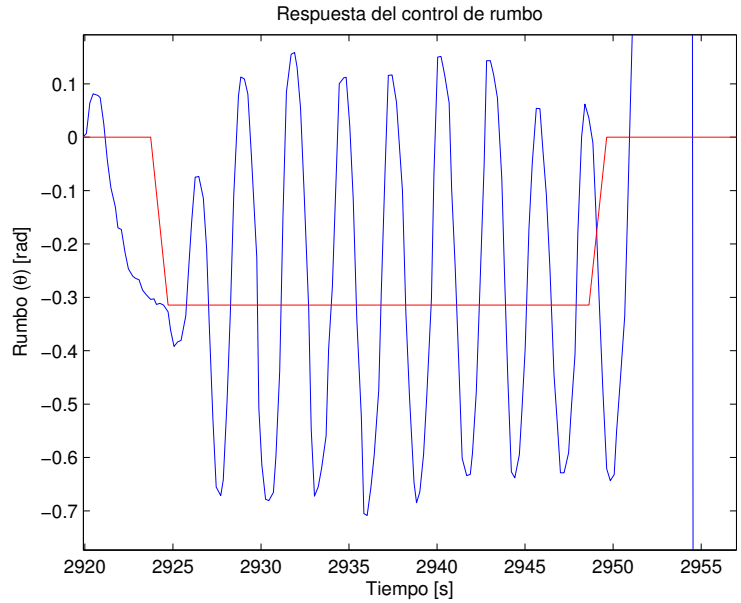
Los experimentos tienen dos objetivos. Por un lado, validar los algoritmos y técnicas desarrolladas en el laboratorio utilizando una plataforma experimental real. Por otro lado, refinar el modelo que se presentó en la sección 4.1. Refinar el modelo permitirá conseguir simulaciones más fiables y así poder afinar los algoritmos antes de los experimentos. Esto es interesante porque los experimentos son costos en cuanto a tiempo y no siempre se tiene todo el tiempo que se querría.

Para estos experimentos se utiliza un módulo que permite enviar señales de motor y/o timón fijas. Esto tiene una ventaja con respecto al joystick ya que se puede controlar exactamente qué señal se envía y los experimentos son repetibles. La figura 5.12 muestra este módulo.

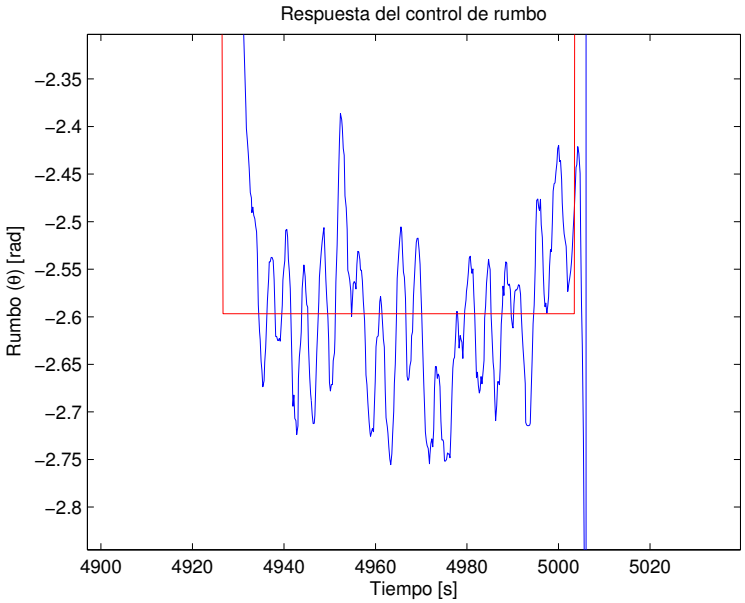
El sistema de propulsión del barco consiste en un motor de continua que hace girar una hélice. El motor se controla utilizando un variador de radio control. La señal que se le envía al variador está relacionada con la potencia que ejerce el motor sobre la hélice. Este sistema es muy complicado de modelar y, tampoco proporcionaría ninguna ventaja. Sin embargo, mejorar el modelo requiere conocer qué fuerza ejerce el motor sobre el barco para cada valor que se le envía.

Para medir la fuerza de empuje del motor se utiliza una galga extensiométrica. Una galga extensiométrica es un sensor que permite medir fuerzas. Se ata un cabo largo a la popa del barco y el otro extremo se ata a la galga. El procedimiento del experimento es el siguiente:

- Se pone el barco en el agua, de forma que el cabo y el rumbo del barco se alineen.
- Se fija una señal de motor.
- Se espera a que la fuerza leída de la galga extensiométrica se estabilice. Esto asegura que el sistema de propulsión está en régimen estacionario.
- Se anota la media de los valores medidos por el sensor en régimen estacionario.
- Se repite el proceso con muchas medidas.



(a) Oscilación cerca del 0



(b) Oscilación lejos del 0

Figura 5.10: Observación detallada de las oscilaciones.

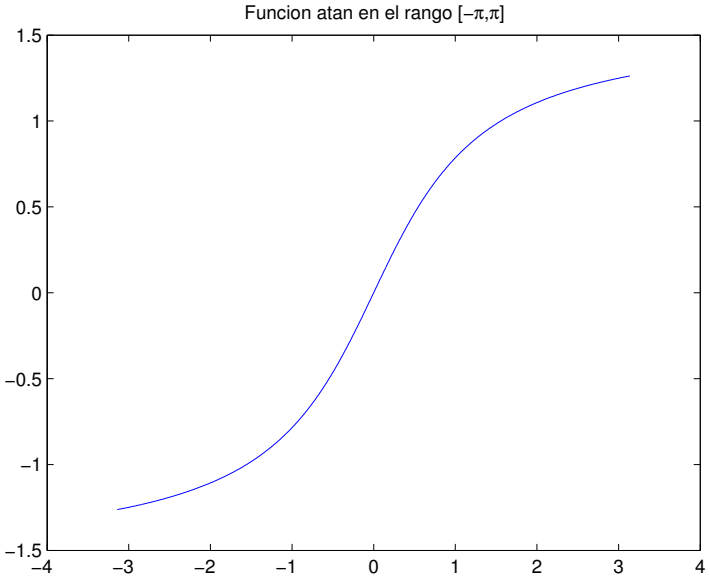


Figura 5.11: Función `atan` en el rango $[-\pi, \pi]$.

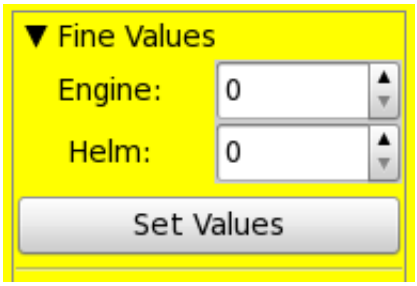


Figura 5.12: Modulo para enviar señales al motor o al timón.

Señal	Fuerza [N]
40	1.2
50	1.7
60	2.6
70	3.5
80	4.2
90	5.2
100	6.5

Cuadro 5.5: Valores de empuje del motor.

El motor comienza a mover la hélice a partir del valor de señal 20, y, por debajo de 40 apenas empuja al barco. La tabla 5.5 muestra los valores de fuerza para cada señal de motor.

Utilizando Matlab se puede hacer un ajuste lineal de estos datos para conseguir la fuerza de empuje como función de la señal del motor; así como el coeficiente de correlación de los datos obtenidos. La figura 5.13 muestra el ajuste. El ajuste da un coeficiente de correlación de 0.9942 lo cual da mucha confianza sobre el mismo.

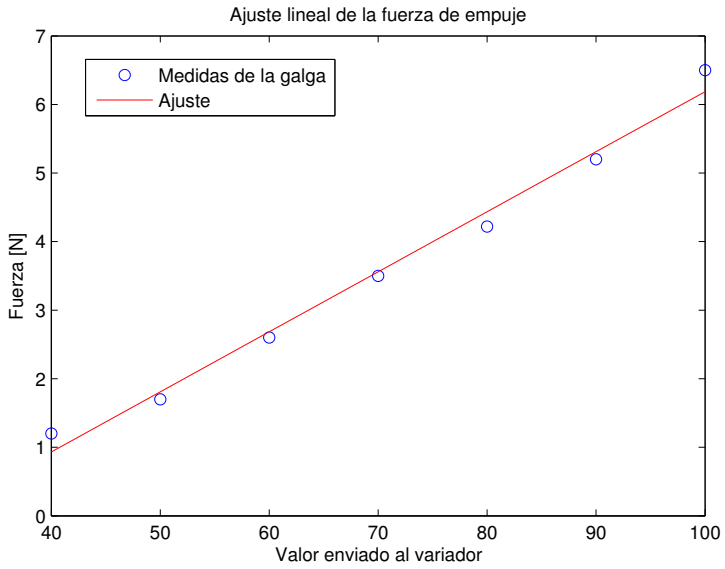


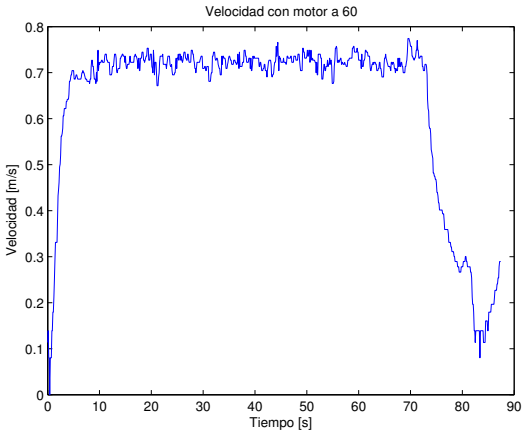
Figura 5.13: Ajuste lineal de la fuerza de empuje. Coeficiente de correlación = 0.9942.

La recta obtenida del ajuste nos permite definir la fuerza del motor (F) en función de la señal del variador (e):

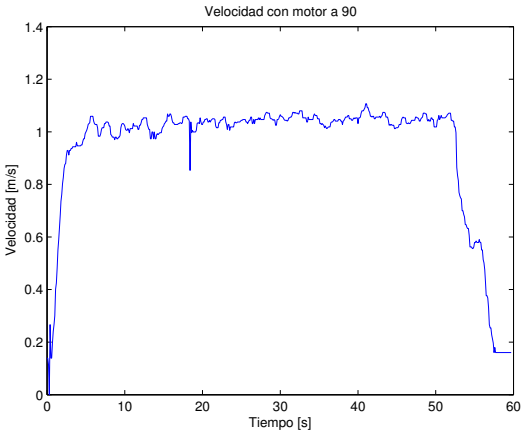
$$F(e) = 0.087571 \cdot e - 2.57 \tag{5.1}$$

Una vez que se ha conseguido una aproximación al modelo de propulsión del barco (5.1) es posible seguir estimando el resto de coeficientes del modelo. Lo siguiente que se va a intentar estimar es la velocidad longitudinal a la que se estabiliza el barco para las distintas señales del motor. El barco tiene dos sensores para medir su velocidad, el GPS y el sensor de presión. A velocidades bajas es más fiable el sensor de presión de modo que se utilizará este. El procedimiento es el siguiente:

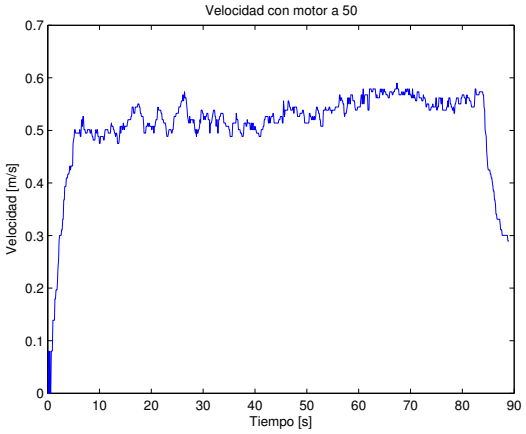
- Se posiciona el barco y se fija una señal del motor.
- Se espera hasta que la velocidad se estabilice. Esto garantiza que se ha pasado el transitorio y el sistema está en régimen estacionario.
- Se para el barco, se le da la vuelta y se repite en sentido contrario. Esto permite eliminar efectos de viento y corriente.
- Repetir el procedimiento para varias señales de motor.



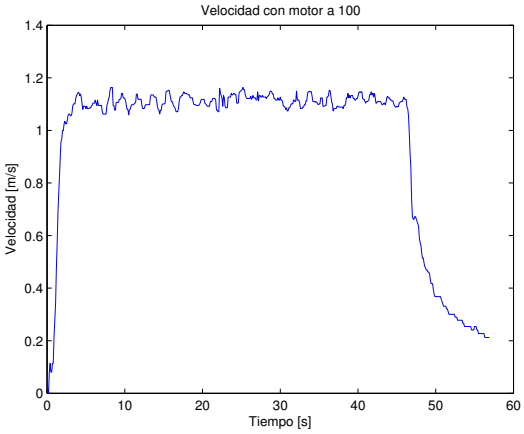
(a) Velocidad con motor a 60



(b) Velocidad con motor a 90



(c) Velocidad con motor a 50



(d) Velocidad con motor a 100

Figura 5.14: Señales del pitot durante los experimentos para medir $v_{estabiliza}$.

Las figuras 5.14a, 5.14b, 5.14c y 5.14d muestran los resultados de algunos de estos experimentos.

Una vez se tienen los datos, se obtiene la media de velocidad. Los datos se presentan en la tabla 5.6.

Señal	Velocidad $[^m/_s]$
40	0.3928
50	0.5515
60	0.7078
70	0.8378
80	0.9367
90	1.0048
100	1.1077

Cuadro 5.6: Valores de velocidad a la que se estabiliza el barco.

Una vez más, se prueba un ajuste lineal de los datos. Se obtiene un coeficiente de correlación de 0.9898 de modo que se confía en el ajuste. La figura 5.15 muestra los datos y la recta de regresión. La recta obtenida es:

$$v_{estabiliza}(e) = 0.011715 \cdot e - 0.028784 \tag{5.2}$$

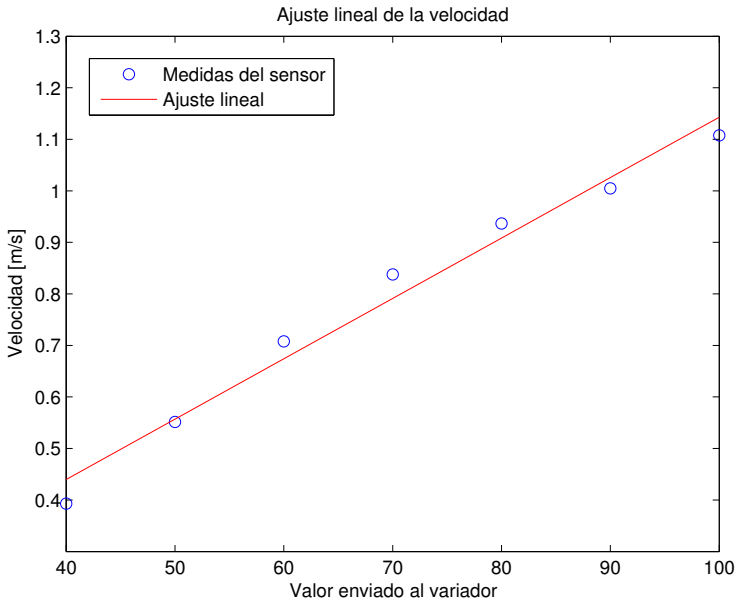


Figura 5.15: Ajuste lineal de los valores a los que se estabiliza la velocidad. Coeficiente de correlación = 0.9898.

En este momento se tienen datos suficientes para empezar a aproximar un modelo para el desplazamiento longitudinal del barco. Para esto, se va a reformular el modelo presentado en la sección 4.1. En este caso, se plantea el modelo en ejes barco. En este modelo, x es el desplazamiento longitudinal del barco.

$$\ddot{x} = (F \cdot \cos\eta - \mu_t \cdot \dot{x}) \cdot \frac{1}{m} \tag{5.3}$$

Viendo (5.3), en régimen estacionario, los únicos valores no conocidos son la masa del barco m y el coeficiente de rozamiento dinámico μ_t. La masa es fácil de medir utilizando una balanza de precisión; se obtiene $m = 4.2 \pm 0.1Kg$.

En régimen estacionario, v es constante de modo que:

$$v = cte. \implies a = \ddot{x} = 0 \tag{5.4}$$

$$0 = (F \cdot \cos\eta - \mu_t \cdot \dot{x}) \cdot \frac{1}{m} \tag{5.5}$$

Dado que durante las medidas el servo del timón estaba en su posición central, $\eta = 0 \implies \cos\eta = 1$. Sabemos que $\dot{x}$ es constante, de modo que se puede reescribir (5.5) poniéndola en función de (5.1) y (5.2):

$$0 = (F(e) - \mu_t \cdot v_{estabiliza}(e)) \cdot \frac{1}{m} \tag{5.6}$$

Despejando (5.6):

$$\frac{F(e)}{m} = \frac{\mu_t \cdot v_{estabiliza}(e)}{m} \tag{5.7}$$

$$F(e) = \mu_t \cdot v_{estabiliza}(e) \tag{5.8}$$

$$\mu_t = \frac{F(e)}{v_{estabiliza}(e)} \tag{5.9}$$

Utilizando los datos de las tablas 5.5 y 5.6 se puede representar μ_t frente a $v_{estabiliza}(e)$. La tabla 5.7 muestra estos datos.

Velocidad $[^m/_s]$	μ_t
0.3928	2.6873
0.5515	2.8767
0.7078	3.5183
0.8378	4.0494
0.9367	4.3938
1.0048	5.0631
1.1077	5.7630

Cuadro 5.7: Valores de μ_t en función de $v_{estabiliza}$.

Se observa que los valores de μ_t varían con la velocidad de modo que se intenta conseguir un modelo para $\mu_t(v)$. Un ajuste lineal da un coeficiente de correlación de 0.9743 así que se acepta el modelo como suficientemente bueno. La figura 5.16 muestra este ajuste que viene dado por la recta (5.10).

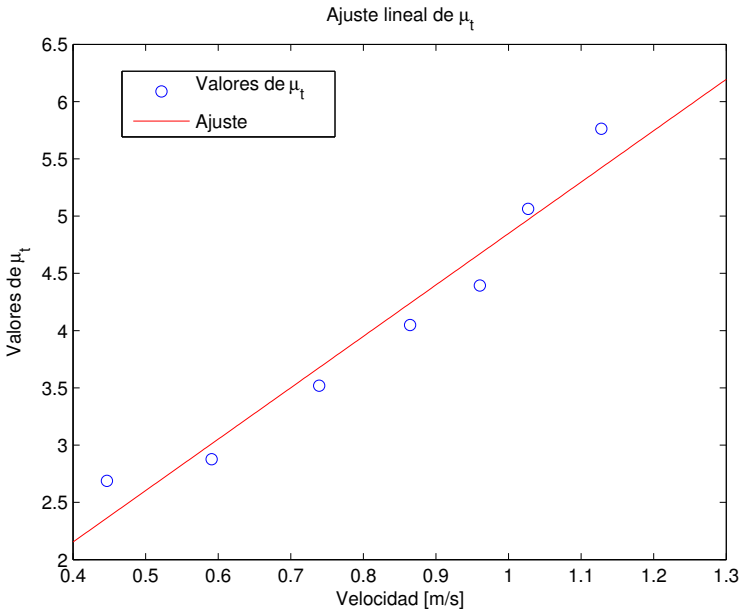


Figura 5.16: Ajuste lineal de $\mu_t(v)$. Coeficiente de correlación de 0.9743.

$$\mu_t(v) = 4.488861 \cdot v + 0.358965 \tag{5.10}$$

Al obtener un modelo lineal para $\mu_t(v)$ hay que reescribir el modelo (5.3) ya que este asumía un sistema de orden 0 para la resistencia al avance. El modelo, incorporando la aproximación para $\mu_t(v)$, se presenta en la ecuación (5.11).

$$\ddot{x} = (F \cdot \cos\eta - \mu_t(\dot{x}) \cdot \dot{x}) \cdot \frac{1}{m} \tag{5.11}$$

Una vez obtenido el modelo, es el momento de comprobar cómo se ajusta a la realidad. Para comprobarlo se toma un registro de velocidad de verdad, como el mostrado en la figura 5.17.

Dado que también existe un registro de los valores que se han enviado al variador, puede *repetirse* el experimento en el simulador. Utilizando los modelos (5.1), (5.11) y (5.10) se hace una simulación. Los resultados, comparados con los datos reales, se presentan en la figura 5.18.

Como puede verse en la figura 5.18 los transitorios medidos del barco se diferencian bastante de los simulados. Esto es normal por varias razones. La primera es que todos los datos se han obtenido en regímenes estacionarios. La segunda razón es que el sistema de propulsión muy probablemente no tenga un transitorio lineal ya que el motor tarda un tiempo en ofrecer su empuje máximo. La tercera y última razón es que la medida del sensor de presión no es muy fiable en los transitorios ya que la variación del cabeceo [2] del barco falsea la medida. Esto no es un problema ya que el objetivo es conseguir un modelo que sea fiable en régimen estacionario y no interesan mucho los transitorios.

[2]En inglés, *pitch*.

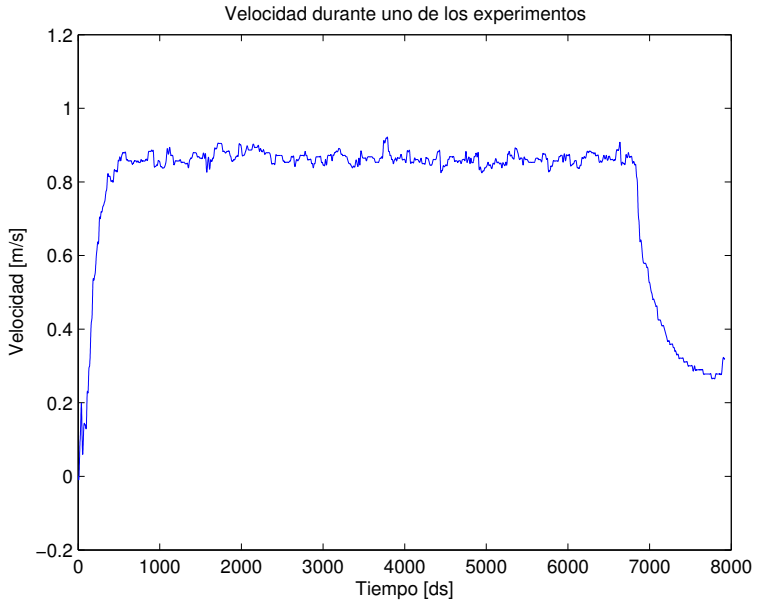


Figura 5.17: Velocidad durante uno de los experimentos.

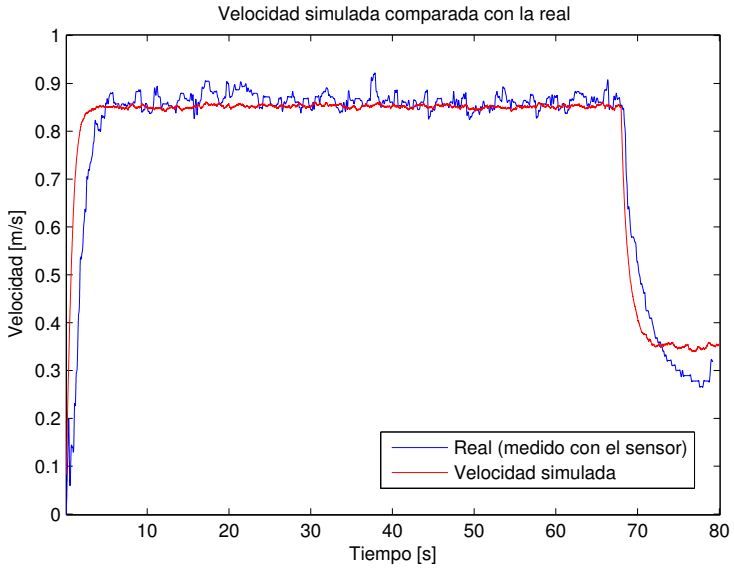


Figura 5.18: Comparación de la simulación con los datos reales.

5.7. Experimentos de control – waypoints

Una vez se ha refinado el modelo y se han ajustado tanto el control de rumbo como el de velocidad, es el momento de plantear el control *ir a un punto* o control *waypoints*. Este control es el primer paso para conseguir que el barco haga trayectorias. El diseño del controlador se ha discutido exhaustivamente en la sección 4.3.4. En dicha sección ya se presentó la necesidad de hacer que la definición de estos waypoints fuera:

$$WP = (\phi, \lambda, R)$$

Donde ϕ y λ son su posición GPS (latitud y longitud) y R es el radio.

Un módulo en la estación de tierra permite interactuar con este sistema. Permite asignar distintos puntos consigna así como controlar el radio. La figura 5.19 muestra una captura de este módulo.

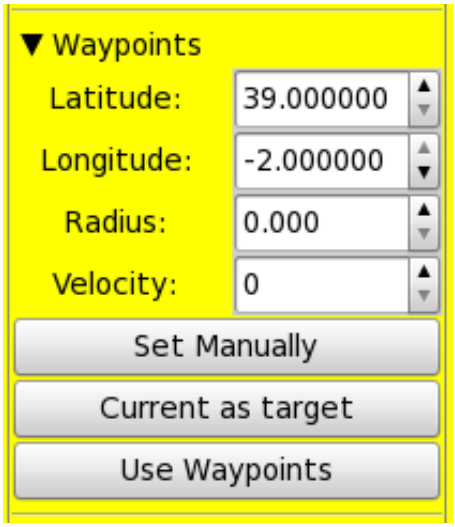


Figura 5.19: Modulo para interactuar con el sistema de control *ir a un punto*.

Para las primeras pruebas de este controlador se plantea un escenario experimental con distintos waypoints en el estanque. La figura 5.20 muestra este escenario. Durante estos experimentos, el chorro del estanque estaba activado, esto crea una corriente radial con centro en el propio chorro.

Es posible ver el efecto que tiene no definir un radio para los waypoints en la figura 5.21. El barco empieza cerca del punto *p4*, a medio camino entre este y el punto *p1* se activa el piloto automático. El barco se aproxima al punto y pasa a menos de un metro de este. Se observa que el barco se queda girando. Se corta el control automático, se cambia el punto de consigna a *p4* y se activa el control automático. Es curioso ver cómo el barco describe una trayectoria casi perfecta, la razón por la que no lo hace es, probablemente, la corriente creada por el chorro.

Una vez más, se ve el mismo efecto con el punto *p4* aunque el barco pasa a menos de dos metros del mismo. Se repite el proceso: se corta el piloto automático y se cambia la consigna al punto *p2* y se vuelve a activar el control. En este caso la trayectoria es perfectamente recta y el barco pasa a poco más de un metro del punto. Una vez más se observa que el barco gira alrededor del punto consigna. Se desactiva el control automático

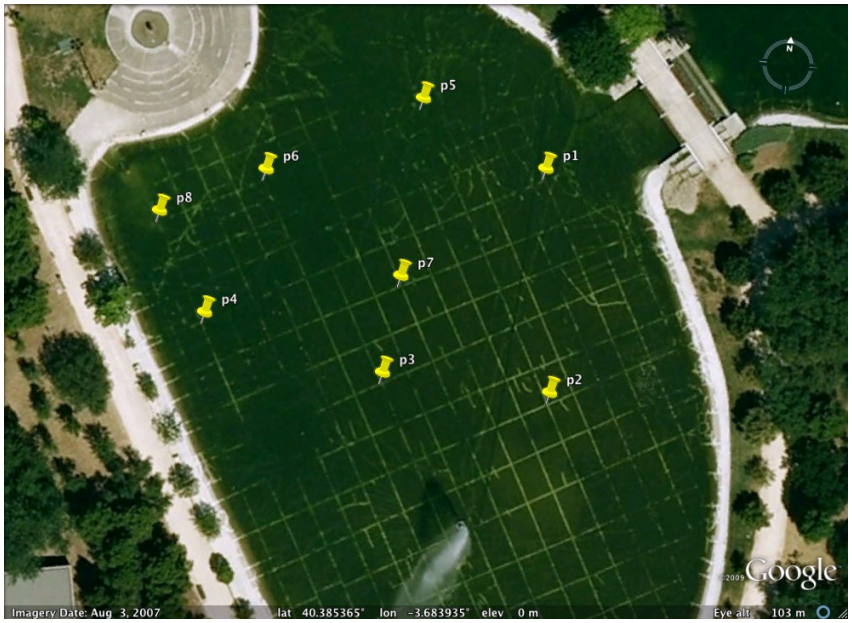


Figura 5.20: Escenario experimental para el control *ir a un punto*

Figura 5.21: El barco con control *ir a un punto*.

y se decide llevar el barco a la orilla (base de experimentos) para cambiar la batería del motor.

Una vez la batería se ha cambiado, se vuelve a dejar el barco en el agua y se le da un pequeño empujón. Se activa el control automático utilizando *p5* como punto consigna. Una vez más, la trayectoria es bastante buena y el barco vuelve a pasar a menos de metro y medio del punto. Como era de esperar, vuelve a girar alrededor del punto consigna.

En este experimento, aunque se había definido un radio, el software tenía un error que resultaba en un cálculo incorrecto de la distancia. El control, aunque pasa muy cerca de los puntos (en algunos casos, a menos de $1m$ de distancia), nunca cree haber llegado al punto. Al no llegar, intenta acercarse un poco más haciendo giros alrededor del punto consigna hasta que el timón entra en saturación y el barco no puede girar más. En ese momento el giro se estabiliza y el barco nunca llega al punto.

La conclusión principal de estos experimentos es que el barco es capaz de ir a un punto y, una vez arreglado el error software, será posible darle distintos puntos al barco para que siga distintas trayectorias.

Antes de volver al campo experimental se hacen pruebas de cálculo de distancia en el parque de la facultad para comprobar que el error ha sido solucionado correctamente.

5.8. Experimentos de trayectorias

Una vez arreglado el error de software, se asume que el control de *waypoints* funciona bien. Si el barco es capaz de ir a un punto y sabe cuando se ha acercado lo suficiente, es posible darle una lista de puntos y que los visite.

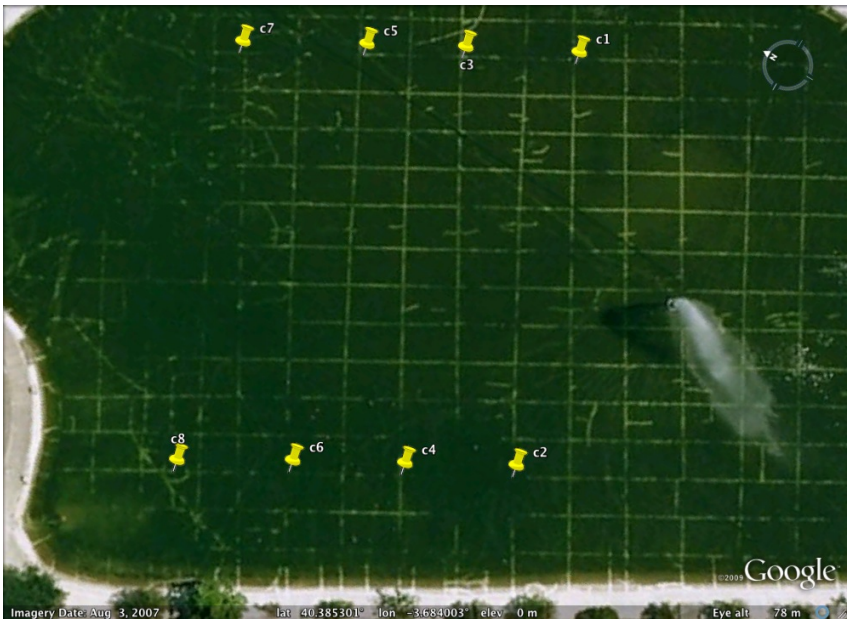
Estos experimentos permiten diseñar maniobras y misiones más elaboradas. Se plantean dos escenarios distintos como puede verse en las figuras 5.22. Con estos experimentos se pretende comprobar la capacidad del vehículo de seguir distintas trayectorias con el objeto de utilizar las maniobras para otros fines. Por ejemplo 5.22a intenta simular las trayectorias utilizadas en barcos dragaminas por medio de pasillos y giros [Alves06], mientras que 5.22b hace una maniobra utilizada por planeadores para perder altura.

Para elegir entre las distintas trayectorias, un módulo de la estación de tierra ha sido desarrollado. Se presenta en la figura 5.23.

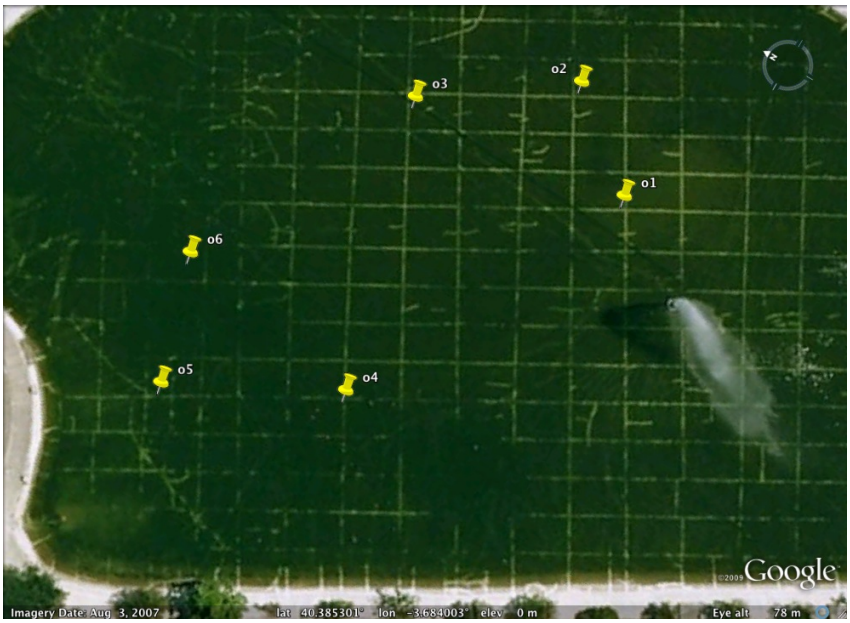
Para la trayectoria planteada en la figura 5.22a se colocó el barco más o menos centrado entre los puntos y se le activó el piloto automático. El resultado se puede ver en la figura 5.24. Para evitar posibles problemas de resolución, se decidió utilizar un radio muy amplio de modo que nunca llega a acercarse mucho a los puntos. Sin embargo, los visita todos muy bien. En algunos casos, parece que el barco hiciera el giro opuesto al que debería hacer. Esto no es así, el control de rumbo hace que el barco siempre gire hacia el lado que minimiza $\varepsilon(t)$. El hecho de que algunos giros parezcan al revés es porque las oscilaciones y las corrientes hacen que el barco llegue con un rumbo aparentemente peor. El control de rumbo, al no poseer conocimiento sobre el siguiente punto, no puede hacer las correcciones pertinentes.

Se decide no reintentar la trayectoria de giros y pasillos dado que se necesitaría un algoritmo algo más sofisticado para conseguir la trayectoria *deseada*.

El escenario presentado en la figura 5.22b se prueba a continuación. Una vez más, se posiciona el barco en el estanque y se activa el piloto automático. El resultado se puede



(a) Pasillos y giros



(b) Maniobra de la pajarita

Figura 5.22: Escenarios de trayectoria planteados. Simulan maniobras reales.

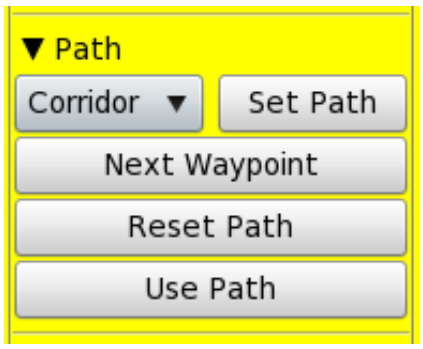


Figura 5.23: Modulo para interactuar con el sistema de control de trayectorias.

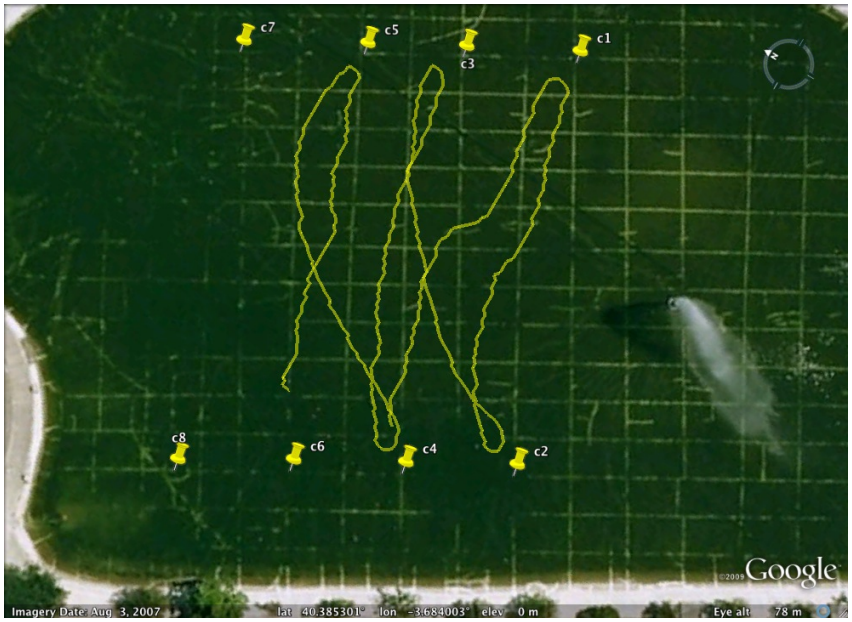


Figura 5.24: El barco siguiendo la trayectoria planteada en 5.22a.

ver en la figura 5.25a. En este caso se ha utilizado el mismo radio que en la figura 5.24. Al ver que la trayectoria que describía el barco se parecía mucho a la trayectoria *perfecta*, se decide disminuir el radio para forzar al barco a que pase más cerca. El resultado se puede ver en la figura 5.25b. Se observa que el barco se ciñe mucho mejor a los puntos y la trayectoria es más parecida a la *ideal*. La figura 5.26 muestra la diferencia entre ambas trayectorias.

La comparación muestra la superioridad de la segunda configuración y se da por buena.

Conviene evaluar el controlador de rumbo durante una trayectoria real donde los cambios de consigna no son entradas escalón. En el caso de la trayectoria de *ochos*, la consigna de rumbo va cambiando a medida que el barco avanza en el experimento. La figura 5.27 muestra el rendimiento del controlador durante parte de este experimento.

Se puede ver que el error es relativamente bajo y solo crece durante los grandes cambios de consigna. Estos cambios de consigna se deben a los cambios de waypoint durante la trayectoria. Tras estos cambios el error vuelve a disminuir y se mantiene, generalmente, por debajo de medio radián. Cuantitativamente, los datos del error ε_θ se pueden resumir en la tabla 5.8. Se considera que el error de rumbo es mínimo.

Variable	Valor
$\overline{\varepsilon_\theta}$	0.2189
σ	0.2404
σ^2	0.0578
Mediana	0.1426
Moda	$4.7099 \cdot 10^{-5}$

Cuadro 5.8: Descripción estadística del error en el control de rumbo.

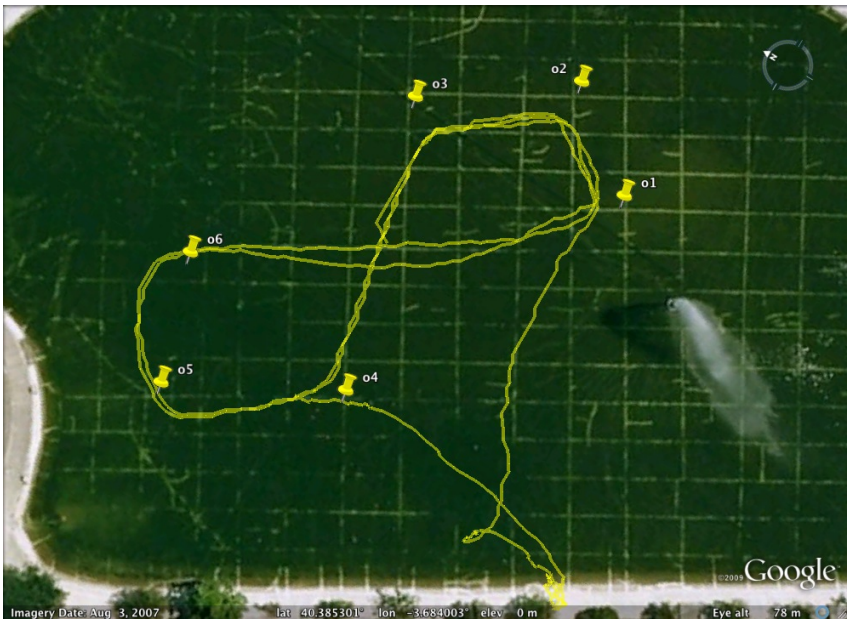
Utilizando el modelo presentado en el capítulo 4 y con los ajustes realizados en la sección 5.6 se pueden realizar validaciones del modelo original. Utilizando las ecuaciones (4.1), (4.2) y (4.3) y un integrador numérico como el descrito en el algoritmo 1 se puede simular la dinámica del barco.

Algorithm 1 Integrador numérico de *Euler*. Calcula $x = \int_0^T v(t)\,dt$

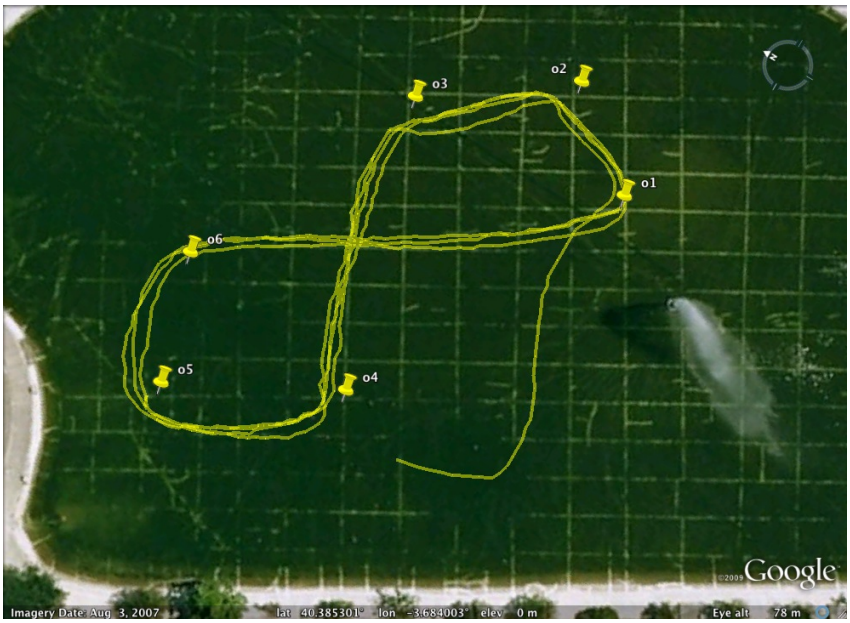
```
x ← 0
t ← 0
while t < T do
    x ← r + v(t) · dt
    t ← t + dt
end while
```

En este momento se puede plantear una simulación con las mismas características que el experimento de la figura 5.25b. Los resultados se presentan en las figuras 5.28.

Como puede verse, cualitativamente, la simulación se parece bastante al experimento real. Se ha decidido no hacer un análisis cuantitativo del error porque aún falta por ajustar la parte del modelo relacionada con el giro del barco. No obstante, los resultados indican que se va por el buen camino en cuanto al modelo.



(a) Con un radio grande



(b) Con un radio más pequeño

Figura 5.25: El barco siguiendo la trayectoria planteada en 5.22b.

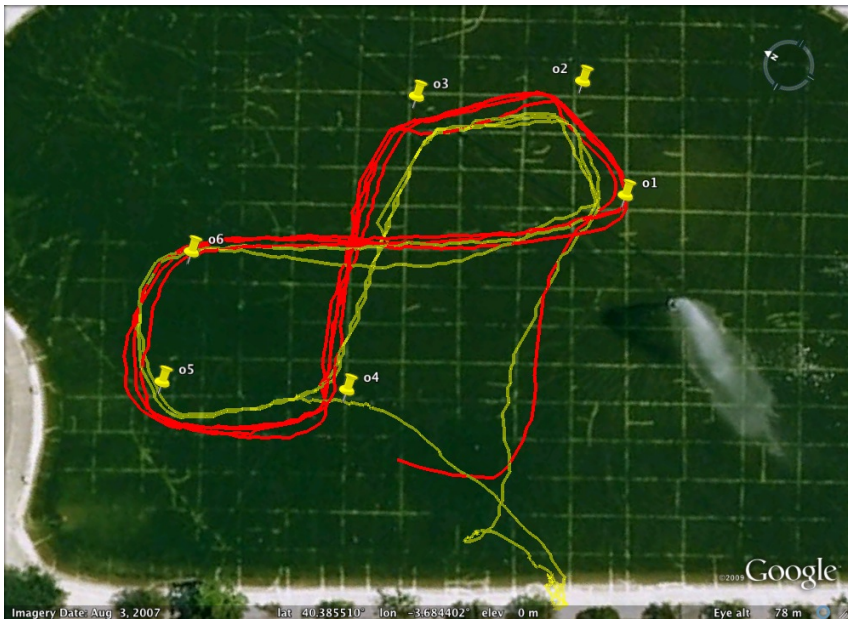


Figura 5.26: Comparación de las trayectorias 5.25a (amarillo) y 5.25b (rojo).

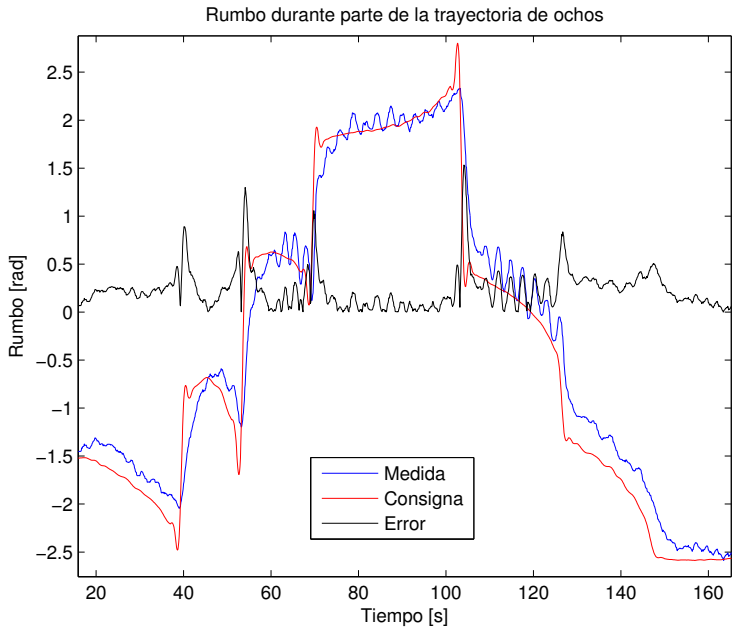
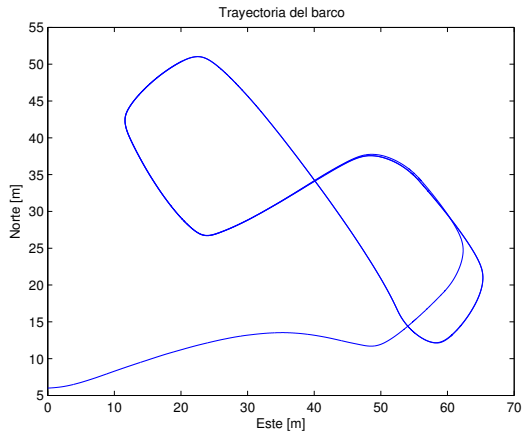
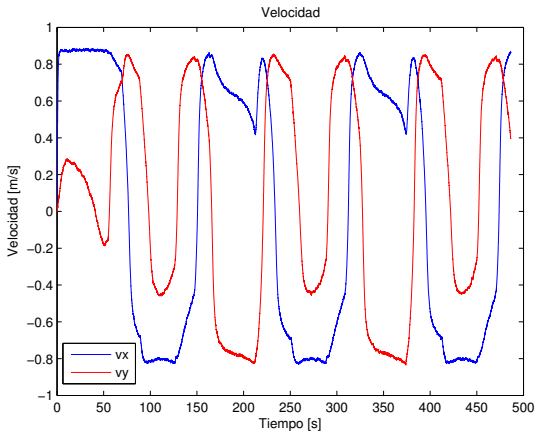


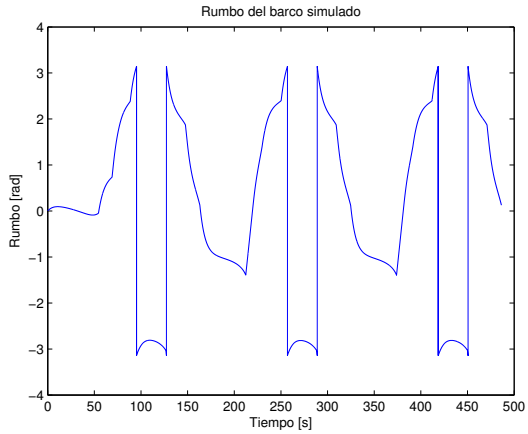
Figura 5.27: Rendimiento del control de rumbo durante el experimento de la figura 5.25b.



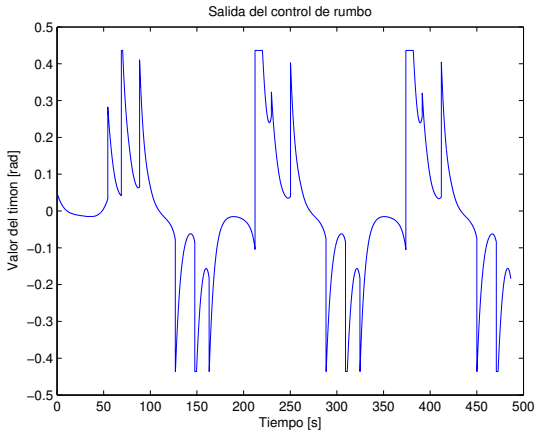
(a) Posición



(b) Velocidad



(c) Rumbo



(d) Timón

Figura 5.28: Resultados de la simulación. Las figuras muestran los distintos aspectos que se pueden simular.

5.9. Experimentos de coordinación – seguidor

El experimento es, conceptualmente, muy simple. Tampoco tiene mucha configuración posible. Cada barco puede seguir a cualquier otro. Un módulo de la estación de tierra permite activar o desactivar este comportamiento (ver figura 5.29).

Figura 5.29: Modulo para interactuar con el comportamiento *seguidor*.

Una vez que un barco tiene el seguidor activado, cada segundo actualiza su consigna del control *ir a un punto* con la posición del otro barco. Además, utiliza la velocidad del otro barco como consigna de su control de velocidad. Esto tiene el efecto de que si uno de los barcos se para, el otro también, evitando una colisión segura.

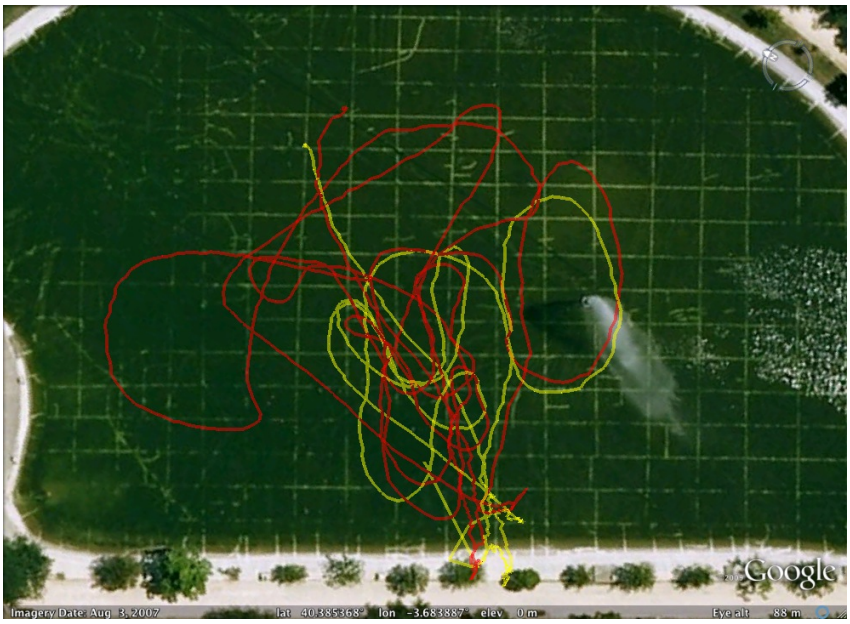
Se hicieron dos experimentos del seguidor, que pueden verse en las figuras 5.30a y 5.30b. En ambas, el barco amarillo sigue al barco rojo.

En el primer caso (5.30a) el barco rojo está en modo manual. Con un joystick se controla y se ve cómo el barco amarillo le sigue. Debido a la saturación del control de velocidad del barco automático, al principio están muy distanciados ya que el barco rojo iba muy rápido. Una vez se detectó el problema, se hizo que el barco rojo no fuera demasiado rápido y pudiera verse cómo el barco amarillo sigue perfectamente al barco rojo. La parte más clara es, quizá, la vuelta por detrás del chorro a la derecha de la imagen.

En el segundo caso (5.30b) el barco rojo está en automático, haciendo la trayectoria de los ochos. En un momento dado, se introduce el barco amarillo en el agua, y se activa el modo seguidor. Puede verse como sigue al barco rojo perfectamente.

En la figura 5.30b puede verse un bucle extraño descrito por el barco rojo. Este bucle es debido a que las saturaciones del control de rumbo deben ser distintas para ambos barcos debido a las diferencias mecánicas entre ellos. No obstante, el control hace todo lo que puede y, eventualmente, llega al punto consigna y sigue al siguiente.

En ambas figuras puede observarse como el barco sigue trayectorias siempre interiores a las descritas al barco seguido. Esto es porque la consigna del control *ir a un punto* es la posición actual del barco y no el camino que ha seguido de modo que en los giros siempre tiende a *recortar* distancias.



(a) Seguido en manual

(b) Seguido en automático

Figura 5.30: Experimentos de seguidor.

Capítulo 6

Conclusiones y trabajos futuros

Una vez se ha analizado el problema y, en base a prototipos, se ha llegado a un modelo final, se realizan los experimentos. El capítulo 5 dedica todo su contenido al análisis e interpretación de los datos obtenidos en los experimentos. Las conclusiones de dichos datos también se presentan en ese capítulo. El presente capítulo se centra en las conclusiones del propio proyecto, cómo se han abordado los problemas presentados en el capítulo 1 y qué resultados se han obtenido.

Copiando del capítulo 1 los dos grandes objetivos del proyecto son:

Construir un sistema de telemetría y control remoto para un barco Debe diseñarse y construirse el software tanto de a bordo como de tierra para poder monitorizar el barco. Además, debe poder controlarse el barco de manera remota utilizando un joystick. Será necesario diseñar e implementar el protocolo de comunicación así como el driver de un enlace de radio. El enlace de radio sirve para comunicar el barco con la estación de tierra.

Sistema de control autónomo El sistema debe permitir que el barco tome sus propias decisiones de cara a misiones y situaciones simples. El barco debe ser capaz de seguir un rumbo, mantener una cierta velocidad y ser capaz de ir a un punto dado. Todos estos comportamientos y sistemas deben poderse activar, desactivar, monitorizar y configurar desde el sistema de telemetría y control remoto (objetivo 1).

El sistema de telemetría y control remoto ha sido ampliamente descrito en los capítulos 2 y 3. Las capacidades han sido demostradas a lo largo del capítulo 5 donde todos y cada uno de los módulos construidos y utilizados se han presentado y probado. El sistema es capaz de hacer las tareas de adquisición y registro de datos. Esos datos pueden ser recuperados para posterior estudio y análisis.

Con respecto al sistema de control autónomo, el capítulo 4 ha presentado las bases del control en lazo cerrado así como un modelo físico del barco. El capítulo 5 presenta los resultados de los controladores diseñados.

El capítulo introductorio planteaba unos problemas que necesitaban solución, en concreto, tres para cada uno de los dos grandes objetivos. Estos eran:

- Definición del protocolo de comunicaciones.

Se ha diseñado, implementado y probado un protocolo de comunicaciones muy simple donde los distintos elementos se intercambian tipos de datos conocidos y con un formato cerrado.

- Formato de los datos en los ficheros de registro.

  Dos distintos tipos de formatos se utilizan. Uno se corresponde casi fielmente al protocolo de comunicaciones. Esto permite guardar muchos datos con poco esfuerzo computacional en el sistema empotrado.

  Por otro lado, los datos son convertidos a un formato que utiliza texto ASCII y marca cada dato con un reloj preciso de modo que sean fáciles de analizar. Este formato permite interactuar con software como Matlab que ayudará en el prototipado de filtros y funciones de control.

- Diseño del software multihilo.

  Se ha presentado y desarrollado un diseño multihilo utilizando el patrón *Observer* como modelo arquitectónico. El sistema es capaz de manejar los datos recibidos por los distintos barcos en tiempo real sin ningún problema.

- Control en lazo cerrado.

  Se han resumido los fundamentos prácticos del control en lazo cerrado así como de los controladores más simples de implementar (PID).

- Planteamiento de un modelo físico y desarrollo de un simulador que permita prototipar controladores y misiones avanzadas.

  El capítulo 4 presenta un modelo físico simple que permite hacer simulaciones. El modelo ha sido refinado en base a experimentos y se ha evaluado frente a datos reales.

- Implementación de un controlador PID y el ajuste del mismo.

  Utilizando controladores PID se consigue que el barco sea capaz de seguir un cierto rumbo así como una cierta velocidad con respecto al agua en la que se mueve. Los controladores implementados son fáciles de entender, ajustar e interpretar.

  Se han implementado mejoras interesantes en los controladores como la aceleración progresiva o la limitación de la actuación mediante la saturación de las señales de salida. Estas mejoras o variantes apenas incrementan la complejidad de los controladores.

- Interactuar (activar, desactivar y configurar) los distintos subsistemas del barco desde la estación de tierra.

  Utilizando una interfaz de usuario simple se puede interactuar con los distintos sistemas del barco. Cada cierto tiempo, el barco envía un paquete de información que incluye el estado de los distintos subsistemas así como sus parámetros. Esto permite al usuario obtener *feedback* casi instantáneo sobre los cambios que hace sobre el barco.

Todos ellos son abordados y descritos con distintos niveles de detalle en esta memoria. Para todos y cada uno de los problemas se ha presentado una solución. No necesariamente la mejor solución, ni la óptima, ni la más novedosa. Se han utilizado las soluciones que eran más viables, más seguras y que ofrecían mejor ratio tiempo / resultado.

Como líneas futuras se plantean algunas ideas desde distintos puntos de vista. Por el lado de la cooperación de robots y sistemas autónomos se plantea un escenario con barcos autónomos para la limpieza de vertidos [Dominguez04]. Otros escenarios incluyen la cooperación entre vehículos aéreos y marinos.

Desde el punto de vista del control automático es posible utilizar distintos controladores basados en técnicas de Inteligencia Artificial. Por ejemplo, se puede utilizar lógica borrosa [GarciaDeMarina10] o redes neuronales. El objetivo sería hacer un control no lineal que permitiera al barco girar mejor a bajas velocidades y más suavemente a altas velocidades.

Desde el punto de vista de trayectorias, [GarciaDeMarina10] presenta una solución utilizada en navegación aérea que puede ser utilizada en barcos.

Durante la realización del proyecto se han escrito varios artículos de congreso relacionados con el mismo. Dos de ellos plantean el desarrollo de un barco autónomo para utilizarlo en tareas de desminado [Pereda10a, Pereda10b] mientras que el tercero, presenta un control utilizando lógica borrosa y una planificación de trayectorias para barcos autónomos [GarciaDeMarina10].

Capítulo 7

Gestión de proyecto

El ciclo de vida del proyecto ha estado basado en la construcción de distintos prototipos. Cada uno de ellos incrementando las capacidades y complejidad del sistema y eliminando los errores encontrados en cada uno de los prototipos desarrollados.

7.1. Planificación

El primer paso a la hora de hacer la planificación ha sido plantear la subdivisión en tareas o Work Breakdown Structure (WBS):

1) Comunicaciones
    a) Tierra - Barco
    b) Barco - Barco
2) Telemetría
    a) Diseño multihilo
    b) Registro de datos
        i) A bordo
        ii) Tierra
    c) Visualización de datos
        i) Gráficas
        ii) Texto
        iii) Google Earth
    d) Repetición de experimentos
3) Control
    a) Remoto (joystick)
    b) Subsistemas desde tierra
        i) Monitorizar

II) Configurar

c) Filtros

d) Control automático

I) Rumbo

a' Implementación

b' Ajuste

II) Velocidad

a' Implementación

b' Ajuste

III) Waypoints

IV) Trayectoria

V) Seguidor

e) Modelo

I) Definición

II) Experimentos

III) Simulación

IV) Ajuste

A cada una de las tareas se le asigna un número de horas y se representa todo en un diagrama GANTT. Dado que el diagrama sería muy grande se ha dividido en las tres grandes tareas: comunicaciones (figura 7.1), telemetría (figura 7.2) y control (figura 7.3).

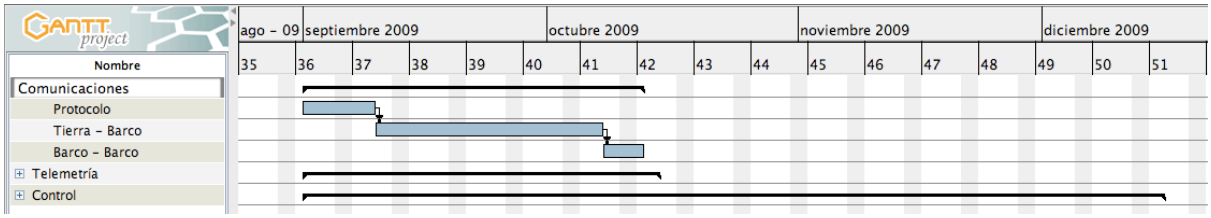


Figura 7.1: Planificación de la tarea de comunicaciones.

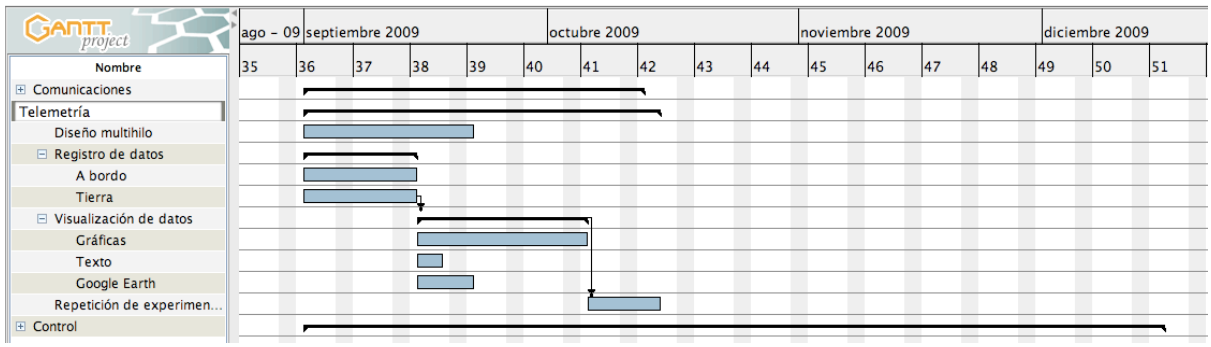


Figura 7.2: Planificación de la tarea del sistema de telemetría.

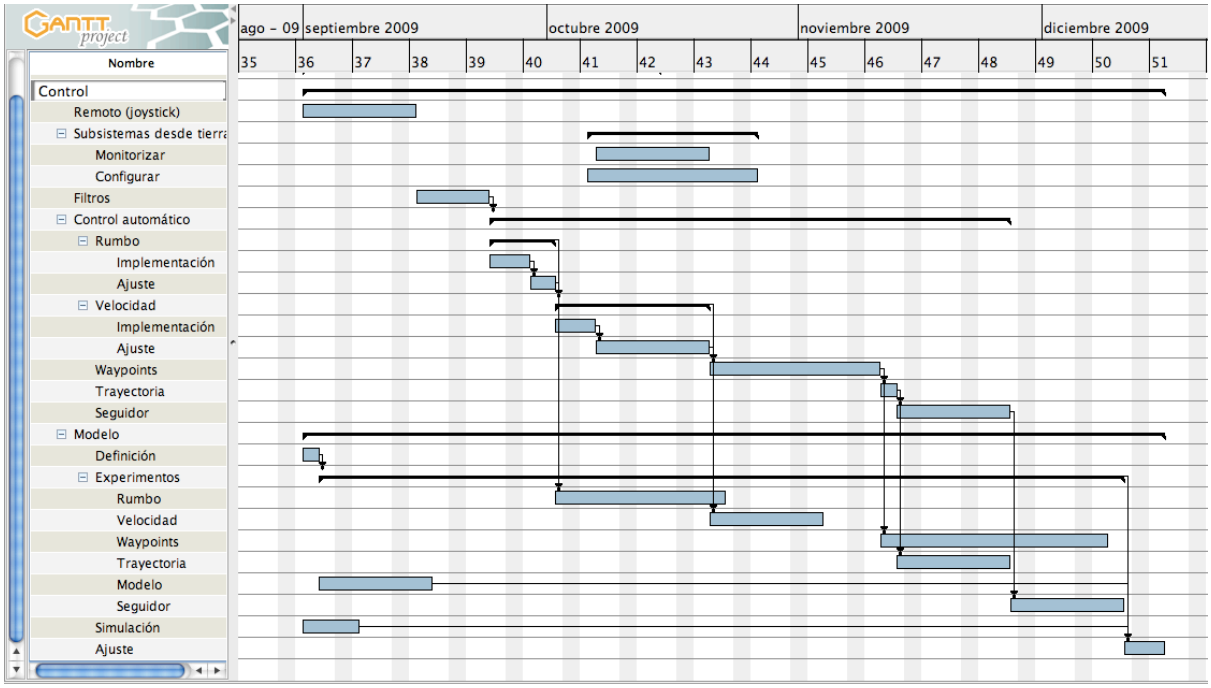


Figura 7.3: Planificación de la tarea del sistema de control.

La planificación da una estima de unos cuatro meses para terminar el proyecto. Sin embargo, la planificación no está particularizada en el caso de que una sola persona tenga que hacer todo el trabajo. En ese caso, no existe la superposición de tareas y la planificación se extendería hasta más o menos un año que es lo que ha durado el desarrollo del proyecto.

Contando con que la jornada laboral tiene 8 horas y la planificación se extiende 15 semanas. El número total de horas planificadas son 600 [1].

7.2. Presupuesto

Dado que la planificación estima unas 600 horas de trabajo, es necesario calcular el coste de personal asociado a este trabajo. La tabla 7.1 recoge este dato.

Concepto	Coste por hora	Horas	Total
Ingeniero	40 €	600	24000 €

Cuadro 7.1: Costes de personal.

Además de las horas de trabajo, se ha incurrido en costes relacionados con la construcción y adquisición de los elementos hardware. En este presupuesto no se cuentan los costes de construcción. Sin embargo, las tablas 7.2, 7.3 y 7.4 detallan los costes de hardware relacionados con cada uno de los elementos.

Se han construido dos barcos, dos copias de la electrónica y una estación de tierra.

[1] $15 \cdot 8 \cdot 5 = 600$

Componente	Precio (Unidad)
Casco	199 €
Motor	185 €
Variador	30 €
Servo	5 €
Total	419 €

Cuadro 7.2: Costes del barco.

Componente	Precio (Unidad)
Microprocesador	46 €
Radio	40 €
Sensor de Presión	10 €
GPS	55 €
SD	15 €
Brújula	20 €
Total	186 €

Cuadro 7.3: Costes del hardware.

Componente	Precio (Unidad)
Joystick	10 €
Radio	40 €
Total	50 €

Cuadro 7.4: Costes de la estación de tierra.

El coste en licencias de software es de 0 €ya que se ha realizado íntegramente con software abierto y gratuito.

Las baterías de los distintos sistemas se engloban en un mismo paquete al que se le asigna un presupuesto de 200 €para todo el proyecto. Además, hay que contar en que existen materiales que no son fáciles de computar como pueden ser tornillos, cinta, cajas estancas, herramientas, etc. a los que se asigna un 1 % del presupuesto.

La tabla 7.5 resume el presupuesto y muestra la cantidad final.

Concepto	Precio unitario	Unidades	Total
Barco	419 €	2	838 €
Electrónica	186 €	2	372 €
Estación de tierra	50 €	1	50 €
Baterías	200 €	1	200 €
Trabajo ingeniero	40 €	600	24000 €
Licencias software	0 €	0	0 €
			25460 €
Otros materiales		1 %	254 €
			25714 €

Cuadro 7.5: Resumen del presupuesto

Al proyecto se le estima un presupuesto de *Veinticinco mil setecientos catorce euros* (25714 €).

Apéndice A

Acrónimos y definiciones

ADC Conversor Analógico Digital

AMSV Autonomous Marine Surface Vehicle

DMA Direct Memory Access

ECEF Earth Centered Earth Fixed

ENU East-North-Up

GCC GNU Compiler Collection

GNU GNU is Not Unix

GPS Global Positioning System

KML Keyhole Markup Language

IEEE Institute of Electrical and Electronics Engineers

NED North-East-Down

PID Proportional Integral Derivative

PWM Modulación por Ancho de Pulsos

SD Tarjeta Secure Digital

SPI Serial Peripheral Interface Bus

TR1 Technical Report 1

TSIP Trimble Standard Interface Protocol

TWI Two Wire Interface

USB Universal Serial Bus

WBS Work Breakdown Structure

Glosario

deriva Separación o desplazamiento de la derrota del barco, producida por la corriente. 35

derrota Trayectoria descrita por una embarcación. 35, 83

waypoint Punto consigna al que debe ir el barco. Si se usa una lista de estos, el barco sigue una cierta trayectoria. 33–35, 37, 59, 61, 64

Bibliografía

[GoF94] Erich Gamma et al., *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional, 1994.

[Li06] Li, Y. and Ang, K.H. and Chong, G.C.Y, *PID control system analysis and design*, IEEE Control Systems Magazine 26(1):pp. 32-41., 2006, (`http://eprints.gla.ac.uk/3815/`).

[Bennet93] Stuart Bennett, *A history of control engineering, 1930-1955.*, IET, 1993, ISBN 9-780863412998.

[Farinelli04] Farinelli, A., L. Iocchi and D. Nardi, *Multirobot Systems: A Classification Focused on Coordination*, IEEE Trans. Systems, Man and Cybernetics, PartB, vol. 34, n. 5, October, p. 2015-2027, 2004.

[Alves06] J. Alves, P. Oliveira, R. Oliveira, A. Pascoal, M. Rufino, L. Sebastião, and C. Silvestre, *Vehicle and mission control of the DELFIM autonomous surface craft*, In Proc. MED2006 – *14th Mediterranean Conference of Control and Automation*, Ancona, Italy, 2006.

[Szarejko99] A. Szarejko and J. Namiesnik, *The Baltic Sea as a dumping site of chemical munitions and chemical warfare agents*, Chemistry and Ecology, vol. 25, pp. 13-26, 2009.

[Choset01] H. Choset, *Coverage for robotics – a survey of recent results*, Annals of Mathematics and Artificial Intelligence, vol. 31, pp. 113-126, 2001.

[Choset03] H. Choset, Y. Zhang, E. Acar and M. Schervish, *Path planning for robotic demining: robust sensor-based coverage of unstructured environments and probabilistic methods*, Intl. J. Robotics Research, vol. 22, n. 7-8, pp. 441-466, 2003.

[Jung09] Y. S. Jung, K. W. Lee, S. Y. Lee, M. H. Choi and B.H. Lee, *An efficient underwater coverage method for multi-AUV with sea current disturbances*, Intl. J. Control, Automation and Systems, vol. 7, n. 4, pp. 615-629, 2009.

[Bertram08] V. Bertram, *Unmanned surface vehicles – a survey*, In Skibsteknisk Selskab, Copenhagen, Denmark, 2008.

[CMWA01] Committee for Mine Warfare Assessment, *Naval Mine Warfare: Operational and Technical Challenges for Naval Forces*, National Academies Press, ISBN 978-0309076852, 2001.

[Esteban00] Esteban, S., J.M. De la Cruz, J.M. Giron-Sierra, B. De Andres, J.M. Diaz and J. Aranda, *Fast ferry vertical acceleration reduction with active flaps and T-foil*, In Proc. IFAC Intl. Symposium Maneuvering and Control of Marine Craft MCMC2000, Aalborg, pp. 233-238, 2000.

[Esteban02] Esteban, S., B. Andres-Toro, E. Besada-Portas, J.M. Giron-Sierra and J.M. De la Cruz, *Multiobjective control of flaps and T-foil in high speed ships*, In Proc. IFAC 2002 World Congress, Barcelona, 2002.

[Giron-Sierra01] Giron-Sierra, J.M., S. Esteban, B. De Andres, J.M. Diaz and J.M. Riola, *Experimental study of controlled flaps and T-foil for comfort improvement of a fast ferry*, In Proc. IFAC Intl. Conf. Control Applications in Marine Systems CAMS2001, Glasgow, 2001.

[Giron-Sierra02] Giron-Sierra, J.M., R. Katebi, J.M. De la Cruz, and S. Esteban, *The control of specific actuators for fast ferry vertical motions damping*, In Proc. IEEE Intl. Conf. CCA/CACSD, Glasgow, 2002.

[Haywood95] Haywood, A.J., A.J. Duncan, K.P. Klaka and J. Bennet, *The development of a ride control system for fast ferries*, Control Engineering Practice, pp. 695-703, 1995.

[Ryle98] Ryle, M., *Smoothing out the ride*, The Motor Ship, January, pp. 23-26, 1998.

[Johansen03] Johansen, V., R. Skjetne and A.J. Sorensen, *Maneuvering of towed interconnected marine systems*, In Proceedings IFAC MCMC 2003, Gerona, Spain, pp. 292-297.

[Soetanto03] Soetanto, D., L. Lapierre and A. Pascoal, *Coordinated motion control of marine robots*, In Proceedings IFAC MCMC 2003, Gerona, Spain, pp. 250-255.

[Stillwell00] Stilwell, D.J. and B.E. Bishop, *Platoons of underwater vehicles*, IEEE Control Systems Magazine, vol. 20, December 2000, pp. 45-52.

[Lloyd98] Lloyd, A.R.J.M, *Seakeeping: Ship Behaviour in Rough Weather*, A.R.M.J. Lloid, Gosport, Hampshire, U.K, 1998.

[Fossen02] Fossen, T.I., *Marine Control Systems*, Marine Cybernetics AS, Trondheim, 2002.

[Dominguez04] Alfonso Domínguez Pérez, *Diseño de un entorno de simulación para maniobras en cooperación de barcos: aplicación del despliegue de barreras de contención rápida*, Proyecto Fin de Carrera, Ingeniería Electrónica, Universidad Complutense de Madrid, 2004.

[Pereda10a] Fernando J. Pereda, Hector García de Marina, Juan Francisco Jiménez, Jose M. Girón-Sierra, *Sea demining with autonomous marine surface vehicles*, Accepted for IEEE SSRR 2010, Bremen, Germany, 2010.

[Pereda10b] Fernando J. Pereda, Hector García de Marina, Juan Francisco Jiménez, Jose M. Girón-Sierra, *A Development Project of Autonomous Marine Surface Vehicles for Sea Demining*, Accepted for ICARCV 2010, Singapore, 2010.

[GarciaDeMarina10] Hector García de Marina, Fernando J. Pereda, Jose M. Girón-Sierra, *Path planning combined with fuzzy control for autonomous ships*, Accepted for CIMSA 2010, Taranto, Italy, 2010.