

UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA



UNIVERSIDAD COMPLUTENSE DE MADRID

FACULTAD DE INFORMÁTICA

Proyecto de Fin de Máster

Máster en Ingeniería de Sistemas y de Control

MODELADO Y CONTROL DE UN CUADRICÓPTERO AR.DRONE

NICOLÁS MONTEAGUDO DURO

Dirigido por DICTINO CHAOS GARCÍA

Curso 2015-2016 - Convocatoria de Septiembre





UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

UNIVERSIDAD COMPLUTENSE DE MADRID

FACULTAD DE INFORMÁTICA

Proyecto de Fin de Máster

Máster en Ingeniería de Sistemas y de Control

MODELADO Y CONTROL DE UN CUADRICÓPTERO AR.DRONE

NICOLÁS MONTEAGUDO DURO

Dirigido por DICTINO CHAOS GARCÍA

Proyecto de Tipo B - Proyecto específico propuesto por el alumno

AUTORIZACIÓN

Autorizo a la Universidad Complutense y a la UNED a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a su autor, tanto la memoria de este Trabajo Fin de Máster, como el código, la documentación y/o el prototipo desarrollado.

Firmado: Nicolás Monteagudo Duro



RESUMEN

El presente Proyecto de Fin del Máster en Ingeniería de Sistemas y de Control ha tenido el objetivo principal de conseguir que un vehículo aéreo no tripulado sea controlado desde un PC de forma autónoma en un entorno cerrado en el que se determina su posición y orientación a través de un sistema de visión compuesto de una cámara instalada en el techo del lugar de experimentación.

Para conseguir esto ha sido necesario realizar un estudio cinemático y dinámico general de los cuadricópteros para entender su comportamiento, y estudiar específicamente el cuadricóptero AR.Drone 2.0 de Parrot, con el que se harán los experimentos reales, para determinar las capacidades que tiene para cumplir con el objetivo de su control autónomo. Se ha creado un modelo simplificado de dicho cuadricóptero y se han diseñado los controladores PID adecuados que consiguen controlarlo para llevarlo a las posiciones de referencia definidas previamente.

La posición y orientación del cuadricóptero se han obtenido del sistema de visión externo apoyándose en un Filtro de Kalman para estimar su estado y que resulta muy útil durante el control autónomo, pero sobre todo cuando se pierden de forma temporal las medidas obtenidas a través del sistema de visión.

Se han creado diferentes estrategias de control con las que se ha demostrado el correcto funcionamiento del controlador diseñado con el sistema real:

- La estabilización en una posición del espacio con una orientación determinada.
- La navegación a través de posiciones previamente programadas, con o sin seguimiento (tracking) entre ellas.
- El seguimiento de trayectorias (tracking) definidas por ecuaciones paramétricas en función del tiempo.

Los resultados obtenidos han sido satisfactorios aunque se recomienda ampliar el lugar de experimentación para permitir vuelos más amplios y seguimientos de trayectorias más suaves, con lo que el error relativo se reduciría bastante. Igualmente se recomienda seguir una estrategia de identificación y control multivariable y no lineal para tener en cuenta dinámicas no modeladas en el modelo creado para el presente proyecto y así perfeccionar el sistema.

El desarrollo del proyecto ha servido, de forma transversal, para crear una plataforma de control para este cuadricóptero que permitirá a investigadores u otras personas interesadas en este campo crear nuevas estrategias de control autónomo o nuevos métodos de navegación que puedan ser validados de forma experimental.

El software de esta plataforma de control ha sido desarrollado en LabView ya que su programación orientada a objetos permite un desarrollo modular, intuitivo y claro del sistema y de los elementos necesarios para su correcto funcionamiento, que unido a las capacidades de conectarse con múltiples dispositivos lo hacen ideal para ampliar la plataforma en desarrollos futuros.

LISTA DE PALABRAS

ingeniería de control; comunicaciones inalámbricas; control remoto; cuadricópteros; vehículos autónomos; vehículos aéreos no tripulados; diseño de circuitos; control en tiempo real; diseño software; modelado cinemático; modelado dinámico; identificación de sistemas; control PID; visión por computador;

DECLARACIÓN JURADA DE AUTORÍA DEL TRABAJO CIENTÍFICO, PARA LA DEFENSA DEL TRABAJO FIN DE MÁSTER

Fecha: 08/09/2016

Quien suscribe:

Autor: **Nicolás Monteagudo Duro**

DNI: **44.510.181-Y**

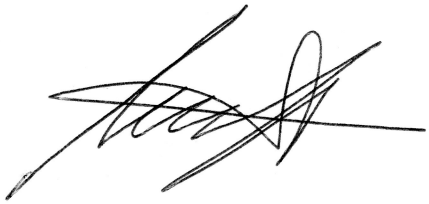
Hace constar que es el autor del trabajo:

Modelado y control de un cuadricóptero AR.Drone

En tal sentido, manifiesto la originalidad de la conceptualización del trabajo, interpretación de datos y la elaboración de las conclusiones, dejando establecido que aquellos aportes intelectuales de otros autores, se han referenciado debidamente en el texto de dicho trabajo.

DECLARACIÓN:

- ✓ Garantizo que el trabajo que remito es un documento original y no ha sido publicado, total ni parcialmente, en otra revista.
- ✓ Certifico que he contribuido directamente al contenido intelectual de este manuscrito, a la génesis y análisis de sus datos, por lo cual estoy en condiciones de hacerme públicamente responsable de él.
- ✓ No he incurrido en fraude científico, plagio o vicios de autoría; en caso contrario, aceptaré las medidas disciplinarias sancionadoras que correspondan.



Firmado: Nicolás Monteagudo Duro

ÍNDICE

AGRADECIMIENTOS **12**
LISTA DE SÍMBOLOS **13**
LISTA DE FIGURAS **22**
LISTA DE TABLAS **27**
1. INTRODUCCIÓN **31**
2. ANTECEDENTES **33**
3. MODELO DE UN CUADRICÓPTERO **34**
3.1. MODELADO CINEMÁTICO 36
3.2. MODELADO DINÁMICO 40
4. DESCRIPCIÓN DEL PARROT AR.DRONE 2.0 ..... **49**
4.1. GENERALIDADES 49
4.1.1. ESTRUCTURA 49
4.1.2. BATERÍA 50
4.1.3. CARACTERÍSTICAS DE VUELO 50
4.1.4. MOTORES 50
4.1.5. PLACAS ELECTRÓNICAS 50
4.1.6. SENSORES 50
4.1.7. COMUNICACIONES 52
4.1.8. ACCESO AL SISTEMA OPERATIVO 52
4.2. ARQUITECTURA DE CONTROL 53
4.2.1. CONTROL EXTERNO 53
4.2.2. CONTROL INTERNO 54
4.3. DESARROLLO DE APLICACIONES 56
4.3.1. OFICIAL 56
4.3.2. NO OFICIAL 56
4.4. COMANDOS 56
4.5. PARÁMETROS DE CONFIGURACIÓN, INFORMACIÓN DE ESTADO Y DATOS DE NAVEGACIÓN 57
5. ENTORNO DE EXPERIMENTACIÓN Y ARQUITECTURA DEL SISTEMA **59**
5.1. LUGAR DE EXPERIMENTACIÓN 59
5.2. ARQUITECTURA DEL SISTEMA 60
5.2.1. CUADRICÓPTERO AR.DRONE 2.0 62
5.2.2. SISTEMA DE VISIÓN 64
5.2.3. EQUIPO INFORMÁTICO 65
6. DESARROLLO DE LA PLATAFORMA SOFTWARE **66**
6.1. PANEL PRINCIPAL 67
6.2. PANELES PARA LA REALIZACIÓN DE PRUEBAS PREVIAS AL USO O AL DESARROLLO DE NUEVAS FUNCIONALIDADES DE LA PLATAFORMA 72
6.2.1. PANEL DE PRUEBA DE LOS CONTROLES 72
6.2.2. PANEL DE PRUEBA DEL TECLADO 72

6.2.3. Panel de prueba de los joysticks 73
6.2.4. Panel de prueba de un controlador PID 74
6.2.5. Panel de prueba de los controladores PID para el cuadricóptero 74
6.2.6. Panel de prueba de trayectorias paramétricas 74
6.3. Estados del sistema 76
6.4. Bucles de ejecución interna 77
7. IDENTIFICACIÓN DEL MODELO 79
7.1. Modelo para el Pitch 80
7.2. Modelo para el Roll 86
7.3. Modelo para el Throttle 91
7.4. Modelo para el Yaw 94
7.5. Modelo de los sensores 96
7.5.1. Modelo de la cámara 97
7.5.2. Modelo del sensor de ultrasonidos del AR.Drone 2.0 99
7.5.3. Modelo de los giróscopos del AR.Drone 2.0 102
7.5.4. Modelo de los acelerómetros del AR.Drone 2.0 104
8. PROPUESTA DE CONTROL 108
8.1. Estimación del estado 108
8.1.1. Descripción del Filtro de Kalman 108
8.1.2. Implementación del Filtro de Kalman 109
8.2. Controlador PID 115
8.2.1. Descripción del control PID 115
8.2.2. Implementación del control PID 116
8.2.2.1. Controlador PID para el Pitch 117
8.2.2.2. Controlador PID para el Roll 118
8.2.2.3. Controlador PID para el Throttle 119
8.2.2.4. Controlador PID para el Yaw 120
8.2.3. Validación de los controladores en el marco de referencia $\boldsymbol{E}$ 120
9. EXPERIMENTOS Y RESULTADOS 124
9.1. Estabilización en un punto 124
9.1.1. Experimento de estabilización sin perturbaciones externas 124
9.1.2. Experimento de estabilización con perturbaciones externas 126
9.2. Navegación a través de Waypoints 127
9.2.1. Experimento sin rotación $\boldsymbol{\psi}$ 128
9.2.2. Experimento con rotación $\boldsymbol{\psi}$ 129
9.3. Seguimiento de trayectorias (tracking) 131
9.3.1. Seguimiento de trayectorias rectas a través de Waypoints 131
9.3.1.1. Experimento sin rotación $\boldsymbol{\psi}$ 131
9.3.1.2. Experimento con rotación $\boldsymbol{\psi}$ 133
9.3.2. Seguimiento de trayectorias paramétricas 135
9.3.2.1. Experimento de seguimiento de una trayectoria elíptica sin rotación $\boldsymbol{\psi}$ 135
9.3.2.2. Experimento de seguimiento de una trayectoria elíptica con rotación $\boldsymbol{\psi}$ 137
9.3.2.3. Experimento de seguimiento de una trayectoria en forma de ocho 139

10. CONCLUSIONES 142
LISTADO DE REFERENCIAS Y BIBLIOGRAFÍA 144
LISTADO DE SIGLAS, ABREVIATURAS Y ACRÓNIMOS 147
ANEXO A. SISTEMA DE VISIÓN 149
A.1. CAPTURA DE LA IMAGEN 149
A.2. EXTRACCIÓN DE PUNTOS DE LA IMAGEN CAPTURADA 149
A.3. CORRECCIONES A LA IMAGEN CAPTURADA 149
A.3.1. DISTORSIÓN 149
A.3.2. PERSPECTIVA 149
A.4. CÁLCULO DE LA POSICIÓN Y ORIENTACIÓN 151
A.5. CÁLCULO DE LAS VELOCIDADES 151
A.6. PROCEDIMIENTO DE CALIBRACIÓN 152
ANEXO B. DESARROLLO EN LABVIEW 153
B.1. DESCRIPCIÓN DE LOS BLOQUES DEL SISTEMA PRINCIPAL 153
B.1.1. BLOQUE PRINCIPAL 153
B.1.2. BLOQUES DE CONTROL MANUAL 154
B.1.3. BLOQUES DE CONTROL AUTOMÁTICO 159
B.1.4. BLOQUES DE LOGS 164
B.1.5. BLOQUES DE PRUEBAS 166
B.1.6. BLOQUES AUXILIARES 170
B.2. DESCRIPCIÓN DE LOS BLOQUES DEL SISTEMA DE VISIÓN 174
B.3. MODIFICACIONES REALIZADAS A LA LIBRERÍA AR.DRONE TOOLKIT 176
B.3.1. BLOQUES MODIFICADOS DEBIDO A INCORRECTA INICIALIZACIÓN Y CONFIGURACIÓN 177
B.3.2. BLOQUES MODIFICADOS DEBIDO A LA INCLUSIÓN DE NUEVOS NAVDATA 180
B.3.3. BLOQUES NUEVOS DEBIDO A INCORRECTA INICIALIZACIÓN Y CONFIGURACIÓN 182
B.3.4. BLOQUES NUEVOS DEBIDO A LA INCLUSIÓN DE FUNCIONALIDAD DE CALIBRACIÓN 184
B.3.5. BLOQUES NUEVOS DEBIDO A LA INCLUSIÓN DE NUEVOS NAVDATA 185
B.4. DESCRIPCIÓN DE LOS BLOQUES DE VARIABLES GLOBALES 186
B.4.1. VARIABLES GLOBALES DEL SISTEMA DE VISIÓN 186
B.4.2. VARIABLES GLOBALES DE LA LIBRERÍA AR DRONE TOOLKIT 186
B.4.3. VARIABLES GLOBALES GENERALES 186
B.4.4. VARIABLES GLOBALES DEL CONTROL AUTOMÁTICO 187
B.4.5. VARIABLES GLOBALES DE LOS DATOS DE NAVEGACIÓN 188
B.4.6. VARIABLES GLOBALES DE LOS DATOS DE LAS GRÁFICAS DE SEGUIMIENTO 189
ANEXO C. MANUAL DE USO BÁSICO DEL SOFTWARE 191
C.1. REQUISITOS PREVIOS 191
C.2. INSTALACIÓN DEL SOFTWARE 191
C.3. CONFIGURACIÓN DE LOS CONTROLES MANUALES 191
C.3.1. CONTROL MANUAL "TECLADO" 193
C.3.2. CONTROL MANUAL "JOYSTICK" 193
C.3.3. CONTROL MANUAL "JOYSTICK + TECLADO 1" 193
C.3.4. CONTROL MANUAL "JOYSTICK + TECLADO 2" 194

C.4. ARRANQUE Y APERTURA DE SESIÓN CON LA CÁMARA Y CON EL AR.DRONE 2.0 194
C.5. USO DE FUNCIONALIDADES PROPIAS DEL AR.DRONE 2.0 195
C.6. FUNCIONAMIENTO DEL MODO AUTOMÁTICO 195
C.7. PARÁMETROS DE CONFIGURACIÓN 196
C.8. FICHEROS DE LOG 196
ANEXO D. PARÁMETROS DE CONFIGURACIÓN, ESTADO Y NAVEGACIÓN DEL AR.DRONE 2.0 201
D.1. PARÁMETROS DE CONFIGURACIÓN 201
D.2. INFORMACIÓN DE ESTADO 204
D.3. DATOS DE NAVEGACIÓN 205

AGRADECIMIENTOS

En primer lugar quiero agradecer a mi director de proyecto, Dictino Chaos, la labor que ha realizado durante todo este tiempo dirigiendo el presente Proyecto Fin de Máster, resolviendo todas aquellas dudas que le he planteado y orientándome en las mejores estrategias a seguir. Pero lo que más le agradezco es la excelente labor como docente que ha demostrado en las asignaturas de este Máster a las que he tenido la suerte de asistir como alumno. Gracias a ellas ha conseguido aumentar más mi interés por el control autónomo de robots y las diferentes estrategias de control.

También me gustaría agradecer a David Moreno, que dirigió mi Proyecto Fin de Carrera de Ingeniero en Informática, por el apoyo técnico que me ha prestado durante la realización de ambos proyectos, pero más aún por el apoyo moral cuando he pasado por momentos complicados derivados de la falta de tiempo al tener que compaginar mi labor profesional con mi vida personal, y con la preparación de oposiciones y la realización de este proyecto.

Por otra parte quiero agradecer al Instituto de Educación Secundaria Azarquiel de Toledo que aceptara mi solicitud para usar su pabellón deportivo para hacer experimentos que permitieran una correcta identificación del modelo. Quiero agradecer en especial a Jesús García Novillo, su director, y a José Javier Lallana Moreno, uno de sus jefes de estudio adjunto, por la cálida bienvenida y por lo cómodo que me hicieron sentir durante la semana que pasé con ellos. No podré olvidarme de las tardes que pasé explicando este proyecto a profesores y alumnos, fue una gran experiencia.

A nivel personal quiero dar las gracias a mis padres y hermanas con quien siempre puedo contar y que me han dado su cariño en todo momento. Igualmente quiero agradecer a mis amigos que muchas veces me han servido de vía de escape para desconectar de la rutina. Pero sobre todo quiero dar las gracias a mi amigo Rubén porque prácticamente todos los días se ha interesado por mi bienestar, por mis estudios y por mis proyectos, y porque a pesar de la distancia y de algunos momentos difíciles ha conseguido que nunca me olvide de lo que significa la amistad.

He querido dejar para el final el agradecimiento más importante, ya que buena parte de los logros que he conseguido en los últimos once años se deben a mi mujer Ana. Ella me ha proporcionado la estabilidad y el apoyo necesarios para abordar todos y cada uno de los proyectos que me he propuesto. Sólo ella sabe el esfuerzo que me ha costado y todo el tiempo que he tenido que dedicar para conseguir el éxito en muchos de estos proyectos Por mucho que lo pueda intentar nunca podré llegar a compensarle por el tiempo que he dejado de estar con ella. No puedo más que dar las gracias por poder compartir la vida con ella.

LISTA DE SÍMBOLOS

Símbolo	Unidad	Descripción
a_{acc}	$[m\ s^{-2}]$	Medida de la aceleración lineal sobre un eje del cuadricóptero determinada por un acelerómetro.
a_{real}	$[m\ s^{-2}]$	Aceleración lineal real del cuadricóptero sobre un eje.
a_x	$[m\ s^{-2}]$	Medida de la aceleración lineal sobre el eje X_B del cuadricóptero determinada por su acelerómetro.
a_{x_real}	$[m\ s^{-2}]$	Aceleración lineal real sobre el eje X_B del cuadricóptero.
a_y	$[m\ s^{-2}]$	Medida de la aceleración lineal sobre el eje Y_B del cuadricóptero determinada por su acelerómetro.
a_{y_real}	$[m\ s^{-2}]$	Aceleración lineal real sobre el eje Y_B del cuadricóptero.
a_z	$[m\ s^{-2}]$	Medida de la aceleración lineal sobre el eje Z_B del cuadricóptero determinada por su acelerómetro.
a_{z_real}	$[m\ s^{-2}]$	Aceleración lineal real sobre el eje Z_B del cuadricóptero.
b	$[N]$	Factor de empuje aerodinámico.
b_{c_acc}	$[m\ s^{-2}]$	Offset constante de la medida de la aceleración lineal del cuadricóptero sobre un eje determinada por un acelerómetro.
b_{v_acc}	$[m\ s^{-2}]$	Offset variable de la medida de la aceleración lineal del cuadricóptero sobre un eje determinada por un acelerómetro.
b_{c_gyro}	$[rad\ s^{-1}]$	Offset constante de la medida de la velocidad angular del cuadricóptero respecto a un eje determinada por un giróscopo.
b_{v_gyro}	$[rad\ s^{-1}]$	Offset variable de la medida de la velocidad angular del cuadricóptero respecto a un eje determinada por un giróscopo.
b_{v_pitch}	$[rad\ s^{-1}]$	Offset variable de la medida de la velocidad angular del cuadricóptero respecto al eje Y_B determinada por su giróscopo.
b_{v_roll}	$[rad\ s^{-1}]$	Offset variable de la medida de la velocidad angular del cuadricóptero respecto al eje X_B determinada por su giróscopo.
b_{v_x}	$[m\ s^{-2}]$	Offset variable de la medida de la aceleración lineal del cuadricóptero sobre el eje X_B determinada por su acelerómetro.
b_{v_y}	$[m\ s^{-2}]$	Offset variable de la medida de la aceleración lineal del cuadricóptero sobre el eje Y_B determinada por su acelerómetro.
b_{v_yaw}	$[rad\ s^{-1}]$	Offset variable de la medida de la velocidad angular del cuadricóptero respecto al eje Z_B determinada por su giróscopo.
b_{v_z}	$[m\ s^{-2}]$	Offset variable de la medida de la aceleración lineal del cuadricóptero sobre el eje Z_B determinada por su acelerómetro.
d	-	Factor de resistencia aerodinámica.
$e(t)$	+	Error entre la referencia y la salida del sistema.
l	$[m]$	Distancia del centro de la hélice al centro del cuadricóptero.
g	$[m\ s^{-2}]$	Constante de la aceleración de la gravedad.
h_{real}	$[m]$	Medida real de la altura del cuadricóptero.
h_{us}	$[m]$	Medida de altura determinada por el sensor de ultrasonidos.
m	$[kg]$	Masa del cuadricóptero.
n_{acc}	$[m\ s^{-2}]$	Ruido de banda ancha de tipo Gaussiano de la medida de la aceleración lineal del cuadricóptero sobre un eje tomada por un acelerómetro.

Símbolo	Unidad	Descripción
n_{cam}	$[m]$	Ruido blanco de tipo Gaussiano de la medida de la cámara en el marco de referencia E.
n_{gyro}	$[rad\ s^{-1}]$	Ruido de banda ancha de tipo Gaussiano de la medida de la velocidad angular del cuadricóptero respecto a un eje tomada por un giróscopo.
n_{pitch}	$[rad\ s^{-1}]$	Ruido de banda ancha de tipo Gaussiano de la medida de la velocidad angular del cuadricóptero respecto al eje Y_B tomada por su giróscopo.
n_{roll}	$[rad\ s^{-1}]$	Ruido de banda ancha de tipo Gaussiano de la medida de la velocidad angular del cuadricóptero respecto al eje X_B tomada por su giróscopo.
n_{us}	$[m]$	Ruido blanco de tipo Gaussiano de la medida del sensor de ultrasonidos.
n_x	$[m\ s^{-2}]$	Ruido de banda ancha de tipo Gaussiano de la medida de aceleración lineal del cuadricóptero sobre el eje X_B tomada por su acelerómetro.
n_{xcam}	$[m]$	Ruido blanco de tipo Gaussiano de la medida de la componente x de la cámara en el marco de referencia E.
n_{ycam}	$[m]$	Ruido blanco de tipo Gaussiano de la medida de la componente y de la cámara en el marco de referencia E.
n_y	$[m\ s^{-2}]$	Ruido de banda ancha de tipo Gaussiano de la medida de aceleración lineal del cuadricóptero sobre el eje Y_B tomada por su acelerómetro.
n_{yaw}	$[rad\ s^{-1}]$	Ruido de banda ancha de tipo Gaussiano de la medida de la velocidad angular del cuadricóptero respecto al eje Z_B tomada por su giróscopo.
n_z	$[m\ s^{-2}]$	Ruido de banda ancha de tipo Gaussiano de la medida de aceleración lineal del cuadricóptero sobre el eje Z_B tomada por su acelerómetro.
$n_{\psi cam}$	$[rad]$	Ruido blanco de tipo Gaussiano de la medida de la orientación por parte de la cámara en el marco de referencia E.
o_{cam}	$[m]$	Offset de la medida de la cámara en el marco de referencia E.
o_{us}	$[m]$	Offset de la medida de la altura del sensor de ultrasonidos.
p	$[rad\ s^{-1}]$	Velocidad angular del cuadricóptero alrededor del eje X_B.
$\dot{p}$	$[rad\ s^{-2}]$	Aceleración angular del cuadricóptero alrededor del eje X_B.
p_{cam}	$[m]$	Posición de un punto determinada por la cámara en el marco de referencia E.
p_{real}	$[m]$	Posición real de un punto en el marco de referencia E.
q	$[rad\ s^{-1}]$	Velocidad angular del cuadricóptero alrededor del eje Y_B.
$\dot{q}$	$[rad\ s^{-2}]$	Aceleración angular del cuadricóptero alrededor del eje Y_B.
r	$[rad\ s^{-1}]$	Velocidad angular del cuadricóptero alrededor del eje Z_B.
$\dot{r}$	$[rad\ s^{-2}]$	Aceleración angular del cuadricóptero alrededor del eje Z_B.
$r(t)$	+	Referencia a seguir por el sistema.
t	$[s]$	Tiempo.
t_{trans}	$[s]$	Tiempo de transición de un punto al siguiente en la navegación a través de Waypoints.
u	$[m\ s^{-1}]$	Velocidad lineal del cuadricóptero en el eje X_B.
$\dot{u}$	$[m\ s^{-2}]$	Aceleración lineal del cuadricóptero en el eje X_B.
u_k	+	Vector de entradas de control en el instante k.
u_{pitch}	-	Entrada de control del Pitch para el AR.Drone 2.0.
u_{roll}	-	Entrada de control del Roll para el AR.Drone 2.0.
$u_{throttle}$	-	Entrada de control del Throttle para el AR.Drone 2.0.
u_{yaw}	-	Entrada de control del Yaw para el AR.Drone 2.0.

Símbolo	Unidad	Descripción
$u_{PID}(t)$	+	Acción de control aplicada al sistema por un controlador PID.
$u(t)$	+	Acción de control aplicada al sistema.
v	$[m\ s^{-1}]$	Velocidad lineal del cuadricóptero en el eje Y_B.
$\dot{v}$	$[m\ s^{-2}]$	Aceleración lineal del cuadricóptero en el eje Y_B.
v_k	+	Ruido aditivo en las medidas del sistema tomadas en el instante k.
v_x^B	$[m\ s^{-1}]$	Velocidad lineal del cuadricóptero en el eje X_B.
v_y^B	$[m\ s^{-1}]$	Velocidad lineal del cuadricóptero en el eje Y_B.
v_z^E	$[m\ s^{-1}]$	Velocidad lineal del cuadricóptero en el eje Z_B.
v_ψ^B	$[rad\ s^{-1}]$	Velocidad angular del cuadricóptero alrededor del eje Z_B.
v_ψ^E	$[rad\ s^{-1}]$	Velocidad angular del cuadricóptero alrededor del eje Z_E.
w	$[m\ s^{-1}]$	Velocidad lineal del cuadricóptero en el eje Z_B.
$\dot{w}$	$[m\ s^{-2}]$	Aceleración lineal del cuadricóptero en el eje Z_B.
w_k	+	Ruido aditivo del sistema en el instante k.
x	$[m]$	Coordenada de la posición en el eje X_E.
$\dot{x}$	$[m\ s^{-1}]$	Velocidad lineal en el eje X_E.
$\ddot{x}$	$[m\ s^{-2}]$	Aceleración lineal en el eje X_E.
x^B	-	Eje de coordenadas de x en el marco de referencia B.
x_{cam}	$[m]$	Coordenada x de la medida de un punto determinada por la cámara en el marco de referencia E.
x_k	+	Estado del sistema en el instante k.
x_{k+1}	+	Estado del sistema en el instante $k+1$.
$\hat{x}_{k\|k}$	+	Estimación del estado del sistema determinada por el Filtro de Kalman en el instante k.
$\hat{x}_{k+1\|k}$	+	Predicción del estado del sistema en el instante $k+1$ con la información disponible del instante k.
$\hat{x}_{k+1\|k+1}$	+	Estimación del estado del sistema resultado de la aplicación del Filtro de Kalman para el instante $k+1$.
x_{real}	$[m]$	Coordenada x real de un punto en el marco de referencia E.
y	$[m]$	Coordenada de la posición en el eje Y_E.
$\dot{y}$	$[m\ s^{-1}]$	Velocidad lineal en el eje Y_E.
$\ddot{y}$	$[m\ s^{-2}]$	Aceleración lineal en el eje Y_E.
y^B	-	Eje de coordenadas de y en el marco de referencia B.
y_{cam}	$[m]$	Coordenada y de la medida de un punto determinada por la cámara en el marco de referencia E.
y_{real}	$[m]$	Coordenada y real de un punto en el marco de referencia E.
$y(t)$	+	Salida del sistema.
z	$[m]$	Coordenada de la posición en el eje Z_E.
$\dot{z}$	$[m\ s^{-1}]$	Velocidad lineal en el eje Z_E.
$\ddot{z}$	$[m\ s^{-2}]$	Aceleración lineal en el eje Z_E.
z^B	-	Eje de coordenadas de z en el marco de referencia B.
z^E	-	Eje de coordenadas de z en el marco de referencia E.
z_{CD}	+	Vector de medidas del sistema cuando se dispone de las medidas de la cámara y del AR.Drone 2.0.

Símbolo	Unidad	Descripción
z_D	+	Vector de medidas del sistema cuando sólo se dispone de las medidas del AR.Drone 2.0.
z_k	+	Medidas del sistema tomadas en el instante k.
z_{k+1}	+	Medidas del sistema tomadas en el instante $k+1$.
ARW	$[rad\ s^{-0.5}]$	Variación aleatoria del ángulo en un giróscopo.
B	-	Marco de referencia fijo del cuadricóptero.
BS	+	Parámetro Bias Stability de un giróscopo o acelerómetro.
$\mathrm{C_B}(\nu)$	+	Matriz de aceleraciones centrífugas y de Coriolis en el marco de referencia B.
$\mathrm{C_H}(\zeta)$	+	Matriz de aceleraciones centrífugas y de Coriolis en el marco de referencia H.
E	-	Marco de referencia inercial de la Tierra. Cuando se está hablando del sistema real de experimentación se refiere al marco de referencia del lugar de experimentación.
E_B	-	Matriz de constantes que relaciona las velocidades de rotación de las hélices con el vector de entradas de control en el marco de referencia B.
$E_H(\xi)$	-	Matriz de constantes que relaciona las velocidades de rotación de las hélices con el vector de entradas de control en el marco de referencia E.
$E(s)$	+	Error entre la referencia y la salida del sistema en el dominio $\mathcal{S}$.
F^B	$[N]$	Vector de fuerzas a las que está sometido el cuadricóptero en el marco de referencia B.
F^E	$[N]$	Vector de fuerzas a las que está sometido el cuadricóptero en el marco de referencia E.
F_G^B	$[N]$	Vector de fuerzas gravitacionales a las que está sometido el cuadricóptero en el marco de referencia B.
F_G^E	$[N]$	Vector de fuerzas gravitacionales a las que está sometido el cuadricóptero en el marco de referencia E.
F_k	-	Matriz de transición de estados del sistema.
$F_{throttle}(z)$	-	Función de transferencia en el dominio $\mathcal{Z}$ entre la velocidad lineal v_z^E y la entrada de control $u_{throttle}$.
$F_{v_x^B_\theta}(z)$	-	Función de transferencia en el dominio $\mathcal{Z}$ entre la velocidad lineal v_x^B y el ángulo θ de Pitch.
$F_{v_y^B_\phi}(z)$	-	Función de transferencia en el dominio $\mathcal{Z}$ entre la velocidad lineal v_y^B y el ángulo ϕ de Roll.
F_x	$[N]$	Fuerza a la que está sometido el cuadricóptero en el eje X_B.
F_y	$[N]$	Fuerza a la que está sometido el cuadricóptero en el eje Y_B.
$F_{yaw}(z)$	-	Función de transferencia en el dominio $\mathcal{Z}$ entre la velocidad angular v_ψ^B y la entrada de control u_{yaw}.
F_z	$[N]$	Fuerza a la que está sometido el cuadricóptero en el eje Z_B.
$F_{\theta_u_{pitch}}(z)$	-	Función de transferencia en el dominio $\mathcal{Z}$ entre el ángulo θ de Pitch y la entrada de control u_{pitch}.
$F_{\phi_u_{roll}}(z)$	-	Función de transferencia en el dominio $\mathcal{Z}$ entre el ángulo ϕ de Roll y la entrada de control u_{roll}.
$\mathrm{G_B}(\xi)$	+	Vector general de fuerzas gravitacionales a las que está sometido el cuadricóptero en el marco de referencia B.

Símbolo	Unidad	Descripción
$\mathrm{G_H}$	+	Vector general de fuerzas gravitacionales a las que está sometido el cuadricóptero en el marco de referencia H.
G_k	-	Matriz de transición de entradas del sistema en el instante k.
$G_{PID}(s)$	-	Función de transferencia en el dominio $\mathcal{S}$ de un controlador PID.
H	-	Marco de referencia híbrido.
H_{CD}	-	Matriz de observaciones del sistema determinada cuando se dispone de las observaciones de la cámara y del AR.Drone 2.0.
H_D	-	Matriz de observaciones del sistema determinada cuando sólo se dispone de las observaciones del AR.Drone 2.0.
H_k	-	Matriz de observaciones del sistema en el instante k.
H_{k+1}	-	Matriz de observaciones del sistema en el instante $k+1$.
I	$[N\ m\ s^2]$	Matriz de inercias del cuadricóptero en el marco de referencia B.
I	-	Matriz identidad.
I_L	$[A]$	Intensidad que circula por cada led del triángulo.
I_{R1}	$[A]$	Intensidad que circula por la resistencia R_1.
I_{XX}	$[N\ m\ s^2]$	Momento de inercia angular del cuadricóptero respecto al eje X_B.
I_{YY}	$[N\ m\ s^2]$	Momento de inercia angular del cuadricóptero respecto al eje Y_B.
I_{ZZ}	$[N\ m\ s^2]$	Momento de inercia angular del cuadricóptero respecto al eje Z_B.
I_{3x3}	-	Matriz identidad de dimensión 3x3.
J_{TP}	$[N\ m\ s^2]$	Momento de inercia rotacional total respecto al eje de rotación de las hélices.
J_Θ	-	Matriz de transformación general.
K_d	-	Ganancia de la parte Derivativa de un controlador PID.
K_i	-	Ganancia de la parte Integral de un controlador PID.
K_{k+1}	-	Ganancia del Filtro de Kalman en el instante $k+1$.
K_p	-	Ganancia de la parte Proporcional de un controlador PID.
L_1	-	Led 1 del triángulo.
L_2	-	Led 2 del triángulo.
L_3	-	Led 3 del triángulo.
$\mathrm{M_B}$	$[N\ m\ s^2]$	Matriz de general de inercias del cuadricóptero en el marco de referencia B.
$\mathrm{M_H}$	$[N\ m\ s^2]$	Matriz de general de inercias del cuadricóptero en el marco de referencia H.
N	-	Constante del filtro Derivativo de un controlador PID.
O_B	$[m]$	Origen del marco de referencia B.
$O_B(v)$	+	Matriz de efectos giroscópicos de la rotación de las hélices en el marco de referencia B.
O_E	$[m]$	Origen del marco de referencia E.
$O_H(\zeta)$	+	Matriz de efectos giroscópicos de la rotación de las hélices en el marco de referencia H.
$\hat{P}_{k\|k}$	+	Estimación de la matriz de covarianzas del error cuadrático medio que el Filtro de Kalman calculó en el instante k.
$\hat{P}_{k+1\|k}$	+	Predicción de la matriz de covarianzas del error cuadrático medio de la predicción $\hat{x}_{k+1\|k}$ gracias a la información disponible en el instante k.
$\hat{P}_{k+1\|k+1}$	+	Estimación de la matriz de covarianzas del error cuadrático medio de la predicción del estado resultado de la aplicación del Filtro de Kalman para el instante k.

Símbolo	Unidad	Descripción
Q	+	Matriz de covarianza del ruido blanco del sistema.
Q_k	+	Matriz de covarianza del ruido blanco del sistema en el instante k.
R_{CD}	+	Matriz de covarianza del ruido blanco en las medidas tomadas del sistema cuando se dispone de las medidas de la cámara y del AR.Drone 2.0
R_D	+	Matriz de covarianza del ruido blanco en las medidas tomadas del sistema cuando sólo se dispone de las medidas del AR.Drone 2.0
R_k	+	Matriz de covarianza del ruido blanco en las medidas tomadas del sistema en el instante k.
R_{k+1}	+	Matriz de covarianza del ruido blanco en las medidas tomadas del sistema en el instante $k+1$.
R_Θ	-	Matriz de rotación (Roll-Pitch-Yaw).
$\dot{R}_\Theta$	-	Derivada de la Matriz de rotación R_Θ.
$R(\phi, X)$	-	Matriz de rotación alrededor del eje X_E.
$R(\theta, Y)$	-	Matriz de rotación alrededor del eje Y_E.
$R(\psi, Z)$	-	Matriz de rotación alrededor del eje Z_E.
R_1	$[\Omega]$	Resistencia para limitar la intensidad que circula a través del triángulo de leds.
S_1	-	Interruptor para encender o apagar el triángulo de leds.
$\mathcal{S}$	-	Dominio de la Transformada de Laplace.
T_d	[s]	Constante de tiempo de la parte Derivativa de un controlador PID.
T_s	[s]	Tiempo de muestreo.
T_Θ	-	Matriz de transformación.
$\dot{T}_\Theta$	-	Derivada de la matriz de transformación T_Θ.
$U_B(\Omega)$	+	Vector de entradas de control en el marco de referencia B.
U_1	$[N]$	Fuerza de empuje aplicada al cuadricóptero para generar un movimiento de Throttle.
U_2	$[N\ m]$	Par aplicado al cuadricóptero para generar un movimiento de Roll.
U_3	$[N\ m]$	Par aplicado al cuadricóptero para generar un movimiento de Pitch.
U_4	$[N\ m]$	Par aplicado al cuadricóptero para generar un movimiento de Yaw.
$U(s)$	+	Acción de control aplicada al sistema en el dominio $\mathcal{S}$.
V^B	$[m\ s^{-1}]$	Vector de velocidades lineales del cuadricóptero en el marco de referencia B.
$\dot{\mathrm{V}}^B$	$[m\ s^{-2}]$	Vector de aceleraciones lineales del cuadricóptero en el marco de referencia B.
V^E	$[m\ s^{-1}]$	Vector de velocidades lineales del cuadricóptero en el marco de referencia E.
V_L	$[V]$	Voltaje directo existente en cada led del triángulo.
V_{R1}	$[V]$	Voltaje existente en la resistencia R_1.
V_1	$[V]$	Voltaje de la batería del AR. Drone 2.0.
VRW	$[m\ s^{-1}]$	Variación aleatoria de la velocidad lineal medida por un acelerómetro.
X	$[m]$	Coordenada de la posición en el eje X_E.
X_B	$[m]$	Eje de coordenadas de la componente x del marco de referencia B.
X_E	$[m]$	Eje de coordenadas de la componente x del marco de referencia E.
Y	$[m]$	Coordenada de la posición en el eje Y_E.
Y_B	$[m]$	Eje de coordenadas de la componente x del marco de referencia B.
Y_E	$[m]$	Eje de coordenadas de la componente y del marco de referencia E.
Z	$[m]$	Coordenada de la posición en el eje Z_E.

Símbolo	Unidad	Descripción
Z_B	$[m]$	Eje de coordenadas de la componente x del marco de referencia B.
Z_E	$[m]$	Eje de coordenadas de la componente z del marco de referencia E.
$\mathcal{Z}$	-	Dominio de la Transformada $\mathcal{Z}$.
α	$[s^{-1}]$	Variable utilizada en la constante de tiempo del filtro derivativo de un controlador PID.
α_{cam}	-	Factor de escala de la medida de la cámara.
α_{us}	-	Factor de escala de la medida del sensor de ultrasonidos.
δt	$[s]$	Periodo de muestreo en el cálculo de la varianza de Allan.
ζ	+	Vector de velocidades general en el marco de referencia H.
θ	$[rad]$	Ángulo de Euler del Pitch.
$\ddot{\theta}$	$[rad\ s^{-2}]$	Aceleración del ángulo de Euler del Pitch.
ν	+	Vector de velocidad general en el marco de referencia B.
$\dot{\nu}$	+	Vector de aceleración general en el marco de referencia B.
ξ	+	Vector de posición general en el marco de referencia E.
$\dot{\xi}$	+	Vector de velocidad general en el marco de referencia E.
σ_{acc}	$[m\ s^{-2}]$	Desviación estándar del ruido de banda ancha existente en la medida de la aceleración lineal sobre un eje determinada por un acelerómetro.
σ_{cam}	+	Desviación estándar del ruido existente en la medida de la cámara.
σ_{gyro}	$[rad\ s^{-1}]$	Desviación estándar del ruido de banda ancha existente en la medida de la velocidad angular respecto a un eje determinada por un giróscopo.
σ_{pitch}	$[rad\ s^{-1}]$	Desviación estándar del ruido de banda ancha existente en la medida de la velocidad angular respecto al eje Y_B determinada por su giróscopo.
σ_{roll}	$[rad\ s^{-1}]$	Desviación estándar del ruido de banda ancha existente en la medida de la velocidad angular respecto al eje X_B determinada por su giróscopo.
σ_{us}	$[m]$	Desviación estándar del ruido existente en la medida del sensor de ultrasonidos.
σ_{vel}	$[m\ s^{-1}]$	Desviación estándar del ruido variable aleatorio existente en la integración de la medida de la aceleración lineal sobre un eje determinada por un acelerómetro.
σ_{vx}^{B}	$[m\ s^{-1}]$	Desviación estándar del ruido en la medida de la velocidad lineal sobre el eje X_B determinada por el AR.Drone 2.0.
σ_{vx}^{E}	$[m\ s^{-1}]$	Desviación estándar del ruido en la medida de la velocidad lineal sobre el eje X_E determinada por la cámara.
σ_{vy}^{B}	$[m\ s^{-1}]$	Desviación estándar del ruido en la medida de la velocidad lineal sobre el eje Y_B determinada por el AR.Drone 2.0.
σ_{vy}^{E}	$[m\ s^{-1}]$	Desviación estándar del ruido en la medida de la velocidad lineal sobre el eje Y_E determinada por la cámara.
σ_{vz}^{B}	$[m\ s^{-1}]$	Desviación estándar del ruido en la medida de la velocidad lineal sobre el eje Z_B determinada por el AR.Drone 2.0.
σ_{vz}^{E}	$[m\ s^{-1}]$	Desviación estándar del ruido en la medida de la velocidad lineal sobre el eje Z_E determinada por la cámara.
σ_{v_acc}	$[m\ s^{-2}]$	Desviación estándar del ruido de paseo aleatorio existente en la medida de la aceleración lineal sobre un eje determinada por un acelerómetro.

Símbolo	Unidad	Descripción
σ_{v_gyro}	$[rad\ s^{-1}]$	Desviación estándar del ruido de paseo aleatorio existente en la medida de la velocidad angular respecto a un eje determinada por un giróscopo.
σ_{v_pitch}	$[rad\ s^{-1}]$	Desviación estándar del ruido de paseo aleatorio existente en la medida de la velocidad angular respecto al eje Y_B determinada por su giróscopo.
σ_{v_roll}	$[rad\ s^{-1}]$	Desviación estándar del ruido de paseo aleatorio existente en la medida de la velocidad angular respecto al eje X_B determinada por su giróscopo.
σ_{v_x}	$[m\ s^{-2}]$	Desviación estándar del ruido de paseo aleatorio existente en la medida de la aceleración lineal sobre el eje X_B determinada por su acelerómetro.
σ_{v_y}	$[m\ s^{-2}]$	Desviación estándar del ruido de paseo aleatorio existente en la medida de la aceleración lineal sobre el eje Y_B determinada por su acelerómetro.
σ_{v_yaw}	$[rad\ s^{-1}]$	Desviación estándar del ruido de paseo aleatorio existente en la medida de la velocidad angular respecto al eje Z_B determinada por su giróscopo.
σ_{v_z}	$[m\ s^{-2}]$	Desviación estándar del ruido de paseo aleatorio existente en la medida de la aceleración lineal sobre el eje Z_B determinada por su acelerómetro.
$\sigma_{v\psi}^{E}$	$[rad\ s^{-1}]$	Desviación estándar del ruido en la medida de la velocidad angular respecto al eje Z_E determinada por la cámara.
σ_x	$[m\ s^{-2}]$	Desviación estándar del ruido de banda ancha existente en la medida de la aceleración lineal sobre el eje X_B determinada por su acelerómetro.
σ_{xcam}	$[m]$	Desviación estándar del ruido existente en la medida de la componente x por parte de la cámara.
σ_y	$[m\ s^{-2}]$	Desviación estándar del ruido de banda ancha existente en la medida de la aceleración lineal sobre el eje Y_B determinada por su acelerómetro.
σ_{yaw}	$[rad\ s^{-1}]$	Desviación estándar del ruido de banda ancha existente en la medida de la velocidad angular respecto al eje Z_B determinada por su giróscopo.
σ_{ycam}	$[m]$	Desviación estándar del ruido existente en la medida de la componente y por parte de la cámara.
σ_z	$[m\ s^{-2}]$	Desviación estándar del ruido de banda ancha existente en la medida de la aceleración lineal sobre el eje Z_B determinada por su acelerómetro.
σ_θ	$[rad\ s^{-1}]$	Desviación estándar del ruido variable aleatorio existente en la integración de la medida de la velocidad angular respecto a un eje determinada por un giróscopo.
$\sigma_{\psi cam}$	$[rad]$	Desviación estándar del ruido existente en la medida de la orientación por parte de la cámara.
τ	$[s]$	Tiempo medio del cálculo de la varianza de Allan.
τ^B	$[N\ m]$	Vector de pares inducidos en el cuadricóptero en el marco de referencia B.
τ^E	$[N\ m]$	Vector de pares inducidos en el cuadricóptero en el marco de referencia E.
τ_{x}	$[N\ m]$	Par inducido en el cuadricóptero alrededor del eje X_B.
τ_{y}	$[N\ m]$	Par inducido en el cuadricóptero alrededor del eje Y_B.
τ_{z}	$[N\ m]$	Par inducido en el cuadricóptero alrededor del eje Z_B.
ϕ	$[rad]$	Ángulo de Euler del Roll.
$\ddot{\phi}$	$[rad\ s^{-2}]$	Aceleración del ángulo de Euler del Roll.
ψ	$[rad]$	Ángulo de Euler del Yaw.
$\dot{\psi}$	$[rad\ s^{-1}]$	Velocidad del ángulo de Euler del Yaw.
$\ddot{\psi}$	$[rad\ s^{-2}]$	Aceleración del ángulo de Euler del Yaw.

Símbolo	Unidad	Descripción
ψ_{cam}	$[rad]$	Orientación del triángulo de leds medida por la cámara en el marco de referencia E.
ψ_m	$[rad]$	Orientación modificada en función de la última medida de orientación de la cámara y de las medidas siguientes de la velocidad v_{ψ}^{B} del cuadricóptero.
ψ_{real}	$[rad]$	Orientación real del triángulo de leds en el marco de referencia E.
ω^B	$[rad\ s^{-1}]$	Vector de velocidades angulares del cuadricóptero en el marco de referencia B.
$\dot{\omega}^B$	$[rad\ s^{-2}]$	Vector de aceleraciones angulares del cuadricóptero en el marco de referencia B.
ω_{gyro}	$[rad\ s^{-1}]$	Medida de la velocidad angular respecto a un eje del cuadricóptero determinada por un giróscopo.
ω_{pitch}	$[rad\ s^{-1}]$	Medida de la velocidad angular respecto al eje Y_B del cuadricóptero determinada por su giróscopo.
ω_{pitch_real}	$[rad\ s^{-1}]$	Velocidad angular real respecto al eje Y_B del cuadricóptero.
ω_{real}	$[rad\ s^{-1}]$	Velocidad angular real del cuadricóptero respecto a un eje.
ω_{roll}	$[rad\ s^{-1}]$	Medida de la velocidad angular respecto al eje X_B del cuadricóptero determinada por su giróscopo.
ω_{roll_real}	$[rad\ s^{-1}]$	Velocidad angular real respecto al eje X_B del cuadricóptero.
ω_{yaw}	$[rad\ s^{-1}]$	Medida de la velocidad angular respecto al eje Z_B del cuadricóptero determinada por su giróscopo.
ω_{yaw_real}	$[rad\ s^{-1}]$	Velocidad angular real respecto al eje Z_B del cuadricóptero.
Γ^E	$[m]$	Vector de posición lineal del cuadricóptero en el marco de referencia E.
$\ddot{\Gamma}^E$	$[m\ s^{-2}]$	Vector de aceleraciones lineales del cuadricóptero en el marco de referencia E.
Θ^E	$[rad]$	Vector de posición angular del cuadricóptero en el marco de referencia E.
$\dot{\Theta}^E$	$[rad\ s^{-1}]$	Vector de velocidades angulares del cuadricóptero en el marco de referencia E.
$\ddot{\Theta}^E$	$[rad^2]$	Vector de aceleraciones angulares del cuadricóptero en el marco de referencia E.
Λ	+	Vector de fuerzas general en el marco de referencia B.
Ω	$[rad\ s^{-1}]$	Vector de velocidades de rotación de las hélices.
Ω_{T}	$[rad\ s^{-1}]$	Suma de las velocidades de rotación de las hélices.
Ω_1	$[rad\ s^{-1}]$	Velocidad de rotación de la hélice 1.
Ω_2	$[rad\ s^{-1}]$	Velocidad de rotación de la hélice 2.
Ω_3	$[rad\ s^{-1}]$	Velocidad de rotación de la hélice 3.
Ω_4	$[rad\ s^{-1}]$	Velocidad de rotación de la hélice 4.
0_{3x1}	-	Vector vertical con cero en sus tres elementos.
0_{3x3}	-	Matriz de dimensión 3x3 con ceros en todos sus elementos.

LISTA DE FIGURAS

Figura 3.1: Tipos de configuración de un cuadricóptero..... 34
Figura 3.2: Configuración en +. Velocidad de los rotores en reposo. 35
Figura 3.3: Configuración en +. Velocidad de los rotores para variar la altura (Throttle)..... 35
Figura 3.4: Configuración en +. Velocidad de los rotores para variar la orientación (Yaw). 35
Figura 3.5: Configuración en +. Velocidad de los rotores para variar la posición en el Eje X (Pitch). 35
Figura 3.6: Configuración en +. Velocidad de los rotores para variar la posición en el Eje Y (Roll). 35
Figura 3.7: Configuración en x. Velocidad de los rotores en reposo..... 36
Figura 3.8: Configuración en x. Velocidad de los rotores para variar la altura (Throttle). 36
Figura 3.9: Configuración en x. Velocidad de los rotores para variar la orientación (Yaw). 36
Figura 3.10: Configuración en x. Velocidad de los rotores para variar la posición en el Eje X (Pitch). 36
Figura 3.11: Configuración en x. Velocidad de los rotores para variar la posición en el Eje Y (Roll)..... 36
Figura 3.12: Marcos de referencia de un cuadricóptero..... 37
Figura 3.13: Rotación de Yaw sobre el eje ZE 37
Figura 3.14: Rotación de Pitch sobre el eje $Y1$ 38
Figura 3.15: Rotación de Roll sobre el eje $X2$ 38
Figura 3.16: Velocidades y sentidos de rotación de las hélices. 43
Figura 4.1: Parrot AR.Drone 2.0 con el casco de interior. 49
Figura 4.2: Parrot AR.Drone 2.0 con el casco de exterior. 49
Figura 4.3: Fusión de sensores 52
Figura 4.4: Captura del acceso por telnet al sistema operativo BusyBox del Parrot AR.Drone 2.0. 53
Figura 4.5: Arquitectura de control interno y fusión de sensores 55
Figura 5.1: Lugar de experimentación. 60
Figura 5.2: Marco de referencia E definido en el lugar de experimentación. 61
Figura 5.3: Arquitectura del sistema. 61
Figura 5.4: Triángulo de LEDs instalados en el Parrot AR.Drone 2.0. 62
Figura 5.5: Detalle de la posición del triángulo de LEDs instalados. 62
Figura 5.6: Esquema eléctrico del triángulo de LEDs. 63
Figura 5.7: Placa de conexión que contiene el interruptor y el potenciómetro. 63
Figura 5.8: Cámara y óptica montada 65
Figura 6.1: Panel principal. 68
Figura 6.2: Pestaña de Datos del panel principal. 68
Figura 6.3: Pestaña de Navegación del panel principal..... 69
Figura 6.4: Parámetros de navegación a través de Waypoints. 70
Figura 6.5 Parámetros de navegación a través de Waypoints con seguimiento de la recta entre ellos..... 70

Figura 6.6: Parámetros de seguimiento de trayectorias paramétricas. 71
Figura 6.7: Pestaña de Video del panel principal. 71
Figura 6.8 Pestaña de Configuración del panel principal. 71
Figura 6.9 Pestaña de Configuración del AR.Drone 2.0 del panel principal. 72
Figura 6.10: Panel para la realización de las pruebas de los controles. 73
Figura 6.11: Panel para la obtención de valores de teclas pulsadas. 73
Figura 6.12: Panel para conocer qué botones y ejes se corresponden con los del joystick. 73
Figura 6.13: Panel de prueba de un controlador PID. 74
Figura 6.14: Panel de prueba de los controladores PID del cuadricóptero. 75
Figura 6.15: Panel para realizar pruebas de las trayectorias paramétricas. 75
Figura 7.1: Respuesta al escalón en la entrada $upitch = -0.5$. 80
Figura 7.2: Régimen estacionario del ángulo θ ante escalones de $upitch$. 81
Figura 7.3: Régimen estacionario de la velocidad lineal vxB ante escalones de $upitch$. 81
Figura 7.4: Comparación entre datos reales y simulados de θ para diferentes escalones de Pitch. 84
Figura 7.5: Comparación entre datos reales y simulados de vxB para diferentes escalones de Pitch. 85
Figura 7.6: Respuesta al escalón en la entrada $uroll = -0.4$. 86
Figura 7.7: Régimen estacionario del ángulo ϕ ante escalones de $uroll$. 87
Figura 7.8: Régimen estacionario de la velocidad lineal vyB ante escalones de $uroll$. 87
Figura 7.9: Comparación entre datos reales y simulados de ϕ para diferentes escalones de Roll. 89
Figura 7.10: Comparación entre datos reales y simulados de vyB para diferentes escalones de Roll. 90
Figura 7.11: Respuesta al escalón en la entrada $uthrottle = 0.6$. 91
Figura 7.12: Régimen estacionario de la velocidad lineal vzE ante escalones de $uthrottle$. 92
Figura 7.13: Comparación entre datos reales y simulados con el modelo para diferentes escalones de Throttle. 93
Figura 7.14: Respuesta al escalón en la entrada $uyaw = -0.5$. 94
Figura 7.15: Régimen estacionario de la velocidad angular $v\psi B$ ante escalones de $uyaw$. 94
Figura 7.16: Comparación entre datos reales y simulados con el modelo para diferentes escalones de Throttle. 96
Figura 7.17: Posición y orientación tomada por la cámara para una posición determinada. 98
Figura 7.18: Ajuste por mínimos cuadrados de la posición y la orientación medidas por la cámara. 99
Figura 7.19: Altura tomada por el sensor de ultrasonidos para una altura fija determinada. 101
Figura 7.20: Ajuste por mínimos cuadrados de la altura medida por el sensor de ultrasonidos. 101
Figura 7.21: Medidas tomadas para el giróscopo del eje xB y su gráfica de la Varianza de Allan. 103
Figura 7.22: Medidas tomadas para el giróscopo del eje yB y su gráfica de la Varianza de Allan. 103
Figura 7.23: Medidas tomadas para el giróscopo del eje zB y su gráfica de la Varianza de Allan. 103
Figura 7.24: Medidas tomadas para el acelerómetro del eje xB y su gráfica de la Varianza de Allan. 106

Figura 7.25: Medidas tomadas para el acelerómetro del eje yB y su gráfica de la Varianza de Allan. 106
Figura 7.26: Medidas tomadas para el acelerómetro del eje zB y su gráfica de la Varianza de Allan. 106
Figura 8.1: Resultado del Filtro de Kalman para x e y en un vuelo de prueba. 113
Figura 8.2: Resultado del Filtro de Kalman para z y ψ en un vuelo de prueba. 114
Figura 8.3: Resultado del Filtro de Kalman para x e y en un vuelo de prueba. 114
Figura 8.4: Resultado del Filtro de Kalman para z y ψ en un vuelo de prueba. 114
Figura 8.5: Esquema básico de control de un proceso por realimentación. 115
Figura 8.6: Esquema de control mediante controladores PID. 117
Figura 8.7: Respuesta al escalón de xB con el controlador PID. 118
Figura 8.8: Respuesta al escalón de yB con el controlador PID. 119
Figura 8.9: Respuesta al escalón de z con el controlador PID. 119
Figura 8.10: Respuesta al escalón de ψB con el controlador PID. 120
Figura 8.11: Respuesta del controlador diseñado a referencias en el marco E. 121
Figura 8.12: Respuesta del controlador diseñado a referencias en el marco E, incluido el Yaw. 121
Figura 8.13: Respuesta del controlador ajustado a referencias en el marco E. 122
Figura 8.14: Respuesta del controlador reajustado a referencias en el marco E. 122
Figura 9.1: Gráfica XY del experimento de estabilización en el punto (0,0,0.8) metros con orientación -1.57 rad. 125
Figura 9.2: Respuesta temporal del experimento de estabilización en el punto (0,0,0.8) metros con orientación -1.57 rad. 125
Figura 9.3: Gráfica XY del experimento de estabilización con perturbaciones externas. 126
Figura 9.4: Respuesta temporal del experimento de estabilización con perturbaciones externas. 127
Figura 9.5: Gráfica XY del primer experimento de navegación a través de Waypoints. 128
Figura 9.6: Respuesta temporal del primer experimento de navegación a través de Waypoints. 129
Figura 9.7: Gráfica XY del segundo experimento de navegación a través de Waypoints. 130
Figura 9.8: Respuesta temporal del segundo experimento de navegación a través de Waypoints. 130
Figura 9.9: Gráfica XY del primer experimento de navegación a través de Waypoints con seguimiento de rectas. 132
Figura 9.10: Respuesta temporal del primer experimento de navegación a través de Waypoints con seguimiento de rectas. 133
Figura 9.11: Gráfica XY del segundo experimento de navegación a través de Waypoints con seguimiento de rectas. 134
Figura 9.12: Respuesta temporal del experimento de navegación a través de Waypoints con seguimiento de rectas. 135
Figura 9.13: Gráfica XY del experimento de seguimiento de trayectoria elíptica. 136
Figura 9.14: Respuesta temporal del experimento de seguimiento de trayectoria elíptica. 137
Figura 9.15: Gráfica XY del experimento de seguimiento de trayectoria elíptica con orientación hacia el centro. 138

Figura 9.16: Respuesta temporal del experimento de seguimiento de trayectoria elíptica con orientación hacia el centro. 139
Figura 9.17: Gráfica XY del experimento de seguimiento de trayectoria de la lemniscata de Gerono. 140
Figura 9.18: Respuesta temporal del experimento de seguimiento de trayectoria de la lemniscata de Gerono. 141
Figura A.1: Transformación general de perspectiva. 150
Figura A.2: Cálculo de la posición y la orientación. 151
Figura B.1: Parte del funcionamiento interno del bloque Principal. 153
Figura B.2: Parte del funcionamiento interno del bloque BuscaJoysticks. 154
Figura B.3: Parte del funcionamiento interno del bloque EstablecerControles. 154
Figura B.4: Parte del funcionamiento interno del bloque ControlesValores. 155
Figura B.5: Parte del funcionamiento interno del bloque TraduceTeclas. 156
Figura B.6: Parte del funcionamiento interno del bloque TraduceJoystickAttack. 157
Figura B.7: Parte del funcionamiento interno del bloque TraduceJoystickAttackTeclas. 157
Figura B.8: Parte del funcionamiento interno del bloque TraduceEmisoraAerofly. 158
Figura B.9: Parte del funcionamiento interno del bloque SumarValoresVuelo. 159
Figura B.10: Parte del funcionamiento interno del bloque GestionaFasesAuto. 159
Figura B.11: Parte del funcionamiento interno del bloque ControlAutomatico. 160
Figura B.12: Parte del funcionamiento interno del bloque ObtenerPosicionActual. 161
Figura B.13: Parte del funcionamiento interno del bloque ObtenerReferenciaNavegacion. 161
Figura B.14: Parte del funcionamiento interno del bloque FiltroKalman. 162
Figura B.15: Parte del funcionamiento interno del bloque PIDBloque. 163
Figura B.16: Funcionamiento interno del bloque TransformaBE. 164
Figura B.17: Funcionamiento interno del bloque TransformaEB. 164
Figura B.18: Parte del funcionamiento interno del bloque CrearLog. 164
Figura B.19: Parte del funcionamiento interno del bloque FormarLineaLog. 165
Figura B.20: Funcionamiento interno del bloque EscribirLog. 166
Figura B.21: Parte del funcionamiento interno del bloque SeleccionarFileDialog. 166
Figura B.22: Parte del funcionamiento interno del bloque PruebaControles. 167
Figura B.23: Funcionamiento interno del bloque PruebaTeclado. 167
Figura B.24: Parte del funcionamiento interno del bloque PruebaJoystick. 168
Figura B.25: Funcionamiento interno del bloque PruebaPID. 168
Figura B.26: Panel frontal del bloque PruebaPIDMulti. 169
Figura B.27: Parte del funcionamiento interno del bloque PruebaTrayectorias. 170
Figura B.28: Parte del funcionamiento interno del bloque AdecuarSentidos. 171
Figura B.29: Parte del funcionamiento interno del bloque AdecuarSentidosInverso. 171

Figura B.30: Funcionamiento interno del bloque DatosEnvioVuelo. 171
Figura B.31: Funcionamiento interno del bloque DibujaGraficas. 172
Figura B.32: Parte del funcionamiento interno del bloque GestionarCalibracion. 173
Figura B.33: Funcionamiento interno del bloque ModificadorAlturaDroneCamara. 173
Figura B.34: Parte del funcionamiento interno del bloque TransformaAnguloRangoPI. 173
Figura B.35: Parte del funcionamiento interno del bloque Unwrap. 174
Figura B.36: Panel frontal del bloque Camara. 175
Figura B.37: Panel frontal del bloque Calibracion. 176
Figura B.38: Parte del funcionamiento interno del bloque ATCONFIG_IDS_modif. 177
Figura B.39: Parte del funcionamiento interno del bloque Initialize Configuration Settings_modif_Multiconfig. 178
Figura B.40: Parte del funcionamiento interno del bloque Initialize Configuration Settings_modif_No_Multiconfig. 179
Figura B.41: Parte del funcionamiento interno del bloque Open_modif. 180
Figura B.42: Parte del funcionamiento interno del bloque OptionParser_modif. 180
Figura B.43: Parte del funcionamiento interno del bloque ParseNavHeader_modif. 181
Figura B.44: Parte del funcionamiento interno del bloque ParseNavData_modif. 181
Figura B.45: Funcionamiento interno del bloque Read NavData_modif. 182
Figura B.46: Funcionamiento interno del bloque ATMISC. 182
Figura B.47: Funcionamiento interno del bloque ATPMODE. 183
Figura B.48: Funcionamiento interno del bloque Envia_PMODE_MISC. 183
Figura B.49: Parte del funcionamiento interno del bloque Envia_Config. 184
Figura B.50: Funcionamiento interno del bloque ATCALIB. 184
Figura B.51: Funcionamiento interno del bloque Calib Drone. 185
Figura C.1: Pantalla principal de la plataforma software. 192
Figura C.2: Ubicación de botones y ejes utilizados del joystick Attack 3 de Logitech. 192

LISTA DE TABLAS

Tabla 5.1: Características eléctricas y ópticas del led LW540A. 64

Tabla 5.2: Valores máximos configurados de las entradas de control. 64

Tabla 7.1: Ajuste de las funciones de transferencia de θ respecto a $upitch$ en el dominio $\mathcal{S}$ para I=DatosIdent y V=DatosValid. 82

Tabla 7.2: Ajuste de las funciones de transferencia de θ respecto a $upitch$ en el dominio $\mathcal{S}$ para I=DatosValid y V=DatosIdent. 82

Tabla 7.3: Ajuste de las funciones de transferencia de θ respecto a $upitch$ en el dominio $\mathcal{Z}-1$ para I=DatosIdent y V=DatosValid. 83

Tabla 7.4: Ajuste de las funciones de transferencia de θ respecto a $upitch$ en el dominio $\mathcal{Z}-1$ para I=DatosValid y V=DatosIdent. 83

Tabla 7.5: Ajuste de las funciones de transferencia de vxB respecto a θ en el dominio $\mathcal{S}$ para I=DatosIdent y V=DatosValid. 83

Tabla 7.6: Ajuste de las funciones de transferencia de vxB respecto a θ en el dominio $\mathcal{S}$ para I=DatosValid y V=DatosIdent. 84

Tabla 7.7: Ajuste de las funciones de transferencia de vxB respecto a θ en el dominio $\mathcal{Z}-1$ para I=DatosIdent y V=DatosValid. 84

Tabla 7.8: Ajuste de las funciones de transferencia de vxB respecto a θ en el dominio $\mathcal{Z}-1$ para I=DatosValid y V=DatosIdent. 85

Tabla 7.9: Ajuste de las funciones de transferencia de ϕ respecto a $uroll$ en el dominio $\mathcal{S}$ para I=DatosIdent y V=DatosValid. 88

Tabla 7.10: Ajuste de las funciones de transferencia de ϕ respecto a $uroll$ en el dominio $\mathcal{S}$ para I=DatosValid y V=DatosIdent. 88

Tabla 7.11: Ajuste de las funciones de transferencia de ϕ respecto a $uroll$ en el dominio $\mathcal{Z}-1$ para I=DatosIdent y V=DatosValid. 88

Tabla 7.12: Ajuste de las funciones de transferencia de ϕ respecto a $uroll$ en el dominio $\mathcal{Z}-1$ para I=DatosValid y V=DatosIdent. 88

Tabla 7.13: Ajuste de las funciones de transferencia de vyB respecto a ϕ en el dominio $\mathcal{S}$ para I=DatosIdent y V=DatosValid. 89

Tabla 7.14: Ajuste de las funciones de transferencia de vyB respecto a ϕ en el dominio $\mathcal{S}$ para I=DatosValid y V=DatosIdent. 89

Tabla 7.15: Ajuste de las funciones de transferencia de vyB respecto a ϕ en el dominio $\mathcal{Z}-1$ para I=DatosIdent y V=DatosValid. 90

Tabla 7.16: Ajuste de las funciones de transferencia de vyB respecto a ϕ en el dominio $\mathcal{Z}-1$ para I=DatosValid y V=DatosIdent. 90

Tabla 7.17: Ajuste de las funciones de transferencia del Throttle en el dominio $\mathcal{S}$ para I=DatosIdent y V=DatosValid. 92

Tabla 7.18: Ajuste de las funciones de transferencia del Throttle en el dominio $\mathcal{S}$ para I=DatosValid y V=DatosIdent. 92

Tabla 7.19: Ajuste de las funciones de transferencia del Throttle en el dominio $\mathcal{Z}-1$ para I=DatosIdent y V=DatosValid. 93

Tabla 7.20: Ajuste de las funciones de transferencia del Throttle en el dominio $\mathcal{Z}-1$ para I=DatosValid y V=DatosIdent. 93

Tabla 7.21: Ajuste de las funciones de transferencia del Yaw en el dominio $\mathcal{S}$ para I=DatosIdent y V=DatosValid. 95

Tabla 7.22: Ajuste de las funciones de transferencia del Yaw en el dominio $\mathcal{S}$ para I=DatosValid y V=DatosIdent. 95

Tabla 7.23: Ajuste de las funciones de transferencia del Yaw en el dominio $\mathcal{Z}-1$ para I=DatosIdent y V=DatosValid. 95

Tabla 7.24: Ajuste de las funciones de transferencia del Yaw en el dominio $\mathcal{Z}-1$ para I=DatosValid y V=DatosIdent. 95

Tabla 7.25: Media y desviación estándar en las medidas de posición y orientación de la cámara. 98

Tabla 7.26: Media y desviación estándar en las medidas de altura del sensor de ultrasonidos. 100

Tabla 7.27: Valores de offsets bc de los giróscopos. 102

Tabla 7.28: Valores de ARW de los giróscopos. 104

Tabla 7.29: Valores de $\sigma gyro$ del ruido blanco de los giróscopos. 104

Tabla 7.30: Valores del parámetro BS de los giróscopos. 104

Tabla 7.31: Valores de σv_gyro del ruido de paseo aleatorio de los giróscopos. 104

Tabla 7.32: Valores de offsets bc_acc de los acelerómetros. 105

Tabla 7.33: Valores de VRW de los acelerómetros. 107

Tabla 7.34: Valores de σacc del ruido blanco de los acelerómetros. 107

Tabla 7.35: Valores del parámetro BS de los acelerómetros. 107

Tabla 7.36: Valores de σv_acc del ruido de paseo aleatorio de los acelerómetros. 107

Tabla 8.1: Parámetros del controlador PID para el Pitch. 117

Tabla 8.2: Características de la respuesta temporal al escalón del controlador PID del Pitch. 118

Tabla 8.3: Parámetros del controlador PID para el Roll. 118

Tabla 8.4: Características de la respuesta temporal al escalón del controlador PID del Roll. 118

Tabla 8.5: Parámetros del controlador PID para el Throttle. 119

Tabla 8.6: Características de la respuesta temporal al escalón del controlador PID del Throttle. 120

Tabla 8.7: Características de la respuesta temporal al escalón del controlador PID del Yaw. 120

Tabla 8.8: Parámetros ajustados del controlador PID para el Pitch. 121

Tabla 8.9: Parámetros ajustados del controlador PID para el Roll. 122

Tabla 8.10: Resumen de los controladores PID diseñados. 123

Tabla 9.1: Valores de los controladores PID utilizados en los experimentos reales. 124

Tabla 9.2: Medidas de error del experimento de estabilización en el punto (0,0,0.8) metros con orientación -1.57 rad. 124

Tabla 9.3: Puntos definidos del primer experimento de navegación a través de Waypoints. 128

Tabla 9.4: Puntos definidos del segundo experimento de navegación a través de Waypoints. 129

Tabla 9.5: Puntos definidos del primer experimento de navegación a través de Waypoints con seguimiento de rectas. 131

Tabla 9.6: Medidas de error del primer experimento de navegación a través de Waypoints con seguimiento de rectas. 132

Tabla 9.7: Puntos definidos del segundo experimento de navegación a través de Waypoints con seguimiento de rectas. 133

Tabla 9.8: Medidas de error del segundo experimento de navegación a través de Waypoints con seguimiento de rectas. 134

Tabla 9.9: Medidas de error del experimento de seguimiento de trayectoria elíptica. 136

Tabla 9.10: Medidas de error del experimento de seguimiento de trayectoria elíptica con orientación hacia el centro. 138

Tabla 9.11: Medidas de error del experimento de seguimiento de trayectoria de la lemniscata de Gerono. 140

Tabla C.1: Valores máximos configurados de las entradas de control en el AR.Drone 2.0. 196

Tabla C.2: Datos almacenados en cada línea del fichero de log. 200

Tabla D.1: Parámetros de configuración del AR.Drone 2.0. 204

Tabla D.2: Información de estado del AR.Drone 2.0. 205

Tabla D.3: Datos de navegación de la opción Navdata Demo del AR.Drone 2.0. 205

Tabla D.4: Datos de navegación de la opción Navdata Cks del AR.Drone 2.0. 206

Tabla D.5: Datos de navegación de la opción Navdata Time del AR.Drone 2.0. 206

Tabla D.6: Datos de navegación de la opción Navdata Raw Measures del AR.Drone 2.0. 206

Tabla D.7: Datos de navegación de la opción Navdata Pressure Raw del AR.Drone 2.0. 206

Tabla D.8: Datos de navegación de la opción Navdata Magneto del AR.Drone 2.0. 207

Tabla D.9: Datos de navegación de la opción Navdata Wind Speed del AR.Drone 2.0. 207

Tabla D.10: Datos de navegación de la opción Navdata Kalman Pressure del AR.Drone 2.0. 208

Tabla D.11: Datos de navegación de la opción Navdata Phys Measures del AR.Drone 2.0. 208

Tabla D.12: Datos de navegación de la opción Navdata Gyros Offsets del AR.Drone 2.0. 208

Tabla D.13: Datos de navegación de la opción Navdata Euler Angles del AR.Drone 2.0. 208

Tabla D.14: Datos de navegación de la opción Navdata References del AR.Drone 2.0. 209

Tabla D.15: Datos de navegación de la opción Navdata Trims del AR.Drone 2.0. 209

Tabla D.16: Datos de navegación de la opción Navdata RC References del AR.Drone 2.0. 209

Tabla D.17: Datos de navegación de la opción Navdata PWM del AR.Drone 2.0. 210

Tabla D.18: Datos de navegación de la opción Navdata Altitude del AR.Drone 2.0. 210

Tabla D.19: Datos de navegación de la opción Navdata Vision Raw del AR.Drone 2.0. 211

Tabla D.20: Datos de navegación de la opción Navdata Vision del AR.Drone 2.0. 211

Tabla D.21: Datos de navegación de la opción Navdata Vision Perf del AR.Drone 2.0. 212

Tabla D.22: Datos de navegación de la opción Navdata Trackers Send del AR.Drone 2.0. 212

Tabla D.23: Datos de navegación de la opción Navdata Vision Detect del AR.Drone 2.0. 212

Tabla D.24: Datos de navegación de la opción Navdata Vision Of del AR.Drone 2.0. 212
Tabla D.25: Datos de navegación de la opción Navdata Watchdog del AR.Drone 2.0. 213
Tabla D.26: Datos de navegación de la opción Navdata ADC Data Frame del AR.Drone 2.0. 213
Tabla D.27: Datos de navegación de la opción Navdata Video Stream del AR.Drone 2.0. 213
Tabla D.28: Datos de navegación de la opción Navdata HDVideo Stream del AR.Drone 2.0. 214
Tabla D.29: Datos de navegación de la opción Navdata Games del AR.Drone 2.0. 214
Tabla D.30: Datos de navegación de la opción Navdata Wifi del AR.Drone 2.0. 214

1. INTRODUCCIÓN

La característica principal de los vehículos aéreos no tripulados (UAV) es que su pilotaje no lo realiza una persona desde el propio vehículo sino que lo hace de forma remota. Este control remoto puede tener diferentes grados de autonomía, desde que el control remoto lo realice por completo una persona hasta que éste sea realizado por computadores.

El objetivo principal del uso de este tipo de vehículos aéreos es realizar tareas que por su dificultad o peligrosidad no son recomendables para las personas, aunque en la actualidad su uso se está expandiendo a otro tipo de actividades como pueden ser las comerciales o de ocio.

De entre todas las aplicaciones que actualmente los UAVs pueden realizar se encuentran las siguientes:

- Militar: En el ámbito militar fue donde se desarrolló inicialmente la tecnología de los UAVs para tareas como el ataque, la defensa o el reconocimiento [1].
- Ocio: Dentro de las aplicaciones recreativas más usuales se encuentra el aeromodelismo y el vuelo de estos aeromodelos en competiciones de habilidad y velocidad [2], o simplemente por diversión.
- Vigilancia aérea: Gracias a los UAVs se pueden vigilar grandes extensiones de terreno para prevenir y ayudar a controlar incendios, problemas en diversos tipos de infraestructuras, como soporte a los agentes de la ley o para la búsqueda y rescate de personas [3].
- Filmación: La grabación de vídeo profesional para su edición o para su emisión en directo es una de las aplicaciones más de moda de los UAVs [4]. Las características de éstos permiten la captura de imágenes desde lugares y ángulos que resultaría imposible obtener de otra forma. Su uso en eventos deportivos es creciente debido a su libertad de movimientos y su rapidez para realizar un seguimiento en directo de los deportistas [5].
- Medio ambiente [6]: Permiten monitorizar diferentes índices de contaminación o polución para crear mapas que faciliten su control. Igualmente permiten controlar y realizar el seguimiento de accidentes industriales con vertidos tóxicos en medios acuáticos y terrestres.
- Logística: Grandes compañías de logística están utilizando UAVs para el envío de paquetería a zonas rurales o de acceso complicado [7]. Igualmente otras compañías lo están haciendo con medicinas o incluso aparatos médicos, como desfibriladores, que pueden ayudar a salvar vidas [8].
- Agricultura: Con los UAVs se puede controlar y monitorizar el estado de los cultivos mediante imágenes multiespectrales, y determinar la eficiencia de regadíos. Igualmente permiten realizar inspecciones visuales en relación a la producción agrícola subvencionada [9].
- Ingeniería civil: Los usos más típicos en este ámbito son los de la inspección de obras desde el aire, la estimación de impacto visual de grandes obras o la realización de mapas del terreno gracias a la fotogrametría [10].
- Investigación científica: Los UAVs pueden utilizarse como apoyo en multitud de aplicaciones de investigación científica sobre todo en entornos de difícil acceso o cuyas condiciones hacen imposible la presencia de personas [11].

El UAV más extendido en la actualidad es el denominado cuadricóptero que dispone de cuatro hélices impulsadas por motores que generan el empuje necesario para despegar y aterrizar verticalmente, volar hacia adelante y hacia atrás, volar de forma lateral y mantenerse en vuelo en una posición determinada. Generalmente dos de estas hélices giran en un sentido y las otras dos en el otro, y en función de las velocidades de giro de cada una de ellas consigue generar las fuerzas y pares necesarios para realizar las maniobras antes descritas.

Los 6 grados de libertad de los que disponen los cuadricópteros, su facilidad de maniobra y el bajo coste de fabricación los hacen ideales para las aplicaciones antes descritas. A ellos se les incorporan diferentes sensores como acelerómetros, giróscopos, sensores de altura, de presión, de viento, etc, con los que se consigue

proporcionar una mayor información a quien controla el vehículo de forma remota, ya sea un humano o una máquina, facilitando así la navegación, la estabilización, etc. Ciertos cuadricópteros disponen de otros tipos de sensores y de dispositivos manipuladores para aplicaciones específicas.

También suelen incorporar cámaras que permitan almacenar el vídeo para su posterior análisis o edición, así como dispositivos de comunicación, Wi-Fi por ejemplo, para enviar las imágenes de la cámara en tiempo real y otros datos que puedan ser útiles para el controlador remoto.

Parrot AR.Drone 2.0 es un cuadricóptero que reúne muchas de estas características [12]. Dispone de multitud de sensores, de dos cámaras y se comunica con su controlador remoto mediante Wi-Fi, con un coste muy bajo. Todas sus características lo hacen ideal para experimentación e investigación científica y por ello ha sido seleccionado para el presente proyecto.

El objetivo principal de este proyecto es conseguir que dicho cuadricóptero sea controlado desde un PC de forma autónoma en un entorno cerrado, bien para mantenerse estable en un punto, navegar a través de puntos previamente programados o seguir una trayectoria definida. Para ello será necesario obtener un modelo matemático que se ajuste lo suficiente al comportamiento real del cuadricóptero y crear las leyes de control adecuadas que posibiliten las diferentes formas de control autónomo indicadas.

Este proyecto no se plantea únicamente con dicho objetivo sino que también pretende proporcionar una plataforma de control para este UAV que permita a investigadores u otras personas interesadas en este campo crear nuevas estrategias de control autónomo o nuevos métodos de navegación que puedan ser validados de forma experimental.

Para determinar la posición real del cuadricóptero se utiliza un sistema de visión externo a él que está compuesto por una cámara instalada en el techo del lugar de experimentación y que proporciona la componente x e y de la posición, así como la orientación del cuadricóptero.

El software de la plataforma de control se desarrollará en LabView ya que su programación orientada a objetos permite un desarrollo intuitivo y claro del sistema y de los elementos necesarios para su correcto funcionamiento. Además, LabView es una herramienta muy adecuada para simulaciones “hardware in the loop”, donde se puede simular el correcto funcionamiento de la aplicación software sin el cuadricóptero real y luego incorporarlo sin modificar el código de la aplicación.

2. ANTECEDENTES

En los últimos 15 años se ha conseguido fabricar cuadricópteros dotándolos de los sensores y equipos necesarios para permitir un vuelo flexible y relativamente estable con un coste bajo. Desde entonces multitud de investigadores han realizado estudios relativos a la cinemática y dinámica de este tipo de vehículos aéreos no tripulados, como por ejemplo en [13] o en [14], para obtener modelos precisos que sirvan para mejorar su manejo. Muchas formas de control han sido utilizadas como los controladores PID o las técnicas LQR como se pone de manifiesto en [15], técnicas adaptativas [16] o control por backstepping [17].

En relación al Parrot AR.Drone se han hecho también múltiples estudios e investigaciones. En [18] se plantea una plataforma que sirva para la investigación y la educación, realizando un control mediante PIDs y con una navegación basada en las imágenes captadas por las cámaras del cuadricóptero. Un estudio parecido [19] utiliza también las cámaras del cuadricóptero pero los algoritmos de visión utilizados necesitan de balizas colocadas en el lugar de experimentación (paredes y suelo) para determinar su posición.

En general la navegación basada en las imágenes de las cámaras del AR.Drone es objetivo de muchos estudios e investigaciones debido a que normalmente no son necesarios sistemas externos que proporcionen una medida real de su posición y orientación. Otro ejemplo de ello es [20].

Además, muchos de estos estudios utilizan el Filtro de Kalman para estimar el estado del cuadricóptero, como por ejemplo [21] y [22] que también utilizan un algoritmo de visión para determinar su posición gracias a las cámaras del AR.Drone, o como [23] donde se utiliza un sistema de visión externo con una cámara Kinect para determinar la posición 3D del cuadricóptero.

El uso de sistemas externos de visión no es muy común en la realización de investigaciones con el cuadricóptero Parrot AR.Drone 2.0 debido a la limitación física que estos sistemas de visión imponen en el área de vuelo del cuadricóptero. A pesar de ello, en el presente proyecto se ha utilizado una cámara situada en el techo del lugar de experimentación para proporcionar una medida real de la posición y orientación del cuadricóptero.

3. MODELO DE UN CUADRICÓPTERO

Un cuadricóptero se compone de cuatro rotores dispuestos simétricamente con respecto a su centro, colocados al final de cuatro brazos, permitiéndole realizar un juego completo de movimientos. Las fuerzas y momentos dominantes que actúan sobre el cuadricóptero se crean mediante los rotores impulsados por motores, que en general son de corriente continua y sin escobillas.

Hay dos tipos de configuración en función de cómo están dispuestos los rotores con respecto a su sistema de coordenadas, estos son en forma de + y en forma de x, tal y como puede verse en la Figura 3.1.

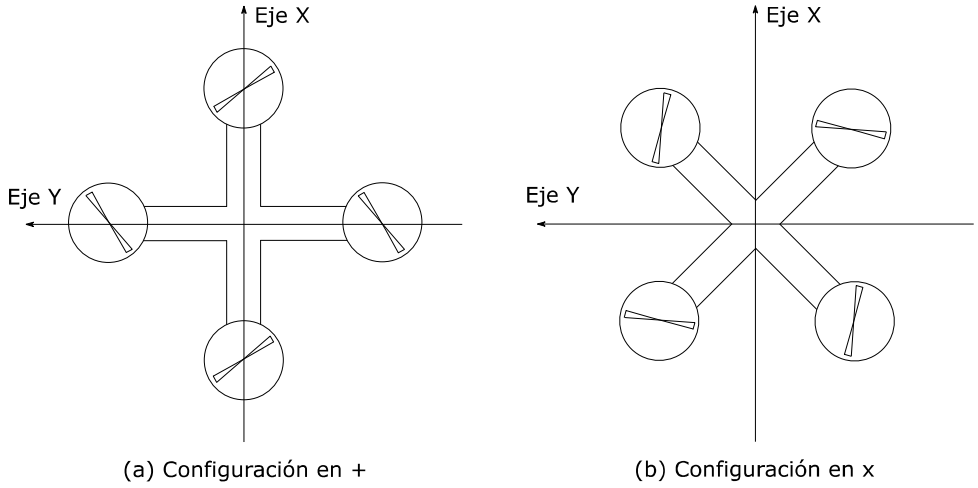


Figura 3.1: Tipos de configuración de un cuadricóptero.

En la configuración en forma de + las dos hélices que se encuentran a lo largo del eje X giran en un sentido mientras que las dos hélices que se encuentran a lo largo del eje Y giran en el sentido contrario. En la configuración en forma de x las dos hélices enfrentadas entre sí giran en un sentido mientras que las otras dos hélices giran en el sentido contrario.

La configuración en forma de x proporciona un mayor momento en comparación con la configuración en forma de + lo que permite incrementar la maniobrabilidad ya que en cualquier movimiento a realizar necesita de una variación en la velocidad de rotación de las cuatro hélices.

Los movimientos básicos de un cuadricóptero y los mecanismos para conseguir dichos movimientos son los siguientes:

- Movimiento en altura. Se consigue aumentando o disminuyendo la potencia de los rotores (Throttle).
- Movimiento en la dirección del eje X. Se consigue a través de un cambio en la velocidad de ciertos rotores que provocan una inclinación del cuadricóptero con respecto a su eje Y. A esta inclinación se le llama Cabeceo o Pitch.
- Movimiento en la dirección del eje Y. Se consigue a través de un cambio en la velocidad de ciertos rotores que provocan una inclinación del cuadricóptero con respecto a su eje X. A esta inclinación se le llama Alabeo o Roll.
- Movimiento de rotación con respecto al eje Z con el que se cambia la Orientación o Heading del cuadricóptero. Se consigue a través de un cambio en la velocidad de ciertos rotores que provocan dicha rotación. Al ángulo de rotación se le llama Guiñada o Yaw.

Combinando estos movimientos básicos se consigue que el cuadricóptero pueda moverse con 6 grados de libertad.

Desde la Figura 3.2 hasta la Figura 3.6 se pueden ver las variaciones de velocidad en los rotores que se necesitan para realizar los movimientos básicos de un cuadricóptero configurado en forma de +. Desde la Figura 3.7 hasta la Figura 3.11 se puede ver lo mismo en caso de que su configuración sea en forma de x.

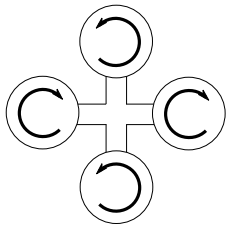


Figura 3.2: Configuración en +. Velocidad de los rotores en reposo.

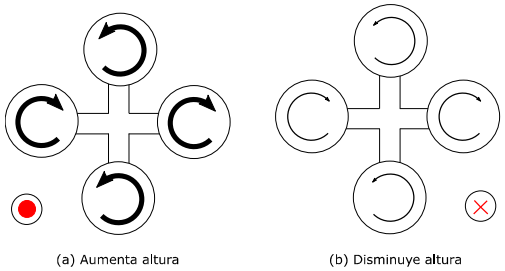


Figura 3.3: Configuración en +. Velocidad de los rotores para variar la altura (Throttle).

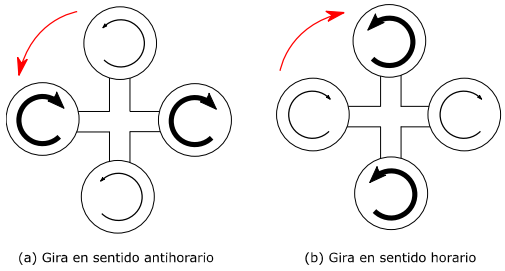


Figura 3.4: Configuración en +. Velocidad de los rotores para variar la orientación (Yaw).

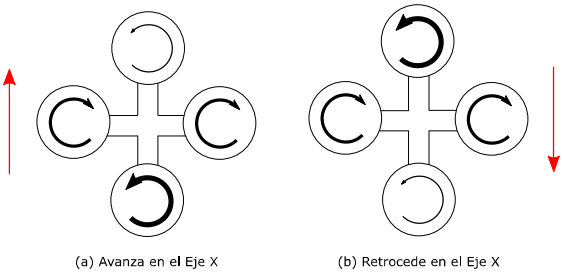


Figura 3.5: Configuración en +. Velocidad de los rotores para variar la posición en el Eje X (Pitch).

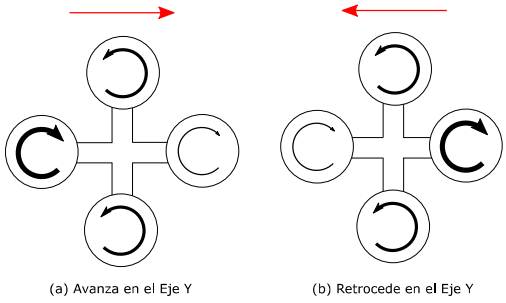


Figura 3.6: Configuración en +. Velocidad de los rotores para variar la posición en el Eje Y (Roll).

Se puede observar que para desplazarse en altura se aumentan o disminuyen las velocidades de giro de los cuatro rotores en la misma medida. Para realizar un movimiento de rotación con respecto al eje Z se aplica la misma variación de velocidades tanto en la configuración en + como en la configuración en x.

Sin embargo la diferencia estriba en los movimientos sobre el eje X (Pitch) y sobre el eje Y (Roll). En la configuración en +, para modificar el Pitch, se disminuye la velocidad del rotor que se encuentra en la posición hacia donde se quiere dirigir el cuadricóptero sobre el eje X y se aumenta la velocidad del rotor enfrentado, mientras que la velocidad de los rotores que se encuentran sobre el eje Y se mantiene constante. En la configuración en x, se disminuye la velocidad de los dos rotores que se encuentran en el semiplano hacia el que se quiere dirigir el cuadricóptero y se aumenta la velocidad de los otros dos rotores. El caso del Roll es similar.

Se puede observar, como se dijo anteriormente, que en el caso de la configuración en x todos los rotores han de variar su velocidad para realizar cualquier movimiento, mientras que en el caso de la configuración en +, los movimientos de Pitch y de Roll se realizan sólo variando la velocidad de dos rotores y manteniendo constante la de los otros dos.

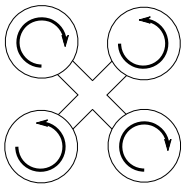


Figura 3.7: Configuración en x. Velocidad de los rotores en reposo.

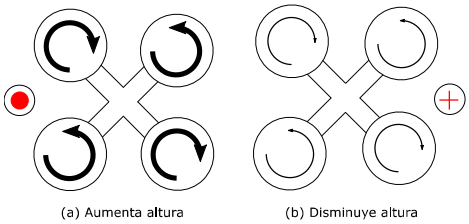


Figura 3.8: Configuración en x. Velocidad de los rotores para variar la altura (Throttle).

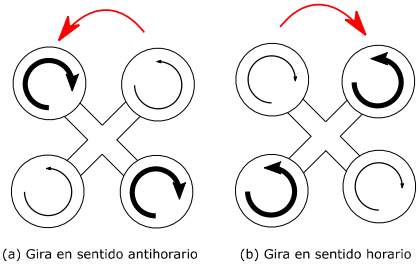


Figura 3.9: Configuración en x. Velocidad de los rotores para variar la orientación (Yaw).

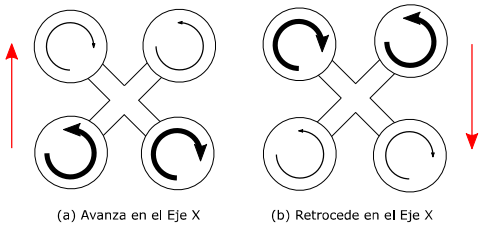


Figura 3.10: Configuración en x. Velocidad de los rotores para variar la posición en el Eje X (Pitch).

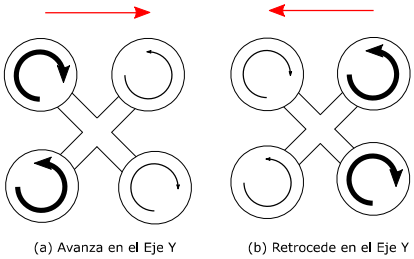


Figura 3.11: Configuración en x. Velocidad de los rotores para variar la posición en el Eje Y (Roll).

Los cuadricópteros se caracterizan por ser vehículos aéreos con un alto comportamiento no lineal, un fuerte acoplamiento y por ser básicamente subactuados al tener 6 grados de libertad para únicamente 4 entradas de control. La dificultad para controlarlos estriba en el acoplamiento cruzado entre los diferentes grados de libertad pero también a que los cuatro rotores montados en él pueden tener diferentes parámetros y a las perturbaciones del entorno, como por ejemplo el viento. Todo esto hace que los cuadricópteros sean muy sensibles a todo tipo de perturbaciones y necesiten de un adecuado control para hacerlos estables.

3.1. Modelado cinemático

El modelo cinemático de un cuadricóptero se va a obtener empleando el método de Newton-Euler [14] para el cuál es necesario definir dos marcos de referencia (Figura 3.12):

- Marco de referencia inercial de la Tierra E (Earth).
- Marco de referencia fijo del cuadricóptero B (Body).

En el marco de referencia E (O_E, X_E, Y_E, Z_E), X_E apunta hacia el Norte, Y_E apunta hacia el Oeste, Z_E apunta hacia arriba con respecto a la tierra y O_E es su origen. Además, en este marco de referencia se utiliza la regla de la mano derecha para determinar si los ángulos de rotación con respecto a cualquier eje son positivos o negativos. Este marco se utiliza para determinar la posición lineal Γ^E y angular Θ^E del cuadricóptero.

En el marco de referencia B (O_B, X_B, Y_B, Z_B), X_B apunta hacia el frente del cuadricóptero, Y_B apunta hacia la izquierda del cuadricóptero, Z_B apunta hacia arriba con respecto al plano $X_B Y_B$ y O_B es su origen que coincide con el centro del cuadricóptero. Igual que en el marco anterior, en este marco de referencia se utiliza la regla de la mano derecha para determinar si los ángulos de rotación con respecto a cualquier eje son positivos o

negativos. En este marco se define la velocidad lineal V^B, la velocidad angular ω^B, las fuerzas F^B y los momentos τ^B del cuadricóptero.

La posición lineal Γ^E se determina a través de las coordenadas del vector que va desde el origen del marco de referencia E hasta el origen del marco de referencia B, tal y como se muestra en la ecuación (3.1).

$$\Gamma^E = [X \quad Y \quad Z]^T \tag{3.1}$$

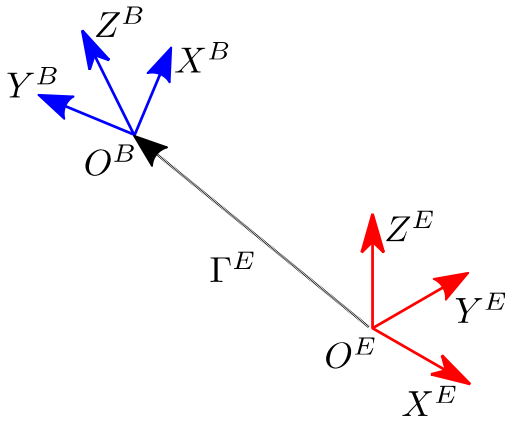
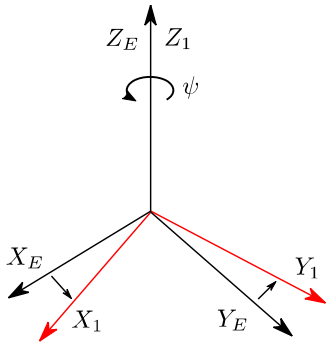


Figura 3.12: Marcos de referencia de un cuadricóptero.

Por su parte, la posición angular Θ^E del cuadricóptero se determina a través de la orientación del marco de referencia B con respecto al E. Esta orientación se obtiene a través de tres rotaciones sucesivas, una para cada eje, que transforman la orientación del marco E a la del B en base a los ángulos de Euler Roll (ϕ), Pitch (θ) y Yaw (ψ). La ecuación (3.2) define la posición angular del cuadricóptero.

$$\Theta^E = [\phi \quad \theta \quad \psi]^T \tag{3.2}$$

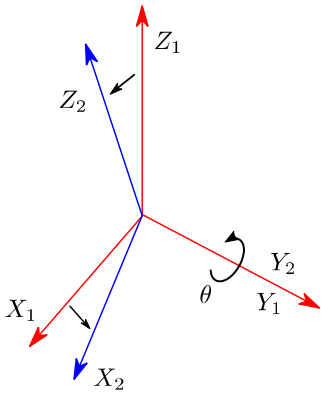
La primera rotación se realiza sobre el eje Z_E con un ángulo de Yaw ψ, tal y como se muestra en la Figura 3.13, obteniendo como resultado la matriz $R(\psi, Z)$ que se puede ver en la ecuación (3.3)



$$R(\psi, Z) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

Figura 3.13: Rotación de Yaw sobre el eje Z_E

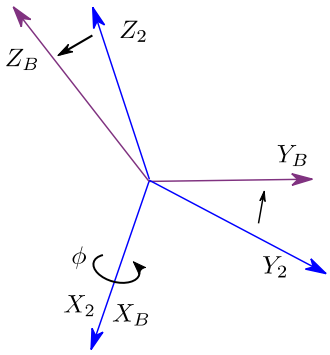
La segunda rotación se realiza sobre el eje Y_1 con un ángulo de Pitch θ, tal y como se muestra en la Figura 3.14, obteniendo como resultado la matriz $R(\theta, Y)$ que se puede ver en la ecuación (3.4).



$$R(\theta, Y) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \quad (3.4)$$

Figura 3.14: Rotación de Pitch sobre el eje Y_1

La última rotación se realiza sobre el eje X_2 con un ángulo de Roll ϕ, tal y como se muestra en la Figura 3.15, obteniendo como resultado la matriz $R(\phi, X)$ que se puede ver en la ecuación (3.5).



$$R(\phi, X) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \quad (3.5)$$

Figura 3.15: Rotación de Roll sobre el eje X_2

Las sucesivas rotaciones forman una matriz de rotación R_Θ como se muestra en la ecuación (3.6).

$$\begin{aligned} R_\Theta &= R(\psi, Z)\, R(\theta, Y)\, R(\phi, X) = \\ &= \begin{bmatrix} \cos\psi\ \cos\theta & -\sin\psi\ \cos\phi + \cos\psi\ \sin\theta\ \sin\phi & \sin\psi\ \sin\phi + \cos\psi\ \sin\theta\ \cos\phi \\ \sin\psi\ \cos\theta & \cos\psi\ \cos\phi + \sin\psi\ \sin\theta\ \sin\phi & -\cos\psi\ \sin\phi + \sin\psi\ \sin\theta\ \cos\phi \\ -\sin\phi & \cos\theta\ \sin\phi & \cos\theta\ \cos\phi \end{bmatrix} \end{aligned} \quad (3.6)$$

Las velocidades lineales V^B y las velocidades angulares ω^B se expresan en el marco de referencia B y se componen según las ecuaciones (3.7) y (3.8).

$$\mathrm{V}^B = [u \quad v \quad w]^T \quad (3.7)$$

$$\omega^B = [p \quad q \quad r]^T \quad (3.8)$$

Se pueden combinar los vectores de posición lineal y de posición angular dando lugar al vector de posición general ξ tal y como se muestra en la ecuación (3.9), e igualmente se pueden combinar los vectores de velocidad lineal y de velocidad angular dando lugar al vector de velocidad general ν tal y como se muestra en la ecuación (3.10).

$$\xi = [\Gamma^E \quad \Theta^E]^T = [X \quad Y \quad Z \quad \phi \quad \theta \quad \psi]^T \tag{3.9}$$

$$\nu = [\mathrm{V}^B \quad \omega^B]^T = [u \quad v \quad w \quad p \quad q \quad r]^T \tag{3.10}$$

La velocidad lineal V^B puede transformarse al marco de referencia E gracias a la matriz de rotación R_Θ según se indica en la ecuación (3.11).

$$\mathrm{V}^E = R_\Theta \mathrm{V}^B \tag{3.11}$$

Igualmente, la velocidad angular ω^B también puede transformarse al marco de referencia E gracias a la matriz de transformación T_Θ según se indica en la ecuación (3.12).

$$\dot{\Theta}^E = T_\Theta \omega^B \tag{3.12}$$

Multiplicando en ambos lados de la ecuación (3.12) por T_Θ^{-1} se obtiene la ecuación (3.13), que representa la transformación de la velocidad angular en el marco de referencia E al marco de referencia B.

$$\omega^B = T_\Theta^{-1} \dot{\Theta}^E \tag{3.13}$$

Sólo en el caso que los ángulos de Euler sean infinitesimales, es cierto que la variación en el tiempo de éstos es equivalente a la velocidad de rotación en el marco de referencia B, por lo que considerando pequeñas variaciones de cada ángulo de Euler se puede determinar el efecto sobre el vector de velocidades angulares ω^B, y con ello obtener la matriz de transformación T_Θ. Teniendo en cuenta las tres rotaciones descritas en las ecuaciones (3.3), (3.4) y (3.5), se puede comprobar que en la transformación inversa, al marco de referencia E, el primer ángulo de Euler (Yaw) conlleva dos rotaciones adicionales, el segundo ángulo de Euler (Pitch) conlleva sólo una rotación adicional, mientras que el tercer ángulo de Euler (Roll) no conlleva rotaciones adicionales. La ecuación (3.14) muestra esta transformación al marco de referencia E.

$$\begin{aligned}
\begin{bmatrix} p \\ q \\ r \end{bmatrix} &= R(\phi, X)^{-1} R(\theta, Y)^{-1} \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} + R(\phi, X)^{-1} \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} \\
&= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}^{-1} \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} \\
&= \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \cos\theta\ \sin\phi \\ 0 & -\sin\phi & \cos\theta\ \cos\phi \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}
\end{aligned} \tag{3.14}$$

Comparando las ecuaciones (3.13) y (3.14) se puede obtener el valor de la matriz de transformación de velocidades angulares del marco de referencia B al E, que se muestra en la ecuación (3.15). Invirtiendo esta matriz se obtiene la matriz de transformación de velocidades angulares del marco de referencia E al B, tal y como se muestra en la ecuación (3.16).

$$T_\Theta^{-1} = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \cos\theta\ \sin\phi \\ 0 & -\sin\phi & \cos\theta\ \cos\phi \end{bmatrix} \tag{3.15}$$

$$T_\Theta = \begin{bmatrix} 1 & \sin\phi\ \tan\theta & \cos\phi\ \tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\ /\cos\theta & \cos\phi\ /\cos\theta \end{bmatrix} \tag{3.16}$$

Las ecuaciones (3.11) y (3.12) que proporcionan la transformación de velocidades lineales y angulares del marco de referencia B al E pueden generalizarse en un único vector $\dot{\xi}$ que resulte de la transformación del vector v tal y como se indica en la ecuación (3.17), donde J_Θ es su correspondiente matriz de transformación y cuyo valor se muestra en la ecuación (3.18) en la que se ha introducido la notación 0_{3x3} que se corresponde con una submatriz de dimensión 3x3 con ceros en todos sus elementos.

$$\dot{\xi} = J_\Theta v \tag{3.17}$$

$$J_\Theta = \begin{bmatrix} R_\Theta & 0_{3x3} \\ 0_{3x3} & T_\Theta \end{bmatrix} \tag{3.18}$$

3.2. Modelado dinámico

El modelo dinámico de un cuadricóptero, al igual que el cinemático se va a obtener empleando el método de Newton-Euler [14], estudiando el efecto de las fuerzas y momentos sobre su movimiento. En este caso, las ecuaciones de movimiento se van a formular con respecto al marco de referencia fijo del cuadricóptero B debido a las siguientes ventajas:

- La matriz de inercias es invariante en el tiempo.
- Se puede aprovechar la simetría del cuadricóptero para simplificar las ecuaciones.
- Las medidas tomadas por el cuadricóptero se convierten fácilmente a dicho marco de referencia.
- Las acciones de control habitualmente son proporcionadas en dicho marco de referencia.

En la aplicación de este enfoque se va a asumir que el origen del marco de referencia B coincide con el centro de masas del cuadricóptero y que los ejes de dicho marco de referencia coinciden con los ejes de inercia principales del cuadricóptero.

El primer axioma de Euler a la segunda ley de movimiento de Newton dice que el momento lineal de un cuerpo es igual al producto de la masa del cuerpo y la velocidad de su centro de masas. De forma equivalente este axioma dice que la fuerza total aplicada a un cuerpo es igual al producto de la masa del cuerpo y la aceleración de su centro de masas. De esta última definición se derivan las componentes lineales de las ecuaciones de movimiento del cuadricóptero, como se ve en la ecuación (3.19).

$$F^E = m\ddot{\Gamma}^E$$

$$R_\Theta F^B = m\dot{\widehat{R_\Theta \mathrm{V}^B}}$$

$$R_\Theta F^B = m\left(R_\Theta \dot{\mathrm{V}}^B + \dot{R}_\Theta \mathrm{V}^B\right)$$

$$R_\Theta F^B = mR_\Theta\left(\dot{\mathrm{V}}^B + \omega^B \times \mathrm{V}^B\right)$$

$$F^B = m\left(\dot{\mathrm{V}}^B + \omega^B \times \mathrm{V}^B\right) \tag{3.19}$$

En ella, m es la masa del cuadricóptero, $\ddot{\Gamma}^E$ es el vector de aceleración lineal del cuadricóptero en el marco de referencia E, F^E es el vector de fuerzas aplicadas al cuadricóptero en el marco de referencia E, F^B es el vector de fuerzas aplicadas al cuadricóptero en el marco de referencia B, $\dot{\mathrm{V}}^B$ es el vector de aceleraciones lineales del cuadricóptero en el marco de referencia B y $\dot{R}_\Theta$ es la derivada de la matriz de rotación R_Θ. Además, para su derivación se ha tenido en cuenta la definición $\omega^B = R_\Theta^{-1}\dot{R}_\Theta$ dada en [24].

El segundo axioma de Euler a la segunda ley de movimiento de Newton dice que la variación del momento angular sobre un punto fijo de un cuerpo en un marco de referencia inercial es igual a la suma de los pares que actúan sobre ese cuerpo en dicho punto. De forma equivalente este axioma dice que, considerando que el

origen del marco de referencia inercial coincide con el centro de masas del cuerpo, la suma de los pares que actúan sobre ese cuerpo en un punto fijo es igual al producto del momento de inercia de dicho cuerpo en su centro de masas y su aceleración angular con respecto al centro de masas. De esta última definición se derivan las componentes angulares de las ecuaciones de movimiento del cuadricóptero, como se ve en la ecuación (3.20).

$$\tau^E = I\ddot{\Theta}^E$$

$$T_\Theta \tau^B = I\dot{\widehat{T_\Theta \omega^B}}$$

$$T_\Theta \tau^B = I\left(T_\Theta \dot{\omega}^B + \dot{T}_\Theta \omega^B\right)$$

$$T_\Theta \tau^B = IT_\Theta \dot{\omega}^B + I\dot{T}_\Theta \omega^B$$

$$T_\Theta \tau^B = T_\Theta\left(I\dot{\omega}^B + \omega^B \times (I\omega^B)\right)$$

$$\tau^B = I\dot{\omega}^B + \omega^B \times (I\omega^B) \qquad (3.20)$$

En ella, I es la matriz de inercias del cuadricóptero en el marco de referencia B, $\ddot{\Theta}^E$ es el vector de aceleraciones angulares del cuadricóptero en el marco de referencia E, $\dot{\omega}^B$ es el vector de aceleraciones angulares del cuadricóptero en el marco de referencia B, τ^E es el vector de pares de fuerza aplicados al cuadricóptero en el marco de referencia E, τ^B es el vector de pares de fuerza aplicados al cuadricóptero en el marco de referencia B y $\dot{T}_\Theta$ es la derivada de la matriz de transformación T_Θ.

Uniendo las ecuaciones (3.19) y (3.20) se consigue describir el movimiento de un cuadricóptero de forma general, tal y como se puede ver en su forma matricial en la ecuación (3.21), donde se ha introducido la notación I_{3x3} que se corresponde con una matriz identidad de dimensión 3x3.

$$\begin{bmatrix} F^B \\ \tau^B \end{bmatrix} = \begin{bmatrix} mI_{3x3} & 0_{3x3} \\ 0_{3x3} & I \end{bmatrix} \begin{bmatrix} \dot{V}^B \\ \dot{\omega}^B \end{bmatrix} + \begin{bmatrix} \omega^B \times (mV^B) \\ \omega^B \times (I\omega^B) \end{bmatrix} \qquad (3.21)$$

A este vector generalizado de fuerzas se le va a denominar Λ y sus componentes son las que se pueden ver en la ecuación (3.22).

$$\Lambda = [F^B \quad \tau^B]^{\mathrm{T}} = [F_{\mathrm{x}} \quad F_{\mathrm{y}} \quad F_{\mathrm{z}} \quad \tau_{\mathrm{x}} \quad \tau_{\mathrm{y}} \quad \tau_{\mathrm{z}}]^{\mathrm{T}} \qquad (3.22)$$

Se puede reescribir, por tanto, la ecuación (3.21) de la manera que se ve en la ecuación (3.23) donde $\dot{\nu}$ es el vector generalizado de aceleraciones en el marco de referencia B, $\mathrm{M_B}$ es la matriz general de inercias del sistema en el marco de referencia B, y $\mathrm{C_B}(\nu)$ es la matriz de aceleraciones centrífugas y de Coriolis en el marco de referencia B.

$$\Lambda = \mathrm{M_B}\dot{\nu} + \mathrm{C_B}(\nu)\nu \qquad (3.23)$$

La matriz de inercias del sistema $\mathrm{M_B}$ se muestra en la ecuación (3.24).

$$\mathrm{M_B} = \begin{bmatrix} mI_{3x3} & 0_{3x3} \\ 0_{3x3} & I \end{bmatrix} = \begin{bmatrix} m & 0 & 0 & 0 & 0 & 0 \\ 0 & m & 0 & 0 & 0 & 0 \\ 0 & 0 & m & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{XX} & 0 & 0 \\ 0 & 0 & 0 & 0 & I_{YY} & 0 \\ 0 & 0 & 0 & 0 & 0 & I_{ZZ} \end{bmatrix} \qquad (3.24)$$

Desarrollando el término $\omega^B \times (m\mathrm{V}^B)$ de la ecuación (3.21) teniendo en cuenta que un producto vectorial puede expresarse como el producto de una matriz anti-simétrica y un vector tal y como se muestra en la ecuación (3.25), se obtiene la equivalencia de la ecuación (3.26).

$$a \times b = [b]_{\times}^{T} a = \begin{bmatrix} 0 & b_3 & -b_2 \\ -b_3 & 0 & b_1 \\ b_2 & -b_1 & 0 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \tag{3.25}$$

$$\omega^B \times (m\mathrm{V}^B) = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times m \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} 0 & mw & -mv \\ -mw & 0 & mu \\ mv & -mu & 0 \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{3.26}$$

Desarrollando el término $\omega^B \times (I\omega^B)$ de la ecuación (3.21) de la misma forma, se obtiene la equivalencia de la ecuación (3.27).

$$\omega^B \times (I\omega^B) = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} I_{XX} & 0 & 0 \\ 0 & I_{YY} & 0 \\ 0 & 0 & I_{ZZ} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} I_{XX} p \\ I_{YY} q \\ I_{ZZ} r \end{bmatrix} = \begin{bmatrix} 0 & I_{ZZ} r & -I_{YY} q \\ -I_{ZZ} r & 0 & I_{XX} p \\ I_{YY} q & -I_{XX} p & 0 \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{3.27}$$

Uniendo los resultados de las ecuaciones (3.26) y (3.27) se obtiene la matriz de aceleraciones centrífugas y de Coriolis $\mathrm{C_B}(\nu)$ que se muestra en la ecuación (3.28).

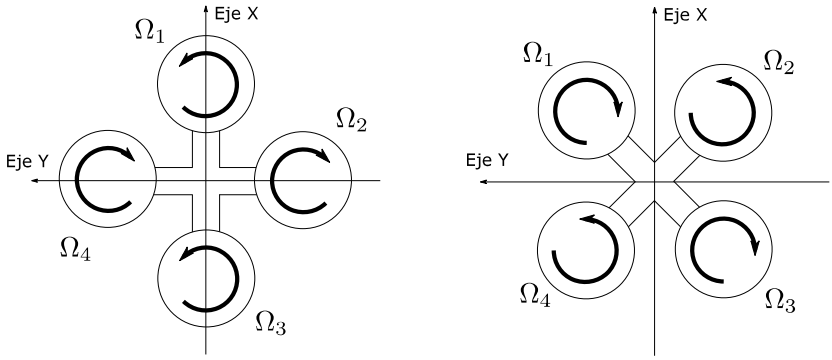
$$\mathrm{C_B}(\nu) = \begin{bmatrix} 0 & 0 & 0 & 0 & mw & -mv \\ 0 & 0 & 0 & -mw & 0 & mu \\ 0 & 0 & 0 & mv & -mu & 0 \\ 0 & 0 & 0 & 0 & I_{ZZ} r & -I_{YY} q \\ 0 & 0 & 0 & -I_{ZZ} r & 0 & I_{XX} p \\ 0 & 0 & 0 & I_{YY} q & -I_{XX} p & 0 \end{bmatrix} \tag{3.28}$$

Las principales fuerzas y pares que actúan sobre el cuadricóptero son tres, esto es, la fuerza gravitacional, los efectos giroscópicos producidos por la rotación de las hélices y las fuerzas y pares producidos directamente por las acciones principales de control del movimiento.

El vector de fuerzas gravitacionales $\mathrm{G_B}(\xi)$ resulta de la aceleración debida a la gravedad y por tanto sólo afecta a las ecuaciones lineales y no a las angulares, ya que no es un par. La obtención de este vector se puede observar en la ecuación (3.29), donde F_G^B es el vector de fuerzas gravitacionales en el marco de referencia B, F_G^E es el vector de fuerzas gravitacionales en el marco de referencia E y la notación 0_{3x1} designa a un vector vertical con cero en sus tres elementos.

$$\mathrm{G_B}(\xi) = \begin{bmatrix} F_G^B \\ 0_{3x1} \end{bmatrix} = \begin{bmatrix} R_\Theta^{-1} F_G^E \\ 0_{3x1} \end{bmatrix} = \begin{bmatrix} R_\Theta^{-1} \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} \\ 0_{3x1} \end{bmatrix} = \begin{bmatrix} mg \sin\theta \\ -mg \cos\theta \sin\phi \\ -mg \cos\theta \sin\phi \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{3.29}$$

La rotación de las hélices produce unos efectos giroscópicos sobre el cuadricóptero. Como se aprecia en la Figura 3.16, dos hélices giran en un sentido y las otras dos giran en el otro sentido con velocidades Ω_k. La suma de las cuatro velocidades puede ser distinta de cero produciendo un desequilibrio. Si además de esto también son distintas de cero las velocidades de rotación de Pitch o de Roll, el cuadricóptero experimenta un par giroscópico de acuerdo a la ecuación (3.30), que afecta únicamente a las ecuaciones angulares y no a las lineales.



(a) Configuración en + (b) Configuración en x

Figura 3.16: Velocidades y sentidos de rotación de las hélices.

$$
\begin{aligned}
O_B(v)\Omega &= \begin{bmatrix} 0_{3x1} \\ -\sum_{k=1}^{4} J_{TP}\left(\omega^B \times \begin{bmatrix}0\\0\\1\end{bmatrix}\right)(-1)^k \Omega_k \end{bmatrix} \\
&= \begin{bmatrix} 0_{3x1} \\ J_{TP}\begin{bmatrix}-q\\p\\0\end{bmatrix}\Omega_T \end{bmatrix} = J_{TP}\begin{bmatrix} 0_{3x1} \\ \begin{bmatrix}-q\\p\\0\end{bmatrix}(-\Omega_1+\Omega_2-\Omega_3+\Omega_4) \end{bmatrix} \\
&= J_{TP}\begin{bmatrix} 0_{3x1} \\ \begin{bmatrix} q\Omega_1 - q\Omega_2 + q\Omega_3 - q\Omega_4 \\ -p\Omega_1 + p\Omega_2 - p\Omega_3 + p\Omega_4 \\ 0 \end{bmatrix} \end{bmatrix} = J_{TP}\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ q & -q & q & -q \\ -p & p & -p & p \\ 0 & 0 & 0 & 0 \end{bmatrix}\Omega
\end{aligned}
\tag{3.30}
$$

En esta ecuación $O_B(v)$ es la matriz de efectos giroscópicos de la rotación de las hélices, Ω es el vector de velocidades de rotación de las cuatro hélices según se indica en la ecuación (3.31), Ω_T es la suma de las velocidades de rotación de las cuatro hélices según se indica en la ecuación (3.32) y J_{TP} es el momento de inercia rotacional total respecto al eje de rotación de las hélices que se obtiene a través de las ecuaciones del motor que genera dicho movimiento de rotación.

$$\Omega = [\Omega_1 \quad \Omega_2 \quad \Omega_3 \quad \Omega_4]^T \tag{3.31}$$

$$\Omega_T = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4 \tag{3.32}$$

Los movimientos principales del cuadricóptero son los identificados anteriormente: Throttle, Roll, Pitch y Yaw. Si se definen las entradas de control del cuadricóptero como las fuerzas o pares inducidos para generar estos movimientos se llega a que:

- U_1 es la fuerza de empuje aplicada al cuadricóptero para generar un movimiento de Throttle.
- U_2 es el par aplicado al cuadricóptero para generar un movimiento de Roll.
- U_3 es el par aplicado al cuadricóptero para generar un movimiento de Pitch.
- U_4 es el par aplicado al cuadricóptero para generar un movimiento de Yaw.

Por tanto se puede definir el vector de entradas de control del cuadricóptero U_B en el marco de referencia B según se indica en la ecuación (3.33).

$$U_B = [0 \quad 0 \quad U_1 \quad U_2 \quad U_3 \quad U_4]^T \tag{3.33}$$

Teniendo en cuenta la teoría aerodinámica, tanto las fuerzas como las pares inducidos son proporcionales al cuadrado de la velocidad de rotación de las hélices, por lo que U_B puede definirse en función de las velocidades de rotación de las hélices tal y como se ve en la ecuación (3.34).

$$U_B(\Omega) = E_B \Omega^2 \tag{3.34}$$

Con esta definición, U_B toma un valor u otro en función de la configuración del cuadricóptero. Para el caso de la configuración en + su valor se puede ver en la ecuación (3.35), mientras que para la configuración en x su valor se puede ver en la ecuación (3.36).

$$U_B(\Omega) = \begin{bmatrix} 0 \\ 0 \\ b\,(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\ b\,l\,(\Omega_2^2 - \Omega_4^2) \\ b\,l\,(\Omega_3^2 - \Omega_1^2) \\ d\,(\Omega_2^2 + \Omega_4^2 - \Omega_1^2 - \Omega_3^2) \end{bmatrix} \tag{3.35}$$

$$U_B(\Omega) = \begin{bmatrix} 0 \\ 0 \\ b\,(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\ b\,l\,(\Omega_2^2 + \Omega_3^2 - \Omega_1^2 - \Omega_4^2) \\ b\,l\,(\Omega_3^2 + \Omega_4^2 - \Omega_1^2 - \Omega_2^2) \\ d\,(\Omega_2^2 + \Omega_4^2 - \Omega_1^2 - \Omega_3^2) \end{bmatrix} \tag{3.36}$$

Por comparación entre las ecuación (3.34), (3.35) y (3.36) se puede obtener el valor de la matriz E_B tanto en la configuración en forma de +, ecuación (3.37), como en la configuración en forma de x, ecuación (3.38).

$$E_B = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ b & b & b & b \\ 0 & b\,l & 0 & -b\,l \\ -b\,l & 0 & b\,l & 0 \\ -d & d & -d & d \end{bmatrix} \tag{3.37}$$

$$E_B = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ b & b & b & b \\ -b\,l & b\,l & b\,l & -b\,l \\ -b\,l & -b\,l & b\,l & b\,l \\ -d & d & -d & d \end{bmatrix} \tag{3.38}$$

La matriz E_B es una matriz constante cuyos valores dependen de la aerodinámica de los rotores, de la forma de las hélices que utiliza el cuadricóptero y de la distancia de éstas a su centro de masas. En este sentido b es el factor de empuje, d es el factor de resistencia y l es la distancia desde el centro de la hélice al centro del cuadricóptero.

La derivación de estos factores a través de la teoría aerodinámica, en la que se combina la teoría del elemento de pala con la teoría de momentos, así como la derivación de las ecuaciones de los motores para la obtención del momento de inercia rotacional total respecto al eje de rotación de las hélices J_{TP} no se va a incluir en esta memoria pero puede consultarse en [14].

El vector generalizado de fuerzas Λ se forma a través de la fuerza gravitacional, los efectos giroscópicos producidos por la rotación de las hélices y las fuerzas y pares producidos directamente por las acciones principales de control del movimiento, según se muestra en la ecuación (3.39).

$$\Lambda = \mathrm{G_B}(\xi) + O_B(\nu)\Omega + U_B(\Omega) \tag{3.39}$$

Igualando las ecuaciones (3.23) y (3.39) se obtiene la ecuación (3.40).

$$\mathrm{G_B}(\xi) + O_B(\nu)\Omega + U_B(\Omega) = \mathrm{M_B}\dot{\nu} + \mathrm{C_B}(\nu)\nu \tag{3.40}$$

Despejando $\dot{\nu}$ de la ecuación (3.40) se obtiene la derivada del vector generalizado de velocidades en el marco de referencia B tal y como se muestra en la ecuación (3.41).

$$\dot{\nu} = \mathrm{M_B^{-1}}\left(-\mathrm{C_B}(\nu)\nu + \mathrm{G_B}(\xi) + O_B(\nu)\Omega + U_B(\Omega)\right) \tag{3.41}$$

Sustituyendo en la ecuación (3.41) los valores proporcionados por las ecuaciones (3.24), (3.28), (3.29), (3.30) y (3.33) se obtiene el sistema de ecuaciones (3.42) que describe la aceleración del cuadricóptero en el marco de referencia B.

$$\begin{cases} \dot{u} &= (v\,r - w\,q) + g\sin\theta \\ \dot{v} &= (w\,p - u\,r) - g\cos\theta\sin\phi \\ \dot{w} &= (u\,q - v\,p) - g\cos\theta\sin\phi + \dfrac{U_1}{m} \\ \dot{p} &= \dfrac{I_{YY} - I_{ZZ}}{I_{XX}} q\,r - \dfrac{J_{TP}}{I_{XX}} q\Omega_{\mathrm{T}} + \dfrac{U_2}{I_{XX}} \\ \dot{q} &= \dfrac{I_{ZZ} - I_{XX}}{I_{YY}} p\,r - \dfrac{J_{TP}}{I_{YY}} p\Omega_{\mathrm{T}} + \dfrac{U_3}{I_{YY}} \\ \dot{r} &= \dfrac{I_{XX} - I_{YY}}{I_{ZZ}} p\,q + \dfrac{U_4}{I_{ZZ}} \end{cases} \tag{3.42}$$

Habitualmente es más útil expresar la dinámica del cuadricóptero con respecto a un marco de referencia híbrido H compuesto de las ecuaciones lineales del marco de referencia E y las ecuaciones angulares del marco de referencia B. De esta forma, el vector generalizado de velocidades en este marco de referencia híbrido se define según la ecuación (3.43).

$$\zeta = [\dot{\Gamma}^E \quad \omega_B]^T = [\dot{x} \quad \dot{y} \quad \dot{z} \quad p \quad q \quad r]^T \tag{3.43}$$

La dinámica del cuadricóptero en el marco de referencia H puede reescribirse en forma matricial según se muestra en la ecuación (3.44), en la que $\dot{\zeta}$ es el vector generalizado de aceleraciones en el marco de referencia H.

$$\mathrm{G_H} + O_H(\zeta)\Omega + E_H(\xi)\Omega^2 = \mathrm{M_H}\dot{\zeta} + \mathrm{C_H}(\zeta)\zeta \tag{3.44}$$

Se puede ver claramente que la matriz de inercias del sistema $\mathrm{M_H}$ es igual tanto en el marco de referencia híbrido H como en el B. Por tanto, esta matriz queda como se indica en la ecuación (3.45).

$$\mathrm{M_H} = \mathrm{M_B} = \begin{bmatrix} mI_{3x3} & 0_{3x3} \\ 0_{3x3} & I \end{bmatrix} = \begin{bmatrix} m & 0 & 0 & 0 & 0 & 0 \\ 0 & m & 0 & 0 & 0 & 0 \\ 0 & 0 & m & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{XX} & 0 & 0 \\ 0 & 0 & 0 & 0 & I_{YY} & 0 \\ 0 & 0 & 0 & 0 & 0 & I_{ZZ} \end{bmatrix} \tag{3.45}$$

Sin embargo la matriz de aceleraciones centrífugas y de Coriolis en el marco de referencia híbrido H es diferente a la del marco de referencia B, siendo la que se muestra en la ecuación (3.46).

$$C_H(\zeta) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I_{ZZ}r & -I_{YY}q \\ 0 & 0 & 0 & -I_{ZZ}r & 0 & I_{XX}p \\ 0 & 0 & 0 & I_{YY}q & -I_{XX}p & 0 \end{bmatrix} \tag{3.46}$$

Con respecto al vector de fuerzas gravitacionales, se ve claramente que sólo afecta a la ecuación lineal que se corresponde con el eje Z del marco de referencia E. Esto puede verse en la ecuación (3.47).

$$G_H = \begin{bmatrix} F_G^E \\ 0_{3x1} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -mg \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{3.47}$$

La matriz de efectos giroscópicos de rotación de las hélices en el marco de referencia híbrido H es exactamente la misma que en el marco de referencia B porque sólo afecta a sus ecuaciones angulares. Esta matriz, por tanto, queda como se muestra en la ecuación (3.48).

$$O_H(\zeta)\Omega = O_B(v)\Omega = J_{TP} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ q & -q & q & -q \\ -p & p & -p & p \\ 0 & 0 & 0 & 0 \end{bmatrix} \Omega \tag{3.48}$$

Por su parte, la matriz E_H referenciada al marco híbrido H es diferente a la matriz E_B referenciada al marco B debido a que la matriz de rotación R_Θ provoca que la entrada de control U_1 afecte a las tres ecuaciones lineales. Esto puede verse en la ecuación (3.49).

$$E_H(\xi)\Omega^2 = \begin{bmatrix} R_\Theta & 0_{3\times3} \\ 0_{3\times3} & I_{3\times3} \end{bmatrix} E_B\Omega^2 = \begin{bmatrix} (\sin\psi\sin\phi + \cos\psi\sin\theta\cos\phi)U_1 \\ (-\cos\psi\sin\phi + \sin\psi\sin\theta\cos\phi)U_1 \\ (\cos\theta\cos\phi)U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} \tag{3.49}$$

Despejando $\dot{\zeta}$ de la ecuación (3.44) se obtiene la derivada del vector generalizado de velocidades en el marco de referencia híbrido H tal y como se muestra en la ecuación (3.50).

$$\dot{\zeta} = M_H^{-1}(-C_H(\zeta)\zeta + G_H + O_H(\zeta)\Omega + E_H(\xi)\Omega^2) \tag{3.50}$$

Sustituyendo en la ecuación (3.50) los valores proporcionados por las ecuaciones (3.45), (3.46), (3.47), (3.48) y (3.49) se obtiene el sistema de ecuaciones (3.51) que describe la aceleración del cuadricóptero en el marco de referencia H. En esta ecuación hay que tener en cuenta que las ecuaciones de las entradas de control y de las velocidades de rotación de las hélices son las mismas que en el marco de referencia B.

$$\begin{cases} \ddot{x} &= (\sin\psi \sin\phi + \cos\psi \sin\theta \cos\phi)\dfrac{U_1}{m} \\ \ddot{y} &= (-\cos\psi \sin\phi + \sin\psi \sin\theta \cos\phi)\dfrac{U_1}{m} \\ \ddot{z} &= -g + (\cos\theta \cos\phi)\dfrac{U_1}{m} \\ \dot{p} &= \dfrac{I_{YY} - I_{ZZ}}{I_{XX}} q\, r - \dfrac{J_{TP}}{I_{XX}} q\Omega_{\mathrm{T}} + \dfrac{U_2}{I_{XX}} \\ \dot{q} &= \dfrac{I_{ZZ} - I_{XX}}{I_{YY}} p\, r - \dfrac{J_{TP}}{I_{YY}} p\Omega_{\mathrm{T}} + \dfrac{U_3}{I_{YY}} \\ \dot{r} &= \dfrac{I_{XX} - I_{YY}}{I_{ZZ}} p\, q + \dfrac{U_4}{I_{ZZ}} \end{cases} \tag{3.51}$$

Es usual que en muchas aplicaciones de control de cuadricópteros se simplifique este modelo a otro que permita una inversión más sencilla. Se suelen realizar las siguientes consideraciones para la simplificación:

- Las contribuciones angulares resultan muy complejas porque se tienen en cuenta varias variables. Muchas de estas provienen del acoplamiento cruzado de las velocidades angulares (efectos giroscópicos y aceleraciones centrífugas y de Coriolis). Si se considera que el movimiento del cuadricóptero está cercano a la estabilidad en un punto (condición de *hovering*), los cambios angulares serán pequeños sobre todo en el caso del Pitch y del Roll. Debido a ello, los efectos giroscópicos y las aceleraciones centrífugas y de Coriolis resultan de un valor mucho más pequeño que las contribuciones angulares principales.
- Las aceleraciones angulares están referenciadas a los ángulos del cuadricóptero medidos en el marco de referencia B y sus valores no son iguales a las aceleraciones de los ángulos de Euler medidos en el marco de referencia E. La relación entre ellos viene definida a través de la matriz de transformación T_Θ. En condición de *hovering* esta matriz de transformación es casi idéntica a la matriz identidad por lo que las ecuaciones angulares pueden referirse directamente respecto a las aceleraciones de los ángulos de Euler.
- El control de bajo nivel del cuadricóptero se encarga de enviar las señales correctas a los rotores para estabilizar las aceleraciones angulares a los valores deseados U_2, U_3 y U_4, y para estabilizar la aceleración lineal en el eje Z al valor deseado U_1. Debido a que son cuatro rotores, no pueden controlarse más de cuatro variables en el bucle de control, por lo que se puede prescindir de dos de las ecuaciones. Habitualmente se decide controlar los ángulos de Euler y la altura, por lo que se prescinde de las ecuaciones de la aceleración angular en el eje X y en el eje Y.

Con estas consideraciones, el sistema de ecuaciones simplificado queda como se muestra en la ecuación (3.52).

$$\begin{cases} \ddot{z} &= -g + (\cos\theta \cos\phi)\dfrac{U_1}{m} \\ \ddot{\theta} &= \dfrac{U_2}{I_{XX}} \\ \ddot{\phi} &= \dfrac{U_3}{I_{YY}} \\ \ddot{\psi} &= \dfrac{U_4}{I_{ZZ}} \end{cases} \tag{3.52}$$

La suposición de que tanto las fuerzas como las pares inducidos son proporcionales al cuadrado de la velocidad de rotación de las hélices que se muestra en la ecuación (3.34) es razonable en condiciones cercanas al hovering, sin embargo en otro tipo de regímenes de vuelo con maniobras más agresivas hay que considerar los efectos que estas fuerzas tienen con respecto a la sustentación durante el movimiento y a la variación del

ángulo de ataque. Estos efectos no van a tenerse en cuenta en este proyecto ya que se asume la condición cercana al hovering, sin embargo se puede consultar sobre ello en [25].

De la misma forma tampoco se va a tener en cuenta el efecto suelo y el efecto techo que se producen cuando el cuadricóptero está volando muy cercano al suelo o muy cercano al techo, aproximadamente a una distancia equivalente al radio del rotor, ya que en los experimentos que se van a realizar no se volará a estas alturas

En el caso de que el cuadricóptero esté volando por debajo de esta distancia con respecto al suelo, la velocidad del flujo de aire inducido se reduce lo que conlleva un aumento del empuje que desplaza al cuadricóptero lejos del suelo. Puede consultarse acerca del efecto suelo en [13] y en [26].

Cuando un cuadricóptero está volando en el interior a una distancia menor que la indicada antes con respecto al techo, éste tiende a desplazarse hacia el techo lo que puede provocar un accidente contra él. Un ejemplo que demuestra experimentalmente este efecto puede consultarse en [27].

4. DESCRIPCIÓN DEL PARROT AR.DRONE 2.0

4.1. Generalidades

El AR.Drone 2.0 de la compañía Parrot [12] es la segunda versión de un cuadricóptero comercial muy vendido en el mercado debido a su precio asequible, a su robustez, a su gran estabilidad y a su fácil manejo a través de smartphones o tablets que lo hacen muy orientado al ocio. Sin embargo ofrece otras muchas características que lo hacen ideal para el control autónomo y por ello está muy presente en investigaciones al respecto, principalmente en aplicaciones que no requieren de maniobras agresivas. Todo esto lo hace perfecto, además, para aquellas personas que quieren iniciarse en el control autónomo de cuadricópteros.

4.1.1. Estructura

El cuadricóptero AR.Drone 2.0 está construido sobre una base de fibra de carbono y con piezas de plástico nylon muy resistente cargado con el 30% de fibra de vidrio de alta calidad. Para mejorar la robustez, utiliza espuma para aislar el centro de inercia de las vibraciones de los motores.



Figura 4.1: Parrot AR.Drone 2.0 con el casco de interior.



Figura 4.2: Parrot AR.Drone 2.0 con el casco de exterior.

Dispone de dos tipos de casco de protección, uno para exterior que puede verse en la Figura 4.1 que no cubre las hélices y uno interior que puede verse en la Figura 4.2 que sí las cubre. Ambos cascos están fabricados con polipropileno expandido inyectado por un molde metálico. Tiene una envergadura de 731mm y un peso de 420 gramos sin contar la batería con la carcasa de interior y 380 con la carcasa de exterior. Sus dimensiones también difieren en el caso de utilizar uno u otro casco. Con el casco de interior mide 517x517mm y con el de exterior mide 450x450mm.

Además es sencillo de modificar y reparar debido a que se dispone de un amplio catálogo de piezas.

4.1.2. Batería

La batería que viene de fábrica con el AR.Drone 2.0 es una LiPo (Polímero de Litio) de 11.1 V de 3 celdas con capacidad de 1000 mAh y capacidad de descarga de 10c. Esta batería pesa unos 100 gramos y proporciona un tiempo de vuelo aproximado de 10 minutos y tarda unos 90 minutos en cargar de manera completa.

Existen otras baterías vendidas por la compañía Parrot con las mismas características pero con una capacidad de 1500 mAh, y un peso algo mayor, lo que proporciona un tiempo de vuelo aproximado de 15 minutos.

Otras compañías tienen disponibles baterías compatibles de muy diversas capacidades y pesos.

4.1.3. Características de vuelo

AR.Drone 2.0 puede volar a una altura máxima de 100 metros aunque está limitada por el alcance del enlace de comunicación Wi-Fi con el control remoto.

Además, puede volar a una velocidad lineal máxima de 11.1 m/s.

4.1.4. Motores

Se compone de cuatro motores de rotor interno sin escobillas con rodamiento de bolas en miniatura y engranajes de Nystrom de bajo ruido para reductor de hélice de 1/8.75. Cada motor está comandado por un microcontrolador AVR de 8 MIPS totalmente reprogramable y resistente al agua.

Las hélices ejercen una alta fuerza de propulsión gracias a la que se mejora la maniobrabilidad. Además, sus ejes son de acero templado.

4.1.5. Placas electrónicas

Dispone de dos placas electrónicas:

- La placa principal incluye un procesador central ARM Cortex A8 de 32 bits a 1 GHz con DSP de vídeo TMS320DMC64x a 800MHz. Ejecuta Linux 2.6.32 junto con el software del AR.Drone 2.0 e incluye una memoria RAM DDR2 de 1 GB a 200 MHz. También incluye un chip Wi-Fi b/g/n de Atheros y un puerto USB 2.0 para instalar las actualizaciones y futuros desarrollos del dispositivo. Está equipada con un sensor de presión e incluye una cámara vertical, que opera a 60 FPS. El sensor de presión y la cámara vertical se utilizan en combinación con la retroalimentación del panel de navegación para proporcionar una absoluta estabilidad. El software que se ejecuta en el procesador de esta placa proporciona:
    - La estabilización y el pilotaje automático del AR.Drone 2.0.
    - La compresión y streaming de vídeo.
    - La gestión de la comunicación Wi-Fi con los controladores externos a través de su propia red de acceso.
- La placa de navegación está equipada con una serie de sensores para asegurar la estabilidad del AR.Drone 2.0 que se detallan en el apartado 4.1.6. Estos sensores proporcionan toda la información al AR.Drone 2.0. La fusión de estos datos proporciona los ángulos de Euler del cuadricóptero que se utilizan para su estabilización. El software de la placa de navegación puede ser actualizado a la vez que se actualiza el software del AR.Drone 2.0.

4.1.6. Sensores

El AR.Drone 2.0 dispone de los siguientes sensores y mecanismos de captación:

- Una unidad de medida inercial miniaturizada que se encuentra en la placa de navegación. Proporciona, entre otras, medidas del cabeceo (pitch), alabeo (roll) y de la guiñada (yaw), y se compone de:
    - Acelerómetro digital de 3 ejes (+/-50 mg) para monitorizar los movimientos de posición.
    - Giróscopo de 3 ejes de precisión (2000º/s) para los movimientos de cabeceo (pitch), alabeo (roll) y de la guiñada (yaw).
    - Magnetómetro de 3 ejes (precisión 6º) para fines de orientación.
- Un sensor de ultrasonidos (frecuencia de emisión de 40 kHz) que proporciona una medida de altitud con respecto al suelo. Rango de detección hasta 6 metros. Este sensor se encuentra en la placa de navegación y además dispone de un nanorevestimiento repelente a los líquidos.
- Un sensor de presión (+/-10 Pa) para permitir medidas de altitud a cualquier altura mayor de 6 metros. Este sensor se encuentra en la placa principal.
- Una cámara que se encuentra en la placa principal situada en la parte inferior del cuadricóptero y que apunta hacia abajo. Proporciona una medida de la velocidad con respecto al suelo. Es una QVGA (320x240) que proporciona un flujo de vídeo a 60 FPS. Las imágenes que proporciona esta cámara pueden ser escaladas para proporcionar un flujo de video a 640x360 o incluso a 1280x720.
- Una cámara que se encuentra en la parte frontal del cuadricóptero y que apunta hacia el frente. Es una HD (1280x720) de 92º de objetivo gran angular diagonal con sensor CMOS. El periodo del flujo de video puede ajustarse entre 15 y 30 FPS, y puede proporcionar imágenes a 1280x720 o a 640x360. Las imágenes están codificadas en H264 y el video se difunde en tiempo real de baja latencia.

Los acelerómetros y giróscopos se encuentran calibrados de fábrica mediante una optimización por mínimos cuadrados, pero además se realiza una calibración de forma previa a cada despegue debido a posibles desajustes en el vuelo o el aterrizaje anterior. Esta calibración sólo realiza correcciones para el Pitch y Roll.

El AR.Drone 2.0 posee algoritmos de fusión de los datos proporcionados por diferentes sensores para determinar una estimación más precisa de la altitud, velocidad, inclinación y orientación. En este sentido, y como puede apreciarse en la Figura 4.3, para estimar los ángulos de inclinación (Roll y Pitch) y de orientación (Yaw) se utilizan las medidas de los acelerómetros fusionadas con las de los giróscopos. Para estimar la velocidad lineal se utilizan las medidas de los acelerómetros fusionadas con los resultados obtenidos de un algoritmo aplicado a las imágenes obtenidas por la cámara vertical. Además, tal y como se pone de manifiesto en [28], la estimación de la velocidad lineal se basa en un modelo aerodinámico implementado internamente y que proporciona una estimación muy precisa. Para estimar la altura se utilizan las medidas del sensor de ultrasonidos fusionadas con las del sensor de presión.

Los giróscopos utilizan las estimaciones de inclinación y orientación como realimentación para establecer sus medidas de forma correcta. Igualmente estas estimaciones junto con las estimaciones de altura son utilizadas por el algoritmo de la cámara vertical para establecer las medidas de velocidad de forma correcta. Por su parte, los acelerómetros utilizan las estimaciones de velocidad como realimentación.

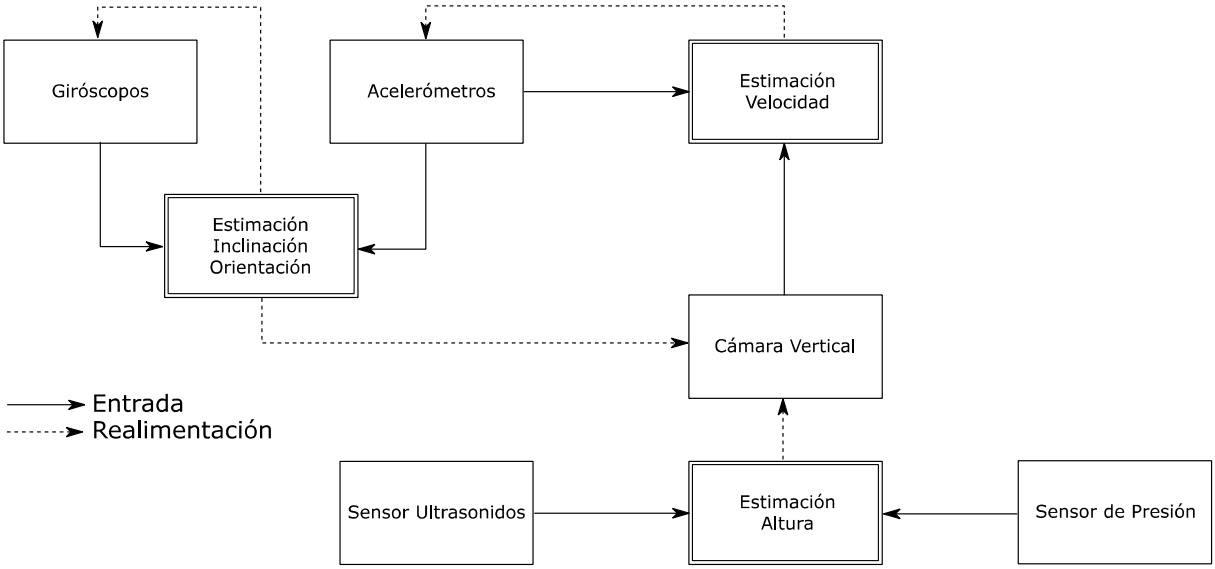


Figura 4.3: Fusión de sensores

4.1.7. Comunicaciones

Como se ha dicho anteriormente, el cuadricóptero AR.Drone 2.0 dispone de un chip Wi-Fi b/g/n gracias al cual puede ser controlado desde cualquier dispositivo que soporte Wi.Fi. En este sentido el AR.Drone 2.0 crea una red Wi-Fi a la que puede conectarse cualquier dispositivo asignándole una IP mediante el servidor DHCP del AR.Drone 2.0.

Dispone de tres servicios de comunicación:

- Para controlar y configurar el AR.Drone 2.0 se envían comandos AT (cadenas de texto) sobre UDP al puerto 5556. La latencia de transmisión de los comandos de control es crítica para la experiencia de usuario. Deben enviarse en bases regulares, usualmente 30 veces por segundo.
- La información del AR.Drone 2.0 (estado, posición, velocidad, etc.) es enviada por él al puerto 5554 del cliente sobre protocolo UDP. Es enviada aproximadamente 15 veces por segundo en modo demo y 200 veces por segundo en modo completo.
- Un flujo de video es enviado por el AR.Drone 2.0 al puerto 5555 del cliente sobre protocolo TCP.

Además, otro canal de comunicación extra puede establecerse en el puerto 5559 sobre protocolo TCP para enviar datos críticos como por ejemplo datos de configuración.

4.1.8. Acceso al Sistema Operativo

El cuadricóptero AR.Drone 2.0 ejecuta Linux 2.6.32 como sistema operativo en forma de BusyBox y crear una red Wi-Fi en la que posee la dirección IP 192.168.1.1.

A través del puerto 21 de su dirección IP puede accederse al servicio de FTP, pero lo que es más importante es su capacidad de conexión vía telnet al puerto 23. Gracias a esta conexión se puede acceder a las configuraciones del AR.Drone 2.0, a los registros de error y a otra información que puede ser útil y que no sería posible averiguar de otra forma. En la Figura 4.4 se muestra una captura del acceso mediante Telnet al sistema operativo donde se ve parte de la estructura de fichero.

Telnet 192.168.1.1

```
BusyBox v1.14.0 () built-in shell (ash)
Enter 'help' for a list of built-in commands.

# ls -ltr
drwxr-xr-x   12 root     root             0 Jan  1  1970 sys
dr-xr-xr-x   70 root     root             0 Jan  1  1970 proc
drwxrwxr-x    2 root     root           240 Jan  1  1970 licenses
drwxr-xr-x    5 root     root          2800 Jan  1  1970 lib
drwxrwxr-x    3 root     root          1256 Jan  1  1970 etc
drwxrwxr-x    4 root     root          5688 Jan  1  1970 bin
drwxrwxr-x    2 root     root          2752 Jan  1  1970 sbin
drwxrwxr-x    2 root     root           160 Jan  1  1970 root
drwxrwxr-x    2 root     root           160 Jan  1  1970 mnt
drwxrwxr-x    8 root     root           544 Jan  1  1970 usr
drwxrwxr-x    2 root     root           352 Jan  1  1970 var
drwxrwxr-x    3 root     root           224 Jan  1  1970 home
drwxr-xr-x    2 root     root           232 Jan  1 00:00 update
drwxrwxrwt    4 root     root          3500 Jan  1 00:00 dev
drwxr-xr-x    3 root     root           536 Jan  1 00:00 firmware
drwxr-xr-x    4 root     root          1096 Jan  1 00:00 data
drwxrwxrwt    3 root     root           200 Jan  1 00:00 tmp
drwxr-xr-x    2 root     root          1064 Jan  1 00:01 factory
# cd data
# ls -ltr
-rw-r--r--    1 root     root          5087 Jan  1 00:00 config.ini.old
-rw-r--r--    1 root     root        508051 Jan  1 00:00 syslog.bin
-rw-r--r--    1 root     root          5095 Jan  1 00:00 config.ini
-rw-r--r--    1 root     root            19 Jan  1 00:00 random_mac.txt
-rw-r--r--    1 root     root             2 Jan  1 00:00 old_adress.txt
drwxr-xr-x    5 root     root           368 Jan  1 00:01 custom.configs
-rw-r--r--    1 root     root            12 Jan  1 00:02 fact_accs_infos.bin
-rw-r--r--    1 root     root            20 Jan  1 00:07 fact_trims.bin
drwxr-xr-x    3 root     root           304 Jan  1 00:10 video
-rw-r--r--    1 root     root       2096628 Jan  1 00:12 syslog.bin.0
-rw-r--r--    1 root     root            20 Jan  1 00:15 trims.bin
-rw-r--r--    1 root     root            12 Jan  1  2000 accs_infos.bin
-rw-r--r--    1 root     root          6292 Jan  1  2000 emergency.bin
# cd custom.configs
# ls -ltr
drwxr-xr-x    2 root     root         47392 Jan  1  2000 sessions
drwxr-xr-x    2 root     root           400 Jan  1  2000 profiles
drwxr-xr-x    2 root     root           400 Jan  1  2000 applis
#
```

Figura 4.4: Captura del acceso por telnet al sistema operativo BusyBox del Parrot AR.Drone 2.0.

4.2. Arquitectura de control

En este apartado se va a describir el control externo del AR.Drone 2.0 que posibilita a un usuario manejarlo desde un dispositivo con capacidades Wi-Fi y por otro lado se va a ver el control interno que posee implementado en su firmware y que hace girar las hélices con las velocidades adecuadas para conseguir la referencia que el usuario ha dado a través del dispositivo externo.

4.2.1. Control externo

El AR.Drone 2.0 está creado para ser controlado de forma remota a través de una conexión Wi-Fi, principalmente mediante smartphones y tablets. Aunque inicialmente sólo podían utilizarse dispositivos Apple de este tipo para el control remoto, como el iPhone o el iPod Touch, después la compañía Parrot lanzó su versión de control remoto para dispositivos Android. La ventaja proporcionada por dispositivos de tipo smartphone o tablet radica en que suelen disponer de acelerómetros con los que es más sencillo para el usuario controlar la inclinación del ángulo de Pitch y de Roll únicamente inclinando el dispositivo de control remoto.

Sin embargo esta capacidad de ser controlado mediante una conexión Wi-Fi permite que cualquier otro dispositivo pueda actuar de control remoto si se conoce el protocolo de comunicación y sus comandos. Parrot ofrece una guía del desarrollador, que se detalla más ampliamente en el apartado 4.3, que proporciona todo lo necesario para crear aplicaciones que controlen el AR.Drone 2.0 desde cualquier dispositivo con capacidad Wi-Fi.

El cuadricóptero AR.Drone 2.0 dispone de varias funciones integradas como la del despegue autónomo, aterrizaje autónomo y la de hovering. Esta última consiste en que el cuadricóptero, gracias a los algoritmos de fusión de los sensores y cámaras de los que dispone, consigue mantenerse estable en el punto del espacio y con la orientación que tenía justo al activar dicha función.

El uso básico por parte de los usuarios comienza por activar la función de despegue autónomo haciendo que el cuadricóptero se sitúe a una altura prefijada quedándose a la espera de las señales de control remoto que el usuario le proporcione. Durante un vuelo el usuario puede activar o desactivar la función de hovering cuando lo desee, y para finalizar el usuario activa la función de aterrizaje autónomo haciendo que el cuadricóptero disminuya su altura suavemente hasta llegar al suelo y apagar los motores.

AR.Drone 2.0 permite realizar el control durante el vuelo a través de cuatro señales de control, cada una de las cuales ejecuta uno de los movimientos básicos de un cuadricóptero. Estas cuatro señales de control se envían de forma conjunta a través del comando AT*PCMD y son:

- Cabeceo (Pitch). Es un valor en el rango [-1..1] que supone un porcentaje del ángulo de inclinación máximo configurado en el AR.Drone 2.0. Un valor negativo hace que el cuadricóptero se desplace hacia adelante a lo largo de su eje X. Un valor positivo hace se desplace hacia atrás a lo largo de su eje X.
- Alabeo (Roll). Es un valor en el rango [-1..1] que supone un porcentaje del ángulo de inclinación máximo configurado en el AR.Drone 2.0. el cuadricóptero se desplace hacia la izquierda a lo largo de su eje Y. Un valor positivo hace se desplace hacia la derecha a lo largo de su eje Y.
- Velocidad angular de rotación sobre el eje Z (Yaw Speed). Es un valor en el rango [-1..1] que supone un porcentaje de la máxima velocidad angular configurada en el AR.Drone 2.0. Un valor negativo hace que el cuadricóptero rote sobre su eje Z en sentido antihorario mientras que un valor positivo hace que rote sobre su eje Z en sentido horario.
- Velocidad vertical (Throttle). Es un valor en el rango [-1..1] que supone un porcentaje de la máxima velocidad vertical configurada en el AR.Drone 2.0. Un valor negativo hace que el cuadricóptero disminuya su altura y un valor positivo hace que aumente su altura.

Si se envían cuatro ceros en este comando el AR.Drone 2.0 no realizará ninguna acción especial sobre sus motores para variar el Pitch, el Roll o el Yaw. Sin embargo debe tenerse en cuenta que de forma autónoma siempre intentará mantener la altura a la que se encuentra si la señal de control de Throttle es 0.

Los valores máximos a los que se hace referencia en las señales de control descritas se configuran mediante un comando AT*CONFIG, modificando los siguientes parámetros:

- *euler_angle_max*: Es el máximo ángulo de inclinación válido tanto para el cabeceo como para el alabeo. Es un parámetro positivo cuyo valor debería estar entre 0 y 0.52 radianes (30 grados).
- *control_vz_max:* Es la máxima velocidad vertical en milímetros por segundo. Los valores recomendados se sitúan entre 200 y 2000. Otros valores podrían causar inestabilidad.
- *control_yaw*: Es la máxima velocidad angular en radianes por segundo. Los valores recomendados se sitúan entre 40 vueltas por segundo (0.7 rad/s aproximadamente) y 350 vueltas por segundo (6.11 rad/s aproximadamente).

4.2.2. Control interno

Como se ha visto en el apartado 4.2.1 el cuadricóptero AR.Drone 2.0 dispone de cuatro entradas de control que establecen los ángulos de inclinación de Pitch y Roll (que provocan un movimiento a lo largo del eje X y a lo largo del eje Y respectivamente), la velocidad angular de rotación del Yaw con respecto al eje Z y la velocidad vertical a lo largo del eje Z.

La Figura 4.5 ha sido obtenida de [28] donde se realiza un estudio de la tecnología interna del cuadricóptero AR.Drone 1.0. A pesar de referirse este estudio a la versión 1.0, la versión 2.0 dispone de una tecnología similar

y, sobre todo, de la misma arquitectura de control interno. En ella se puede ver que para que el cuadricóptero pueda llegar a las consignas establecidas por el usuario, dispone de diferentes lazos de control interno y algoritmos que hacen uso de las medidas que le ofrecen sus sensores y de las imágenes captadas por su cámara vertical. Tras una serie de correcciones, este control interno determina la velocidad de rotación que debe tener cada una de las hélices y envía la señal adecuada a los motores para conseguirla.

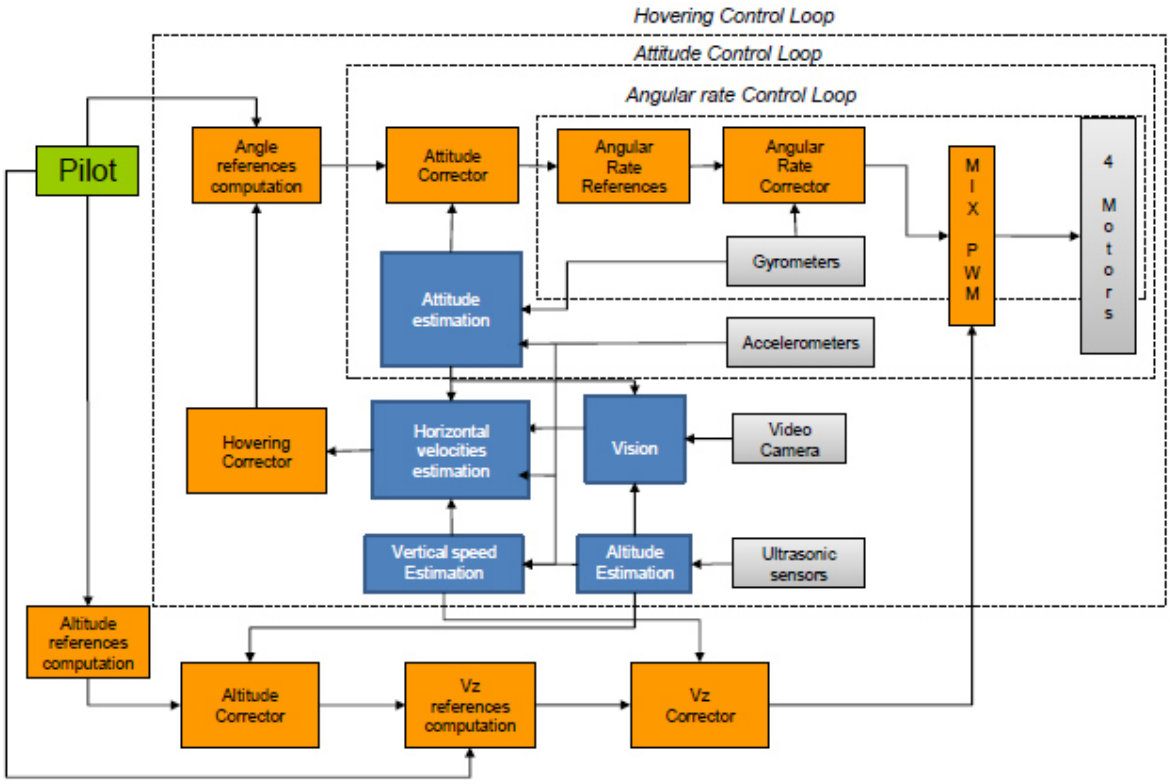


Figura 4.5: Arquitectura de control interno y fusión de sensores

El control interno está formado por varios lazos de control. El lazo de control de la inclinación y orientación está a su vez formado por un lazo de control de velocidades angulares. El primero de ellos calcula una consigna de las velocidades angulares gracias a la diferencia entre la consigna de Pitch, Roll y Yaw Speed con respecto a la inclinación y orientación estimadas. En este lazo de control se utiliza un controlador PI para conseguir su objetivo. El segundo lazo de control, el de las velocidades angulares, controla las señales enviadas a los motores a través de sencillos controladores P.

Otro lazo de control es el de hovering. Cuando el usuario tiene activada la función de hovering o cuando no está enviando ninguna señal de control al cuadricóptero, éste asigna de forma automática su consigna de velocidades lineales y de velocidad Yaw Speed a 0. Además, este lazo realiza el control durante la transición del modo de vuelo libre al modo hovering. En este lazo de control se utiliza un controlador PI para conseguir su objetivo.

El último lazo de control es el de la altura que calcula una corrección de las velocidades de los motores a través de la diferencia entre la consigna de altura y las estimaciones de altura y de velocidad vertical. En este lazo de control se utiliza un controlador PID para conseguir su objetivo.

4.3. Desarrollo de aplicaciones

4.3.1. Oficial

La compañía Parrot proporciona un SDK oficial con el que desarrollar fácilmente aplicaciones para controlar el AR.Drone. La versión estable más reciente del SDK en el momento de escribir esta memoria es la 2.0.1 [29], con la que se puede controlar las versiones 1.0 y 2.0 de AR.Drone. La última versión con la que no se puede controlar el AR.Drone 2.0 es la SDK 1.8 [30], que únicamente sirve para el AR.Drone 1.0.

El SDK puede utilizarse en dispositivos Apple iOS, Android, y en cualquier dispositivo PC que tenga conectividad Wi-Fi. En la documentación relativa al SDK 1.8 se indica que puede usarse en un PC con Linux o Windows, e incluso se proporciona un ejemplo de aplicación para Windows realizada con Microsoft Visual C++ 2008 Express Edition. Sin embargo, el SDK 2.0.1 indica que únicamente puede utilizarse con PCs Linux.

En caso de no poder utilizar el SDK o bien si se ha de desarrollar una aplicación en un entorno específico, se puede desarrollar un middleware propio para controlar el AR.Drone 2.0. Este middleware, para comunicarse con el cuadricóptero, utilizará los canales de comunicación descritos en el apartado 4.1.7 enviando los comandos que se especifican en el apartado 4.4 de esta misma memoria. Para conseguir un control satisfactorio Parrot aconseja que los comandos AT se envíen cada 30 ms, y para prevenir que el AR.Drone 2.0 considere que la conexión con el cliente se ha perdido, deben enviarse dos comandos consecutivos en menos de 2 segundos.

4.3.2. No oficial

Diversos desarrolladores no vinculados con Parrot han realizado sus propias aplicaciones middleware para controlar el cuadricóptero AR.Drone 2.0 en diferentes entornos y lenguajes de programación. De entre todos, se pueden destacar los siguientes:

- *ARDrone-Control-.NET*: Proyecto realizado en la plataforma .NET [31].
- *Javadrone*: Proyecto realizado en lenguaje Java [32].
- *AR Drone Simulink Development Kit:* Proyecto realizado para su funcionamiento en Matlab Simulink [33].
- ardrone_autonomy: Proyecto realizado para su funcionamiento en Robot Operating System (ROS) [34].
- tum_ardrone: Proyecto realizado para su funcionamiento en Robot Operating System (ROS) [35].
- *AR Drone Toolkit para LabVIEW - LVH:* Proyecto realizado para su funcionamiento en LabVIEW [36][37][38].

El presente proyecto se va a realizar en LabVIEW debido a que es un lenguaje que aporta simplicidad y rapidez en el desarrollo de interfaces gráficas, integración directa con el hardware y sobre todo una fácil creación de hilos y tareas de control en Tiempo Real. Pero también por aprovechar por completo el desarrollo ya existente del sistema de visión [39], y de la base y parte del control manual del desarrollo de una plataforma de control para un UAV Alien JumpJet de la empresa Snelflight [40].

Como se decidió realizar el desarrollo del presente proyecto en LabVIEW se van a utilizar las librerías que proporciona el middleware *AR Drone Toolkit para LabVIEW - LVH.*

4.4. Comandos

Cuando no se utilizan las librerías proporcionadas oficialmente por Parrot en su SDK para controlar el cuadricóptero, se puede realizar el control enviando directamente a los canales de comunicación Wi-Fi diversos comandos [29]. Estos comandos se denominan AT y se forman en texto plano codificados en ASCII de 8 bits y terminados con un carácter de retorno de carro CR. Un comando AT consiste en los tres caracteres AT*

seguidos del nombre del comando, un signo igual, un número secuencial y, opcionalmente, una lista de parámetros separados por comas cuyo significado depende del comando en particular.

Los parámetros pueden ser de tres tipos diferentes:

- Entero con signo con una representación decimal.
- Cadena de caracteres entre comillas.
- Valor en coma flotante de precisión sencilla con una representación IEEE-754. Sin embargo en los comandos AT estos valores deben ser convertidos a un entero con signo de 32 bits.

Un paquete UDP enviado al cuadricóptero puede contener uno o más de estos comandos separados por caracteres de retorno de carro CR.

El número secuencial debe incrementarse en uno cada vez que se envíe un nuevo comando al cuadricóptero teniendo en cuenta que el primer comando enviado debe tener un 1 en su número secuencial. A continuación se puede ver un ejemplo de todo lo dicho hasta ahora:

AT*PCMD_MAG=21625,1,0,0,0,0,0,0<CR>AT*REF=21626,290717696<CR>

Los comandos mal formados o aquellos cuya longitud sea mayor de 1024 caracteres son ignorados por el cuadricóptero.

A continuación se describen los comandos más importantes:

- AT*REF. Se utiliza para controlar el comportamiento básico del cuadricóptero en cuanto a despegue o aterrizaje, parada de emergencia o reinicialización.
- AT*PCMD. Se utiliza para enviar las señales de control de movimiento al cuadricóptero.
- AT*PCMD_MAG. Igual que el anterior pero proporciona parámetros para controlar el cuadricóptero en el modo de Control Absoluto, en el que se considera que todas las señales de control se envían respecto al marco de referencia del control remoto en lugar de al marco de referencia del cuadricóptero.
- AT*FTRIM. Se utiliza para indicar al cuadricóptero que se encuentra en reposo horizontal, lo que éste usará para determinar sus referencias internas en relación a los giróscopos. Debe enviarse siempre antes de comenzar un vuelo pero nunca durante él.
- AT*CALIB. Se utiliza para calibrar el magnetómetro. Debe enviarse cuando el cuadricóptero se encuentra en vuelo y éste hace automáticamente una rotación completa durante un breve momento.
- AT*CONFIG. Se utiliza para cambiar o establecer el valor a una opción de configuración.
- AT*CONFIG_IDS. Se utiliza para cambiar o añadir el valor a una opción de configuración cuando el cuadricóptero se encuentra en el modo de Multiconfiguración. Este modo permite al cuadricóptero almacenar configuraciones diferentes según la aplicación, usuario y/o sesión, y recuperarlas de una manera sencilla sin necesidad de reenviar de nuevo los valores de configuración.
- AT*COMWDG. Reinicializa el watchdog de las comunicaciones.

4.5. Parámetros de configuración, información de estado y datos de navegación

AR.Drone 2.0 dispone de multitud de parámetros de configuración divididos en diferentes categorías. Para conocer la lista de todos ellos con su descripción se puede consultar el apartado D.1 del ANEXO D.

AR.Drone 2.0 tiene la capacidad de enviar datos de navegación que pueden ser utilizados por el cliente para conocer los valores de las variables internas que maneja. Toda esta información puede resultar muy útil para el desarrollador de aplicaciones o para el investigador que requiere de su análisis para realizar un control autónomo. Los datos de navegación se envían en estructuras denominadas opciones de Navdata dentro de una estructura que las contiene. Hay multitud de opciones de Navdata y la que más suele utilizarse es la Navdata

Demo, que proporciona los valores de las variables de navegación básicas. Cuando se configura el AR.Drone 2.0 para que únicamente envíe este Navdata al cliente, lo hace a razón de 15 veces por segundo. Sin embargo, cuando se configura para recibir todos los Navdata existentes, lo hace a razón de 200 veces por segundo.

Para conocer en más detalle todos los Navdata que el AR.Drone 2.0 puede enviar al dispositivo de control remoto, así como las variables que contiene, se puede consultar el apartado D.3 del ANEXO D.

Cada vez que el AR.Drone 2.0 envía datos de navegación al cliente incluye información sobre su estado. Esta información se envía en una palabra de 32 bits donde cada uno de ellos tiene un significado. Para conocerlos se puede consultar el apartado D.2 del ANEXO D.

5. ENTORNO DE EXPERIMENTACIÓN Y ARQUITECTURA DEL SISTEMA

La identificación de los parámetros del modelo del AR.Drone 2.0 se consigue gracias a experimentos realizados en los que se registran múltiples datos. Estos datos son analizados generalmente mediante computadoras y el uso de software específico para ello, como Matlab de la compañía Mathworks, proporcionando un modelo que simula con suficiente precisión el comportamiento del cuadricóptero ante las señales de control que recibe.

Con este modelo de cuadricóptero obtenido experimentalmente se pueden realizar simulaciones de diferentes leyes de control con diferentes parámetros para obtener un controlador que consiga que el modelo del cuadricóptero se comporte tal y como se requiere.

Aunque la simulación es una herramienta indispensable en el diseño de leyes de control debido a que permite analizar de forma rápida y segura la respuesta del sistema, resulta prácticamente imposible modelar todos los detalles de un sistema físico real. Por ello se hace necesaria la utilización de un prototipo o un sistema real para experimentar y validar las leyes de control diseñadas ajustándolas ante posibles perturbaciones y dinámicas no modeladas, retardos en el sistema, incertidumbre en los parámetros, etc.

Como se ha visto en el capítulo anterior el AR.Drone 2.0 proporciona al dispositivo de control remoto información sobre velocidades lineales, angulares, etc. Estas velocidades pueden integrarse y obtener la posición en la que supuestamente se encuentra el cuadricóptero junto con la orientación de su parte frontal. Sin embargo la odometría presenta el problema de la acumulación de error al realizar la integración de las velocidades y por tanto no es un sistema efectivo para el control del posicionamiento o seguimiento de trayectorias de un vehículo autónomo. En ciertas aplicaciones este sistema puede resultar útil si se combina con otros métodos o sensores que le proporcionen de una forma más fiable la posición en la que se encuentra cada cierto tiempo, eliminando así el error acumulado. Este sistema de navegación se denomina *dead reckoning* o navegación por estima.

En el presente proyecto se va a aprovechar el trabajo realizado en investigaciones anteriores no relacionadas con vehículos aéreos no tripulados, utilizando las instalaciones y parte del hardware y software implementado en ellas. Por ello se realizarán los experimentos en el interior, evitando así las perturbaciones provocadas por el viento, y se utilizará un sistema de visión externo que proporciona el posicionamiento X e Y del vehículo así como su orientación en el marco de referencia E.

Las estrategias de control que se diseñen y validen en experimentos realizados en el interior utilizando este sistema de visión externo son fácilmente adaptables a vuelos en el exterior gracias a receptores GPS por ejemplo.

A continuación se va a analizar en detalle el entorno donde se van a realizar los experimentos con el cuadricóptero AR.Drone 2.0, tanto para identificar su modelo como para la validación de las leyes de control diseñadas.

5.1. Lugar de experimentación

El lugar donde se realizarán los experimentos es el hall de la sexta planta del edificio de la Escuela Técnica Superior de Ingeniería Informática de la Universidad Nacional de Educación a Distancia. La Figura 5.1 muestra una fotografía de este hall, que dispone de un sistema de visión que se compone de una cámara instalada en el techo apuntando hacia el suelo, y de un ordenador conectado a la cámara que trata las imágenes captadas. Aprovechando dicho sistema, este lugar ha sido y sigue siendo utilizado por investigadores y alumnos para realizar sus pruebas, prácticas o proyectos relacionados con el control autónomo y la navegación de vehículos.

El techo del hall se encuentra a una altura de 2.9 metros y la cámara instalada permite captar un área de observación de aproximadamente 3x4 metros de suelo desde una altura de 2.7 metros. Sin embargo hay que tener en cuenta que debido al sistema de visión, el área de observación varía en función de la altura de vuelo del cuadricóptero. A mayor altura de vuelo, la cámara puede captar menos área de observación, por lo que el vuelo del cuadricóptero se debe limitar a una altura determinada para permitir un área de observación lo suficientemente grande como para realizar los experimentos y que el AR.Drone 2.0 no se salga de ella.



Figura 5.1: Lugar de experimentación.

Debido a ello, si se considera una altura máxima de vuelo de 1.2 metros, la cámara permite captar un área de aproximadamente 1.6x2.2 metros. Aunque el espacio total en el que el AR.Drone puede volar durante los experimentos es muy pequeño, es suficiente para realizar las pruebas de control autónomo básicas consideradas en este proyecto.

El marco de referencia E definido en el lugar de experimentación puede verse en la Figura 5.2 y en él ha de tenerse en cuenta que la orientación 0° se encuentra sobre el eje X^E en su sentido negativo, y que la velocidad angular con respecto al eje Z^E (velocidad de giro del ángulo de Yaw) no se rige por la regla de la mano derecha y, por tanto, es positiva cuando gira en el sentido de las agujas del reloj y negativa cuando gira en el sentido contrario al de las agujas del reloj. Esta diferencia con respecto al marco de referencia E utilizado en el capítulo 3 para derivar el modelo general de un cuadricóptero deberá tenerse en cuenta de aquí en adelante.

5.2. Arquitectura del sistema

La arquitectura del sistema se muestra en la Figura 5.3 donde se pueden ver todos sus componentes. Consta de un cuadricóptero AR.Drone 2.0, una cámara que permite determinar la posición XY y la orientación del cuadricóptero por medio de un sistema de visión artificial, y de un ordenador, dotado de una tarjeta de adquisición de imágenes y de un dispositivo USB que permite comunicarse a través de redes Wi-Fi, en el que se implementan los algoritmos de visión y de control del cuadricóptero.



Figura 5.2: Marco de referencia E definido en el lugar de experimentación.

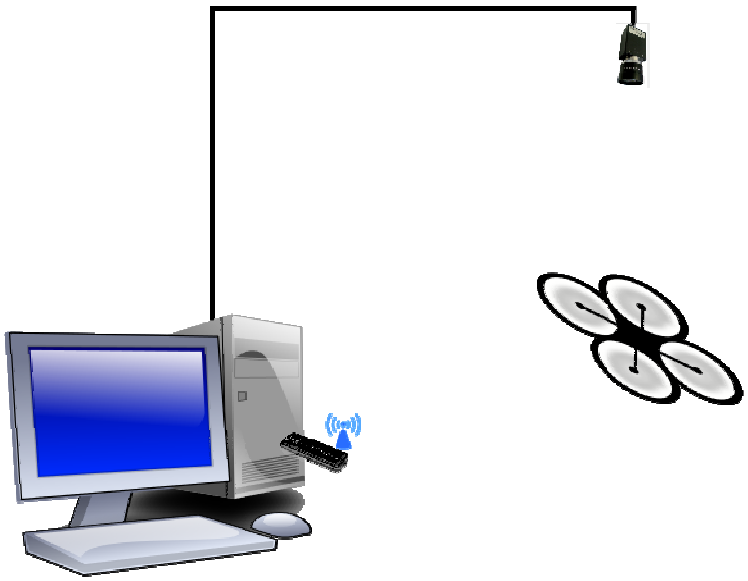


Figura 5.3: Arquitectura del sistema.

5.2.1. Cuadricóptero AR.Drone 2.0

El vehículo aéreo no tripulado que se va a utilizar en este proyecto es el AR.Drone 2.0 cuyas características han sido descritas en el capítulo 4. Sin embargo es necesario realizar una modificación sobre él para que el sistema de visión pueda determinar su posición XY y su orientación. Para ello se han instalado tres leds de alto brillo sobre el casco de interior dispuestos en forma de triángulo isósceles, tal y como se muestra en la Figura 5.4 y en la Figura 5.5, que hacen la función de baliza en el sistema de visión.



Figura 5.4: Triángulo de LEDs instalados en el Parrot AR.Drone 2.0.

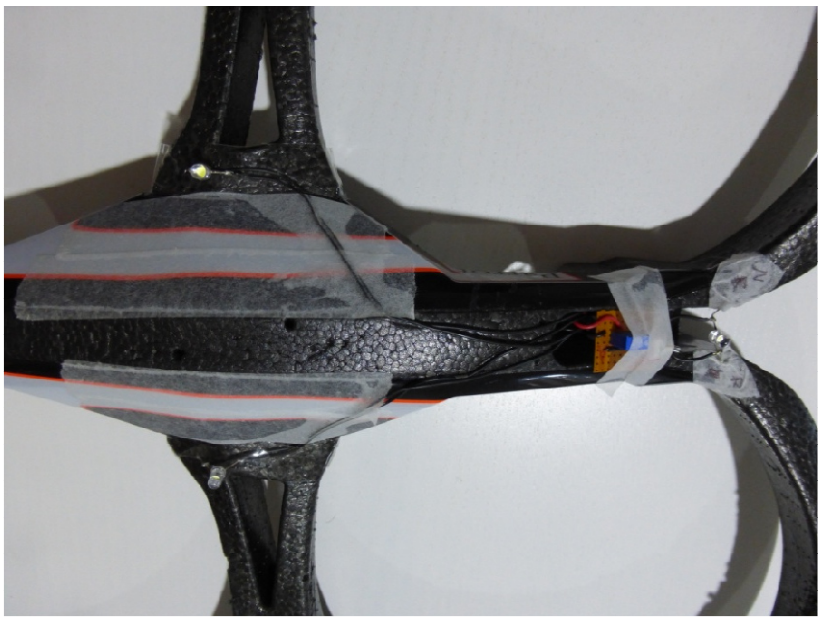


Figura 5.5: Detalle de la posición del triángulo de LEDs instalados.

El lado mayor del triángulo mide 20.2 cm y el lado menor 11.1 cm de tal modo que la relación de aspecto es 20.2/11.1. El centro del cuadricóptero se encuentra a 1 cm del punto medio de los dos leds que conforman el lado menor del triángulo, en sentido hacia el tercer led. Éste se encuentra dispuesto en la parte delantera del eje X del cuadricóptero con lo que el sistema de visión puede determinar la orientación.

Los leds están alimentados por la propia batería del AR.Drone 2.0 que proporciona una tensión V_1 de 11.1 V y están dispuestos eléctricamente según el esquema de la Figura 5.6. En él se puede ver que los tres leds se encuentran dispuestos en paralelo, y en serie con ellos se dispone de un potenciómetro R_1 ajustado a un valor de resistencia que permita circular la corriente necesaria para encender los leds con el brillo suficiente. El circuito se completa con un interruptor S_1 que permite realizar el encendido o apagado de los leds. El circuito físico resultante se ha implementado en una pequeña placa agujereada como se muestra en la Figura 5.7, a la que le llegan los cables desde la batería y desde el que parten los cables hacia los leds.

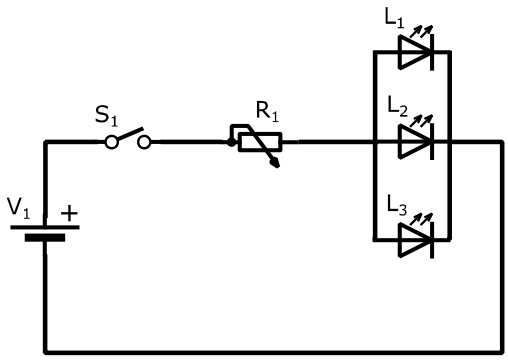


Figura 5.6: Esquema eléctrico del triángulo de LEDs.

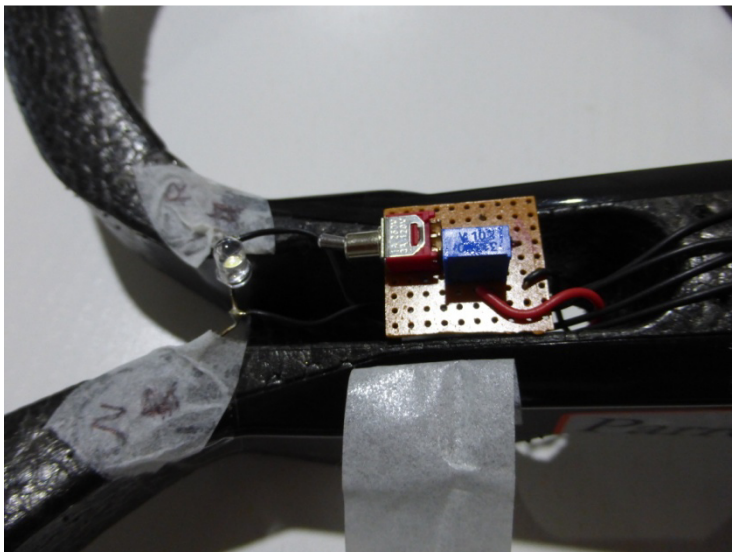


Figura 5.7: Placa de conexión que contiene el interruptor y el potenciómetro.

Se han utilizado leds LW540A que emiten luz blanca de alta luminosidad con un ángulo de 40°. Es necesaria la emisión de luz en un ángulo tal que la cámara capte la luz aún cuando el cuadricóptero está inclinado durante el vuelo. El resto de características eléctricas y ópticas se muestran en la Tabla 5.1 donde se puede ver que la condición típica de uso consiste en hacer pasar una intensidad I_L de 20 mA. En estas condiciones la caída de tensión directa V_L en el led es de 3.3 V.

Teniendo esto en cuenta, y según la disposición eléctrica de los componentes, la caída de tensión en la resistencia R_1 es igual a $V_{R1} = V_1 - V_L = 7.8$ V, y la intensidad que debe pasar por ella es de $I_{R1} = 3I_L = 60$ mA. Por la Ley de Ohm, el valor de la resistencia $R_1 = V_{R1}/I_{R1} = 130\ \Omega$.

Parameter	Symbol	Condition	Min	Typ	Max	Unit
Luminous Intensity *3	I_V	If=20mA	3500	6000		mcd
Luminous Flux	Φ_V	If=20mA	-	4	-	lm
Chromaticity Coordinate*4	x, y	If=20mA	X=0.31, y=0.31			-
Forward Voltage *5	V_F	If=20mA	-	3.3	3.8	V
View Angle	$2\theta_{½}$	If=20mA	40			Deg.
Optical Efficiency	η_{elc}	If=20mA	-	60	-	lm/W
Reverse Current	I_R	V_R = 5V	-	-	5	μA

Tabla 5.1: Características eléctricas y ópticas del led LW540A.

Como se ha indicado en el apartado 4.2.1 las entradas de control externo del AR.Drone 2.0 se encuentran en el rango $[-1..1]$, donde los valores límite se corresponden con un valor máximo configurado. Tras realizar unos vuelos iniciales de prueba y teniendo en cuenta que el presente proyecto no está pensado para que el cuadricóptero realice vuelos con inclinaciones agresivas o altas velocidades, se configuran estos valores máximos del AR.Drone 2.0 según se muestra en la Tabla 5.2, manteniéndose así en todas las fases del presente proyecto.

Configuración	Valor
Máxima inclinación de Pitch	15 °
Máxima inclinación de Roll	15 °
Máxima velocidad vertical de Throttle	0.7 m/s
Máxima velocidad angular de Yaw	100 °/s

Tabla 5.2: Valores máximos configurados de las entradas de control.

Se ha elegido una frecuencia de 50 Hz en el envío de las señales de control, enviando así una señal nueva cada 20 ms. Este valor ha sido elegido tras realizar diversas pruebas mediante control manual, y permite una experiencia de usuario muy buena, reaccionando el cuadricóptero sin retraso a las órdenes del control manual. Una frecuencia menor podría introducir retrasos en la reacción mientras que una frecuencia mayor podría no dar suficiente tiempo al software a ejecutarse. Además, no sería recomendable una frecuencia mayor de 200 Hz debido a que los datos de navegación son enviados por el AR.Drone 2.0 con esta frecuencia tal y como se indica en el apartado 4.5.

5.2.2. Sistema de visión

El sistema de visión externo se encuentra descrito en el ANEXO A, y su desarrollo ha sido fruto del trabajo realizado en [39].

De forma breve, este sistema consiste en la captura de imágenes a través de una cámara situada en el techo que capta las luces del triángulo de leds instalado en el cuadricóptero, extracción de la posición de los puntos de luz de la imagen, corrección de la distorsión y la perspectiva que la propia cámara introduce en la imagen capturada, determinación de la posición y orientación del centro del cuadricóptero a través de la trigonometría y de la transformación de dichos puntos al sistema de referencia E, así como de la estimación de sus velocidades lineales y angulares.

Para la adquisición de imágenes se utiliza la cámara CCD en color de alta resolución de Basler A631fc. Aunque se trata de una cámara en color se trabaja con ella en modo monocromo para aprovechar al máximo la resolución espacial y la velocidad de disparo.

En estas condiciones se obtienen imágenes con una resolución de 1388x1038 pixeles a una velocidad de 18.7 fps. La cámara se encuentra montada en un soporte anclado al techo del hall de tal modo que se realiza la observación del suelo desde una posición cenital. La cámara se conecta con el ordenador mediante un cable de enlace Firewire (IEEE 1394a) de 10 metros de longitud que se encuentra anclado al techo.



Figura 5.8: Cámara y óptica montada

Sobre esta cámara se monta una óptica de gran apertura Pentax de 4.8mm de distancia focal y 1:1.8 de apertura. Con este sistema es posible cubrir un área de observación de aproximadamente 3x4 metros en el suelo desde una altura de observación de 2.7m. No obstante, el disponer de un campo visual tan grande (aproximadamente 86º) tiene como contrapartida una gran distorsión en la imagen que debe ser corregida por el algoritmo de visión artificial.

5.2.3. Equipo informático

El PC con el que se adquieren y tratan las imágenes de la cámara y con el que se realiza el control del cuadricóptero dispone de un procesador Intel Core i3-2100 de 2 núcleos y 4 subprocesos a 3.10 Ghz y con 4 GB de memoria principal. El sistema operativo instalado es Microsoft Windows 7 Professional de 64 bits con Service Pack 1.

Para la adquisición de imágenes se utiliza una tarjeta PCI de adquisición (*frame grabber*) Matrox Meteor2-1394 controlada a través de unas librerías especiales de procesamiento de imagen MIL (Matrox Image Library) modificadas que se adjuntan junto al resto del código fuente del proyecto.

Para la comunicación con el cuadricóptero se dispone de un dispositivo USB conectado al ordenador que permite conectarse a redes Wi-Fi y comunicarse a través de ellas.

6. DESARROLLO DE LA PLATAFORMA SOFTWARE

El software del sistema ha sido desarrollado en LabVIEW 2013 utilizando funcionalidades proporcionadas por los módulos *Vision Development Module*, *Control Design and Simulation Module* y *MathScript RT Module*.

Las razones por las que se ha elegido este lenguaje son la simplicidad y rapidez en el desarrollo de interfaces gráficas, la integración directa con el hardware y sobre todo la simplicidad a la hora de crear hilos y tareas de control en Tiempo Real.

Otra razón por la que se ha decidido usar este lenguaje es que así puede aprovecharse por completo el desarrollo ya existente realizado en LabVIEW para el sistema de visión [39], así como la base y parte del control manual realizado en LabVIEW por el mismo autor del presente proyecto en el desarrollo de una plataforma de control para el vehículo aéreo no tripulado *Alien JumpJet* de la empresa *Snelflight* y que puede consultarse en [40].

Como se ha indicado en el apartado 4.3.2 se ha utilizado la librería *AR Drone Toolkit para LabVIEW - LVH* que proporciona las funciones necesarias para comunicarse con el AR.Drone 2.0 y así poder configurarlo, manejarlo y obtener datos de navegación sin necesidad de implementar toda esta funcionalidad. Aunque la última versión oficial de esta librería es la v0.1.0.34 [37], en el presente proyecto se ha utilizado la versión v0.1.0.36 que se encuentra en fase beta pero es muy estable, y que fue liberada por su autor a través del foro oficial de dicha librería en la página de National Instruments [38].

La novedad de esta última versión es que permite la captura de imágenes de las cámaras del AR.Drone 2.0 y pueden ser tratadas con algoritmos de visión para muy diferentes aplicaciones. A pesar de que en el presente proyecto no se utiliza esta característica, se ha visto adecuado dejar el software preparado para el tratamiento de las imágenes de las cámaras del cuadricóptero, por si fuera necesaria en posteriores investigaciones y desarrollos.

La instalación de la librería *AR Drone Toolkit para LabVIEW – LVH* se realiza a través del gestor de paquetes *VI Package Manager*.

Desafortunadamente la librería contiene diferentes errores en lo relativo a la inicialización y configuración del cuadricóptero que ha sido necesario subsanar mediante la modificación de algunos de sus bloques y la creación de otros nuevos. Además, ha sido necesario crear nuevos bloques para la obtención de datos de navegación ya que la librería sólo tenía implementada la obtención de los datos de navegación básicos. Las modificaciones de bloques de esta librería y los nuevos bloques creados en relación a ella se detallan en los apartados B.3, B.4.2 y B.4.5 del ANEXO B.

El desarrollo de la plataforma software que permite el control del cuadricóptero AR.Drone 2.0 ha sido realizado de forma modular, diferenciándose cuatro grandes sistemas:

- Sistema principal.
- Sistema de visión.
- Librería *AR.Drone Toolkit* junto con sus modificaciones.
- Variables globales.

El sistema principal, a su vez, consta de los siguientes subsistemas claramente diferenciados, y que para su funcionamiento se apoyan en otros bloques auxiliares:

- Módulo Principal.
- Módulo de Control Manual.
- Módulo de Control Automático.
- Módulo de Logs.
- Paneles de realización de pruebas.

En este capítulo se va a describir brevemente el desarrollo realizado poniendo especial énfasis en las interfaces con el usuario. Se van a detallar los estados por los que puede pasar el sistema y se va a describir de forma básica el funcionamiento interno. Toda la información aquí mostrada se completa gracias a la descripción del código fuente y el funcionamiento interno que se detalla en el ANEXO B, y gracias al manual de uso básico de la plataforma de control que se encuentra en el ANEXO C.

6.1. Panel principal

En la Figura 6.1 se puede ver el panel principal de la plataforma de control. En su parte superior se dispone de los siguientes controles:

- Botón "STOP": Sirve para terminar la ejecución del programa. Es conveniente haber cerrado la conexión con el AR.Drone de forma previa.
- Botón "Abrir Sesión": Sirve para establecer la conexión con el AR. Drone 2.0.
- Botón "Cerrar Sesión": Sirve para cerrar la conexión con el AR.Drone 2.0.
- Switch "Volar-Aterrizar": Sirve para dar la orden de despegue o de aterrizaje al AR.Drone 2.0.
- Switch "Hover": Sirve, en vuelo, para dar la orden al AR.Drone 2.0 de mantenerse en el estado de hovering o no mediante la funcionalidad que dispone de forma interna.
- Botón "EMERGENCIA": Cuando el AR.Drone 2.0 no despega a causa de algún problema o choque en el vuelo anterior, este botón sirve para volver su estado a la normalidad. Igualmente este botón se puede utilizar mientras está en vuelo para indicar al AR.Drone 2.0 que pare sus motores, por lo que hay que usarlo sólo en casos excepcionales.
- Botón "CALIBRAR": Sirve para calibrar el magnetómetro del AR.Drone 2.0. Sólo funciona cuando está en vuelo y en modo hovering.
- Lista desplegable "Control": Desde aquí se puede seleccionar la forma de controlar manualmente el AR.Drone 2.0: Teclado, Joystick, Joystick+Teclado 1, Joystick+Teclado 2 o Automático.
- Sliders "Pitch", "Roll", "Vertical Speed" y "Yaw Speed": Son las señales de control (multiplicadas por 10000) que se envían al AR.Drone 2.0. Pueden moverse con el ratón o a través del control manual seleccionado en la lista desplegable.
- Switch "AUTOMÁTICO": Mientras el AR.Drone 2.0 se encuentra en vuelo sirve para pasar al modo de control autónomo sin necesidad de seleccionarlo en la lista desplegable.

También en la parte superior se encuentra el indicador del estado en el que se encuentra el sistema, el switch "AUTOMÁTICO" que se utiliza para cambiar al modo de control autónomo y junto a él se dispone del indicador del estado del modo Automático en el que se encuentra el sistema.

La pantalla principal se encuentra conformada por una serie de pestañas que se describe a continuación:

- "Datos": En la Figura 6.2 se muestran todos los datos que pueden visualizarse en esta pestaña. A la izquierda se encuentra el estado del AR.Drone 2.0 y a continuación se encuentran los datos de posición, orientación y velocidad, entre otros, que proporciona el sistema de visión. Por último se encuentran los datos de navegación NavData proporcionados por el AR.Drone 2.0 que han sido implementados en este proyecto.

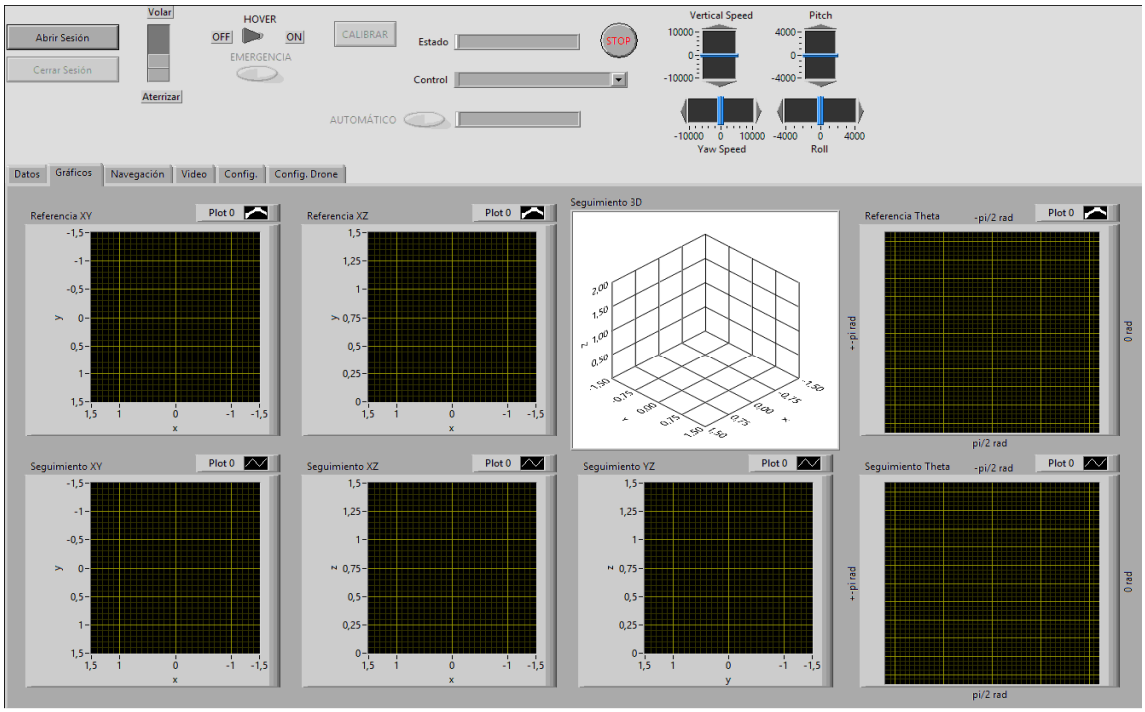


Figura 6.1: Panel principal.



Figura 6.2: Pestaña de Datos del panel principal.

- "Gráficos": En la Figura 6.1 se muestran los diferentes gráficos con los que se puede realizar el seguimiento de la posición y de la orientación del AR.Drone 2.0. El primer gráfico a la izquierda en la parte superior indica, mediante un punto, la referencia XY que en cada instante de tiempo debe seguir el cuadricóptero si se encuentra activado el modo Automático. Debajo de este gráfico se muestra la trayectoria XY que ha seguido el cuadricóptero. El segundo gráfico en la parte superior indica, igualmente, la referencia XZ y el gráfico de debajo de él muestra la trayectoria XZ que ha seguido el cuadricóptero. El tercer gráfico en la parte superior muestra la trayectoria en 3 dimensiones XYZ que ha seguido el cuadricóptero. En su parte inferior se en encuentra el gráfico que muestra la trayectoria YZ seguida por el cuadricóptero. El último gráfico de la parte superior indica, mediante un punto, la referencia de la orientación que en cada instante debe seguir el cuadricóptero si se encuentra activado el modo automático. El gráfico de la parte inferior muestra la trayectoria de la orientación seguida por el cuadricóptero.
- "Navegación": En la Figura 6.3 se muestran las diferentes opciones de navegación que dispone el programa para el control autónomo del AR.Drone 2.0. En su parte superior se encuentra el switch "Seguimiento Yaw" que indica si se quiere o no controlar el Yaw, el switch "Usar KF" que indica si se quiere o no usar el Filtro de Kalman, y un selector de lista desplegable que contiene los tipos de navegación posible: "Estabilización", "Waypoints", "Waypoints Tracking" y "Tracking". Para cada uno de ellos se corresponde una pestaña en la que se configuran sus parámetros, y que se describen a continuación:

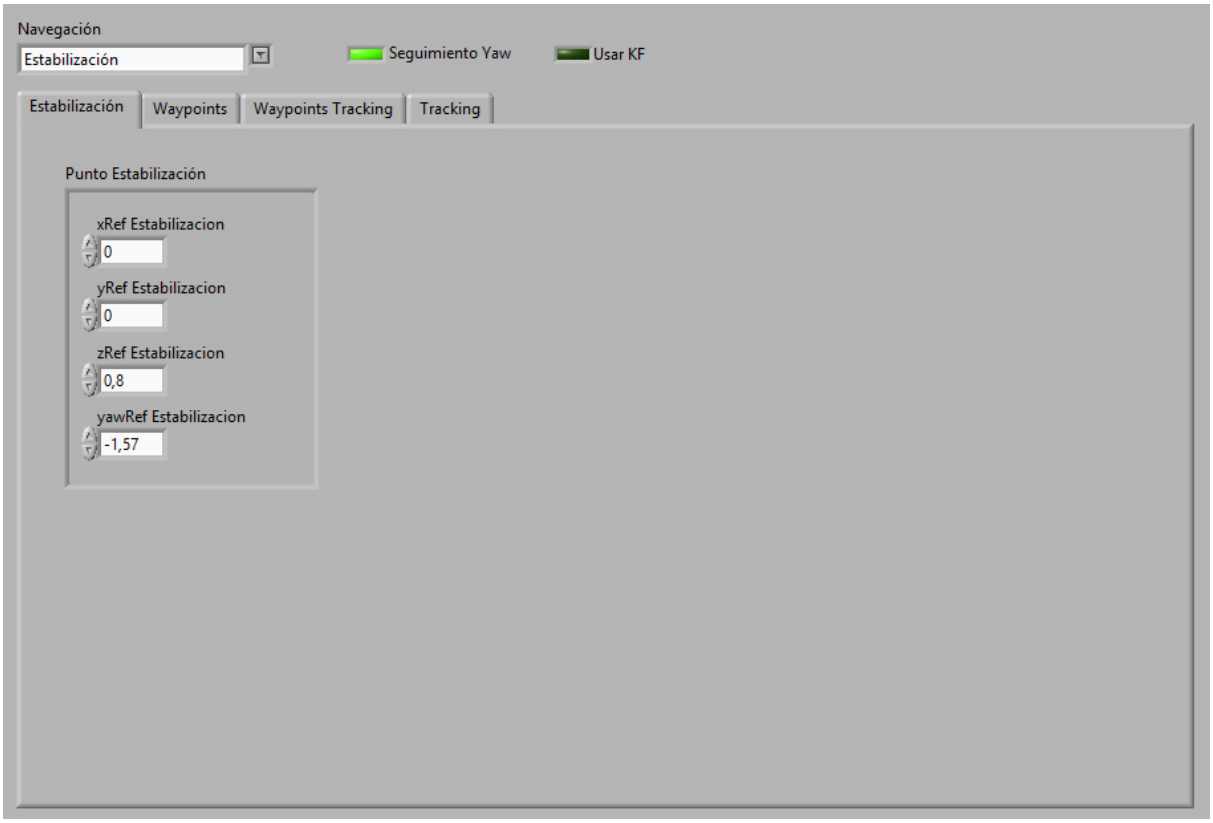


Figura 6.3: Pestaña de Navegación del panel principal.

- "Estabilización": En la Figura 6.3 se pueden ver los parámetros configurables para este tipo de control autónomo, que simplemente es un punto de referencia junto con su orientación.
- "Waypoints": En la Figura 6.4 se pueden ver los parámetros configurables para este tipo de control autónomo. Se configuran los distintos puntos de referencia por los que el AR.Drone 2.0 debe ir pasando, el tiempo de permanencia en cada uno de estos puntos y las tolerancias

máximas que determinan cuándo el cuadricóptero ha llegado al punto y comienza su tiempo de permanencia en él.

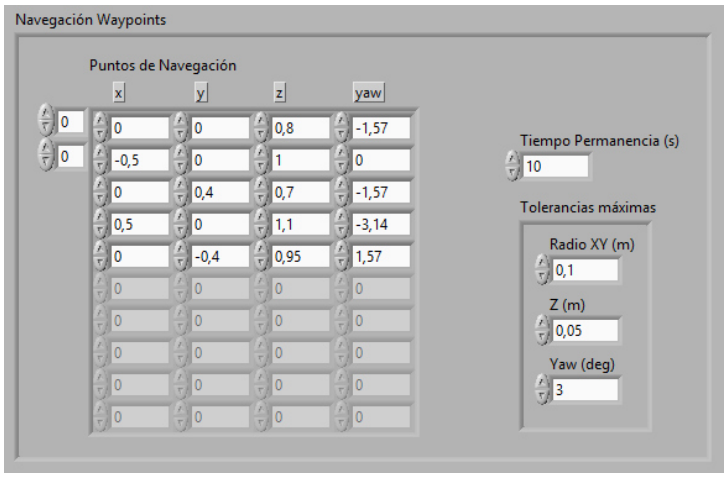


Figura 6.4: Parámetros de navegación a través de Waypoints.

- "Waypoints Tracking": En la Figura 6.5 se pueden ver los parámetros configurables para este tipo de control autónomo. Son los mismos que en el caso de la navegación a través de "Waypoints" pero se añade el tiempo que debe tardar en llegar de un punto al siguiente punto. Durante este tiempo, el cuadricóptero sigue la referencia en línea recta que une ambos puntos.

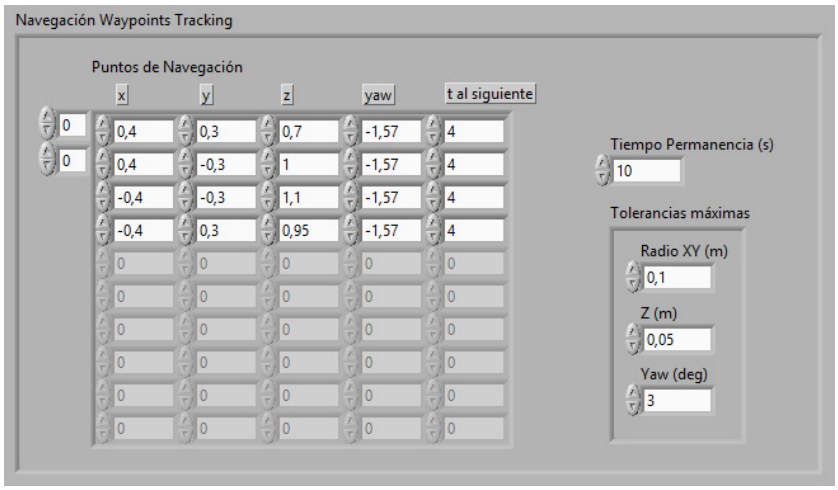


Figura 6.5 Parámetros de navegación a través de Waypoints con seguimiento de la recta entre ellos.

- "Tracking": En la Figura 6.6 se pueden ver los parámetros configurables para el tipo de control autónomo de seguimiento de trayectorias paramétricas. La trayectoria a seguir por el cuadricóptero viene aquí definida por las ecuaciones en función del tiempo t desde el que comenzó el estado "Automático" y de un valor k que se incrementa conforme va pasando el tiempo de incremento definido como parámetro. El parámetro de "Adelanto" define el tiempo de adelanto para calcular la referencia, mientras que el switch "Yaw Tangencial" define si se utiliza la ecuación escrita o si la referencia del Yaw debe ser tal que la parte frontal del AR.Drone 2.0 siempre esté mirando de forma tangencial a la trayectoria seguida según x e y.

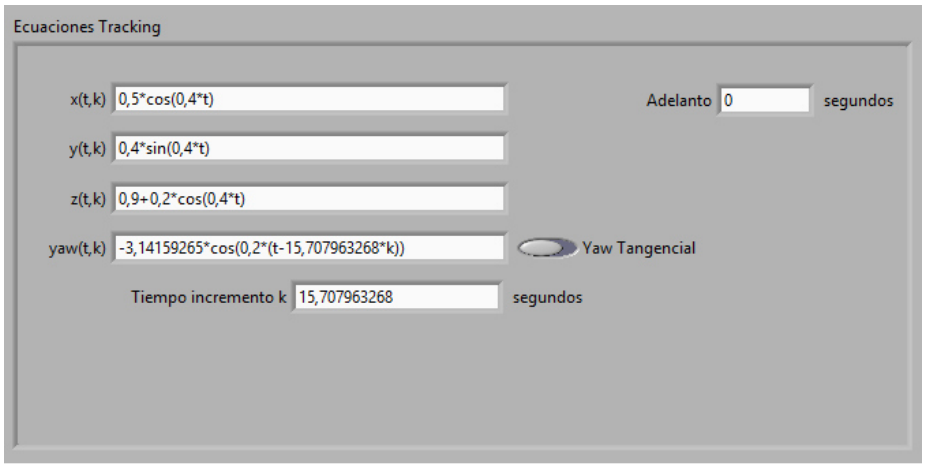


Figura 6.6: Parámetros de seguimiento de trayectorias paramétricas.

- "Video": En la Figura 6.7 se muestra el vídeo que envía el AR.Drone 2.0 a través de sus cámaras, aunque actualmente esta funcionalidad se encuentra deshabilitada porque no se usa en el presente proyecto.

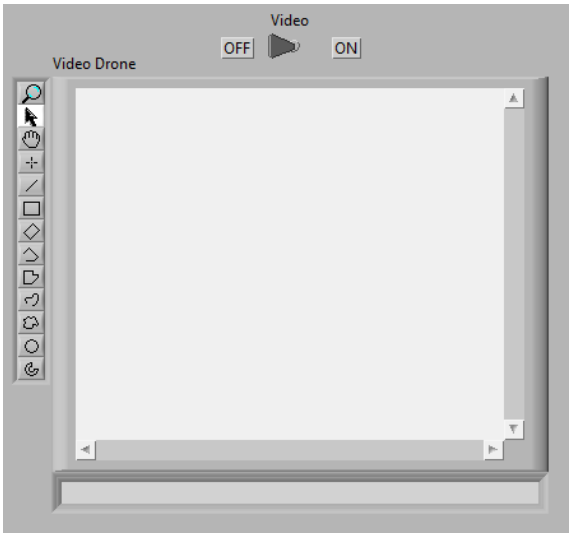


Figura 6.7: Pestaña de Video del panel principal.

- "Config.": En la Figura 6.8 se muestra el contenido de la pestaña de configuración, donde simplemente es posible configurar el incremento de valores del control manual por milisegundo. Aquí también se muestra el nombre del fichero de log que se ha generado para la actual sesión abierta.

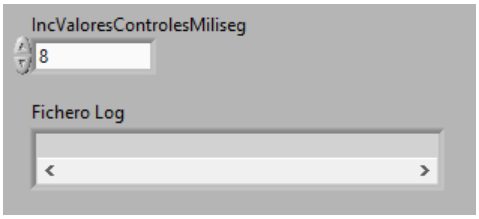


Figura 6.8 Pestaña de Configuración del panel principal.

- "Config. Drone": En la Figura 6.9 se muestra el contenido de la pestaña de configuración del AR.Drone 2.0. En ella pueden configurarse los parámetros básico del cuadricóptero: Direcciones IP, Modo de vuelo Interior/Exterior, Vídeo de qué cámara enviar, altura máxima de vuelo, así como los valores máximos de inclinación de Pitch y Roll, de velocidad vertical y de Yaw. Se recomienda no cambiar estos valores ya que la identificación y el control autónomo se han hecho en función de ellos.

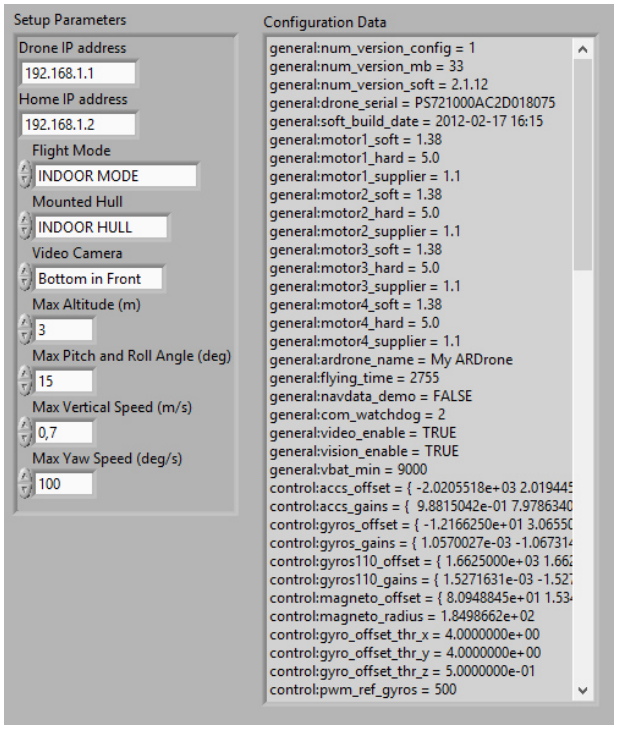


Figura 6.9 Pestaña de Configuración del AR.Drone 2.0 del panel principal.

6.2. Paneles para la realización de pruebas previas al uso o al desarrollo de nuevas funcionalidades de la plataforma

En algunos casos resulta útil la posibilidad de hacer pruebas de funcionalidades de la plataforma sin tener que arrancar el software o sin tener que poner en vuelo el cuadricóptero. En este apartado se van a detallar los paneles que posee la plataforma, que pueden ejecutarse de forma independiente a ella, y que permiten la realización de pruebas con diferentes objetivos.

6.2.1. Panel de prueba de los controles

Este panel, que se muestra en la Figura 6.10, permite realizar pruebas de los diferentes controles conectados al PC. Con él se puede verificar que las teclas pulsadas y los ejes movidos son interpretados correctamente por la plataforma y así evitar sorpresas desagradables cuando el cuadricóptero ya se encuentra en vuelo.

Este panel también resulta útil cuando se está incorporando un nuevo joystick o control manual distinto de los implementados en la plataforma por defecto. De esta forma puede probarse que la implementación creada en LabView para el nuevo joystick es interpretada de forma correcta por la plataforma.

6.2.2. Panel de prueba del teclado

Este panel, que se muestra en la Figura 6.11, permite conocer los valores internos de LabView con los que se corresponden las pulsaciones de teclas en el teclado. Esto resulta útil cuando se requiere implementar nuevas funcionalidades en la plataforma que necesiten del uso de nuevas teclas.

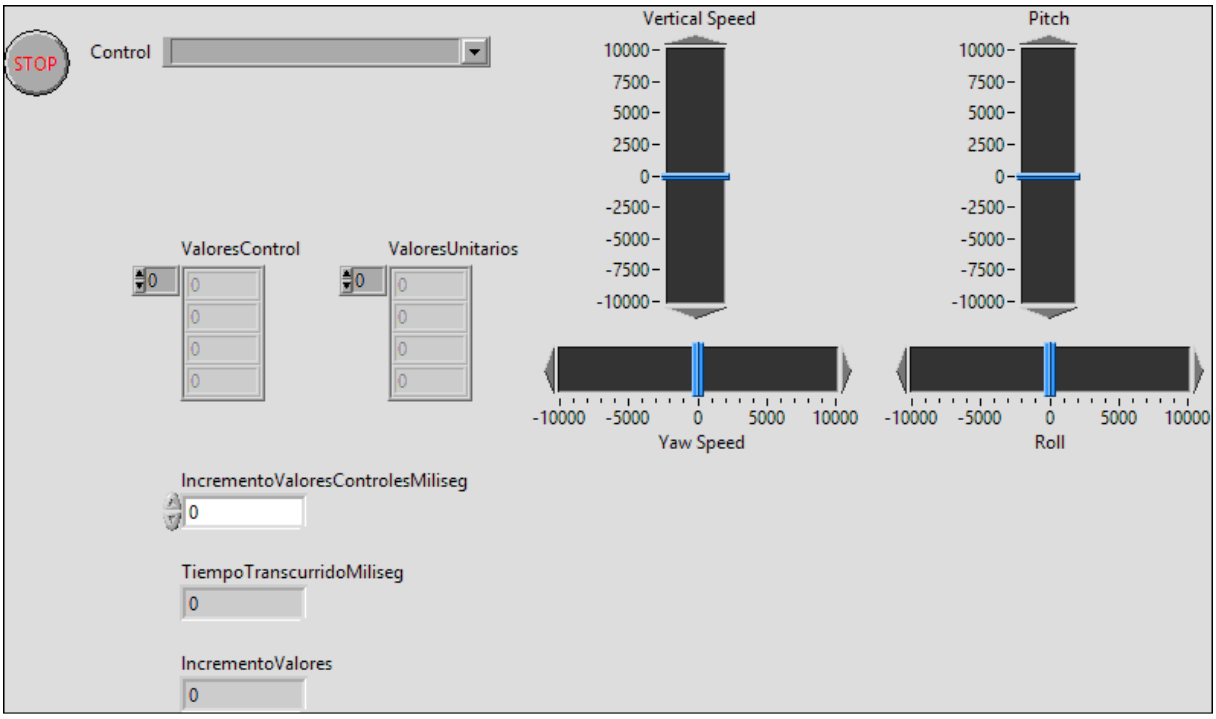


Figura 6.10: Panel para la realización de las pruebas de los controles.



Figura 6.11: Panel para la obtención de valores de teclas pulsadas.

6.2.3. Panel de prueba de los joysticks

Este panel, que se muestra en la Figura 6.12, permite conocer los valores internos de LabView con los que se corresponden las pulsaciones de botones de un joystick o el movimiento de sus ejes. Esto resulta útil cuando se requiere implementar nuevas funcionalidades en la plataforma que necesiten del uso de nuevos botones o ejes de un joystick ya incorporado en ella. Igualmente es útil cuando se necesita usar un nuevo joystick en la plataforma y deben crearse los bloques que interpreten sus acciones.

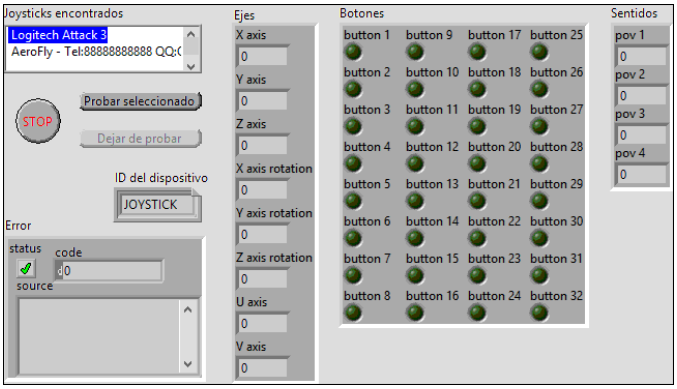


Figura 6.12: Panel para conocer qué botones y ejes se corresponden con los del joystick.

6.2.4. Panel de prueba de un controlador PID

Este panel, que se muestra en la Figura 6.13, permite realizar pruebas de un controlador PID implementado en su diagrama de bloques y verificar que la respuesta temporal y la acción de control son las esperadas cuando se aplican a un modelo también implementado en su diagrama de bloques.

Este panel resulta útil cuando se ha creado un controlador PID en alguna herramienta externa, como Simulink de Matlab, y antes de introducirlo en el control autónomo de la plataforma se desea verificar que su respuesta es la esperada.

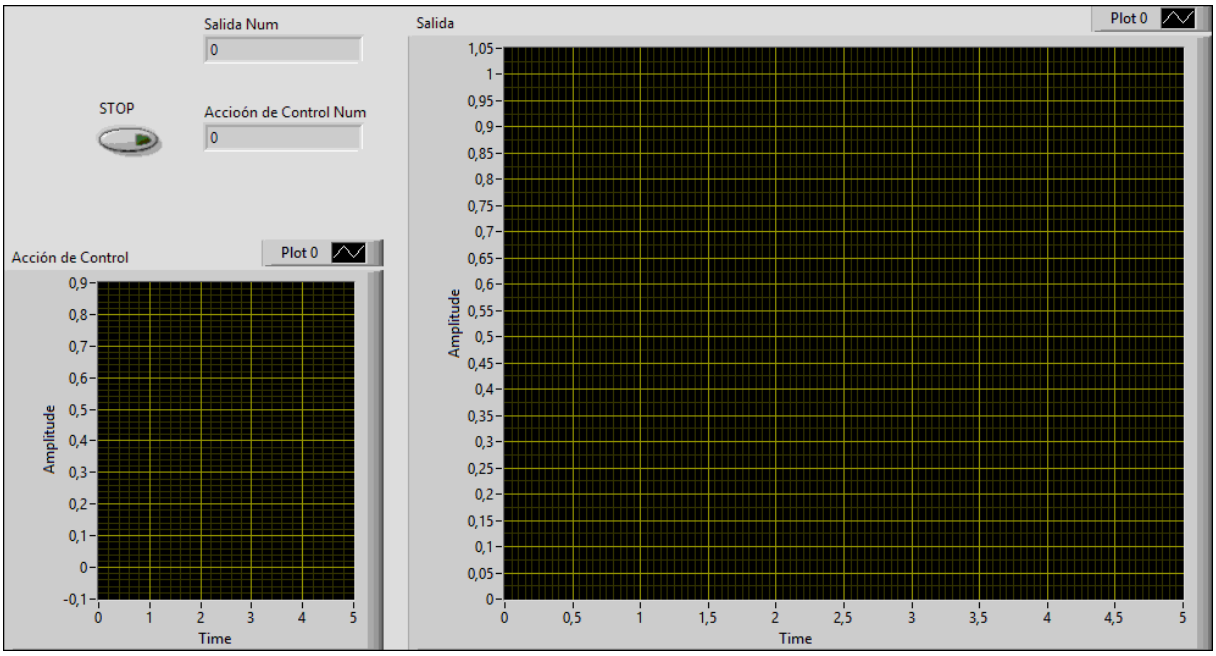


Figura 6.13: Panel de prueba de un controlador PID.

6.2.5. Panel de prueba de los controladores PID para el cuadricóptero

Este panel, que se muestra en la Figura 6.14, permite realizar pruebas de los controladores PID implementados en su diagrama de bloques y verificar que las respuestas temporales y las acciones de control son las esperadas cuando se aplican al modelo del cuadricóptero que se encuentra implementado en su diagrama de bloques.

Su utilidad es similar que en el caso anterior pero proporciona en una sola prueba las respuestas del Pitch, Roll, Throttle y Yaw del cuadricóptero.

6.2.6. Panel de prueba de trayectorias paramétricas

Este panel, que se muestra en la Figura 6.15, permite realizar pruebas de las trayectorias paramétricas que puede seguir el cuadricóptero al activar el modo de navegación "Tracking". La realización de estas pruebas de forma previa a la activación de dicho modo de navegación en automático es indispensable para evitar sorpresas desagradables si las ecuaciones no han sido definidas correctamente.

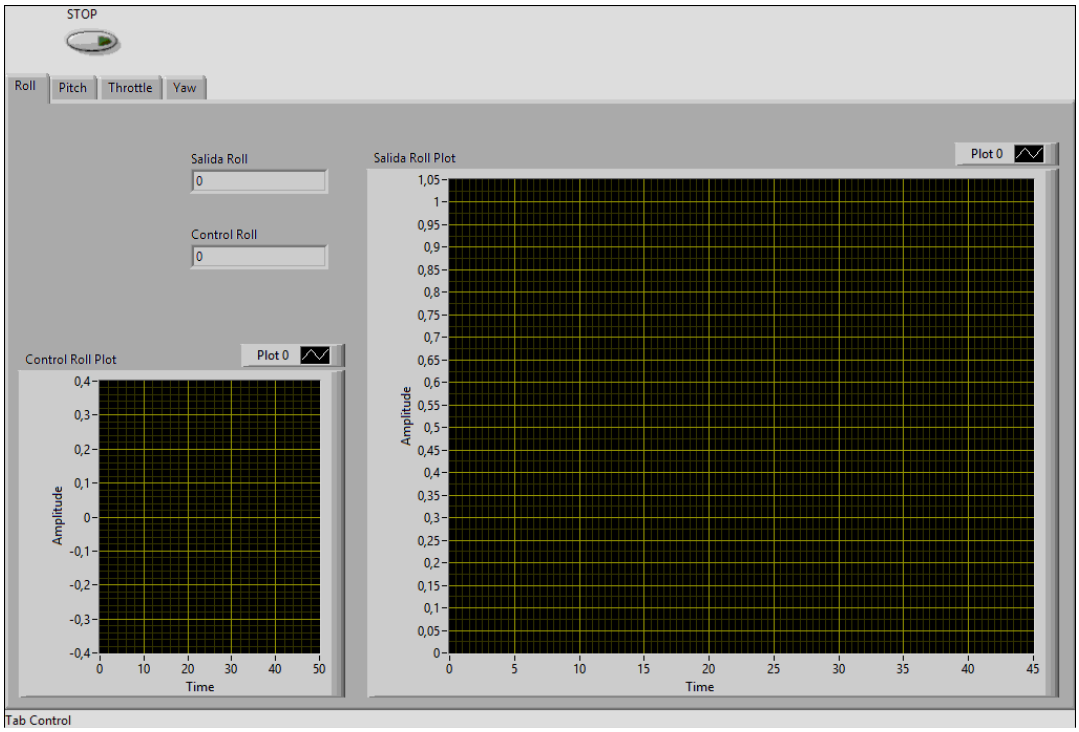


Figura 6.14: Panel de prueba de los controladores PID del cuadricóptero.

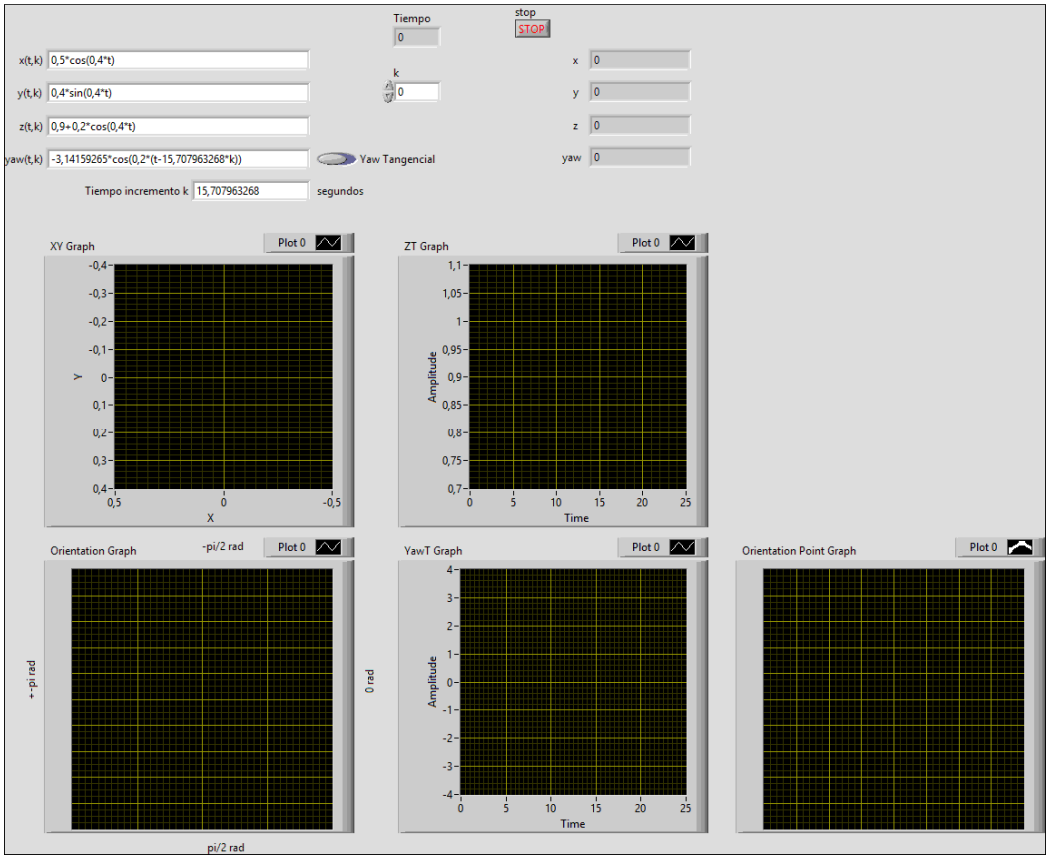


Figura 6.15: Panel para realizar pruebas de las trayectorias paramétricas.

6.3. Estados del sistema

La plataforma se basa en una máquina de estados, de modo que se realizan unas tareas u otras en función del estado en el que se encuentra el sistema. A continuación se describen las tareas que se realizan en cada estado:

- "Iniciando": Este es el estado inicial del sistema y es donde se inicializan ciertos valores de configuración y de los controles existentes en el panel principal. Tras realizar estas acciones el sistema pasa de forma automática al estado "ReposoPre".
- "ReposoPre": En él se habilitan o deshabilitan de forma conveniente los controles del panel principal y se inicializan ciertas configuraciones y valores de dichos controles. A continuación se pasa de forma automática al estado "Reposo".
- "Reposo": Este es el estado normal del sistema cuando no se encuentra conectado al AR.Drone 2.0. En él se está a la espera de la pulsación en el botón "Abrir Sesión" e inmediatamente al pulsarlo se pasa al estado "AbriendoSesionPre".
- "AbriendoSesionPre": Se prepara el sistema de forma previa a la apertura de sesión habilitando o deshabilitando de forma conveniente los controles del panel principal, e inicializando ciertas configuraciones y valores de dichos controles. Tras finalizar estas acciones el sistema pasa automáticamente al estado "AbriendoSesion".
- "AbriendoSesion": En este estado se realizan las siguientes tareas:
    - Se obtiene una referencia al bloque de la cámara y se ejecuta para así arrancar el sistema de visión.
    - A continuación se abre una sesión con el AR.Drone 2.0.
    - Seguidamente se crea un fichero de log cuyo nombre contiene la fecha y hora con minutos y segundos donde se almacenarán los datos obtenidos.
    - Por último se pasa al sistema al estado "MarchaPre" de forma automática.
- "MarchaPre": En este estado previo se habilitan o deshabilitan de forma conveniente los controles del panel principal, inicializándose ciertas configuraciones y valores de dichos controles. Posteriormente se inicializan las gráficas donde se mostrará el seguimiento visual del cuadricóptero. Después se inicializan ciertas variables globales para, por último, pasar de forma automática al estado "Marcha".
- "Marcha": Este es el estado normal cuando existe una sesión abierta con el AR.Drone 2.0. En él se realizan las siguientes tareas:
    - Se gestiona la pulsación del botón "CALIBRAR".
    - En cada instante de tiempo se obtienen los valores del control seleccionado en la lista desplegable.
    - A la vez se gestionan las fases del modo de control Automático, así como la navegación cuando dicho modo está activado.
    - Se obtiene o estima la posición actual del cuadricóptero.
    - Si el modo automático se encuentra activado, se obtienen las acciones determinadas por el controlador.
    - Tras realizar todo lo anterior se envían al AR.Drone 2.0 las acciones de control, ya sea del modo Automático o del control manual.
    - En paralelo se forma una línea de log con los datos actuales y se actualizan las gráficas de seguimiento con los datos actuales.

- Por último se gestiona la pulsación del botón "Cerrar Sesión", y si se ha pulsado se pasa al estado "CerrandoSesionPre".

- "CerrandoSesionPre": En este estado se habilitan o deshabilitan de forma conveniente los controles del panel principal, inicializándose ciertas configuraciones y valores de dichos controles. A continuación se pasa de forma automática al estado "CerrandoSesion".
- "CerrandoSesion": Durante el tiempo que se encuentra el sistema en este estado se cierra la conexión con el AR.Drone 2.0 y se envía, a través de una variable global una señal de STOP global para detener la ejecución del bloque de la cámara. A continuación se pasa de forma automática al estado "ReposoPre".
- "Salir": A este estado se llega cuando se ha pulsado el botón "STOP" de la pantalla principal. En él se intenta cerrar la conexión con el AR.Drone 2.0 por si siguiera abierta y se gestiona la salida de forma ordenada de los diferentes hilos de ejecución del programa.

La plataforma dispone de otra máquina de estados para gestionar el modo Automático. Estos estados sólo tienen sentido cuando el sistema se encuentra en "Marcha" y su descripción es la siguiente:

- "Manual": Indica que el control autónomo no se encuentra activado. Si durante este estado se pulsa el switch "AUTOMÁTICO" se pasa al estado de "Iniciando Auto".
- "Iniciando Auto": Se inicializan ciertas variables globales relacionadas con la navegación y el control automático, y a continuación se pasa automáticamente al estado "Esperando Auto".
- "Esperando Auto": Este estado tiene una duración de 1 segundo durante el cual se inicializa y comienza a ejecutarse el Filtro de Kalman para que pueda ofrecer mejores estimaciones una vez iniciado el control autónomo. Al terminar este tiempo se pasa automáticamente al estado "Automático" y se cambia el desplegable de control a "Automático".
- "Automático": Durante el tiempo que se encuentra en este estado se calculan y se envían las acciones de control determinadas por el controlador implementado. Si en este estado el switch "AUTOMÁTICO" es desactivado se pasa al estado "Manual".

6.4. Bucles de ejecución interna

El desarrollo del sistema se ha planteado de modo que se ejecuten de forma continua cuatro bucles en otros tantos hilos de ejecución paralelos que realizan diferentes funciones. Son:

- Bucle de ejecución principal del sistema.
- Bucle de actualización de datos de navegación.
- Bucle de actualización de vídeo proveniente del AR.Drone 2.0.
- Bucle de escritura del fichero de log.

El bucle de ejecución principal se ejecuta cada 20 ms y realiza las tareas en función del estado en el que se encuentra el sistema, y que se describió anteriormente. En él también se realiza la gestión de errores del AR.Drone 2.0, la gestión de errores de los controles manuales (teclado) en caso de no disponer de ninguno de ellos, y la gestión de la finalización ordenada del programa.

El bucle de actualización de los datos de navegación NavData se ejecuta de forma continua y se encarga de obtener los datos que el AR.Drone 2.0 envía al puerto de comunicaciones correspondiente, tratarlos y actualizar las variables globales que los almacenan. Las tareas de este bucle sólo se realizan si el sistema se encuentra en el estado "Marcha".

El bucle de actualización del vídeo del AR.Drone 2.0 se ejecuta también de forma continua y se encarga de obtener los datos que el AR.Drone 2.0 envía al puerto de comunicaciones correspondiente, tratarlos y

actualizar el bloque donde pueden verse las imágenes de las cámaras del cuadricóptero. Las tareas de este bucle sólo se realizan si el sistema se encuentra en el estado "Marcha".

El último bucle, de escritura de log, se ejecuta cada segundo y guarda en el fichero de log las líneas que se han ido formando en el bucle principal desde su última ejecución. Para ello se utilizan los bloques de colas de los que dispone LabView, de modo que desde el bucle principal se encolan las líneas que deben escribirse, y en el bucle de escritura de log se vacía la cola escribiéndolas al fichero. Este bucle se ejecuta independientemente del estado en el que se encuentre el programa ya que durante su finalización puede necesitar esperar a que transcurra un segundo más y almacenar el resto de líneas de log que quedaban. Si se ejecuta cuando aún no se ha abierto sesión con el cuadricóptero y, por tanto, aún no se ha creado el fichero de log, no pasa nada porque en dicho bucle se verifica si existe o no antes de intentar escribir en él.

De forma previa al comienzo de ejecución de los bucles se realiza la inicialización de la cola para el log, la inicialización de los controles manuales existentes (teclado, joysticks, etc), el rellenado de las listas desplegables con sus valores correspondientes, la inicialización de diversas líneas de tiempo utilizadas en diferentes partes del programa, y por último se inicializa el estado del sistema a "Iniciando".

7. IDENTIFICACIÓN DEL MODELO

Una simulación precisa y el diseño de controladores efectivos dependen en un alto grado de la construcción de modelos que se asemejen al sistema real a controlar y, por tanto, dependen también de la correcta estimación de los parámetros de dichos modelos. A esta estimación de los parámetros de los modelos se le suele denominar identificación del modelo del sistema.

Hay diferentes métodos para identificar los parámetros del modelo del cuadricóptero. Uno de ellos consiste en realizar mediciones para determinar cada uno de los parámetros tal y como se realiza en [41], donde ejecuta experimentos para medir la masa del cuadricóptero, los momentos de inercia, el factor de empuje aerodinámico y el factor de resistencia, entre otros. Para el cálculo con este método de ciertos parámetros es necesario el uso de instrumental específico.

Otro método, más sofisticado, consiste en el uso de filtros de partículas, como por ejemplo los estimadores de Monte Carlo, a través de datos reales tomados de vuelos de prueba. Un ejemplo de este método de identificación de parámetros puede consultarse en [23], donde los vuelos de prueba proporcionan estimaciones reales del estado que posibilitan al filtro de partículas ajustar su distribución y afinar los parámetros reales. Para ello se utilizó un sofisticado sistema de cámaras de infrarrojos que llenaban el lugar de experimentación con rayos infrarrojos y capturaban las posiciones de pequeñas bolas reflectantes instaladas en el cuadricóptero, con lo que se obtenían datos muy precisos del cuadricóptero.

El método más sencillo para identificar el modelo del AR.Drone 2.0 y que además parece el más adecuado debido al control interno que posee, consiste en considerarlo como una caja negra y realizar experimentos en los que se varíen las entradas y se registren los efectos que provocan en el estado del cuadricóptero para su posterior análisis. Este método, por su sencillez y sus buenos resultados, ha sido utilizado en diferentes sistemas reales como puede consultarse en [42], [18], [19], [43] y [44] entre otros.

Este último método es el que va a utilizarse en el presente proyecto debido a que caracterizando las entradas y salidas se modelan de forma inherente las dinámicas complejas, como por ejemplo el tiempo de respuesta de los motores, el efecto pala, el par giroscópico de los rotores y el factor de empuje aerodinámico. De la misma manera se modela de forma inherente el bucle cerrado de control interno que posee el AR.Drone 2.0.

La manera más común de variar las entradas para registrar el estado del cuadricóptero es mediante la aplicación de escalones en ellas, aunque también se pueden aplicar, por ejemplo, rampas o señales sinusoidales de frecuencia fija o de frecuencia variable (chirp).

En la identificación del modelo del cuadricóptero se va a seguir la estrategia de identificación de aplicar escalones a cada entrada y comprobar la variación en el estado del cuadricóptero, a través de los datos de navegación que proporciona, para obtener una función de transferencia que modele con la suficiente precisión la relación entre ellos. Debe aclararse que los experimentos de identificación no han sido realizados en el lugar de experimentación debido a que las limitaciones de espacio no permitían que el cuadricóptero llegase, en muchas ocasiones, a regímenes estacionarios. Debido a ello se han realizado estos experimentos en el pabellón deportivo del Instituto de Educación Secundaria Azarquiel de Toledo que han cedido el uso de sus instalaciones para esta fase del presente proyecto.

El envío de las entradas de control y el registro de datos se realiza con un periodo de muestreo de 20 ms tal y como se ha detallado en el apartado 5.2.1. En este sentido, se ha decidido obtener las funciones de transferencia discretas de los modelos en el dominio $\mathcal{Z}$.

Los modelos obtenidos se han recopilado en un fichero de Matlab que se adjunta junto al resto del código fuente de este proyecto, y cuyo nombre es *MCompleto.mat*.

Por otro lado, hay que tener en cuenta que para mejorar el control autónomo se va a obtener una estimación del estado del cuadricóptero de forma previa a establecer la acción de control. Dicha estimación va a ser proporcionada por el Filtro de Kalman que se describe en el apartado 8.1. Éste, aparte de otras variables, utiliza una matriz de covarianzas del error en las medidas de los sensores para proporcionar su estimación. Es por ello que es necesario modelar el ruido de los sensores que proporcionan las medidas de posición y orientación,

inclinación y velocidad. En este capítulo se caracterizará, con dicho objetivo, el ruido de los giróscopos, acelerómetros y del sensor de ultrasonidos del cuadricóptero, así como el ruido en las medidas del sistema de visión.

7.1. Modelo para el Pitch

Tras realizar un primer experimento aplicando un escalón de valor arbitrario en la entrada de control u_{pitch}, se verifica que las variables del estado del cuadricóptero que se encuentran afectadas, y que podrían ser significativas en la identificación del modelo, son la velocidad lineal en el eje x^B y el ángulo de inclinación θ del Pitch.

El ángulo de inclinación θ del Pitch se obtiene del dato de navegación *theta* de *NavData Demo*, mientras que la velocidad lineal en el eje x^B se obtiene del dato de navegación *vx* de *NavData Demo*.

En la Figura 7.1 se puede ver claramente que ante una entrada de control u_{pitch} de -0.5 el ángulo de inclinación θ llega a estabilizarse en -0.246 radianes (-14.1 °) y la velocidad lineal v_x^B llega a estabilizarse en 2.95 m/s.

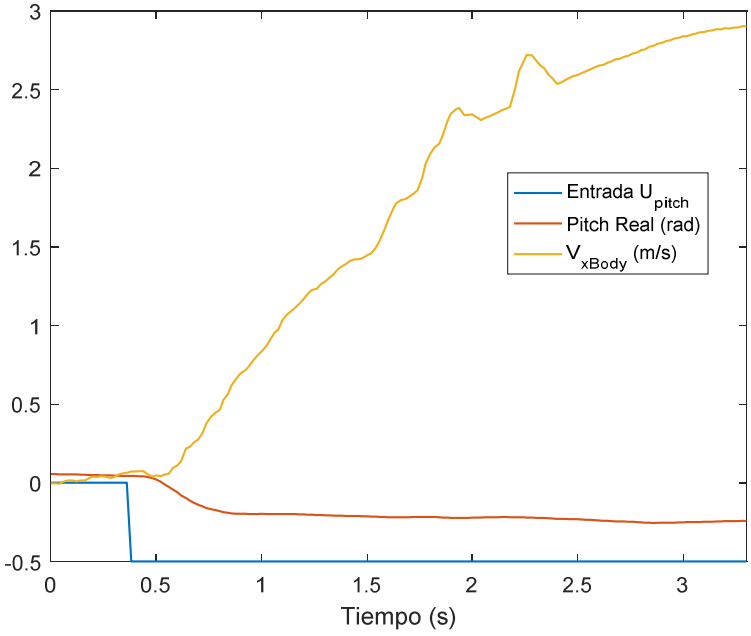


Figura 7.1: Respuesta al escalón en la entrada $u_{pitch} = -0.5$.

Para comprobar que este comportamiento es similar ante diferentes valores del escalón de u_{pitch} se realizan 20 experimentos en total variando u_{pitch} desde -1 hasta 1 en incrementos de 0.1. Para cada experimento se obtiene el valor de θ y de v_x^B en el régimen estacionario y como puede observarse en la Figura 7.2 y en la Figura 7.3, se produce una saturación en los valores estacionarios a partir de una entrada de u_{pitch} de $|0.4|$ aproximadamente.

Para no tener que crear un modelo en el que se tenga en cuenta la saturación existente se va a hacer trabajar al cuadricóptero dentro de la zona lineal. En este sentido, se han realizado más experimentos con escalones de u_{pitch} desde -0.45 hasta 0.45 en incrementos de 0.1, para disponer de datos más completos en el momento de la identificación.

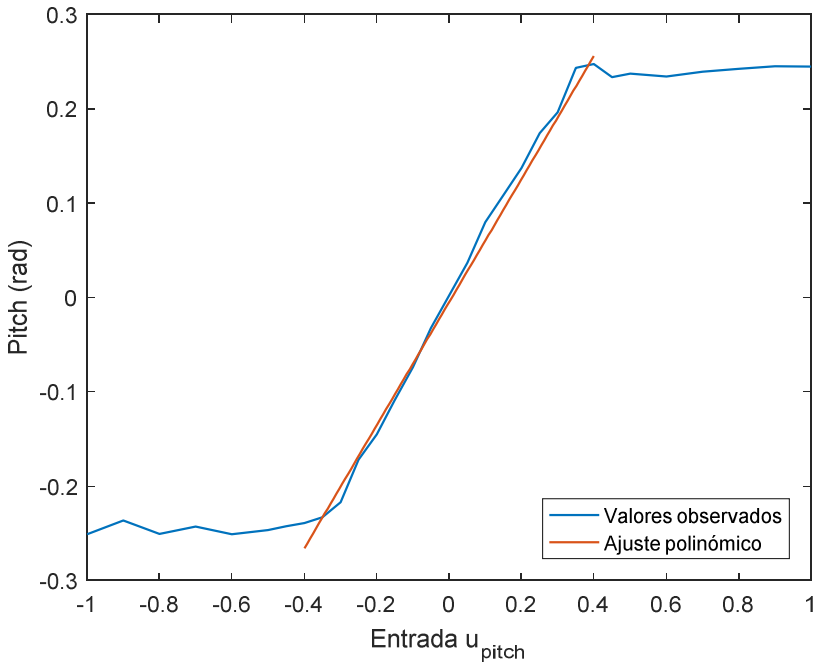


Figura 7.2: Régimen estacionario del ángulo θ ante escalones de u_{pitch}.

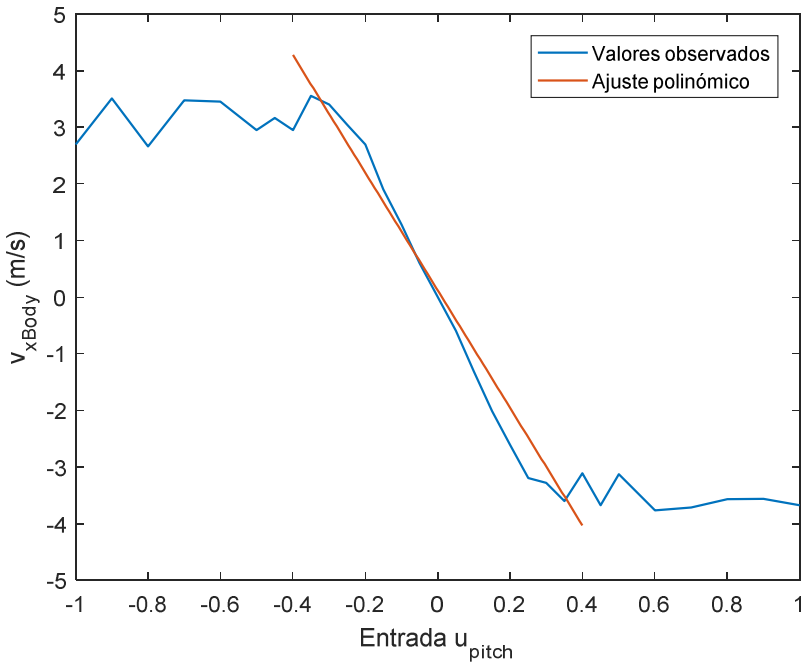


Figura 7.3: Régimen estacionario de la velocidad lineal v_x^B ante escalones de u_{pitch}.

Cuando se aplica una señal de control u_{pitch} el cuadricóptero modifica su ángulo de inclinación θ para llegar al valor de consigna y esto provoca una velocidad lineal v_x^B, Por ello, se va a comenzar realizando la identificación de una función de transferencia que relacione la entrada u_{pitch} con la inclinación θ, y a continuación se realizará la identificación de una función de transferencia que relacione dicha inclinación con el valor de v_x^B.

En todos los experimentos indicados se han aplicados dos escalones del mismo valor para poder tener datos para identificar así como datos para la posterior validación del modelo identificado. Esta identificación ha sido

realizada mediante la herramienta *System Identification* de Matlab, creando un objeto *iddata* con los datos de u_{pitch} y θ del primer escalón de cada experimento para la identificación, y otro objeto *iddata* con los datos de u_{pitch} y θ del segundo escalón de cada experimento, teniendo en cuenta que los experimentos incluidos son aquellos en los que se trabaja en la zona lineal, es decir, $u_{pitch} \in [-0.4, 0.4]$. Estos objetos han sido creados mediante el script denominado *Crea_iddata_Ident_Pitch_Escalones_U_Uref.m* que se adjunta junto al resto del código fuente del proyecto. También se han creado scripts similares para el Roll, Throttle y Yaw.

Igualmente se ha creado un script de Matlab para realizar de forma automática la identificación de la función de transferencia en el dominio $\mathcal{S}$ variando los ceros y polos del sistema entre 0 y 2 con el objeto *iddata* que contiene los datos de identificación obteniendo el porcentaje de ajuste de la curva estimada con la real para los datos de identificación. El nombre de dicho script es *Genera_TF_Ident_Pitch_Escalones_U_Uref.m*, e igualmente se han creado las versiones para el Roll, Throttle y Yaw).

En este momento hay que hacer inciso para indicar que la herramienta *System Identification* de Matlab ha sido configurada para que estime las condiciones iniciales al generar las funciones de transferencia. Por ello en las figuras de identificación del Pitch, así como del Roll, Throttle y Yaw, mostradas en este capítulo se puede ver que la gráfica real y la del modelo estimado no suelen comenzar con la misma condición inicial. Pero además hay que tener en cuenta que las medidas tomadas durante los experimentos de identificación son ruidosas y a veces se encuentran afectadas por perturbaciones, como se pone de manifiesto en dichas figuras.

En otro script, denominado *Obtiene_Mejor_Fit_Ident_Pitch_Escalones_U_Uref.m,* se aplica esta función de transferencia al objeto *iddata* que contiene los datos de validación para obtener el porcentaje de ajuste de la curva estimada con la real para los datos de validación. Existen, además, versiones de este script para el Roll, Throttle y Yaw. En la Tabla 7.1 se muestran los porcentajes de ajuste obtenidos tras ejecutar dicho script, donde el porcentaje de ajuste final viene dado por la fórmula de la ecuación (7.1) en la que se da más peso al ajuste obtenido de los datos de validación con respecto al de los datos de identificación.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	32,35	41,08	38,46
0	1	63,97	66,84	65,98
1	1	64,61	65,96	65,55
0	2	63,92	67,28	66,27
1	2	76,20	78,17	77,58
2	2	71,15	74,41	73,43

Tabla 7.1: Ajuste de las funciones de transferencia de θ respecto a u_{pitch} en el dominio $\mathcal{S}$ para I=DatosIdent y V=DatosValid.

$$\text{Ajuste} = 0.3\text{AjusteI} + 0.7\text{AjusteV} \tag{7.1}$$

Este mismo proceso se ha repetido tomando el objeto *iddata* que contiene los datos de validación para realizar la identificación, mientras que el objeto *iddata* que contiene los datos de identificación ha sido utilizado para la validación. Los scripts utilizados en este caso han sido *Genera_TF_Ident_Pitch_Escalones_U_Uref_tfz.m* y *Obtiene_Mejor_Fit_Ident_Pitch_Escalones_U_Uref_tfz.m*, para los que existen también versiones para el Roll, Pitch y Yaw. De esta forma se obtienen los ajustes mostrados en la Tabla 7.2.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	42,23	32,97	35,75
0	1	68,60	62,41	64,27
1	1	69,22	62,11	64,25
0	2	12,83	8,53	9,82
1	2	71,92	66,87	68,38
2	2	42,96	37,10	38,86

Tabla 7.2: Ajuste de las funciones de transferencia de θ respecto a u_{pitch} en el dominio $\mathcal{S}$ para I=DatosValid y V=DatosIdent.

Igualmente se ha hecho este proceso en el dominio $\mathcal{Z}^{-1}$ y se han obtenido los porcentajes de ajuste mostrados en la Tabla 7.3 y en la Tabla 7.4.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	32,35	41,08	38,46
0	1	64,04	66,83	65,99
1	1	63,95	66,88	66,00
0	2	64,12	67,41	66,42
1	2	64,01	67,36	66,36
2	2	77,69	78,19	78,04

Tabla 7.3: Ajuste de las funciones de transferencia de θ respecto a u_{pitch} en el dominio $\mathcal{Z}^{-1}$ para I=DatosIdent y V=DatosValid.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	42,23	32,97	35,75
0	1	68,31	62,56	64,28
1	1	68,60	62,41	64,27
0	2	69,70	62,39	64,58
1	2	69,87	62,26	64,54
2	2	70,35	61,80	64,36

Tabla 7.4: Ajuste de las funciones de transferencia de θ respecto a u_{pitch} en el dominio $\mathcal{Z}^{-1}$ para I=DatosValid y V=DatosIdent.

Con estos datos y visualizando las curvas en Matlab se decide usar como modelo de θ respecto a u_{pitch} el modelo de la Tabla 7.3 para 1 cero y 2 polos en el dominio $\mathcal{Z}^{-1}$ que proporciona un ajuste bastante bueno con un orden bajo. Se transforma al dominio $\mathcal{Z}$ quedando la función de transferencia con ningún cero y 2 polos indicada en (7.2).

$$F_{\theta_u_{pitch}}(z) = \frac{0.009145}{z^2 - 1.622z + 0.6365} \tag{7.2}$$

En la Figura 7.4 se muestra una comparativa entre los datos reales de θ y los obtenidos con el modelo para diferentes escalones donde se puede ver que el ajuste del modelo es bastante aproximado.

Para la identificación de la función de transferencia que relaciona θ con el valor de v_x^B se van a aprovechar los experimentos realizados antes y se procederá de la misma forma. Los scripts creados para ello han sido *Crea_iddata_Ident_Pitch_V_U.m*, *Genera_TF_Ident_Pitch_V_U.m*, *Obtiene_Mejor_Fit_Pitch_V_U.m*, *Genera_TF_Ident_Pitch_V_U_tfz.m* y *Obtiene_Mejor_Fit_Pitch_V_U_tfz.m*. Para estos scripts se han creado también las versiones del Roll.

Con esto, los ajustes obtenidos se muestran en la Tabla 7.5, en la Tabla 7.6, en la Tabla 7.7 y en la Tabla 7.8.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	30,31	23,97	25,87
0	1	63,77	58,99	60,42
1	1	63,66	58,25	59,87
0	2	73,80	68,54	70,12
1	2	78,31	74,06	75,33
2	2	78,61	74,74	75,90

Tabla 7.5: Ajuste de las funciones de transferencia de v_x^B respecto a θ en el dominio $\mathcal{S}$ para I=DatosIdent y V=DatosValid.

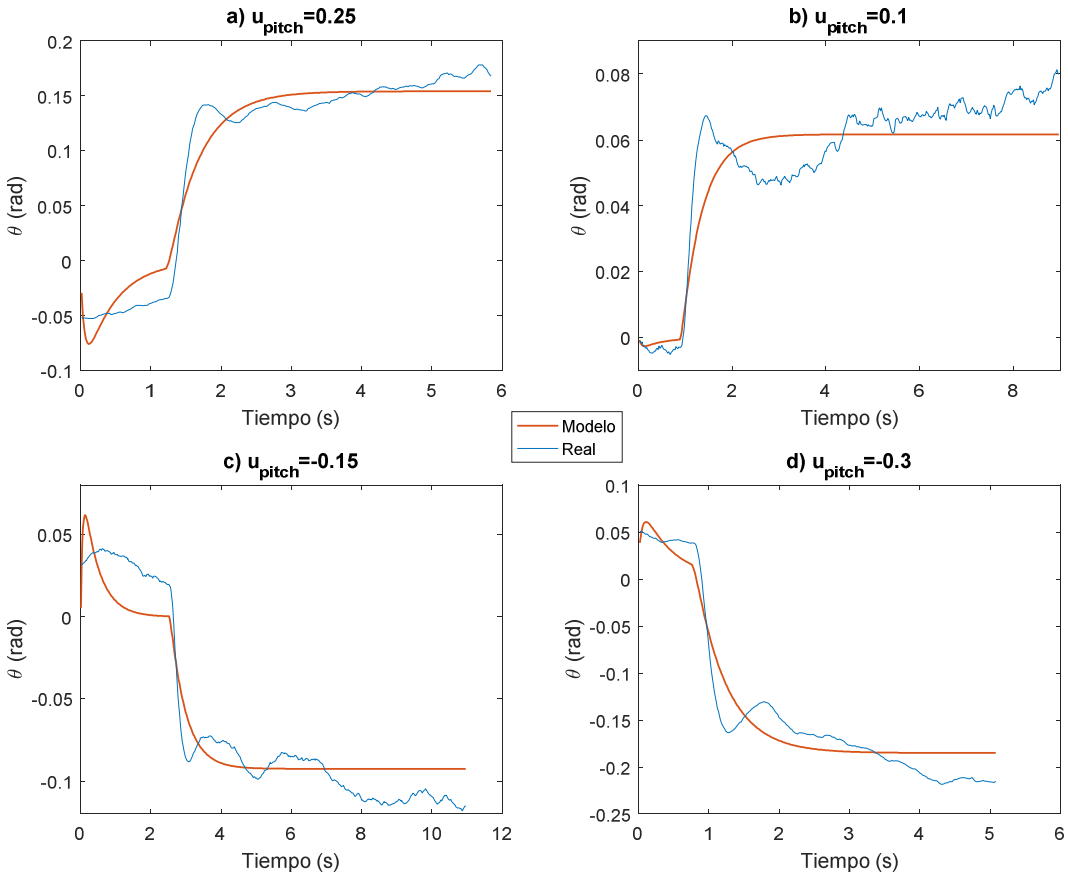


Figura 7.4: Comparación entre datos reales y simulados de θ para diferentes escalones de Pitch.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	26,20	28,34	27,70
0	1	62,45	63,47	63,17
1	1	60,77	63,67	62,80
0	2	72,98	72,30	72,50
1	2	77,76	79,12	78,71
2	2	76,84	78,09	77,71

Tabla 7.6: Ajuste de las funciones de transferencia de v_x^B respecto a θ en el dominio $\mathcal{S}$ para I=DatosValid y V=DatosIdent.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	30,31	23,97	25,87
0	1	63,89	58,96	60,44
1	1	63,74	58,95	60,39
0	2	71,05	65,39	67,09
1	2	70,83	65,21	66,90
2	2	75,17	69,51	71,21

Tabla 7.7: Ajuste de las funciones de transferencia de v_x^B respecto a θ en el dominio $\mathcal{Z}^{-1}$ para I=DatosIdent y V=DatosValid.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	26,20	28,34	27,70
0	1	62,44	63,66	63,29
1	1	62,44	63,50	63,18
0	2	62,42	63,72	63,33
1	2	62,62	63,94	63,54
2	2	61,07	64,02	63,13

Tabla 7.8: Ajuste de las funciones de transferencia de v_x^B respecto a θ en el dominio $\mathcal{Z}^{-1}$ para I=DatosValid y V=DatosIdent.

Con estos datos y visualizando las curvas en Matlab se decide usar como modelo de v_x^B respecto a θ el modelo de la Tabla 7.8 para 1 cero y 1 polo en el dominio $\mathcal{Z}^{-1}$ que proporciona un ajuste bastante bueno con un orden bajo. Se transforma al dominio $\mathcal{Z}$ quedando la función de transferencia con ningún cero y 1 polo indicada en (7.3).

$$F_{v_{x_\theta}^B}(z) = \frac{-0.2343}{z - 0.9876} \tag{7.3}$$

En la Figura 7.5 se muestra una comparativa entre los datos reales de v_x^B y los obtenidos con el modelo para diferentes escalones donde se puede ver que el ajuste del modelo es bastante aproximado.

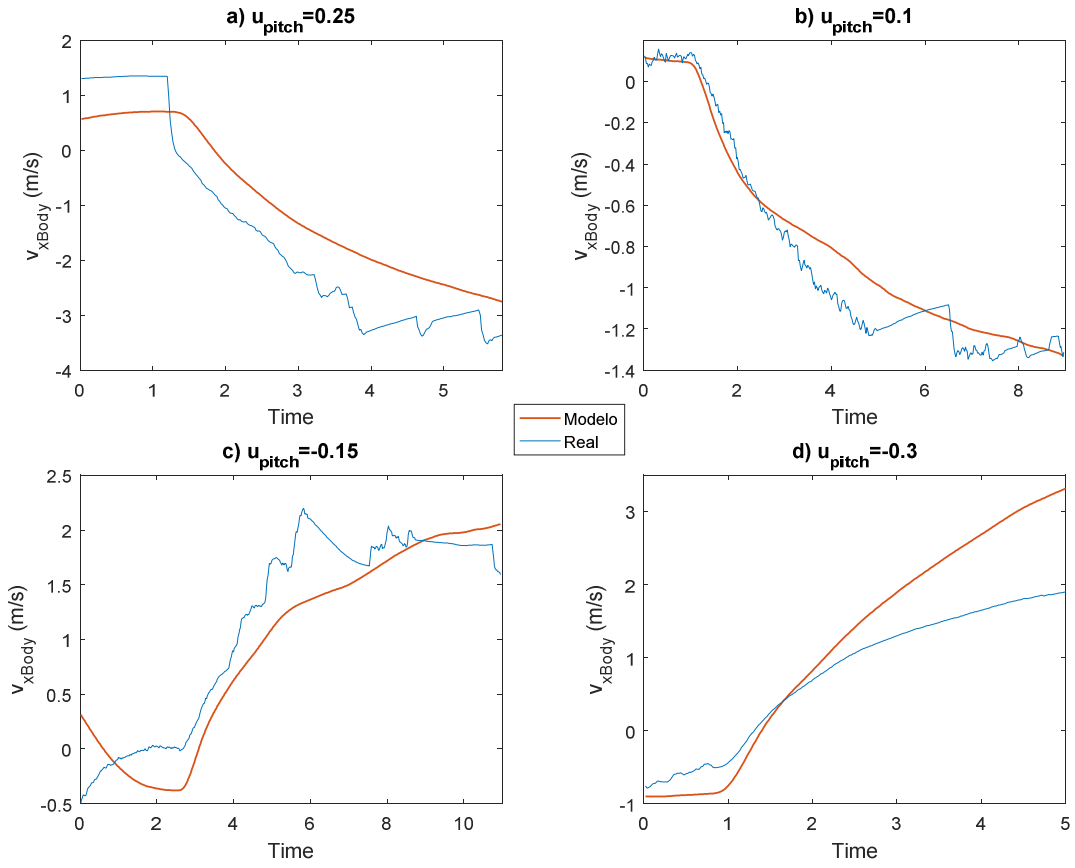


Figura 7.5: Comparación entre datos reales y simulados de v_x^B para diferentes escalones de Pitch.

7.2. Modelo para el Roll

Aplicando la misma estrategia de identificación seguida en el Pitch para el Roll se verifica que las variables de estado del cuadricóptero afectadas ante variaciones en la entrada de control u_{roll}, y que podrían ser significativas para la identificación, son la velocidad lineal en el eje y^B y el ángulo de inclinación ϕ del Roll.

El ángulo de inclinación ϕ del Roll se obtiene del dato de navegación *phi* de *NavData Demo*, mientras que la velocidad lineal en el eje y^B se obtiene del dato de navegación *vy* de *NavData Demo*.

En la Figura 7.6 se puede ver claramente que ante una entrada de control u_{roll} de -0.4 el ángulo de inclinación ϕ llega a estabilizarse en -0.19 radianes (-10.9 °) y la velocidad lineal v_y^B llega a estabilizarse en -1.91 m/s aproximadamente.

Se realizan 20 experimentos en total variando u_{roll} desde -1 hasta 1 en incrementos de 0.1. Para cada experimento se obtiene el valor de ϕ y de v_y^B en el régimen estacionario y como puede observarse en la Figura 7.7 y en la Figura 7.8, se produce una saturación en los valores estacionarios a partir de una entrada de u_{roll} de |0.4| aproximadamente.

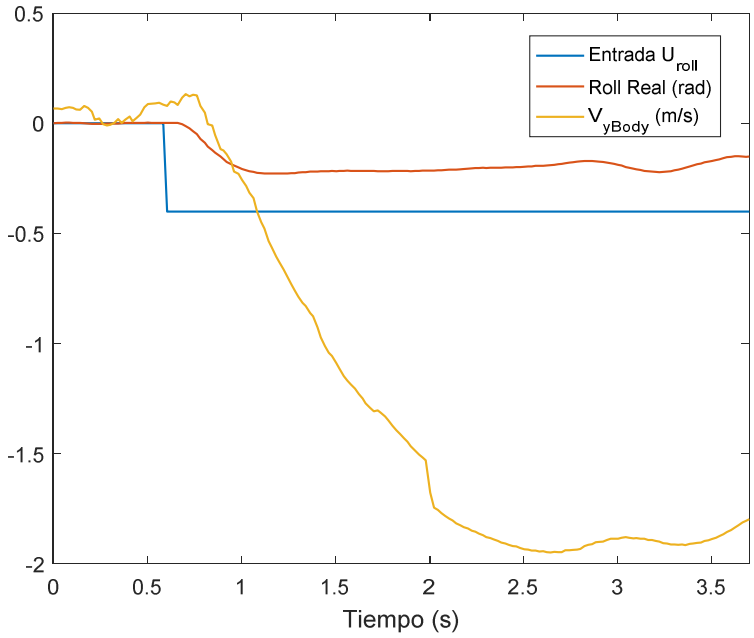


Figura 7.6: Respuesta al escalón en la entrada $u_{roll} = -0.4$.

Para no tener que crear un modelo en el que se tenga en cuenta la saturación existente se va a hacer trabajar al cuadricóptero dentro de la zona lineal. En este sentido, se han realizado más experimentos con escalones de u_{roll} desde -0.45 hasta 0.45 en incrementos de 0.1, para disponer de datos más completos en el momento de la identificación.

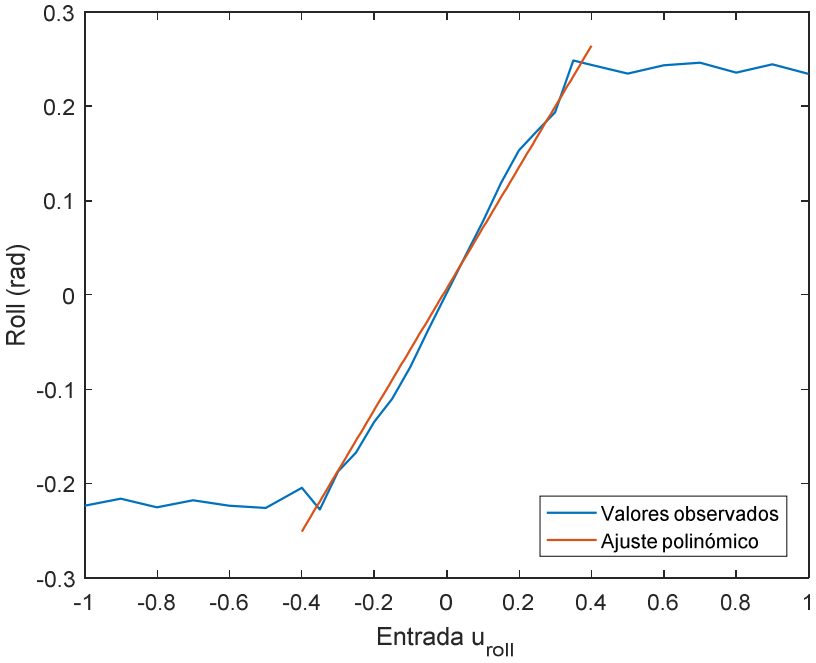


Figura 7.7: Régimen estacionario del ángulo ϕ ante escalones de u_{roll}.

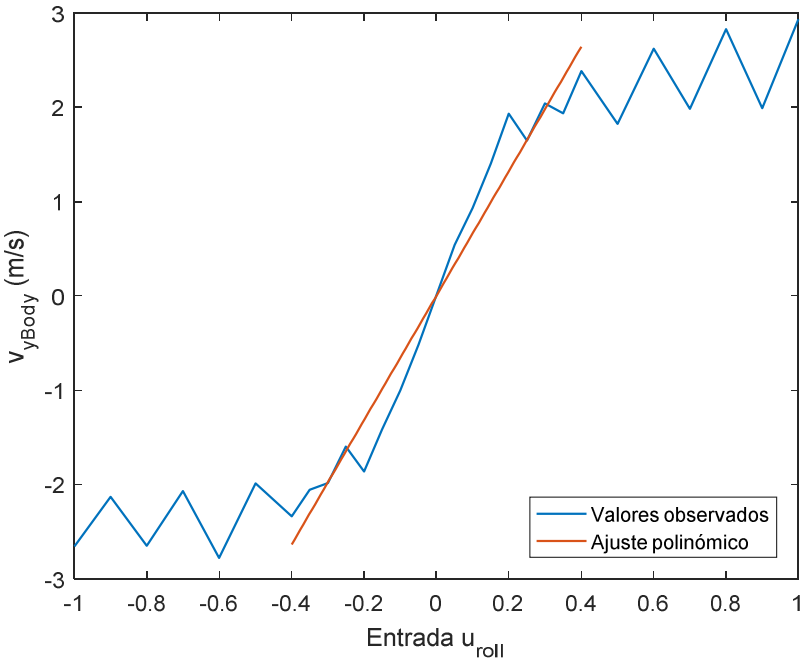


Figura 7.8: Régimen estacionario de la velocidad lineal v_y^B ante escalones de u_{roll}.

Al igual que en el caso del Pitch, en la identificación del modelo se va a comenzar realizando la identificación de una función de transferencia que relacione la entrada u_{roll} con la inclinación ϕ, y a continuación se realizará la identificación de una función de transferencia que relacione dicha inclinación con el valor de v_y^B.

Realizando el mismo conjunto de experimentos que con el modelo del Pitch y teniendo en cuenta que también se va a hacer trabajar al modelo del Roll en la zona lineal, es decir $u_{roll} \in [-0.4, 0.4]$, se obtienen los resultados que se muestran en la Tabla 7.9, en la Tabla 7.10, en la Tabla 7.11 y en la Tabla 7.12.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	33,28	33,23	33,24
0	1	60,99	61,63	61,44
1	1	60,59	61,18	61,00
0	2	17,45	10,21	12,38
1	2	36,94	35,96	36,25
2	2	-32,41	-34,88	-34,14

Tabla 7.9: Ajuste de las funciones de transferencia de ϕ respecto a u_{roll} en el dominio $\mathcal{S}$ para I=DatosIdent y V=DatosValid.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	33,77	33,63	33,67
0	1	62,11	61,24	61,50
1	1	62,27	61,43	61,68
0	2	-17,07	-8,73	-11,24
1	2	76,25	74,59	75,08
2	2	64,06	63,17	63,44

Tabla 7.10: Ajuste de las funciones de transferencia de ϕ respecto a u_{roll} en el dominio $\mathcal{S}$ para I=DatosValid y V=DatosIdent.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	33,28	33,23	33,24
0	1	60,88	61,52	61,33
1	1	60,97	61,58	61,40
0	2	61,39	61,87	61,73
1	2	61,44	61,92	61,77
2	2	61,54	62,00	61,86

Tabla 7.11: Ajuste de las funciones de transferencia de ϕ respecto a u_{roll} en el dominio $\mathcal{Z}^{-1}$ para I=DatosIdent y V=DatosValid.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	33,77	33,63	33,67
0	1	62,01	61,14	61,40
1	1	62,10	61,24	61,50
0	2	62,38	61,65	61,87
1	2	62,42	61,69	61,91
2	2	62,50	61,78	61,99

Tabla 7.12: Ajuste de las funciones de transferencia de ϕ respecto a u_{roll} en el dominio $\mathcal{Z}^{-1}$ para I=DatosValid y V=DatosIdent.

Con estos datos y visualizando las curvas en Matlab se decide usar como modelo ϕ respecto a u_{roll} el modelo de la Tabla 7.12 para 1 cero y 2 polos en el dominio $\mathcal{Z}^{-1}$ que proporciona un ajuste bastante bueno con un orden bajo. Se transforma al dominio $\mathcal{Z}$ quedando la función de transferencia con ningún cero y 2 polos indicada en (7.4).

$$F_{\phi_u_{roll}}(z) = \frac{0.007383}{z^2 - 1.767z + 0.7788} \tag{7.4}$$

En la Figura 7.11 se muestra una comparativa entre los datos reales de ϕ y los obtenidos con el modelo para diferentes escalones donde se puede ver que el ajuste del modelo es bastante aproximado.

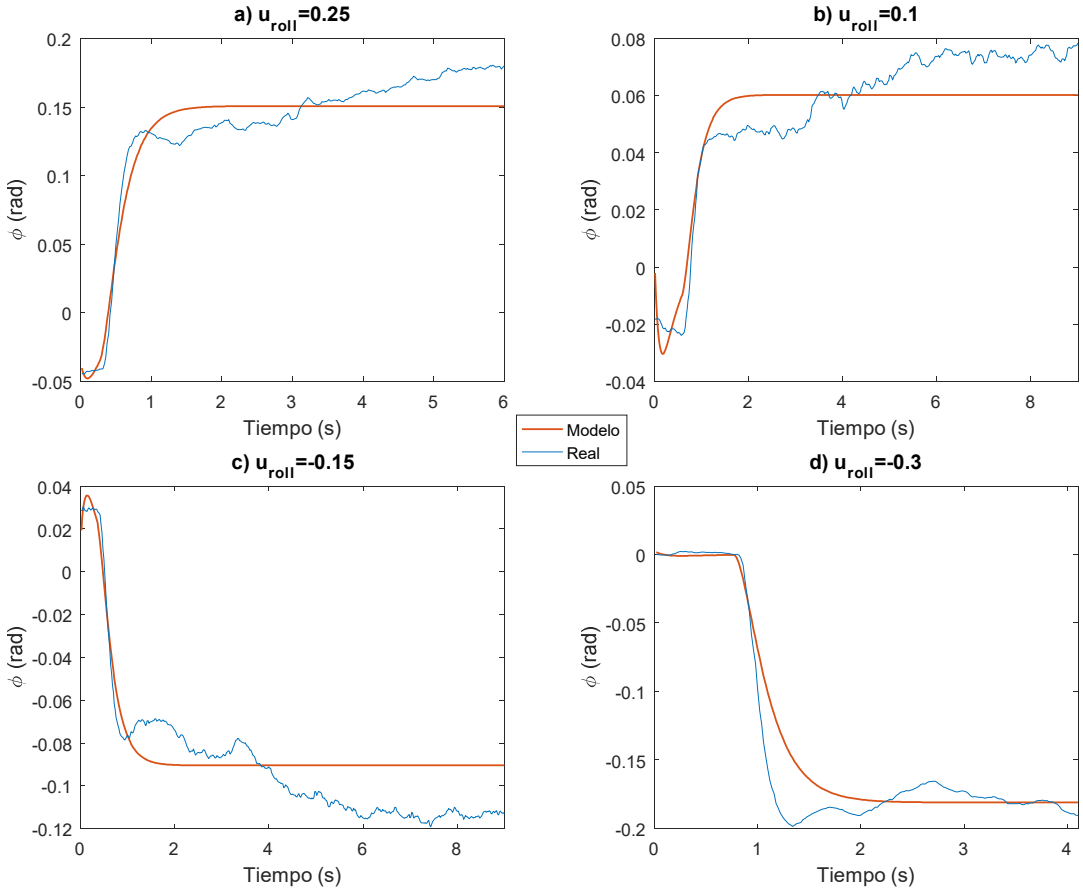


Figura 7.9: Comparación entre datos reales y simulados de ϕ para diferentes escalones de Roll.

Para la identificación de la función de transferencia que relaciona ϕ con el valor de v_y^B se van a aprovechar los experimentos realizados antes y se procederá de la misma forma. Con esto, los ajustes obtenidos se muestran en la Tabla 7.13, en la Tabla 7.14, en la Tabla 7.15 y en la Tabla 7.16.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	26,87	26,84	26,85
0	1	59,78	53,96	55,71
1	1	60,15	54,43	56,15
0	2	61,02	60,35	60,55
1	2	66,61	65,69	65,97
2	2	65,87	65,10	65,33

Tabla 7.13: Ajuste de las funciones de transferencia de v_y^B respecto a ϕ en el dominio S para I=DatosIdent y V=DatosValid.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	27,36	26,44	26,72
0	1	54,48	58,81	57,51
1	1	54,50	59,15	57,75
0	2	62,70	62,79	62,76
1	2	63,59	63,51	63,53
2	2	64,12	64,19	64,17

Tabla 7.14: Ajuste de las funciones de transferencia de v_y^B respecto a ϕ en el dominio S para I=DatosValid y V=DatosIdent.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	26,87	26,84	26,85
0	1	59,88	54,06	55,81
1	1	59,77	53,94	55,69
0	2	59,90	54,32	55,99
1	2	59,84	54,01	55,76
2	2	60,21	54,51	56,22

Tabla 7.15: Ajuste de las funciones de transferencia de v_y^B respecto a ϕ en el dominio $\mathcal{Z}^{-1}$ para I=DatosIdent y V=DatosValid.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	27,36	26,44	26,72
0	1	54,54	58,86	57,56
1	1	54,46	58,75	57,46
0	2	54,60	58,98	57,66
1	2	54,52	58,89	57,58
2	2	67,29	67,56	67,48

Tabla 7.16: Ajuste de las funciones de transferencia de v_y^B respecto a ϕ en el dominio $\mathcal{Z}^{-1}$ para I=DatosValid y V=DatosIdent.

Con estos datos y visualizando las curvas en Matlab se decide usar como modelo de v_y^B respecto a ϕ el modelo de la Tabla 7.16 para 1 cero y 1 polo en el dominio $\mathcal{Z}^{-1}$ que proporciona un ajuste bastante bueno con un orden bajo. Se transforma al dominio $\mathcal{Z}$ quedando la función de transferencia con ningún cero y 1 polo indicada en (7.5).

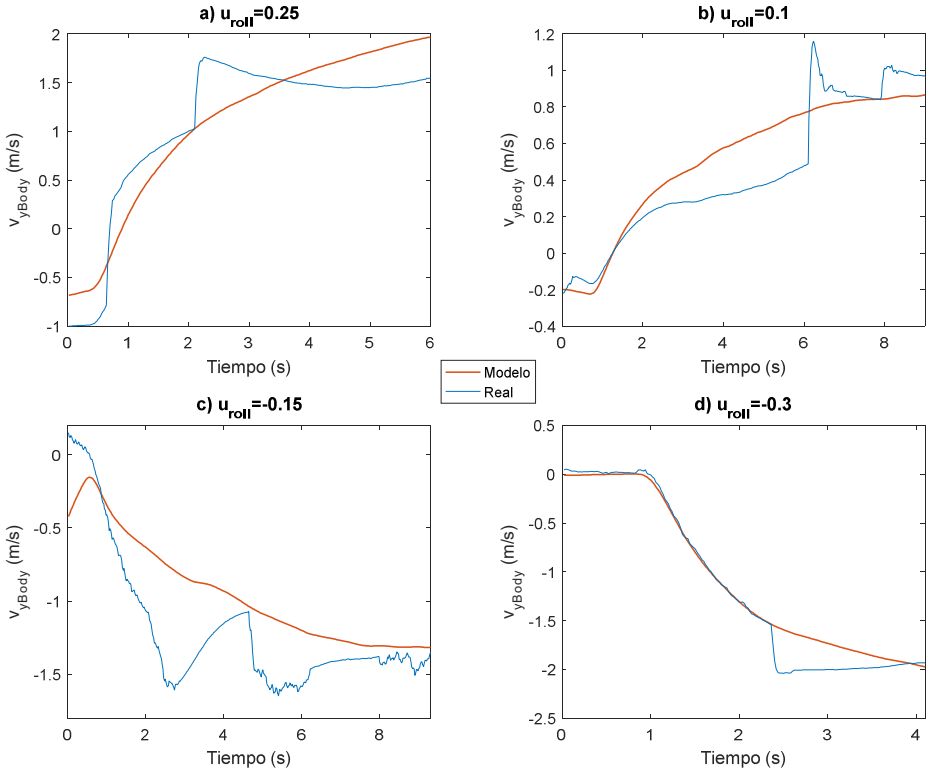


Figura 7.10: Comparación entre datos reales y simulados de v_y^B para diferentes escalones de Roll.

$$F_{v_y^B_\phi}(z) = \frac{0.2107}{z - 0.9819} \tag{7.5}$$

En la Figura 7.10 se muestra una comparativa entre los datos reales de v_y^B y los obtenidos con el modelo para diferentes escalones donde se puede ver que el ajuste del modelo es bastante aproximado. A la vista de las gráficas mostradas en esta figura puede intuirse lo complejo de modelar que es el comportamiento, en general, del cuadricóptero ya que puede estar afectado por perturbaciones o incluso el control interno del AR.Drone 2.0 puede introducir acciones extrañas.

7.3. Modelo para el Throttle

Aplicando la misma estrategia de identificación seguida en los apartados anteriores se verifica que la variable de estado del cuadricóptero afectada ante variaciones en la entrada de control $u_{throttle}$, y que podría ser significativa para la identificación, es la velocidad lineal en el eje z^E.

En los datos de navegación *NavData Demo* la variable *vz* siempre está a cero, seguramente debido a algún problema interno del firmware del AR.Drone 2.0. Para conseguir una medida de la velocidad lineal en el eje z^B se podría realizar la derivada de la altura que proporcionan los sensores del cuadricóptero y que se adquiere a través del dato de navegación *altitude* del *NavData Demo*. Sin embargo, tras un estudio más profundo de los *NavData* se comprobó la existencia del dato de navegación *altitude_vz* del *NavData Altitude* que proporciona la velocidad requerida de forma directa. Por tanto se usará este último dato como la variable que define la velocidad v_z^E teniendo en cuenta que se proporciona invertido, es decir, la velocidad se incrementa cuando el cuadricóptero desciende y se decrementa cuando asciende.

En la Figura 7.11 se puede ver claramente que ante una entrada de control $u_{throttle}$ de 0.6 la velocidad lineal v_z^E llega a estabilizarse en 0.55 m/s aproximadamente.

Se realizan 20 experimentos en total variando $u_{throttle}$ desde -1 hasta 1 en incrementos de 0.1. Para cada experimento se obtiene el valor de v_z^E en el régimen estacionario y como puede observarse en la Figura 7.12 no se produce saturación, a diferencia de los casos del Pitch y el Roll. Por tanto se puede hacer trabajar al Throttle con todos los posibles valores de su entrada de control, es decir, $u_{throttle} \in [-1,1]$.

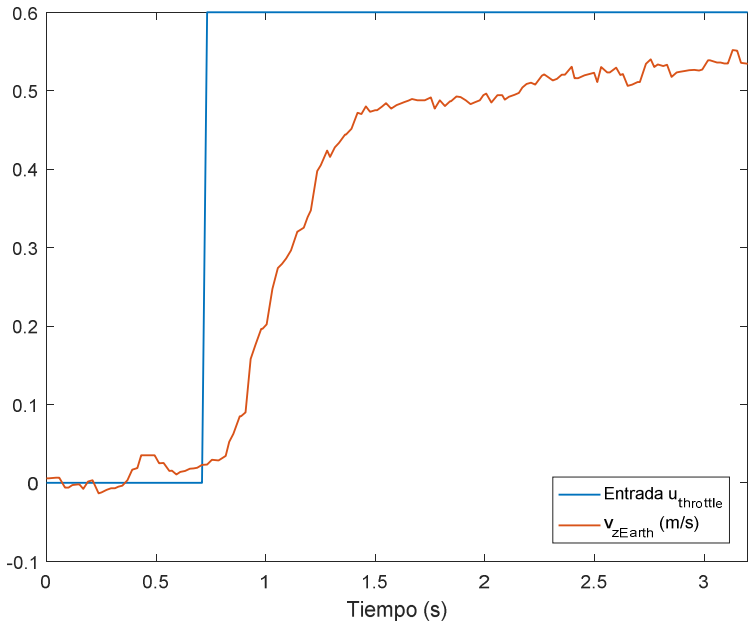


Figura 7.11: Respuesta al escalón en la entrada $u_{throttle} = 0.6$.

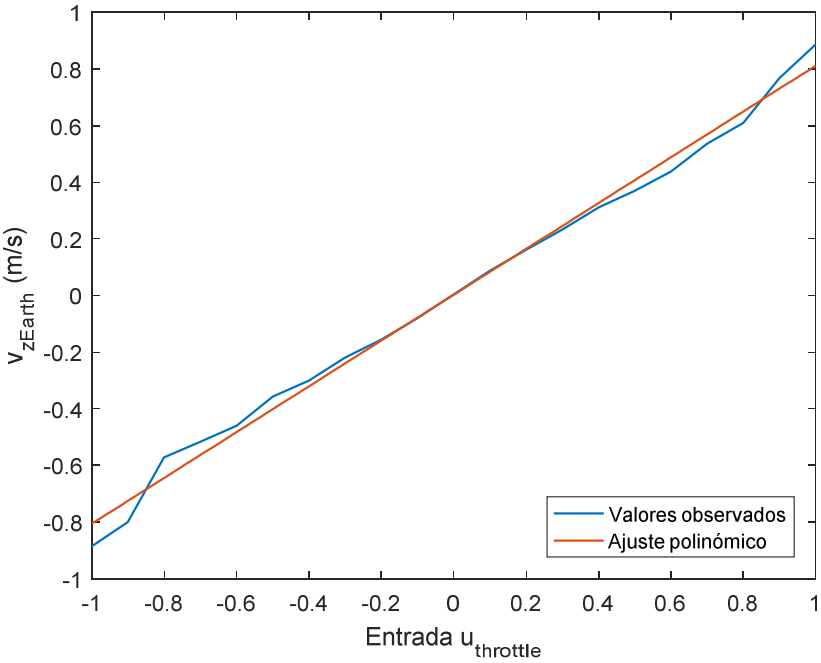


Figura 7.12: Régimen estacionario de la velocidad lineal v_z^E ante escalones de $u_{throttle}$.

El modelo del Throttle consistirá en una función de transferencia que relacione la entrada $u_{throttle}$ con el valor de v_z^E, que permitirá establecer posteriormente un control adecuado.

Con la misma metodología usada en el caso del modelo del Pitch y del Roll, se obtienen los resultados que se muestran en la Tabla 7.17, en la Tabla 7.18, en la Tabla 7.19 y en la Tabla 7.20.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	15,84	13,23	14,02
0	1	41,26	35,95	37,54
1	1	41,33	34,56	36,59
0	2	41,36	36,29	37,81
1	2	47,53	41,33	43,19
2	2	44,47	39,34	40,88

Tabla 7.17: Ajuste de las funciones de transferencia del Throttle en el dominio $\mathcal{S}$ para I=DatosIdent y V=DatosValid.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	12,23	14,98	14,16
0	1	32,54	38,67	36,83
1	1	40,80	40,51	40,60
0	2	32,83	38,49	36,79
1	2	52,49	54,28	53,75
2	2	50,74	50,49	50,57

Tabla 7.18: Ajuste de las funciones de transferencia del Throttle en el dominio $\mathcal{S}$ para I=DatosValid y V=DatosIdent.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	15,84	13,23	14,02
0	1	41,07	35,57	37,22
1	1	45,55	40,29	41,87
0	2	41,26	35,93	37,53

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
1	2	43,56	38,69	40,15
2	2	55,44	50,44	51,94

Tabla 7.19: Ajuste de las funciones de transferencia del Throttle en el dominio $\mathcal{Z}^{-1}$ para I=DatosIdent y V=DatosValid.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	12,23	14,98	14,16
0	1	32,26	38,57	36,68
1	1	34,70	40,75	38,94
0	2	32,51	38,65	36,81
1	2	34,52	40,45	38,67
2	2	53,13	54,16	53,85

Tabla 7.20: Ajuste de las funciones de transferencia del Throttle en el dominio $\mathcal{Z}^{-1}$ para I=DatosValid y V=DatosIdent.

Con estos datos y visualizando las curvas en Matlab se decide usar como modelo de Throttle el modelo de la Tabla 7.20 para 2 ceros y 2 polos en el dominio $\mathcal{Z}^{-1}$ que proporciona un ajuste suficientemente bueno con un orden bajo. Se transforma al dominio $\mathcal{Z}$ quedando la función de transferencia con 1 cero y 2 polos indicada en (7.6).

$$F_{throttle}(z) = \frac{0.01825z - 0.01813}{z^2 - 1.978z + 0.9786} \tag{7.6}$$

En la Figura 7.13 se muestra una comparativa entre los datos reales de v_z^E y los obtenidos con el modelo para diferentes escalones donde se puede ver que el ajuste del modelo es bastante aproximado.

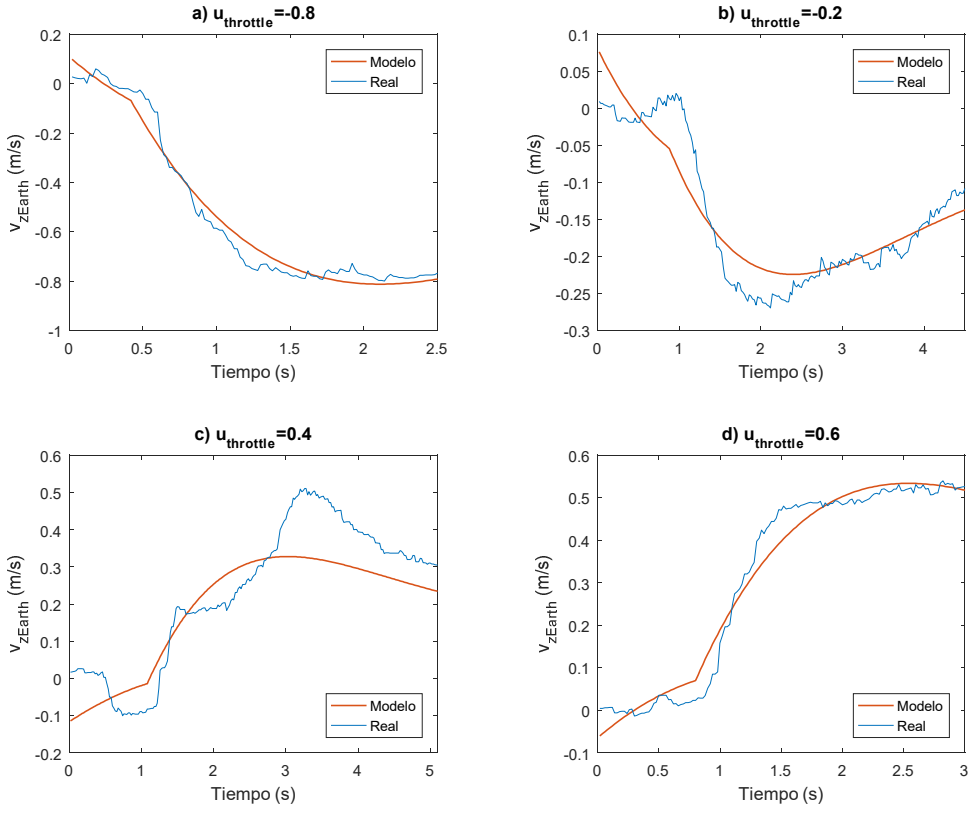


Figura 7.13: Comparación entre datos reales y simulados con el modelo para diferentes escalones de Throttle.

7.4. Modelo para el Yaw

Por último, para el caso del Yaw, se verifica que la variable de estado del cuadricóptero afectada ante variaciones en la entrada de control u_{yaw}, y que podría ser significativa para la identificación, es la velocidad angular con respecto al eje z^B que puede considerarse igual a la velocidad angular con respecto al eje z^E según lo indicado en el apartado 3.2 por la asunción de la condición de hovering.

El tercer componente del vector *phys_gyros* de los datos de navegación *NavData Phys Measures* proporciona el valor de la velocidad angular con respecto al eje z^B. Según la definición del marco de referencia B y la definición del marco de referencia E del lugar de experimentación, la velocidad angular v_ψ^B está invertida con respecto a la velocidad angular v_ψ^E. Aunque la identificación se va a realizar con respecto al marco de referencia B, esta diferencia deberá tenerse en cuenta de aquí en adelante.

En la Figura 7.14 se puede ver claramente que ante una entrada de control u_{yaw} de -0.5 la velocidad angular v_ψ^B llega a estabilizarse en -0.81 rad/s aproximadamente.

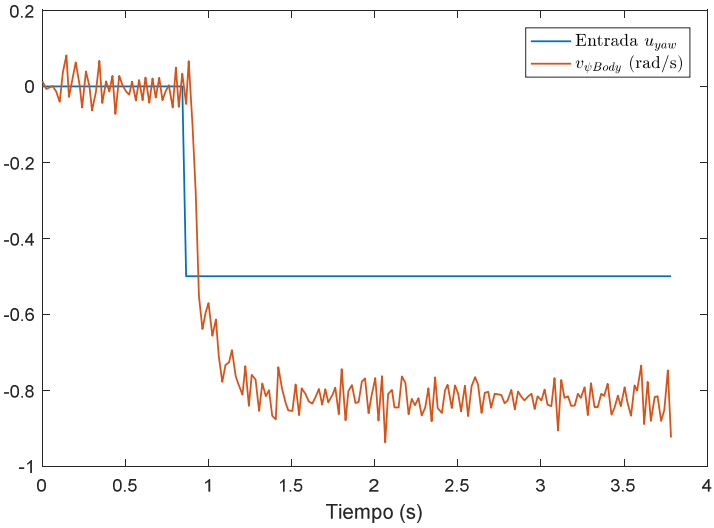


Figura 7.14: Respuesta al escalón en la entrada $u_{yaw} = -0.5$.

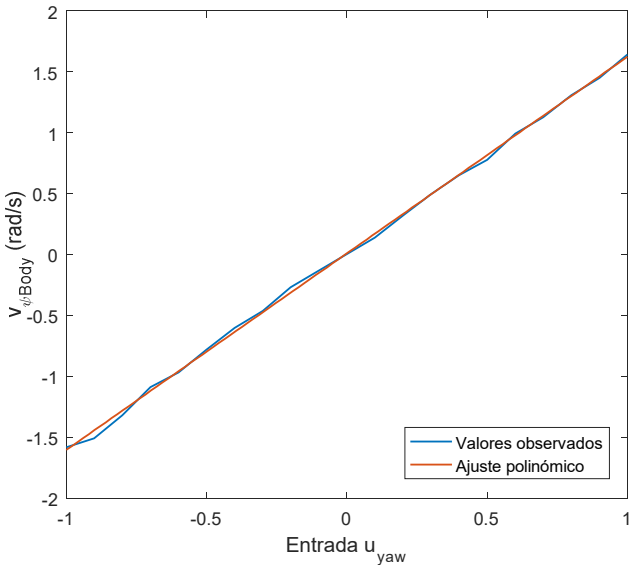


Figura 7.15: Régimen estacionario de la velocidad angular v_ψ^B ante escalones de u_{yaw}.

Se realizan 20 experimentos en total variando u_{yaw} desde -1 hasta 1 en incrementos de 0.1. Para cada experimento se obtiene el valor de v_{ψ}^{B} en el régimen estacionario y como puede observarse en la Figura 7.15 no se produce saturación, a diferencia de los casos del Pitch y el Roll. Por tanto se puede hacer trabajar al Yaw con todos los posibles valores de su entrada de control, es decir, $u_{yaw} \in [-1,1]$.

El modelo del Yaw consistirá en una función de transferencia que relacione la entrada u_{yaw} con el valor de v_{ψ}^{B}, que permitirá establecer posteriormente un control adecuado.

Con la misma metodología usada en la identificación de los modelos anteriores, se obtienen los resultados que se muestran en la Tabla 7.21, en la Tabla 7.22, en la Tabla 7.23 y en la Tabla 7.24.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	47,39	50,06	49,26
0	1	68,33	69,39	69,07
1	1	47,61	48,75	48,41
0	2	68,91	70,11	69,75
1	2	71,41	70,84	71,01
2	2	68,82	68,78	68,79

Tabla 7.21: Ajuste de las funciones de transferencia del Yaw en el dominio $\mathcal{S}$ para I=DatosIdent y V=DatosValid.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	50,43	47,12	48,11
0	1	69,81	68,17	68,66
1	1	70,13	68,48	68,98
0	2	70,55	68,71	69,26
1	2	71,74	71,27	71,41
2	2	70,97	70,14	70,39

Tabla 7.22: Ajuste de las funciones de transferencia del Yaw en el dominio $\mathcal{S}$ para I=DatosValid y V=DatosIdent.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	47,39	50,06	49,26
0	1	66,95	68,01	67,69
1	1	68,30	69,34	69,03
0	2	68,25	69,42	69,07
1	2	68,78	69,94	69,59
2	2	69,10	70,35	69,98

Tabla 7.23: Ajuste de las funciones de transferencia del Yaw en el dominio $\mathcal{Z}^{-1}$ para I=DatosIdent y V=DatosValid.

Ceros	Polos	% Ajuste I	% Ajuste V	% Ajuste
0	0	50,43	47,12	48,11
0	1	68,42	66,82	67,30
1	1	69,76	68,13	68,62
0	2	69,85	68,06	68,60
1	2	70,38	68,58	69,12
2	2	70,48	69,02	69,46

Tabla 7.24: Ajuste de las funciones de transferencia del Yaw en el dominio $\mathcal{Z}^{-1}$ para I=DatosValid y V=DatosIdent.

Con estos datos y visualizando las curvas en Matlab se decide usar como modelo de Yaw el modelo de la Tabla 7.21 para ningún cero y 1 polo en el dominio $\mathcal{S}$ que proporciona un ajuste bastante bueno con un orden bajo. Se discretiza el modelo transformándolo al dominio $\mathcal{Z}$ y queda así la función de transferencia con ningún cero y 1 polo indicada en (7.7).

$$F_{yaw}(z) = \frac{0.2926}{z - 0.8204} \tag{7.7}$$

En la Figura 7.16 se muestra una comparativa entre los datos reales de v_{ψ}^{B} y los obtenidos con el modelo para diferentes escalones donde se puede ver que el ajuste del modelo es muy aproximado.

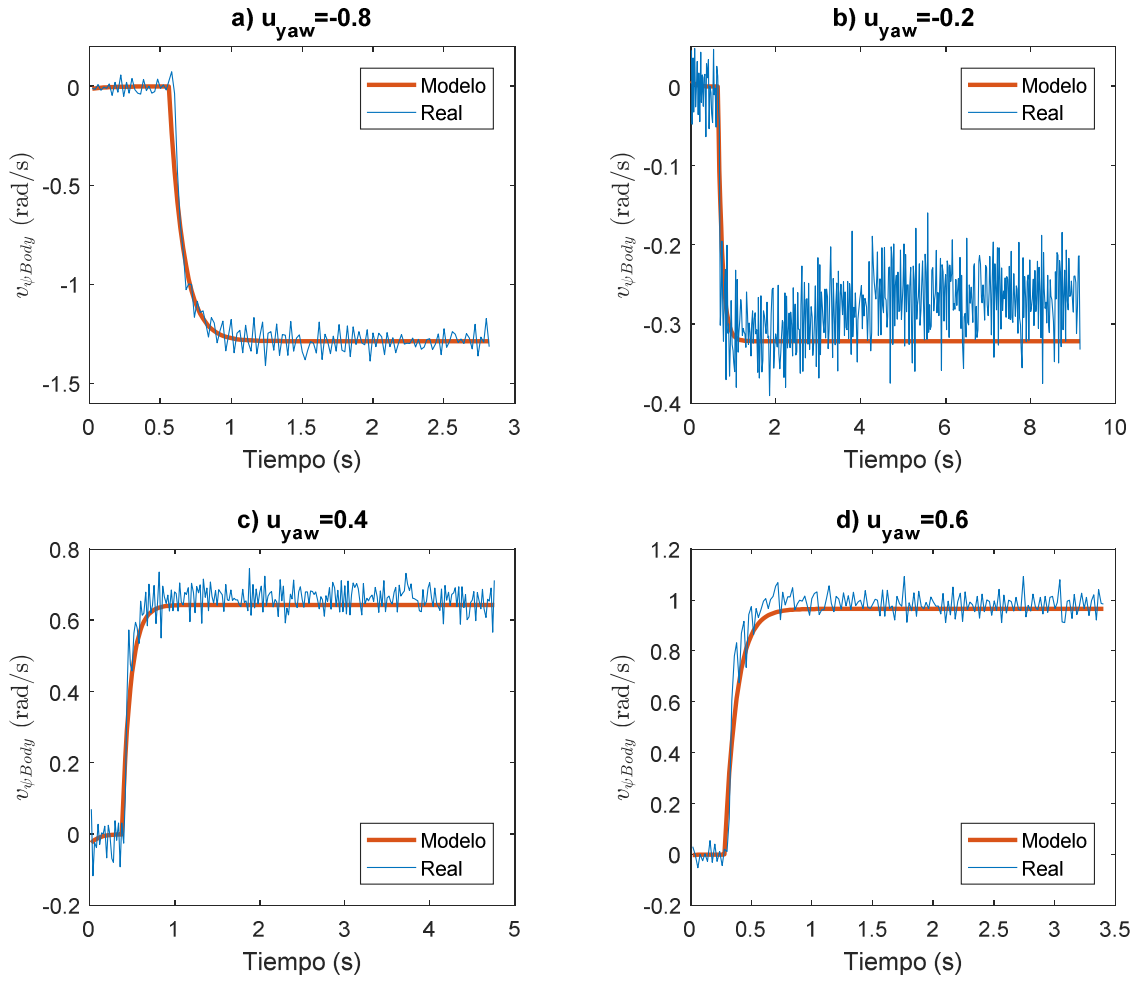


Figura 7.16: Comparación entre datos reales y simulados con el modelo para diferentes escalones de Throttle.

7.5. Modelo de los sensores

En este apartado se va a obtener un modelo de los sensores que proporcionan las medidas de posición y orientación, inclinación y velocidad. En cada uno de estos modelos se caracterizará el ruido existente en sus medidas, lo que resultará útil para la implementación del Filtro de Kalman que estimará, durante el control autónomo, el estado del cuadricóptero.

Las gráficas de la varianza de Allan que se incluyen en este apartado han sido obtenidas en Matlab gracias a la función creada por Fabian Czerwinski [45], y que se adjunta con el resto del código fuente del proyecto con su respectiva licencia de redistribución y uso.

7.5.1. Modelo de la cámara

El sistema de visión se explica con detalle en el ANEXO A y se realiza su calibración con frecuencia por lo que las medidas que se obtienen a través de él resultan lo suficientemente fiables. Sin embargo la cámara digital, como cualquier otro dispositivo sensor que capta magnitudes físicas, incorpora de forma inherente errores en la medida, entre otros [46]:

- Distorsión.
- Factor de escala.
- Errores de cuantificación.
- Ruido Gaussiano.
- Ruido Impulsional.
- Ruido de disparo.

Los errores como la distorsión en barril o el escalado ya han sido tenidos en cuenta para este sistema de visión tal y como se explica en el ANEXO A.

Los errores de cuantificación se deben al redondeo en la conversión analógica a digital y habitualmente son despreciables debido a que el resto de ruidos son mucho mayores sobre todo en la actualidad donde las cámaras digitales disponen de sensores que almacenan más bits por pixel.

El resto de errores son difícilmente medibles y por ello se va a considerar que la medida de la cámara presenta un ruido blanco de tipo Gaussiano que agrupe todos aquellos errores en la medida provenientes de fuentes de ruido.

De esta forma, el modelo de medida de la cámara puede simplificarse según se describe en la ecuación (7.8).

$$p_{cam} = \alpha_{cam} \cdot p_{real} + o_{cam} + n_{cam} \tag{7.8}$$

En dicha ecuación, p_{cam} es la posición que la cámara determina, p_{real} es la posición real, α_{cam} es el factor de escala de la medida de la cámara, o_{cam} es el offset de la medida de la cámara y n_{cam} es el ruido blanco de tipo Gaussiano que la cámara introduce en la medida.

Para identificar los parámetros del modelo se han realizado 20 experimentos. En cada uno de ellos se ha colocado el cuadricóptero con el triángulo de leds encendido en una posición, se ha tomado la medida real de dicha posición y se han registrado los datos que proporciona la cámara durante 180 segundos mientras el cuadricóptero se mantiene fijo en ella.

El resultado de uno de estos experimentos puede verse en la Figura 7.17, donde se aprecia la posición y la orientación que interpreta la cámara cuando el cuadricóptero se encuentra en la posición x_{real}=0.521 metros e y_{real}=0.355 metros con una orientación ψ_{real}=-0.66 radianes.

Tomando la media y la desviación estándar de la posición y la orientación de cada experimento se obtienen los valores mostrados en la Tabla 7.25.

x_{real} (m)	x_{cam} (m)	σ_{xcam} (m)	y_{real} (m)	y_{cam} (m)	σ_{ycam} (m)	ψ_{real} (rad)	ψ_{cam} (rad)	$\sigma_{\psi cam}$ (rad)
-0,085	-0,0846	2,43E-05	-0,033	-0,0311	3,40E-05	-1,60	-1,6061	1,92E-04
0,145	0,1431	1,31E-05	0,096	0,0978	1,01E-04	-1,25	-1,2547	2,13E-04
0,350	0,3528	2,05E-05	0,235	0,2337	3,22E-05	-0,97	-0,9735	3,87E-04
0,521	0,5250	4,75E-05	0,355	0,3541	3,93E-05	-0,66	-0,6615	2,34E-04
-0,113	-0,1110	8,25E-06	0,405	0,4043	2,84E-05	-0,35	-0,3468	1,71E-04
-0,833	-0,8320	2,49E-05	0,381	0,3833	2,51E-05	-0,03	-0,0233	1,24E-04
-0,631	-0,6292	2,91E-05	0,221	0,2193	1,46E-05	0,27	0,2832	5,24E-04
-0,375	-0,3763	1,52E-05	0,055	0,0562	3,65E-05	0,60	0,5865	1,50E-04

x_{real} (m)	x_{cam} (m)	σ_{xcam} (m)	y_{real} (m)	y_{cam} (m)	σ_{ycam} (m)	ψ_{real} (rad)	ψ_{cam} (rad)	$\sigma_{\psi cam}$ (rad)
-0,855	-0,8580	3,21E-05	-0,125	-0,1245	2,99E-05	0,90	0,9055	2,74E-04
-0,884	-0,8865	5,28E-05	-0,644	-0,6463	1,06E-05	1,22	1,2362	2,08E-04
-0,635	-0,6331	4,97E-06	-0,460	-0,4641	6,29E-06	1,57	1,5441	8,11E-05
-0,322	-0,3196	1,09E-05	-0,255	-0,2537	2,29E-05	1,84	1,8170	3,64E-04
-0,105	-0,1055	1,42E-06	-0,586	-0,5828	9,43E-07	2,12	2,1588	8,20E-06
0,587	0,5858	2,73E-05	-0,541	-0,5443	7,48E-05	2,44	2,4646	2,28E-04
0,424	0,4268	7,36E-06	-0,407	-0,4059	4,59E-05	2,80	2,7693	2,03E-04
0,174	0,1751	6,29E-07	-0,262	-0,2603	2,07E-06	3,14	3,0944	7,85E-05
0,355	0,3560	3,99E-05	-0,037	-0,0377	1,12E-05	-2,90	-2,8944	5,22E-04
-0,077	-0,0757	2,11E-06	0,245	0,2463	9,11E-07	-2,55	-2,5769	2,04E-04
-0,515	-0,5136	5,91E-06	-0,070	-0,0697	7,88E-06	-2,23	-2,2547	6,95E-04
-0,114	-0,1118	2,05E-05	-0,387	-0,3852	7,16E-05	1,91	-1,9367	1,51E-04

Tabla 7.25: Media y desviación estándar en las medidas de posición y orientación de la cámara.

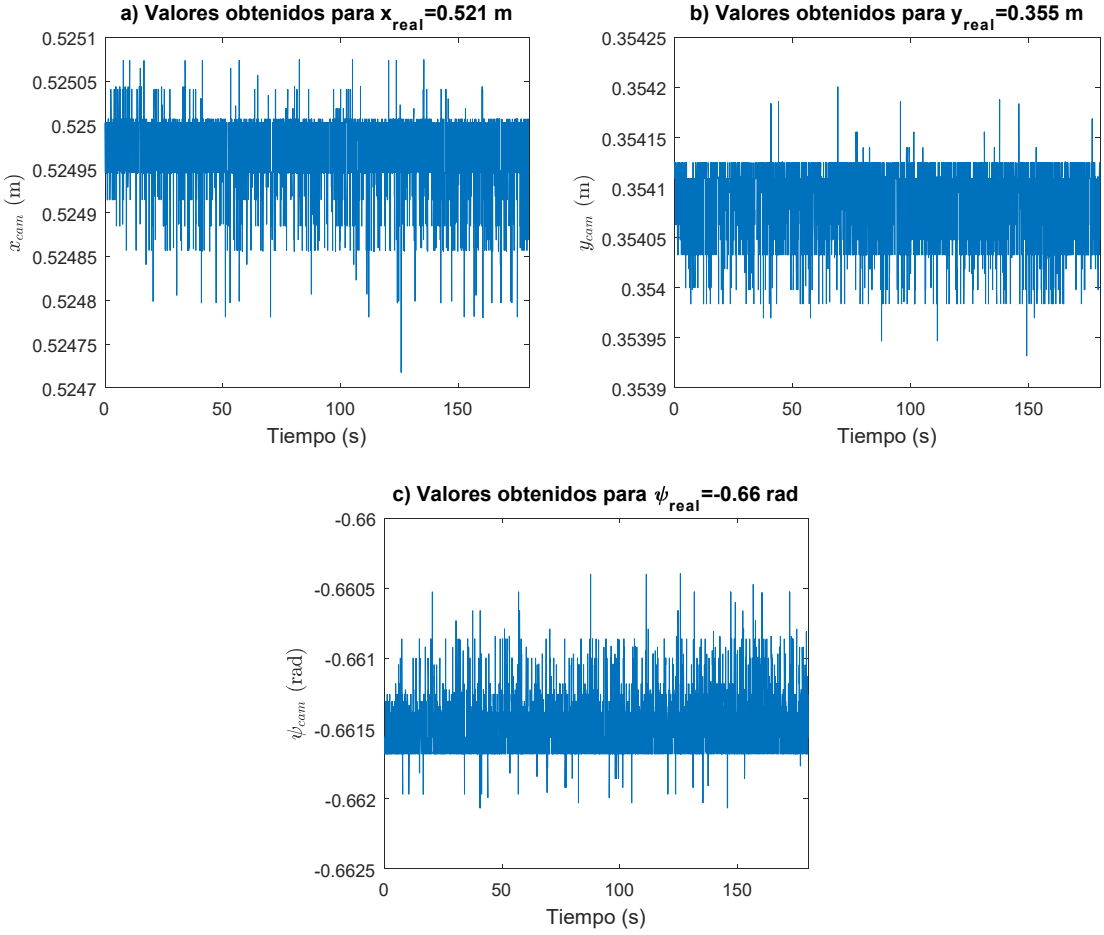


Figura 7.17: Posición y orientación tomada por la cámara para una posición determinada.

Con estos datos se pueden obtener los parámetros del modelo simplificado de la cámara a través de la línea recta que más se ajusta a los puntos obtenidos. Mediante un sencillo comando de Matlab se obtienen estas rectas con el método de los mínimos cuadrados. En la Figura 7.18 se puede ver el ajuste para cada componente

de la posición y para la orientación, mientras que en el conjunto de ecuaciones (7.9) se pueden ver las rectas que mejor se ajustan.

$$\begin{aligned} x_{cam} &= 1.0015x_{real} + 0.001 \\ y_{cam} &= 1.0008y_{real} + 0.00025 \\ \psi_{cam} &= 1.0103\psi_{real} - 0.2002 \end{aligned} \tag{7.9}$$

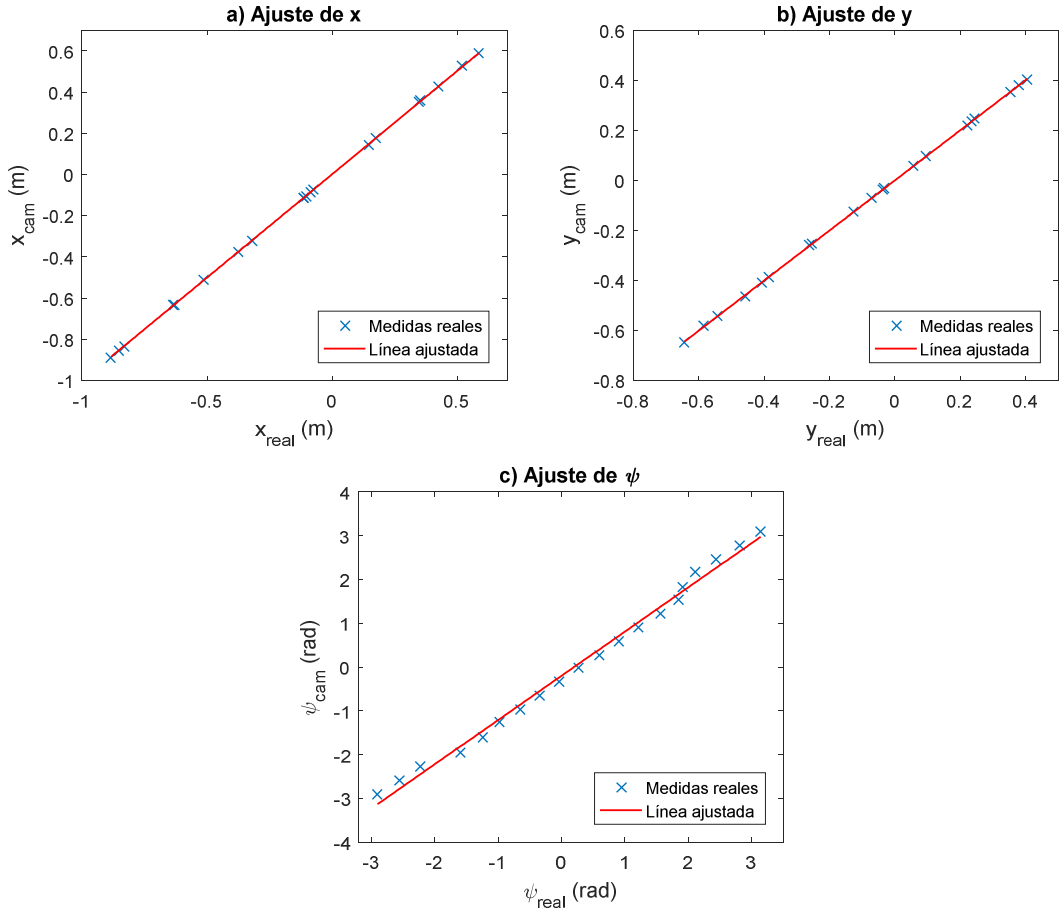


Figura 7.18: Ajuste por mínimos cuadrados de la posición y la orientación medidas por la cámara.

Por tanto, el modelo de la cámara queda como se muestra en el conjunto de ecuaciones (7.10) donde n_{cam} es un ruido blanco de tipo Gaussiano de desviación estándar σ_{cam} igual a la media de las desviaciones estándar de las medidas tomadas.

$$\begin{aligned} x_{cam} &= 1.0015x_{real} + 0.001 + n_{cam(\sigma_{xcam}=1.94\text{e-}05)} \\ y_{cam} &= 1.0008y_{real} + 0.00025 + n_{cam(\sigma_{ycam}=2.98\text{e-}05)} \\ \psi_{cam} &= 1.0103\psi_{real} - 0.2002 + n_{cam(\sigma_{\psi cam}=2.51\text{e-}04)} \end{aligned} \tag{7.10}$$

7.5.2. Modelo del sensor de ultrasonidos del AR.Drone 2.0

El sensor de ultrasonidos del AR.Drone 2.0 puede modelarse de forma sencilla con un modelo similar al de la cámara en el que se tenga en cuenta un factor de escala, un offset y un ruido blanco de tipo Gaussiano tal y como se muestra en la ecuación (7.11).

$$h_{us} = \alpha_{us} \cdot h_{real} + o_{us} + n_{us} \quad (7.11)$$

En dicha ecuación, h_{us} es la altura que el sensor de ultrasonidos determina, h_{real} es la altura real, α_{us} es el factor de escala de la medida del sensor de ultrasonidos, o_{us} es el offset de la medida del sensor de ultrasonidos y n_{us} es el ruido blanco de tipo Gaussiano que el sensor de ultrasonidos introduce en la medida.

Para identificar los parámetros del modelo se han realizado 12 experimentos. En cada uno de ellos se ha colocado el cuadricóptero a una altura, se ha tomado la medida real de dicha altura y se han registrado los datos que proporciona el AR.Drone 2.0 durante 180 segundos mientras el cuadricóptero se mantiene fijo en ella. Hay que tener en cuenta que la variable *altitude* de los datos de navegación *Navdata Demo* no proporciona ningún valor si los motores del cuadricóptero no están en marcha y no es posible realizar de forma correcta este tipo de identificación con él en vuelo (a no ser que se dispusiera de otros sistemas externos que la pudieran determinar, que no es el caso de este proyecto). Por tanto se ha utilizado la variable *alt_temp_raw* del *NavData Raw Measures* (idéntica a la variable *altitude_raw* del *NavData Altitude*) que proporciona una medida de la altura en bruto sin filtrar. En las condiciones en las que se va a realizar estos experimentos, la medida de altura en bruto y la medida filtrada deberían tener el mismo valor.

El resultado de uno de estos experimentos puede verse en la Figura 7.19, donde se aprecia la altura determinada por el cuadricóptero cuando éste se encuentra a un altura h_{real}=0.818 metros.

Tomando la media y la desviación estándar de la altura de cada experimento se obtienen los valores mostrados en la Tabla 7.26.

h_{real} (m)	h_{us} (m)	σ_{hus} (m)
0	0,237	4,98E-04
0,090	0,239	4,87E-04
0,150	0,239	4,92E-04
0,184	0,239	2,43E-04
0,195	0,241	4,97E-04
0,247	0,257	4,99E-04
0,298	0,288	1,02E-04
0,470	0,440	1,97E-04
0,696	0,656	4,63E-04
0,818	0,769	8,66E-04
1,012	0,949	3,31E-04
1,148	1,073	6,63E-04

Tabla 7.26: Media y desviación estándar en las medidas de altura del sensor de ultrasonidos.

En estos datos se observa que cuando el cuadricóptero se encuentra a una altura real entre 0 y 0.247 metros, el sensor de ultrasonidos siempre proporciona un valor cercano a los 0.24 metros. Para el resto de valores se puede obtener la línea recta que más se ajusta a dichos puntos. Mediante un sencillo comando de Matlab se obtiene esta recta con el método de los mínimos cuadrados. En la Figura 7.20 se puede ver este ajuste mientras que en la ecuación (7.12) se ve la recta que mejor se ajusta.

$$h_{us} = 0.9166 h_{real} + 0.0192 \quad (7.12)$$

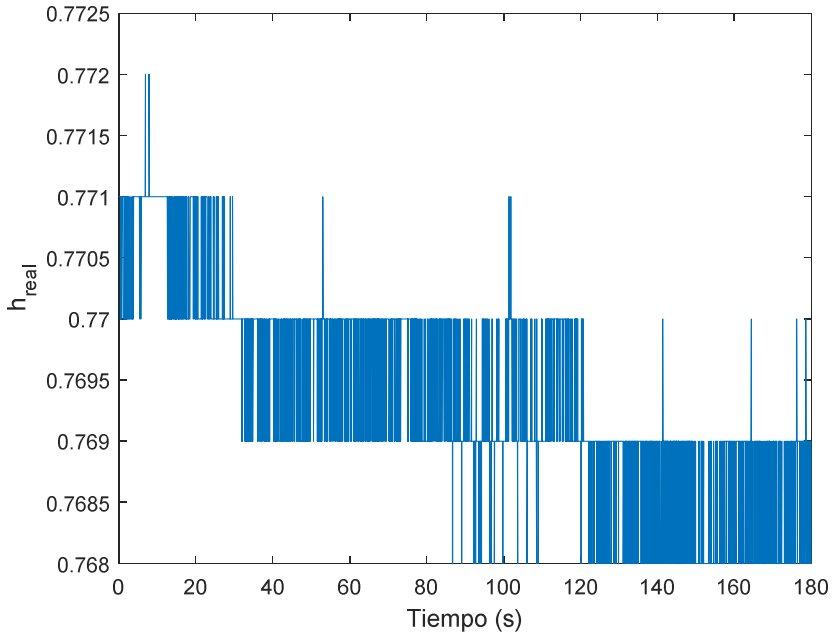


Figura 7.19: Altura tomada por el sensor de ultrasonidos para una altura fija determinada.

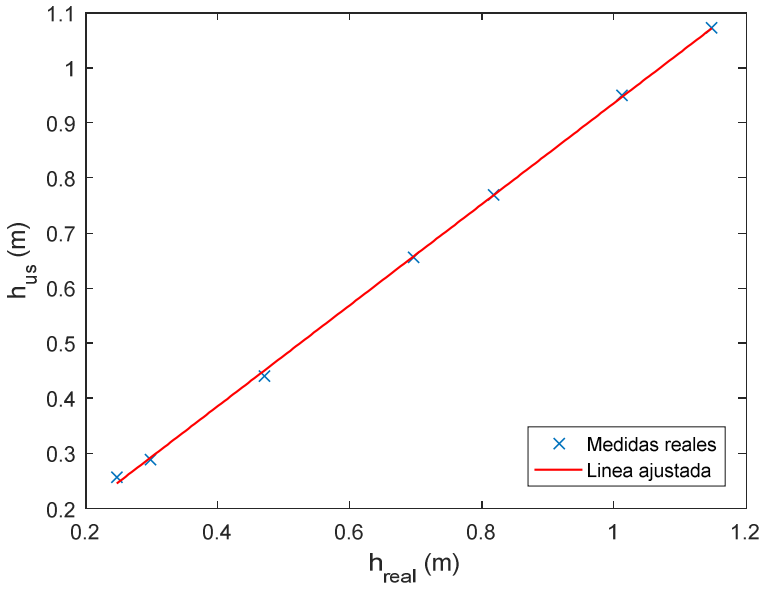


Figura 7.20: Ajuste por mínimos cuadrados de la altura medida por el sensor de ultrasonidos.

Por tanto, el modelo del sensor de ultrasonidos queda como se muestra en la ecuación (7.13) donde n_{us} es un ruido blanco de tipo Gaussiano de desviación estándar σ_{us} igual a la media de las desviaciones estándar de las medidas tomadas.

$$\begin{cases} h_{us} = 0.24 + n_{us(\sigma_{us}=4.43e-04)} & (h_{real} < 0.247) \\ h_{us} = 0.9166 h_{real} + 0.0192 + n_{us(\sigma_{us}=4.46e-04)} & (h_{real} \geq 0.247) \end{cases} \quad (7.13)$$

7.5.3. Modelo de los giróscopos del AR.Drone 2.0

Los giróscopos de una Unidad de Medida Inercial o IMU pueden modelarse de forma sencilla teniendo en cuenta tres fuentes de error tal y como se indica en [47] y en [41]: un offset variable, un offset constante y un error aleatorio.

Este sencillo modelo se muestra en la ecuación (7.14), donde ω_{gyro} es la velocidad angular que el giróscopo determina, ω_{real} es la velocidad angular real, b_{c_gyro} es un offset de la medida del giróscopo, b_{v_gyro} es un offset variable de la medida del giróscopo y n_{gyro} es un ruido de banda ancha.

$$\omega_{gyro} = \omega_{real} + b_{c_gyro} + b_{v_gyro} + n_{gyro} \tag{7.14}$$

El offset constante b_{c_gyro} del giróscopo puede determinarse como el valor medio del dato que proporciona el giróscopo cuando no existe rotación ninguna durante un experimento de larga duración.

Por su parte el offset variable b_{v_gyro} se debe principalmente al ruido de fluctuación en la electrónica y en otros componentes. Es un tipo de ruido que es más importante a bajas frecuencias y puede modelarse como un ruido de paseo aleatorio cuya desviación estándar se muestra en la ecuación (7.15).

$$\sigma_{v_gyro} = BS\sqrt{\delta t/\tau} \tag{7.15}$$

En dicha ecuación BS es el parámetro *Bias Stability* definido por el fabricante como el punto más bajo del gráfico de la Varianza de Allan, y evalúa la estabilidad/inestabilidad del offset variable del giróscopo dentro del tiempo medio τ de Allan con un periodo de muestreo δt.

El ruido de banda ancha n_{gyro} se debe a ruido termo-mecánico que fluctúa a frecuencias más altas que el periodo de muestreo. Por tanto puede modelarse como ruido blanco de media cero y con una desviación estándar σ_{gyro}. Integrando este ruido en el tiempo, se produce un ruido variable aleatorio en el ángulo cuya desviación estándar σ_θ satisface la ecuación (7.16).

$$\sigma_\theta(t) = \sigma_{gyro}\sqrt{\delta t \cdot t} \tag{7.16}$$

Los fabricantes de giróscopos definen habitualmente la Variación Aleatoria del Ángulo (ARW) según la ecuación (7.17).

$$ARW = \sigma_\theta(1) = \sigma_{gyro}\sqrt{\delta t} \tag{7.17}$$

Por tanto puede estimarse la desviación estándar del ruido blanco a través de la ARW, que puede determinarse mediante la Varianza de Allan, y del periodo de muestreo.

Para obtener, por tanto, los parámetros del modelo de los giróscopos es necesario realizar un experimento en el que se mantenga el cuadricóptero parado durante un largo tiempo mientras se registran los valores que proporcionan los giróscopos, obtener la media de dichos valores y realizar el análisis de la Varianza de Allan.

La Figura 7.21, la Figura 7.22 y la Figura 7.23 muestran los datos tomados por los giróscopos del cuadricóptero en dicho experimento, así como la gráfica de sus Varianzas de Allan, teniendo en cuenta que la frecuencia de muestreo ha sido de 50 Hz. Haciendo las medias de todas las medidas tomadas se determinan los valores de offsets b_{c_gyro} que se muestran en la Tabla 7.27.

Giróscopo	b_{c_gyro} (rad/s)
Eje x^B	-3.41e-05
Eje y^B	-2.66e-04
Eje z^B	-2.24e-04

Tabla 7.27: Valores de offsets b_c de los giróscopos.

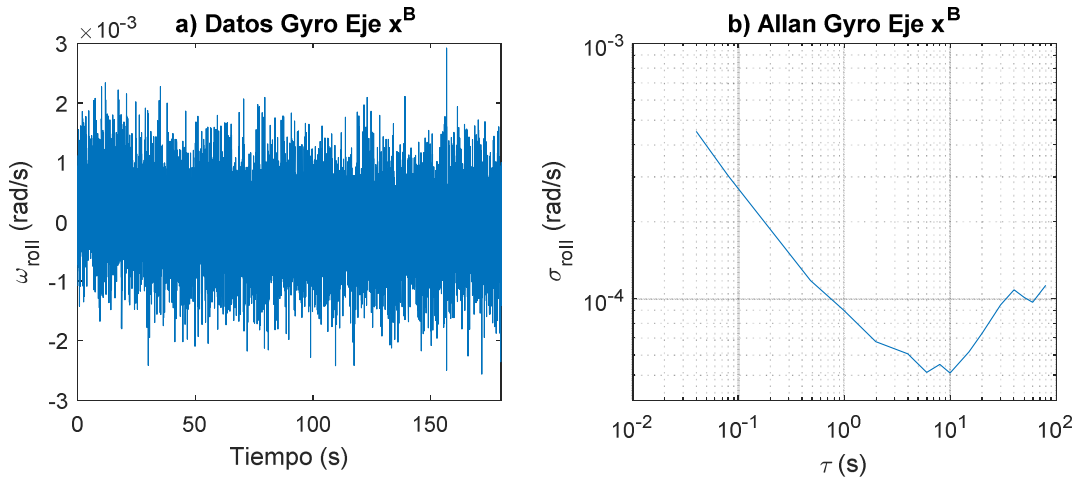


Figura 7.21: Medidas tomadas para el giróscopo del eje x^B y su gráfica de la Varianza de Allan.

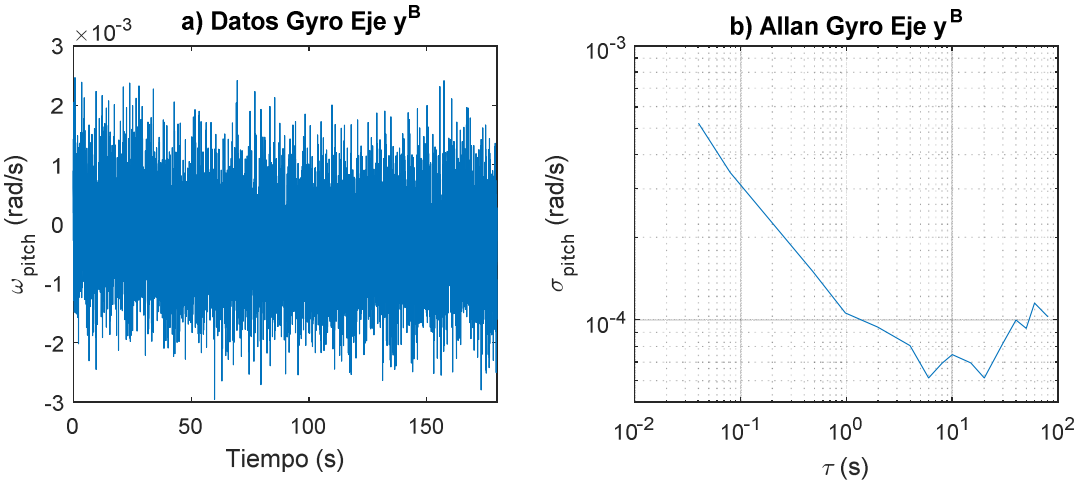


Figura 7.22: Medidas tomadas para el giróscopo del eje y^B y su gráfica de la Varianza de Allan.

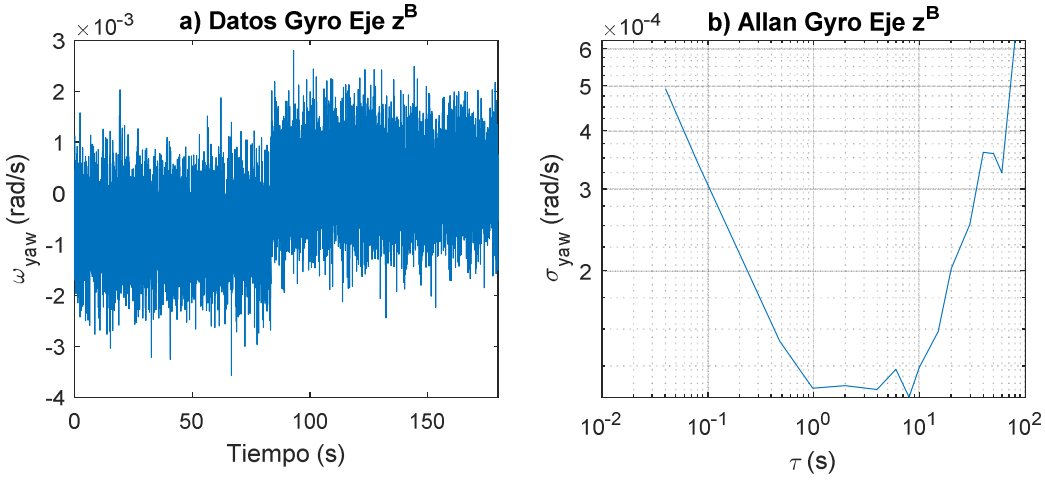


Figura 7.23: Medidas tomadas para el giróscopo del eje z^B y su gráfica de la Varianza de Allan.

La Variación Aleatoria del Ángulo (ARW) del ruido de un giróscopo se obtiene a través del valor de la desviación estándar de Allan con el tiempo medio τ=1 s. Tras realizar el análisis se determinan sus valores para los giróscopos, que se muestran en la Tabla 7.28.

Giróscopo	ARW $(\mathrm{rad}/\sqrt{s})$
Eje x^B	-3.41e-05
Eje y^B	-2.66e-04
Eje z^B	-2.24e-04

Tabla 7.28: Valores de ARW de los giróscopos.

Con estos valores, y gracias a la ecuación (7.17), es posible calcular la desviación estándar σ_{gyro} del ruido blanco n_{gyro} de los giróscopos. Estas desviaciones estándar se muestran en la Tabla 7.29.

Giróscopo	σ_{gyro} (rad/s)
Eje x^B	6.37e-04
Eje y^B	7.48e-04
Eje z^B	7.91e-04

Tabla 7.29: Valores de σ_{gyro} del ruido blanco de los giróscopos.

Como se ha dicho anteriormente el parámetro BS se define como el punto más bajo del gráfico de la Varianza de Allan. Tras analizar las gráficas se obtienen los valores de este parámetro y el τ de Allan en el que se encuentra, que se muestran en la Tabla 7.30.

Giróscopo	BS (rad/s)	τ (s)
Eje x^B	5.11e-05	10
Eje y^B	6.14e-05	6
Eje z^B	1.07e-04	8

Tabla 7.30: Valores del parámetro BS de los giróscopos.

Con estos valores, y gracias a la ecuación (7.15), es posible calcular la desviación estándar σ_{v_gyro} del ruido de paseo aleatorio por el cual se define el offset variable b_{v_gyro} de los giróscopos. Estas desviaciones estándar se muestran en la Tabla 7.31.

Giróscopo	σ_{v_gyro} (rad/s)
Eje x^B	2.29e-06
Eje y^B	3.55e-06
Eje z^B	5.36e-06

Tabla 7.31: Valores de σ_{v_gyro} del ruido de paseo aleatorio de los giróscopos.

El modelo de los giróscopos queda de forma definitiva tal y como se muestra en el conjunto de ecuaciones (7.18).

$$\begin{aligned}
\omega_{roll} &= \omega_{roll_real} - 3.41\text{e-}05 + b_{v_roll(\sigma_{v_roll}=2.29\text{e-}06)} + n_{roll(\sigma_{roll}=6.37\text{e-}04)} \\
\omega_{pitch} &= \omega_{pitch_real} - 2.66\text{e-}04 + b_{v_pitch(\sigma_{v_pitch}=3.55\text{e-}06)} + n_{pitch(\sigma_{pitch}=7.48\text{e-}04)} \\
\omega_{yaw} &= \omega_{yaw_real} - 2.24\text{e-}04 + b_{v_yaw(\sigma_{v_yaw}=5.36\text{e-}06)} + n_{yaw(\sigma_{yaw}=7.91\text{e-}04)}
\end{aligned} \tag{7.18}$$

7.5.4. Modelo de los acelerómetros del AR.Drone 2.0

Los acelerómetros de una Unidad de Medida Inercial o IMU pueden modelarse de la misma sencilla forma que los giróscopos tal y como se indica en [47]: un offset variable, un offset constante y un error aleatorio.

Este sencillo modelo se muestra en la ecuación (7.19), donde a_{acc} es la aceleración que el giróscopo determina, a_{real} es la aceleración real, b_{c_acc} es un offset de la medida del acelerómetro, b_{v_acc} es un offset variable de la medida del acelerómetro y n_{acc} es un ruido de banda ancha.

$$a_{acc} = a_{real} + b_{c_acc} + b_{v_acc} + n_{acc} \tag{7.19}$$

El offset constante b_{c_acc} del acelerómetro puede determinarse como el valor medio del dato que proporciona el acelerómetro cuando no existe movimiento ninguno durante un experimento de larga duración.

Por su parte el offset variable b_{v_acc} se debe principalmente al ruido de fluctuación en la electrónica y en otros componentes. Es un tipo de ruido que es más importante a bajas frecuencias y puede modelarse como un ruido de paseo aleatorio cuya desviación estándar se muestra en la ecuación (7.20).

$$\sigma_{v_acc} = BS\sqrt{\delta t/\tau} \tag{7.20}$$

En dicha ecuación BS es el parámetro *Bias Stability* definido por el fabricante como el punto más bajo del gráfico de la Varianza de Allan, y evalúa la estabilidad/inestabilidad del offset variable del acelerómetro dentro del tiempo medio τ de Allan con un periodo de muestreo δt.

El ruido de banda ancha n_{acc} se debe a ruido termo-mecánico que fluctúa a frecuencias más altas que el periodo de muestreo. Por tanto puede modelarse como ruido blanco de media cero y con una desviación estándar σ_{acc}. Integrando este ruido en el tiempo, se produce un ruido variable aleatorio en la velocidad cuya desviación estándar σ_{vel} satisface la ecuación (7.21).

$$\sigma_{vel}(t) = \sigma_{acc}\sqrt{\delta t \cdot t} \tag{7.21}$$

Los fabricantes de acelerómetros definen habitualmente la Variación Aleatoria de la Velocidad (VRW) según la ecuación (7.22).

$$VRW = \sigma_{vel}(1) = \sigma_{acc}\sqrt{\delta t} \tag{7.22}$$

Por tanto puede estimarse la desviación estándar del ruido blanco a través de la VRW, que puede determinarse mediante la Varianza de Allan, y del periodo de muestreo.

Para obtener, por tanto, los parámetros del modelo de los acelerómetros es necesario realizar un experimento en el que se mantenga el cuadricóptero parado durante un largo tiempo mientras se registran los valores que proporcionan los acelerómetros, obtener la media de dichos valores y realizar el análisis de la Varianza de Allan.

La Figura 7.24, la Figura 7.25 y la Figura 7.26 muestran los datos tomados por los acelerómetros del cuadricóptero en dicho experimento, así como la gráfica de sus Varianzas de Allan, teniendo en cuenta que la frecuencia de muestreo ha sido de 50 Hz. Haciendo las medias de todas las medidas tomadas se determinan los valores de offsets b_{c_acc} que se muestran en la Tabla 7.32.

Acelerómetro	b_{c_acc} (m/s^2)
Eje x^B	0.7056
Eje y^B	0.1118
Eje z^B	-9.7570

Tabla 7.32: Valores de offsets b_{c_acc} de los acelerómetros.

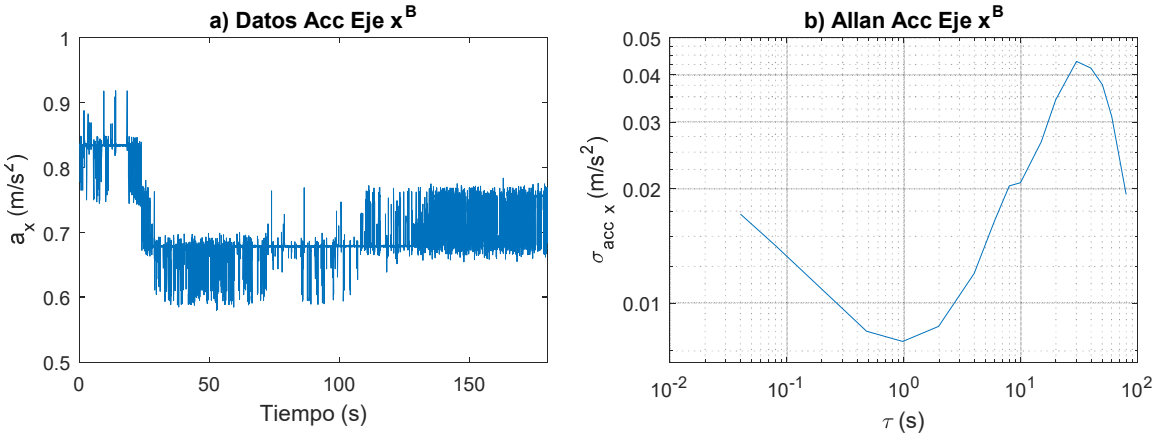


Figura 7.24: Medidas tomadas para el acelerómetro del eje x^B y su gráfica de la Varianza de Allan.

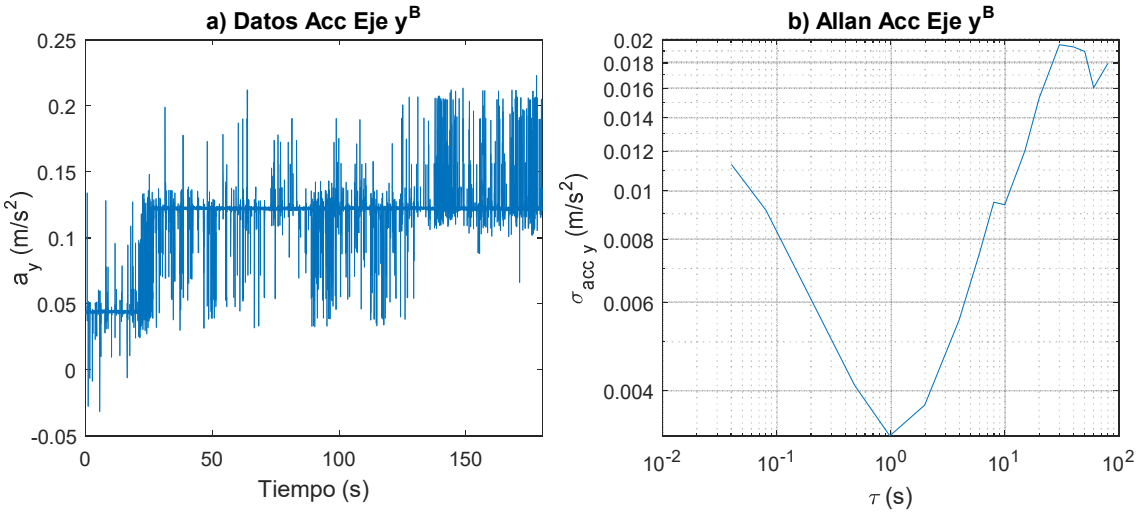


Figura 7.25: Medidas tomadas para el acelerómetro del eje y^B y su gráfica de la Varianza de Allan.

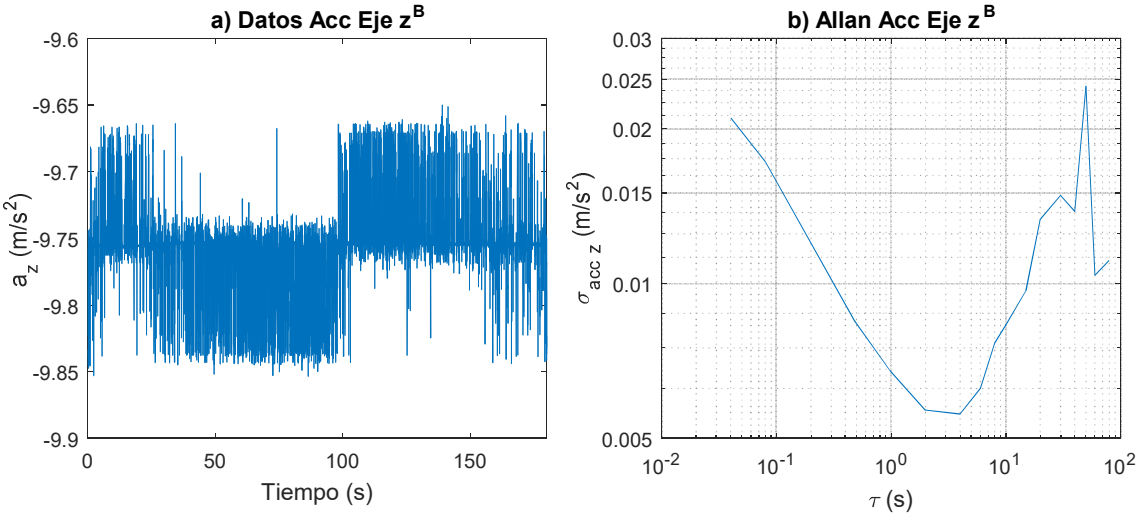


Figura 7.26: Medidas tomadas para el acelerómetro del eje z^B y su gráfica de la Varianza de Allan.

La Variación Aleatoria de la Velocidad (VRW) del ruido de un acelerómetro se obtiene a través del valor de la desviación estándar de Allan con el tiempo medio τ=1 s. Tras realizar el análisis se determinan sus valores para los acelerómetros, que se muestran en la Tabla 7.33.

Acelerómetro	VRW (m/s)
Eje x^B	7.94e-03
Eje y^B	3.25e-03
Eje z^B	6.76e-03

Tabla 7.33: Valores de VRW de los acelerómetros.

Con estos valores, y gracias a la ecuación (7.22), es posible calcular la desviación estándar σ_{acc} del ruido blanco n_{acc} de los acelerómetros. Estas desviaciones estándar se muestran en la Tabla 7.34.

Acelerómetro	σ_{acc} (m/s^2)
Eje x^B	5.61e-02
Eje y^B	2.30e-02
Eje z^B	4.78e-02

Tabla 7.34: Valores de σ_{acc} del ruido blanco de los acelerómetros.

Como se ha dicho anteriormente el parámetro BS se define como el punto más bajo del gráfico de la Varianza de Allan. Tras analizar las gráficas se obtienen los valores de este parámetro y el τ de Allan en el que se encuentra, que se muestran en la Tabla 7.35.

Acelerómetro	BS (m/s^2)	τ (s)
Eje x^B	7.94e-03	1
Eje y^B	3.25e-03	1
Eje z^B	5.58e-03	4

Tabla 7.35: Valores del parámetro BS de los acelerómetros.

Con estos valores, y gracias a la ecuación (7.20), es posible calcular la desviación estándar σ_{v_acc} del ruido de paseo aleatorio por el cual se define el offset variable b_{v_acc} de los acelerómetros. Estas desviaciones estándar se muestran en la Tabla 7.36.

Acelerómetro	σ_{v_acc} (m/s^2)
Eje x^B	1.13e-03
Eje y^B	4.64e-04
Eje z^B	3.95e-04

Tabla 7.36: Valores de σ_{v_acc} del ruido de paseo aleatorio de los acelerómetros.

El modelo de los acelerómetros queda de forma definitiva tal y como se muestra en el conjunto de ecuaciones (7.23).

$$\begin{aligned} a_x &= a_{x_real} + 0.7056 + b_{v_x(\sigma_{v_x}=1.13e-03)} + n_{x(\sigma_x=5.61e-02)} \\ a_y &= a_{y_real} + 0.1118 + b_{v_y(\sigma_{v_y}=4.64e-04)} + n_{y(\sigma_y=2.30e-02)} \\ a_z &= a_{z_real} - 9.7570 + b_{v_z(\sigma_{v_z}=3.95e-04)} + n_{z(\sigma_z=4.78e-02)} \end{aligned} \quad (7.23)$$

8. PROPUESTA DE CONTROL

El modelo obtenido en el capítulo anterior va a utilizarse para crear una ley de control que permita llevar el sistema a un estado estable o a seguir determinadas referencias de navegación. En el presente proyecto se ha optado por elegir una estrategia de control desacoplado, de modo que una variable de la salida del sistema sea controlada únicamente por un controlador.

Por su sencillez y facilidad de diseño y ajuste se va a proponer el uso de controladores PID y se va a implementar un Filtro de Kalman para estimar las variables del estado del cuadricóptero de una manera fiable, sobre todo en el caso de pérdida temporal de las medidas proporcionadas por el sistema de visión.

8.1. Estimación del estado

En este proyecto se va a utilizar el Filtro de Kalman discreto para obtener una estimación del estado del cuadricóptero, para filtrar las medidas obtenidas a través del cuadricóptero y del sistema de visión externo, así como para realizar la fusión de las medidas de los diferentes sensores.

8.1.1. Descripción del Filtro de Kalman

El Filtro de Kalman, [48], [49] y [50], es un estimador del estado recursivo de tipo Bayesiano que utiliza datos de entrada con ruido e incertidumbre y obtiene una óptima estimación del estado para sistemas lineales, a través de un algoritmo con una eficiencia computacional bastante buena. Un sistema lineal puede representarse según el sistema (8.1).

$$x_{k+1} = F_k x_k + G_k u_k + w_k \tag{8.1}$$

En dicho sistema x_k es el estado del sistema en el instante k, u_k es el vector de entradas de control, w_k es el ruido aditivo del proceso, G_k es la matriz de transición de entradas y F_k es la matriz de transición de estados.

Además, las observaciones del estado se realizan a través de medidas del sistema que puede representarse por una ecuación lineal como se indica en (8.2).

$$z_k = H_k x_k + v_k \tag{8.2}$$

En esta ecuación z_k son las medidas u observaciones realizadas en el instante k, H_k es la matriz de observaciones y v_k es el ruido aditivo de las medidas.

Para la utilización de este Filtro de Kalman se asume que el ruido del proceso w_k y el ruido de las medidas v_k son procesos de ruido blanco no correlacionados, de media cero, y con matrices de covarianza conocidas Q_k y R_k, respectivamente, simétricas semidefinidas positivas.

El algoritmo de la estimación mediante el Filtro de Kalman se compone de dos fases. En la primera fase, denominada de "Predicción", se predice el estado y la varianza en el instante $k+1$ en función de la información que se dispone en el instante k. Esta fase se traduce en las ecuaciones mostradas en (8.3) y (8.4).

$$\hat{x}_{k+1|k} = F_k \hat{x}_{k|k} + G_k u_k \tag{8.3}$$

$$\hat{P}_{k+1|k} = F_k \hat{P}_{k|k} F_k^T + Q_k \tag{8.4}$$

En dichas ecuaciones $\hat{x}_{k+1|k}$ es la predicción del estado en función de la información de la que se dispone en el instante k, $\hat{x}_{k|k}$ es la estimación del estado que el Filtro de Kalman calculó en la última iteración, $\hat{P}_{k+1|k}$ es la predicción de la matriz de covarianzas del error cuadrático medio de la predicción $\hat{x}_{k+1|k}$ en función de la información de la que se dispone en el instante k, y $\hat{P}_{k|k}$ es la estimación de la matriz de covarianzas del error cuadrático medio de la predicción $\hat{x}_{k|k}$ que el Filtro de Kalman calculó en la última iteración.

En la segunda fase, denominada de "Actualización", se actualiza el estado y la matriz de covarianzas del error en la predicción usando una combinación del estado predicho y de las observaciones z_{k+1}. Esta fase se traduce en las ecuaciones mostradas en (8.6) y (8.7), teniendo en cuenta la ganancia del Filtro de Kalman K_{k+1} expresada en la ecuación (8.5).

$$K_{k+1} = \hat{P}_{k+1|k} H_{k+1}^T \left[H_k \hat{P}_{k+1|k} H_{k+1}^T + R_{k+1} \right]^{-1} \tag{8.5}$$

$$\hat{x}_{k+1|k+1} = \hat{x}_{k+1|k} + K_{k+1} \left[z_{k+1} - H_{k+1} \hat{x}_{k+1|k} \right] \tag{8.6}$$

$$\hat{P}_{k+1|k+1} = (I - K_{k+1} H_{k+1}) \hat{P}_{k+1|k} (I - K_{k+1} H_{k+1})^T + K_{k+1} R_{k+1} K_{k+1}^T \tag{8.7}$$

En estas ecuaciones $\hat{x}_{k+1|k+1}$ es la estimación del estado resultado de la aplicación del Filtro de Kalman, $\hat{P}_{k+1|k+1}$ es la estimación de la matriz de covarianzas del error de la predicción del estado resultado de la aplicación del Filtro de Kalman e I es la matriz identidad.

8.1.2. Implementación del Filtro de Kalman

Para implementar el Filtro de Kalman en el control del cuadricóptero AR.Drone 2.0, se va a definir el estado x_k en función de la posición y orientación del cuadricóptero y su velocidad tal y como se muestra en la ecuación (8.8).

$$x_k = [x \quad y \quad z \quad \psi \quad \dot{x} \quad \dot{y} \quad \dot{z} \quad \dot{\psi}]^T \tag{8.8}$$

En dicha ecuación x, y y z es la posición del cuadricóptero expresada en metros, mientras que $\dot{x}$, $\dot{y}$ y $\dot{z}$ son las velocidades lineales respectivas expresadas en m/s. Por su parte ψ es el ángulo de orientación del cuadricóptero expresado en radianes mientras que $\dot{\psi}$ es la velocidad angular de dicha orientación expresada en rad/s. El estado se define con respecto al marco de referencia del lugar de experimentación E.

El modelo del proceso se puede definir como una implementación discreta de un modelo cinemático de velocidad constante tal y como se describe en [49] según el sistema de ecuaciones (8.9), donde δt es el incremento de tiempo existente entre el instante k y el instante $k+1$.

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ z_{k+1} \\ \psi_{k+1} \\ \dot{x}_{k+1} \\ \dot{y}_{k+1} \\ \dot{z}_{k+1} \\ \dot{\psi}_{k+1} \end{bmatrix} = \begin{bmatrix} x_k + \delta t \dot{x}_k \\ y_k + \delta t \dot{y}_k \\ z_k + \delta t \dot{z}_k \\ \psi_k + \delta t \dot{\psi}_k \\ \dot{x}_k \\ \dot{y}_k \\ \dot{z}_k \\ \dot{\psi}_k \end{bmatrix} \tag{8.9}$$

En este modelo se asume que los cambios de velocidad en el intervalo de tiempo entre el instante k y el instante $k+1$ es un ruido blanco. Para poder aplicar esta asunción el incremento de tiempo debería ser lo suficientemente pequeño y la covarianza de las velocidades del Filtro de Kalman debería reflejar sus posibles cambios entre ambos instantes.

De este sistema se deriva fácilmente la matriz de transición de estados F_k, que queda como se muestra en la ecuación (8.10).

$$F_k = \begin{bmatrix} 1 & 0 & 0 & 0 & \delta t & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & \delta t & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & \delta t & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & \delta t \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{8.10}$$

El sistema de visión externo proporciona valores para todas las medidas excepto las relacionadas con la altura, pero lo hace con una frecuencia menor (18.7 Hz) que las medidas que se reciben a través del cuadricóptero (50 Hz). Del cuadricóptero se obtiene la altura z y su velocidad $\dot{z}$, la velocidad angular v_ψ^B y las velocidades lineales v_x^B y v_y^B que son fácilmente transformadas al marco de referencia del lugar de experimentación E para obtener las velocidades $\dot{\psi}$, $\dot{x}$ e $\dot{y}$ respectivamente. El sistema de visión externo proporciona unas medidas muy fiables pero cuando no se dispone de ellas, bien porque no ha sido recibida una nueva medida, bien porque el cuadricóptero se ha salido del área de observación de la cámara o bien porque el sistema de visión ha fallado por no determinar correctamente el triángulo de luz, se pueden utilizar las medidas que proporciona el cuadricóptero. El Filtro de Kalman puede utilizarse, por tanto, para fusionar las medidas proporcionadas por diferentes sensores y así realizar una mejor estimación en base a los valores muy fiables del sistema de visión y a la odometría usando las medidas, algo menos fiables, del cuadricóptero.

Por tanto, se dispondrá de diferentes vectores y matrices de observación en función de si se dispone de medidas del sistema de visión en una iteración o no, utilizándose unas u otras en la iteración actual del Filtro de Kalman.

El vector de medidas proporcionadas por el AR.Drone 2.0 se muestra en la ecuación (8.11), mientras que su matriz de observaciones se muestra en la ecuación (8.12).

$$z_D = \begin{bmatrix} z \\ \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} z^B \\ v_x^B \cos\psi_m - v_y^B \sin\psi_m \\ -v_x^B \sin\psi_m - v_y^B \cos\psi_m \\ v_z^B \\ -v_\psi^B \end{bmatrix} \tag{8.11}$$

$$H_D = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{8.12}$$

En la transformación de velocidades lineales del marco de referencia B al marco de referencia E de la ecuación (8.11) se ha utilizado el ángulo modificado ψ_m. La razón es que la medida del ángulo ψ es proporcionada de forma fiable por el sistema de visión, pero no puede utilizarse de forma directa en la transformación porque dicha medida podría no ser correcta y estar desactualizada durante múltiples iteraciones. Por ello, cuando no se dispone de la medida del ángulo ψ a través del sistema de visión, en cada iteración se calcula un nuevo ángulo modificado ψ_m que tiene en cuenta el ángulo modificado de la iteración anterior, la velocidad angular v_ψ^B y el tiempo transcurrido entre la medida actual y la anterior. Cuando en una iteración se dispone de las medidas del sistema de visión, el ángulo modificado ψ_m se resetea al valor ψ que proporciona dicho sistema. El cálculo de este ángulo modificado se puede ver en la ecuación (8.13).

$$\psi_m(k+1) = \psi_m(k) - \big(t(k+1) - t(k)\big)v_\psi^B(k+1) \tag{8.13}$$

Por su parte, el vector de medidas proporcionadas por el sistema de visión se muestra en la ecuación (8.14), mientras que su matriz de observaciones se muestra en la ecuación (8.15). En ellas se tiene en cuenta que en cada iteración siempre se dispone de las medidas que proporciona el cuadricóptero, por lo que este vector y esta matriz se encuentran ampliados con la altura y su velocidad.

$$z_{CD} = \begin{bmatrix} x \\ y \\ z \\ \psi \\ \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} x^E \\ y^E \\ z^B \\ \psi^E \\ v_x^E \\ v_y^E \\ v_z^B \\ v_\psi^E \end{bmatrix} \tag{8.14}$$

$$H_{CD} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{8.15}$$

Para completar el Filtro de Kalman falta por definir las matrices de covarianzas Q_k y R_k. Esta última es la matriz de covarianzas del ruido de las medidas proporcionadas por los sensores, y da información de cómo de fiables son estas medidas. Para establecer los valores de esta matriz se pueden utilizar los modelos de los sensores que se determinaron en el apartado 7.5 considerando que el ruido en las medidas de un sensor no está correlacionado con el de ningún otro.

Con respecto a las medidas que proporciona el AR.Drone 2.0, la medida de la altura z^B es proporcionada a través de su sensor de ultrasonidos y se determinó que este sensor está afectado por un ruido de tipo Gaussiano n_{us} con una desviación estándar $\sigma_{us} = 4.46\text{e-}04$ m, por lo que su varianza resulta ser de $\sigma_{us}^2 = 1.99\text{e-}07$ m^2. La medida de la velocidad v_z^B es calculada internamente por el AR.Drone 2.0, según lo visto en los apartados 4.1.6 y 4.2.2, a través de la medida de altura. Asumiendo que la velocidad es obtenida a través de una derivada de primer orden, el ruido propagado de la medida de la altura a la medida de su velocidad tendrá una desviación estándar de $\sigma_{vz}^B = 2\sigma_{us}/\delta t = 8.92\text{e-}04/0.02 = 4.46\text{e-}02$ m/s. Con esto, la varianza resulta ser de $\sigma_{vz}^{B\,2} = 1.99\text{e-}03$ (m/s)^2.

En estos mismos apartados se ha visto que las velocidades angulares se determinan a través de los giróscopos del AR.Drone 2.0. Para el caso de la velocidad angular v_ψ^B se determinó que el giróscopo correspondiente está afectado por un ruido de tipo Gaussiano n_{yaw} con una desviación estándar $\sigma_{yaw} = 7.91\text{e-}04$ rad/s, por lo que su varianza resulta ser de $\sigma_{yaw}^2 = 6.26\text{e-}07$ (rad/s)^2.

De igual forma, las velocidades lineales v_x^B y v_y^B son determinadas por el AR.Drone 2.0 mediante una fusión de las medidas de los giróscopos, de los acelerómetros y de su cámara vertical. Se desconoce cómo es el algoritmo que fusiona estas medidas por lo que es complicado saber el ruido que afecta a estas medidas. Por ello se va a asumir que la desviación estándar del error de estas medidas sea la mayor de entre el giróscopo y el acelerómetro correspondiente. La desviación estándar del error en la medida de v_x^B se asume, por tanto, de un

valor $\sigma_{vx}^{B} = 5.61\text{e-}02$ m/s y su varianza de $\sigma_{vx}^{B\,2} = 3.15\text{e-}03\ (m/s)^2$. Por su parte, la desviación estándar del error en la medida de v_y^B se asume de un valor $\sigma_{vy}^{B} = 2.30\text{e-}02$ m/s y su varianza de $\sigma_{vy}^{B\,2} = 5.29\text{e-}04\ (m/s)^2$.

Con respecto a las medidas que proporciona el sistema de visión externo, se determinó que la medida de la posición x^E está afectada por un ruido de tipo Gaussiano n_{xcam} con una desviación estándar $\sigma_{xcam} = 1.94\text{e-}05$ m, por lo que su varianza resulta ser de $\sigma_{xcam}^2 = 3.76\text{e-}10\ m^2$. La medida de la posición y^E está afectada por un ruido de tipo Gaussiano n_{ycam} con una desviación estándar $\sigma_{ycam} = 2.98\text{e-}05$ m, por lo que su varianza resulta ser de $\sigma_{ycam}^2 = 8.88\text{e-}10\ m^2$. Y por último, la medida del ángulo de orientación ψ^E está afectada por un ruido de tipo Gaussiano $n_{\psi cam}$ con una desviación estándar $\sigma_{\psi cam} = 2.51\text{e-}04$ rad, por lo que su varianza resulta ser de $\sigma_{\psi cam}^2 = 6.30\text{e-}08\ rad^2$.

Las medidas de velocidad que proporciona el sistema de visión se obtienen a través de la derivada de primer orden de la posición y de la orientación. El ruido propagado de la medida x^E a la medida de velocidad v_x^E tendrá, por tanto, una desviación estándar de $\sigma_{vx}^E = 2\sigma_{xcam}/\delta t = 3.88\text{e-}05/0.0535 = 7.26\text{e-}04$ m/s. Con esto, la varianza resulta ser de $\sigma_{vx}^{E\,2} = 5.27\text{e-}07\ (\text{m/s})^2$. El ruido propagado de la medida y^E a la medida de velocidad v_y^E tendrá, de la misma manera, una desviación estándar de $\sigma_{vy}^E = 2\sigma_{ycam}/\delta t = 5.96\text{e-}05/0.0535 = 1.11\text{e-}03$ m/s. Con esto, la varianza resulta ser de $\sigma_{vy}^{E\,2} = 1.24\text{e-}06\ (\text{m/s})^2$. Por último, el ruido propagado de la medida ψ^E a la medida de velocidad v_ψ^E tendrá una desviación estándar de $\sigma_{v\psi}^E = 2\sigma_{\psi cam}/\delta t = 5.02\text{e-}04/0.0535 = 9.38\text{e-}03$ rad/s. Con esto, la varianza resulta ser de $\sigma_{v\psi}^{E\,2} = 8.80\text{e-}05\ (\text{rad/s})^2$.

Al igual que sucede con el vector de medidas y con la matriz de observaciones, la matriz R_k tiene una forma u otra en función de si en la iteración actual del Filtro de Kalman se dispone de las medidas del sistema de visión o no. Por tanto, cuando no se dispone de estas medidas la matriz R_k tendrá los valores que se indican en la ecuación (8.16).

$$R_D = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.99\text{e-}07 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3.15\text{e-}03 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 5.29\text{e-}04 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1.99\text{e-}03 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6.26\text{e-}07 \end{bmatrix} \quad (8.16)$$

Mientras que si se dispone de las medidas del sistema de visión, la matriz R_k tendrá los valores que se indican en la ecuación (8.17).

$$R_{CD} = \begin{bmatrix} 3.76\text{e-}10 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 8.88\text{e-}10 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.99\text{e-}07 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 6.30\text{e-}08 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5.27\text{e-}07 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.24\text{e-}06 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1.99\text{e-}03 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 8.80\text{e-}05 \end{bmatrix} \quad (8.17)$$

La matriz de covarianzas de ruido del proceso Q_k es difícil de diseñar por los siguientes motivos [51]:

- Las matemáticas necesarias para diseñarla requieren de grandes conocimientos en teoría de la señal.
- Se está intentando modelar un ruido para el que se dispone de poca información debido a que el modelo del proceso no es ni completo ni perfecto, conteniendo así dinámicas no modeladas. Por tanto

no se puede discernir fácilmente si lo que se ha modelado como ruido del proceso es una dinámica no modelada o es un ruido realmente.

Aunque existen métodos [52] para estimar los valores de esta matriz, lo habitual es realizar simulaciones a través de datos obtenidos experimentalmente variando los valores de la matriz y observando los resultados, tal y como se indica en [51]. En este sentido hay que comprender que si el valor de Q_k es demasiado pequeño con respecto al valor de R_k entonces el Filtro de Kalman confiará mucho más en su modelo de predicción que en las medidas observadas. Si por el contrario Q_k es demasiado grande con respecto al valor de R_k entonces el Filtro de Kalman confiará mucho más en las medidas observadas que en su modelo de predicción y, por tanto, estará más influenciado por el ruido de los sensores.

Tras diversas pruebas modificando valores de la matriz Q_k y observando los resultados obtenidos al ejecutar el Filtro de Kalman sobre datos experimentales de vuelos de prueba, se llega a los valores que se muestran en la ecuación (8.18) y que ofrecen unos resultados satisfactorios.

$$Q = \begin{bmatrix} 5e\text{-}12 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1e\text{-}11 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 5e\text{-}02 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 5e\text{-}10 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1e\text{-}08 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1e\text{-}08 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 5e\text{-}05 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1e\text{-}08 \end{bmatrix} \quad (8.18)$$

Estos resultados pueden verse desde la Figura 8.1 hasta la Figura 8.4 donde se ha seleccionado una parte de un vuelo de prueba en el que el cuadricóptero se salió dos veces seguidas del área de observación de la cámara. En relación a la altura y su velocidad que pueden verse en estas figuras hay que tener en cuenta que dichas medidas están proporcionadas por el AR.Drone 2.0 y, por tanto, siempre se encuentran disponibles aunque se salga del área de observación.

Las simulaciones del Filtro de Kalman han sido realizadas con Matlab mediante los ficheros *SimulaKalman.m* y *kalmanf.m* que se adjuntan con el resto del código fuente del proyecto.

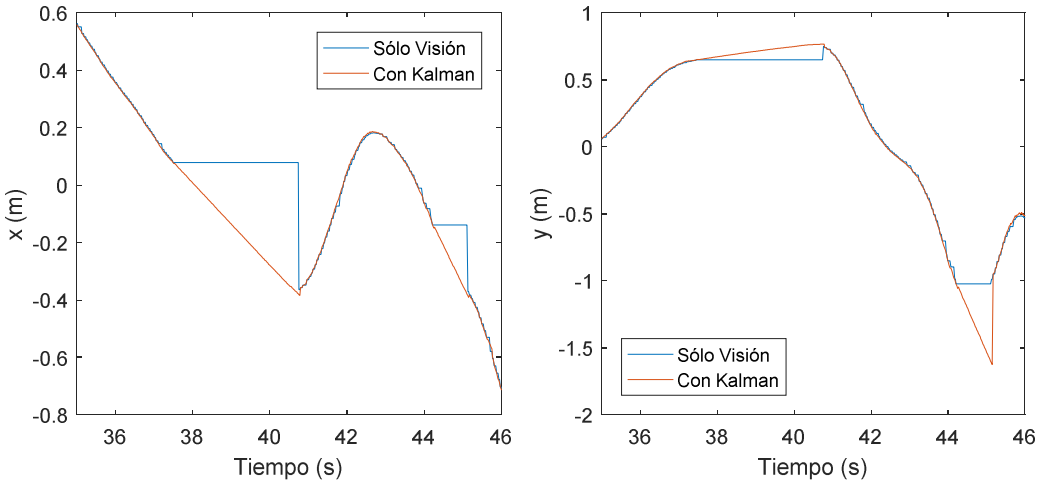


Figura 8.1: Resultado del Filtro de Kalman para x e y en un vuelo de prueba.

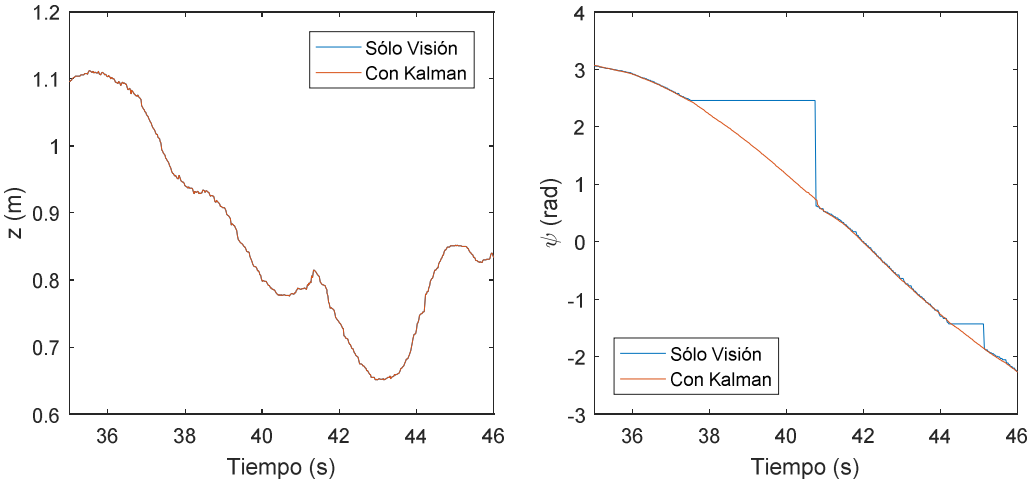


Figura 8.2: Resultado del Filtro de Kalman para z y ψ en un vuelo de prueba.

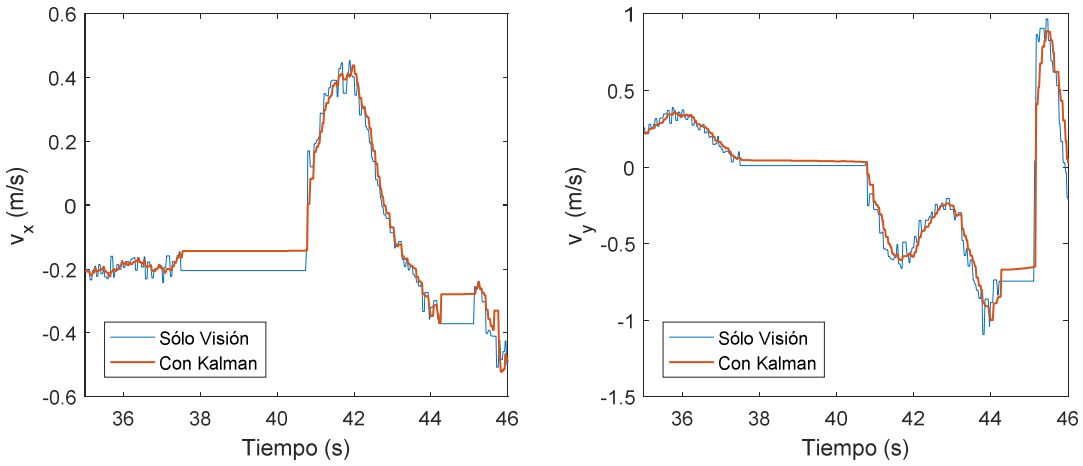


Figura 8.3: Resultado del Filtro de Kalman para ẋ e ẏ en un vuelo de prueba.

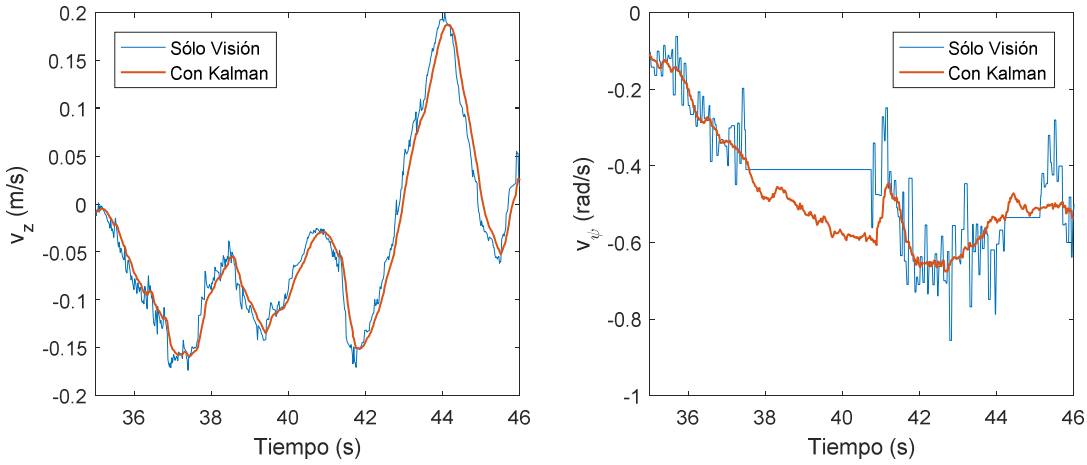


Figura 8.4: Resultado del Filtro de Kalman para ż y ψ̇ en un vuelo de prueba.

8.2. Controlador PID

Este apartado se inicia describiendo brevemente en qué consiste y las formas de diseñar un controlador PID, y continua con el diseño de los controladores PID para los modelos obtenidos en el capítulo anterior. Como el diseño se va a realizar en el marco de referencia B deberá posteriormente validarse y ajustarse si fuera necesario en el marco de referencia E.

8.2.1. Descripción del control PID

El esquema básico de control de procesos que se muestra en la Figura 8.5 se basa en la realimentación de la salida $y(t)$ de un sistema para obtener la diferencia existente con la referencia $r(t)$. Esta diferencia, o error, es utilizada como entrada al controlador para determinar la acción de control u(t) que lleve al proceso a alcanzar dicha referencia.

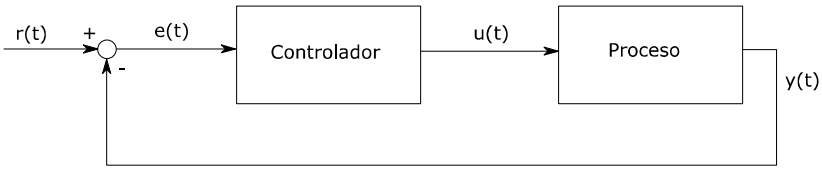


Figura 8.5: Esquema básico de control de un proceso por realimentación.

El control PID (Proporcional-Integral-Derivativo) es uno de los algoritmos de control más comunes en el control de bucles por realimentación y su implementación es muy sencilla. Se basa en la determinación de una acción de control en función del error existente entre la referencia y la salida, de la integral de dicho error y de su derivada. Su formulación en función del tiempo se muestra en la ecuación (8.19), donde $e(t) = r(t) - y(t)$.

$$u_{PID}(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt} \tag{8.19}$$

Este controlador puede formularse como una función de transferencia en el dominio $\mathcal{S}$ tal y como se muestra en la ecuación (8.20).

$$G_{PID}(s) = \frac{U(s)}{E(s)} = K_p + \frac{K_i}{s} + K_d s \tag{8.20}$$

Esta función de transferencia se conoce como "ideal" porque presenta ganancias muy elevadas a altas frecuencias. Para atenuar este comportamiento se filtra la acción derivativa por un sistema de primer orden con una constante de tiempo αT_d. Esta nueva acción actuará como verdadera derivada sólo a frecuencias bajas y su ganancia a altas frecuencias estará limitada a K_p/α. Denominando $N = 1/\alpha T_d$ se obtiene la función de transferencia del controlador PID con el filtro de la acción derivativa que se muestra en la ecuación (8.21).

$$G_{PID}(s) = K_p + \frac{K_i}{s} + \frac{K_d N s}{s + N} \tag{8.21}$$

Esta función de transferencia puede transformarse al dominio $\mathcal{Z}$ discretizándola con varios métodos: el método trapezoidal, el método de Euler hacia adelante y el método de Euler hacia atrás. La función de transferencia del controlador PID discretizado con el método de Euler hacia adelante con un tiempo de muestreo T_s queda como se muestra en la ecuación (8.22).

$$G_{PID}(z) = K_p + K_i \frac{T_s}{z - 1} + K_d \frac{N(z - 1)}{z + (NT_s - 1)} \tag{8.22}$$

En estas ecuaciones K_p es la ganancia de la parte Proporcional del controlador, K_i es la ganancia de la parte Integral y K_d es la ganancia de la parte Derivativa. Estas ganancias, junto a N, son los parámetros que deben ajustarse en el diseño del controlador.

Hay múltiples formas de realizar el diseño de un controlador PID tal y como puede consultarse, por ejemplo, en [53] y en [54]. La elección de la forma de diseñar un controlador debe basarse en el objetivo principal del control que habitualmente es el seguimiento de una referencia o el rechazo de las perturbaciones a través de unas especificaciones determinadas. Dichas especificaciones podrían ser características de la respuesta en el tiempo del sistema para entradas típicas, características de la respuesta en frecuencia o una determinada atenuación de los errores de medida.

Por ello hay dos métodos usuales para realizar el diseño de controladores PID:

- Diseño a través de características deseadas de la respuesta en el tiempo. Si no se conoce la función de transferencia del proceso se pueden usar las reglas de Ziegler-Nichols para ajustar los parámetros del controlador PID. Si se conoce la función de transferencia del proceso o si puede obtenerse un modelo aproximado, se pueden utilizar muchos enfoques tanto analíticos como gráficos para realizar el diseño como por ejemplo el método de la asignación de polos. Este método permite asignar los polos del sistema combinado con el controlador para que tenga las características deseadas.
- Diseño a través de características deseadas de la respuesta en la frecuencia. En este método se utilizan compensadores de adelanto o de atraso de fase apoyados en los diagramas de Bode o de Nyquist.

En el diseño de los controladores PID del presente proyecto se va a utilizar la herramienta Simulink de Matlab que dispone de algoritmos de ajuste automático de los parámetros para cumplir con unas especificaciones en la respuesta temporal o en frecuencia.

8.2.2. Implementación del control PID

En este proyecto se va a realizar un control desacoplado de modo que cada entrada de control del AR.Drone 2.0 sea determinada por un controlador en base a una única variable de estado del AR.Drone 2.0 y su referencia. Por tanto la entrada de control u_{pitch} será determinada en base a la coordenada x^B del cuadricóptero, la entrada de control u_{roll} será determinada en base a la coordenada y^B del cuadricóptero, la entrada de control $u_{throttle}$ será determinada en base a la coordenada z^E, mientras que la entrada de control u_{yaw} será determinada en base a la coordenada ψ^E.

El esquema de control implementado en este proyecto se puede ver en la Figura 8.6 donde los bloques del área del Modelo se corresponden con los modelos en el dominio $\mathcal{Z}$ obtenidos en el capítulo 7. El error entre las referencias y las salidas en el marco de referencia E es transformado al marco de referencia B como entrada a los controladores diseñados en el marco de referencia B. Las salidas de los modelos en respuesta a las acciones de control son las velocidades en el marco de referencia B, que son transformadas al marco de referencia E. Mediante la integración de estas velocidades se obtiene las posición, orientación y altura respecto al marco de referencia E. La referencia a seguir por el cuadricóptero, por tanto, se proporciona con respecto al marco de referencia E. Los controladores PID tienen su salida limitada a los posibles valores de las entradas de los modelos según se identificó en el capítulo 7.

El esquema de control utilizado en Simulink se adjunta junto al resto del código fuente del proyecto y se denomina *EsquemaControl.slx*.

En el diseño de los controladores PID se ha decidido aplicar la referencia directamente en el marco de referencia B para simplificar este diseño evitando así las no linealidades introducidas por las transformaciones. Tras la obtención de los controladores diseñados para seguir referencias en el marco de referencia B, se introducirán en el esquema de control de la Figura 8.6, para su simulación, validación y ajuste en caso necesario.

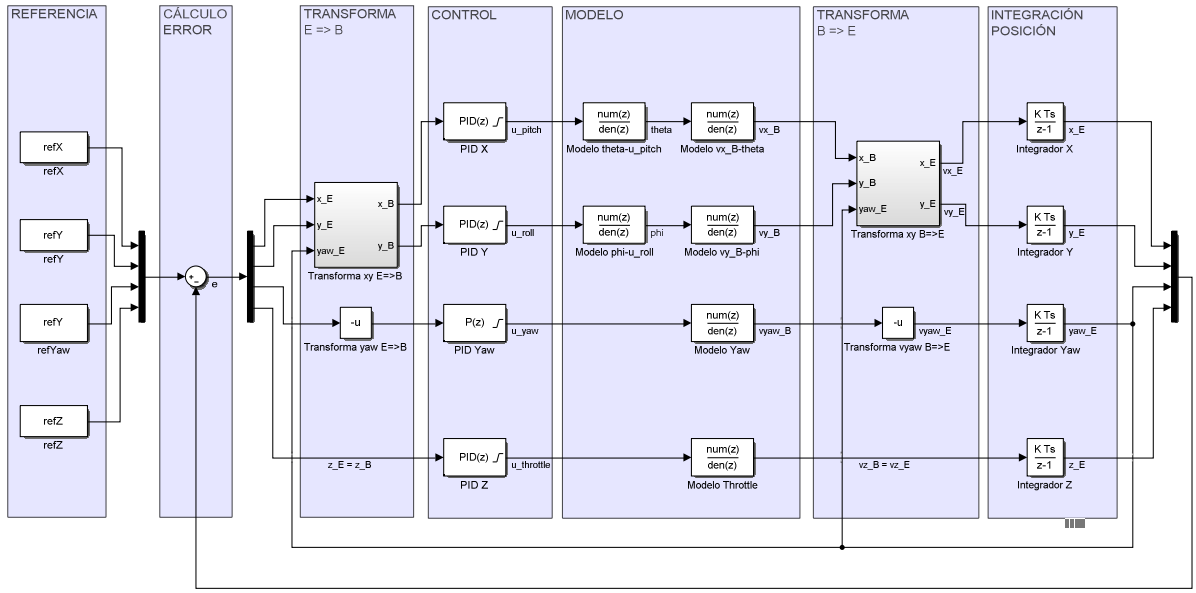


Figura 8.6: Esquema de control mediante controladores PID.

Como se ha dicho anteriormente, se van a utilizar las funcionalidades que proporciona Simulink para ajustar de forma automática los parámetros de los controladores PID para cumplir con unos requisitos en la respuesta. Sin embargo este modo automático no funciona bien cuando en el lazo de control existen no linealidades, como por ejemplo la limitación de la salida del PID. Por ello es necesario realizar los ajustes automáticos sin limitar esta salida y después hacer una simulación con la salida del PID limitada para comprobar si la respuesta es acorde a los requisitos. Si no fuera así se volvería a ajustar y nuevamente a verificar la simulación hasta que la respuesta se encuentre dentro de las especificaciones requeridas.

Los esquemas de control utilizados en Simulink para ajustar los controladores en el marco de referencia B se adjuntan con el resto del código fuente del proyecto y se llaman *PID_x_Tuning.slx*, *PID_y_Tuning.slx*, *PID_z_Tuning.slx* y *PID_yaw_Tuning.slx*.

8.2.2.1. Controlador PID para el Pitch

Se va a diseñar un controlador PID para el Pitch que permita una velocidad lineal aproximada v_x^B de 0.5 m/s, con un sobrepico menor del 10% y cuyo tiempo de establecimiento no supere los 6 segundos.

Teniendo en cuenta lo indicado anteriormente se ha obtenido un controlador PID con los parámetros que se indican en la Tabla 8.1 para realizar el seguimiento de la referencia de la componente x de la posición en el marco de referencia B.

K_p	K_i	K_d	N
-0.2	-2.70e-04	-0.2	20

Tabla 8.1: Parámetros del controlador PID para el Pitch.

La respuesta al escalón así como la acción de control de este controlador se muestran en la Figura 8.7 y poseen las características temporales que se muestran en la Tabla 8.2 y que entran dentro de los parámetros de diseño iniciales, siendo por tanto suficientes para realizar el seguimiento de la referencia de la componente x^B. En la figura indicada se puede ver que la pendiente de la respuesta al escalón es tal que cumple con el requisito de velocidad lineal de 0.5 m/s.

Tiempo de subida (s)	1.33
Tiempo de establecimiento (s)	5.40
Pico (%)	8.48
Tiempo de pico (s)	2.86

Tabla 8.2: Características de la respuesta temporal al escalón del controlador PID del Pitch.

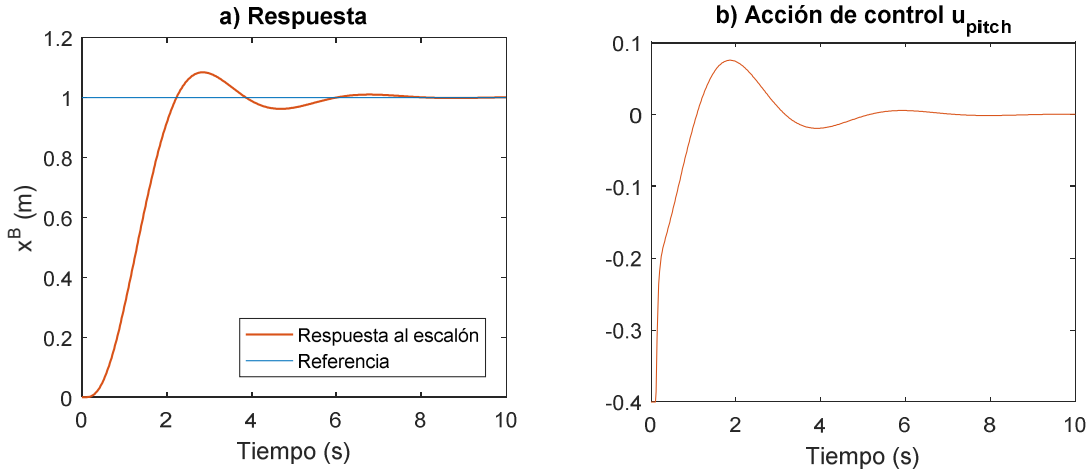


Figura 8.7: Respuesta al escalón de x^B con el controlador PID.

8.2.2.2. Controlador PID para el Roll

En el diseño del controlador PID para el Roll se van a utilizar los mismos requisitos que en el caso del Pitch, es decir, debe permitir una velocidad lineal aproximada v_y^B de 0.5 m/s, con un sobrepico menor del 10% y cuyo tiempo de establecimiento no supere los 6 segundos.

De esta forma se ha obtenido un controlador PID con los parámetros que se indican en la Tabla 8.3 para realizar el seguimiento de la referencia de la componente y de la posición en el marco de referencia B.

K_p	K_i	K_d	N
0.25	5.10e-04	0.2	12

Tabla 8.3: Parámetros del controlador PID para el Roll.

La respuesta al escalón así como la acción de control de este controlador se muestran en la Figura 8.8 y poseen las características temporales que se muestran en la Tabla 8.4 y que entran dentro de los parámetros de diseño iniciales, siendo por tanto suficientes para realizar el seguimiento de la referencia de la componente y^B. En la figura indicada se puede ver que la pendiente de la respuesta al escalón es tal que cumple con el requisito de velocidad lineal de 0.5 m/s.

Tiempo de subida (s)	1.25
Tiempo de establecimiento (s)	3.25
Pico (%)	4.49
Tiempo de pico (s)	2.68

Tabla 8.4: Características de la respuesta temporal al escalón del controlador PID del Roll.

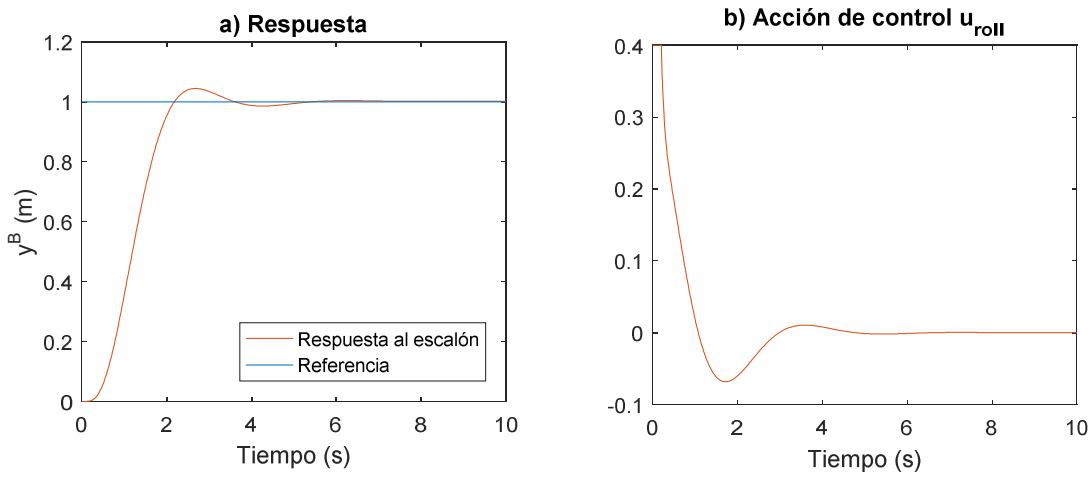


Figura 8.8: Respuesta al escalón de y^B con el controlador PID.

8.2.2.3. Controlador PID para el Throttle

Los requisitos de la respuesta temporal del modelo para diseñar el controlador PID del Throttle deberán ser tales que permitan una velocidad lineal v_z^B de 0.5 m/s aproximadamente, con un sobrepico menor del 10% y cuyo tiempo de establecimiento no supere los 6 segundos.

Así se ha obtenido un controlador PID con los parámetros que se indican en la Tabla 8.5 para realizar el seguimiento de la referencia de la altura z.

K_p	K_i	K_d	N
5	0.036	1.5	20

Tabla 8.5: Parámetros del controlador PID para el Throttle.

La respuesta al escalón así como la acción de control de este controlador se muestran en la Figura 8.9 y poseen las características temporales que se muestran en la Tabla 8.6 y que entran dentro de los parámetros de diseño iniciales, siendo por tanto suficientes para realizar el seguimiento de la referencia de la altura z. En la figura indicada se puede ver que la pendiente de la respuesta al escalón es tal que cumple con el requisito de velocidad lineal de 0.5 m/s.

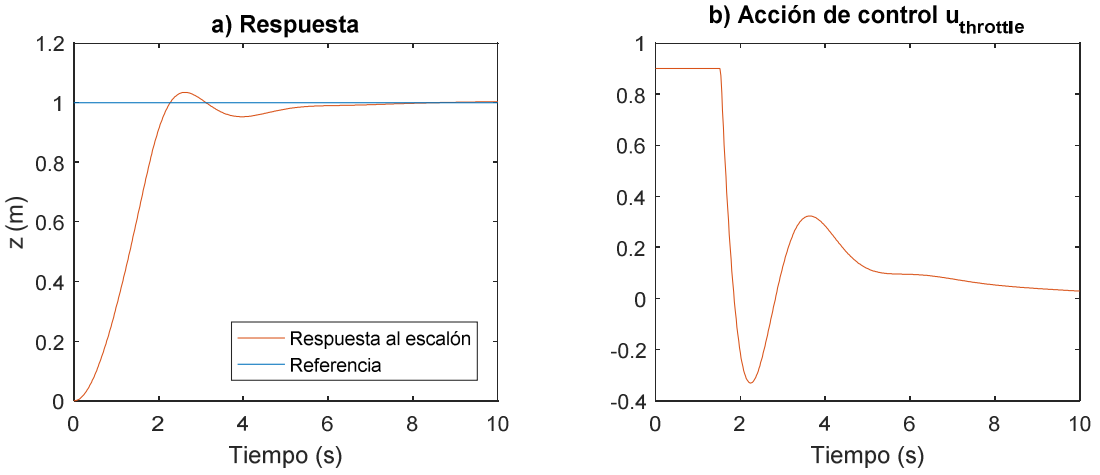


Figura 8.9: Respuesta al escalón de z con el controlador PID.

Tiempo de subida (s)	1.44
Tiempo de establecimiento (s)	5.06
Pico (%)	3.45
Tiempo de pico (s)	2.62

Tabla 8.6: Características de la respuesta temporal al escalón del controlador PID del Throttle.

8.2.2.4. Controlador PID para el Yaw

En el diseño del controlador PID para el Yaw se van a tener en cuenta los requisitos de velocidad angular v_{ψ}^{B} de 1 rad/s, con un sobrepico menor del 10% y cuyo tiempo de establecimiento no supere los 6 segundos.

La dinámica del Yaw es tal que ha permitido obtener un controlador P, sin acción Integral ni Derivativa, con una ganancia $K_p = 2$, suficiente para cumplir con los requisitos y realizar el seguimiento de la referencia de ángulo de orientación ψ en el marco de referencia B.

La respuesta al escalón así como la acción de control de este controlador se muestran en la Figura 8.10 y poseen las características temporales que se muestran en la Tabla 8.7 y que son suficientes para realizar el seguimiento de la referencia del ángulo de orientación ψ^B. En la figura indicada se puede ver que la pendiente de la respuesta al escalón es tal que cumple con el requisito de velocidad angular de 1 rad/s.

Tiempo de subida (s)	0.63
Tiempo de establecimiento (s)	0.97
Pico (%)	0.78
Tiempo de pico (s)	1.26

Tabla 8.7: Características de la respuesta temporal al escalón del controlador PID del Yaw.

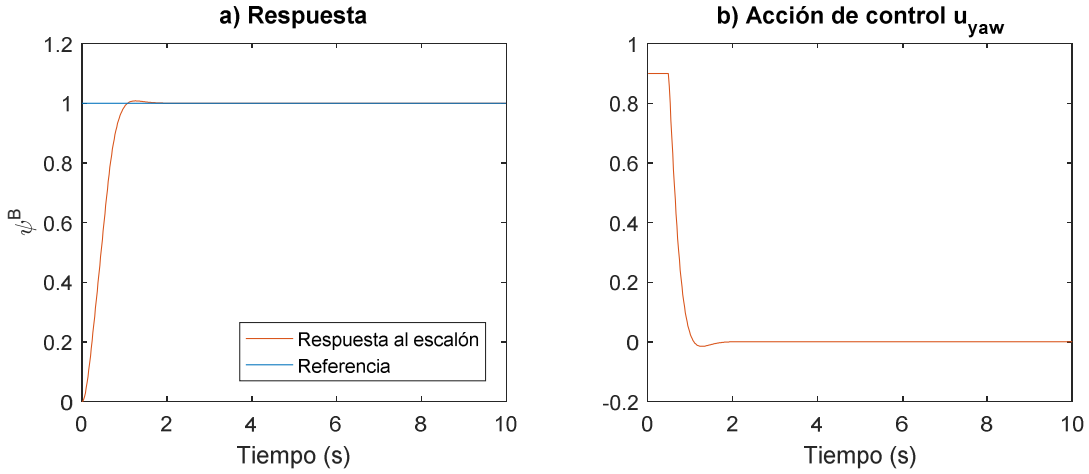


Figura 8.10: Respuesta al escalón de ψ^B con el controlador PID.

8.2.3. Validación de los controladores en el marco de referencia E

Los controladores obtenidos en los apartados anteriores para seguir referencias del marco de referencia B deben ser validados con el esquema de control de la Figura 8.6 en que las referencias se proporcionan, de forma conjunta, respecto al marco de referencia E y son comparadas con las salidas del modelo transformadas también a dicho marco E.

Haciendo una simulación aplicando un escalón unitario a las entradas u_{pitch}, u_{roll} y $u_{throttle}$ se obtienen los resultados que se muestran en la Figura 8.11 donde se puede ver que el comportamiento es similar al obtenido en el diseño de los controladores en el marco B.

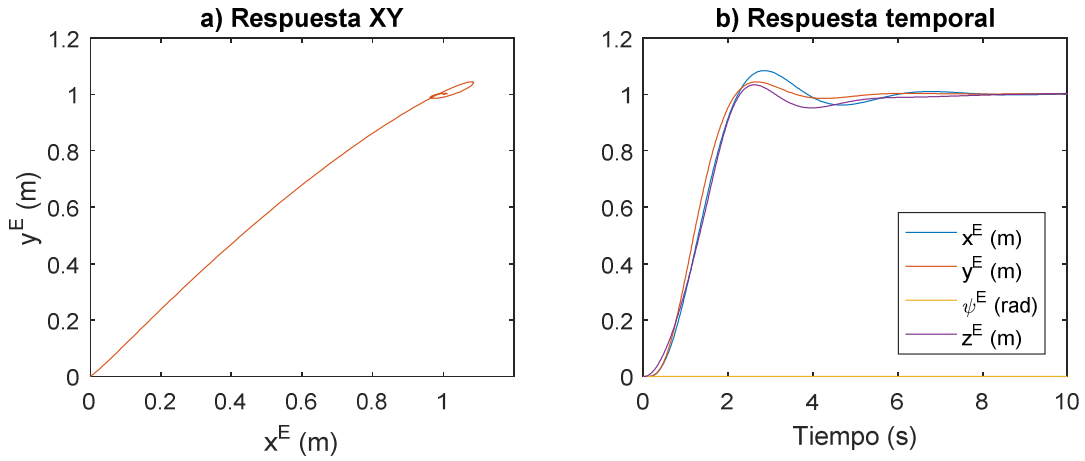


Figura 8.11: Respuesta del controlador diseñado a referencias en el marco E.

Sin embargo cuando se aplica a la vez un escalón unitario a la entrada u_{yaw} se obtienen los resultados que se muestran en la Figura 8.12 donde se puede ver que el comportamiento no es similar al obtenido en el diseño de los controladores en el marco B ya que la respuesta de y^E presenta un pico mucho mayor que el esperado mientras la respuesta de x^E está sobreamortiguada. Si el escalón unitario de u_{yaw} fuera negativo este mismo comportamiento se daría al revés, es decir, y^E presentaría la respuesta sobreamortiguada mientras que x^E presentaría un pico mucho mayor que el esperado.

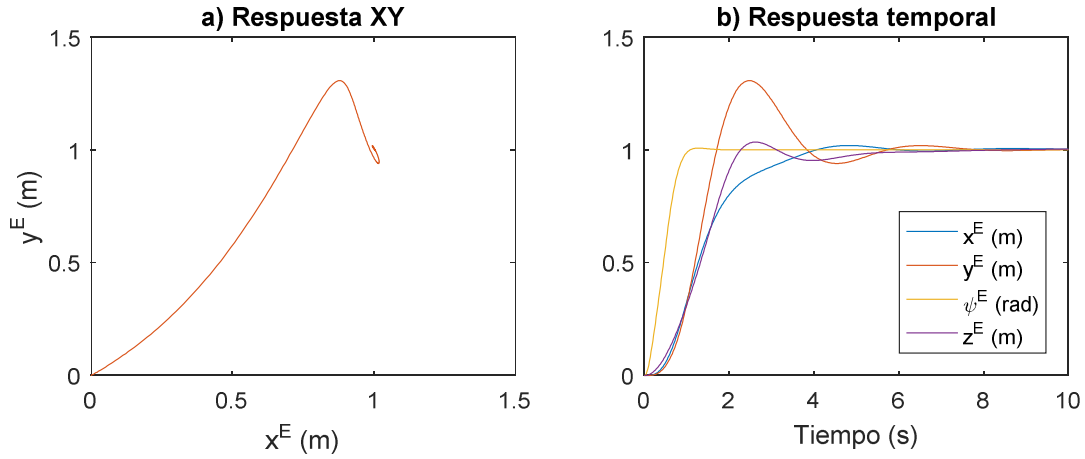


Figura 8.12: Respuesta del controlador diseñado a referencias en el marco E, incluido el Yaw.

Para conseguir que estos comportamientos no sean tan acusados se han ajustado las ganancias de los controladores del Pitch y del Roll, teniendo en cuenta no sólo las respuestas ante escalones sino también las respuestas ante los diferentes tipos de experimentos que se muestran en el capítulo 9.

Los parámetros ajustados del controlador del Pitch quedan como se puede ver en la Tabla 8.8, mientras que los del controlador del Roll pueden verse en la Tabla 8.9.

K_p	K_i	K_d	N
-0.15	-2.70e-04	-0.2	20

Tabla 8.8: Parámetros ajustados del controlador PID para el Pitch.

K_p	K_i	K_d	N
0.2	5.10e-04	0.2	12

Tabla 8.9: Parámetros ajustados del controlador PID para el Roll.

Con estos parámetros se consigue que el comportamiento del sobrepico no sea tan acusado con lo que el cuadricóptero no se desvía tanto de la referencia tal y como puede verse en la Figura 8.13.

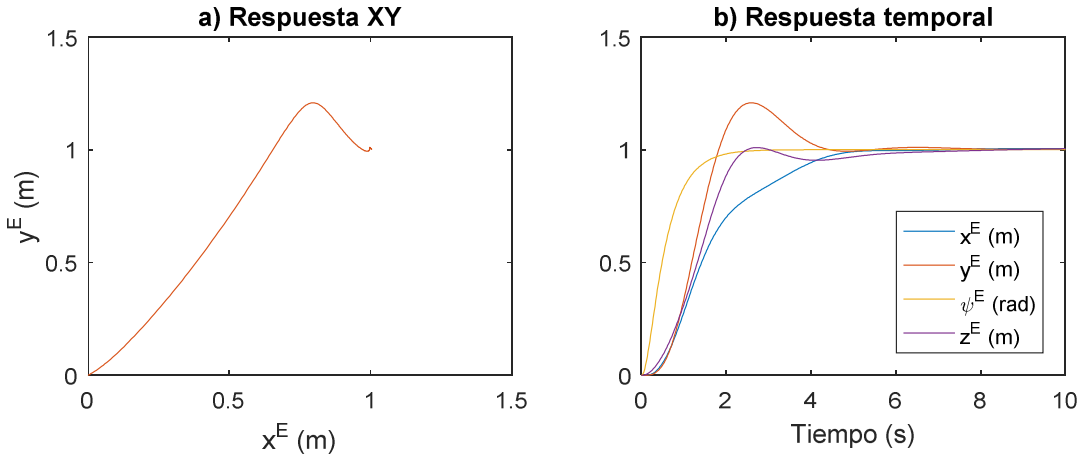


Figura 8.13: Respuesta del controlador ajustado a referencias en el marco E.

El sobrepico existente en el Roll o en el Pitch es de aproximadamente un 25%, valor que no cumple con los requisitos de diseño iniciales del 10%, por lo que se reajusta la ganancia del controlador del Yaw a $K_p = 1$ y se limita la acción de control a un valor entre $[-0.5,0.5]$ para disminuir el efecto que el giro del Yaw, a través de las no linealidades de las transformaciones entre los marcos de referencia, introduce en la respuesta del Roll y el Pitch. Con este nuevo reajuste se consiguen las respuestas mostradas en la Figura 8.14.

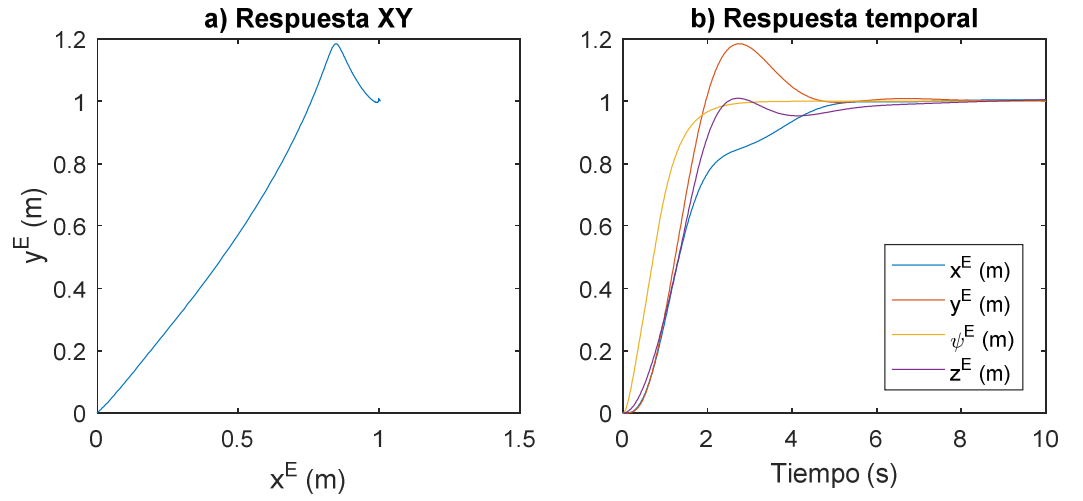


Figura 8.14: Respuesta del controlador reajustado a referencias en el marco E.

La disminución en la velocidad de giro v_{ψ}^{E} del Yaw a 0.8 rad/s consigue hacer disminuir el sobrepico del Roll o Pitch al 18%. Aunque ambos valores se encuentran fuera de los límites establecidos en los requisitos iniciales del diseño, se ha llegado a un término medio aceptable, siendo este controlador reajustado un buen candidato a ser probado experimentalmente.

En la Tabla 8.10 se recopilan los valores definitivos de las ganancias y las limitaciones en la acción de los controladores PID diseñados.

	K_p	K_i	K_d	N	*Acción de control*
Pitch	-0.15	-2.70e-04	-0.2	20	$[-0.4,0.4]$
Roll	0.2	5.10e-04	0.2	12	$[-0.4,0.4]$
Throttle	5	0.036	1.5	20	$[-0.9,0.9]$
Yaw	1	0	0	0	$[-0.5,0.5]$

Tabla 8.10: Resumen de los controladores PID diseñados.

9. EXPERIMENTOS Y RESULTADOS

Tras la obtención de un modelo del sistema real y de un controlador que resulte satisfactorio para controlar de forma autónoma dicho modelo, es necesario validarlo de forma experimental por lo que han sido diseñados diferentes experimentos con dicho objetivo.

Tras realizar diversas pruebas de estos experimentos se ha tenido que reajustar el controlador PID del Throttle bajando su ganancia K_p a 1, eliminado la acción derivativa del PID y limitando la acción del controlador a $[-0.5,0.5]$, consiguiendo evitar altas velocidades verticales y grandes oscilaciones. Con esto, los valores de las ganancias y las limitaciones en la acción de los controladores PID utilizados en los experimentos reales se muestran en la Tabla 9.1.

	K_p	K_i	K_d	N	*Acción de control*
Pitch	-0.15	-2.70e-04	-0.2	20	$[-0.4,0.4]$
Roll	0.2	5.10e-04	0.2	12	$[-0.4,0.4]$
Throttle	1	0.036	0	0	$[-0.5,0.5]$
Yaw	1	0	0	0	$[-0.5,0.5]$

Tabla 9.1: Valores de los controladores PID utilizados en los experimentos reales.

9.1. Estabilización en un punto

En este apartado se van a ver los resultados de dos experimentos de estabilización del cuadricóptero en un punto. El primero de ellos es un experimento básico sin ninguna perturbación externa, mientras que en el segundo se le dan varios empujones para ver cómo reacciona y si intenta volver al punto de consigna.

9.1.1. Experimento de estabilización sin perturbaciones externas

El experimento más sencillo a realizar es la estabilización del cuadricóptero en un punto con una orientación determinada. En este experimento la referencia durante todo el tiempo de vuelo es siempre fija. En la Figura 9.1 se muestra la gráfica XY del experimento de estabilización en el punto (0,0,0.8) metros y con una orientación de -1.57 radianes. El punto inicial de este experimento viene determinado por el momento del vuelo en el que se cambia al modo automático desde el software de control. En este caso el punto inicial desde el que se inicia el modo automático ha sido (-0.99,-0.02,0.68) metros con una orientación de -1.53 radianes. En esta figura puede verse la simulación realizada para este experimento con el mismo punto inicial pero sin tener en cuenta las velocidades que tuviera el cuadricóptero en dicho momento.

En la Figura 9.2 se muestran las gráficas de respuesta temporal donde se aprecia que la estabilización en el punto es bastante buena. Teniendo en cuenta los últimos 20 segundos del experimento se calcula el error máximo entre la respuesta real y la referencia, así como la raíz de su error cuadrático medio (RMSE), que se muestran en la Tabla 9.2.

	x^E *(m)*	y^E *(m)*	ψ^E *(rad)*	z^E *(m)*
Error máximo	0.0773	0.0704	0.0776	0.0290
RMSE	0.0410	0.0336	0.0612	0.0092

Tabla 9.2: Medidas de error del experimento de estabilización en el punto (0,0,0.8) metros con orientación -1.57 rad.

En dicha tabla puede verse que el error máximo en las componentes de la posición x e y es menor de 8 centímetros, lo cual es un valor muy bueno para este tipo de cuadricóptero. En cuanto a la orientación, su error máximo es menor de 5° que resulta bastante aceptable. Mientras que el error máximo en la altura es también muy bueno de 3 centímetros.

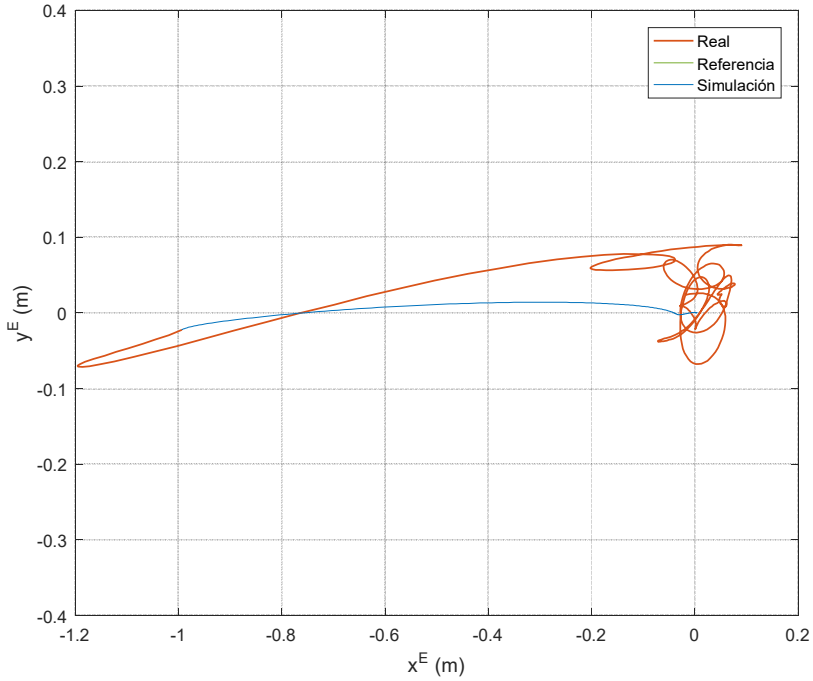


Figura 9.1: Gráfica XY del experimento de estabilización en el punto (0,0,0.8) metros con orientación -1.57 rad.

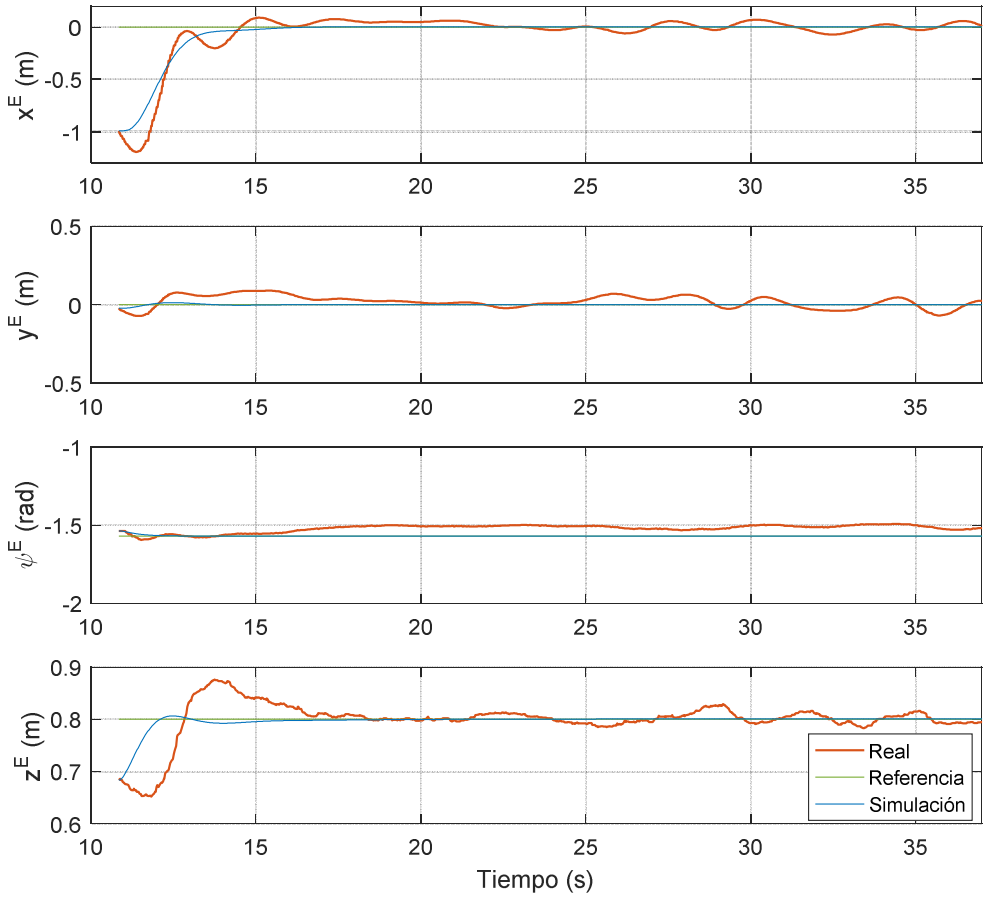


Figura 9.2: Respuesta temporal del experimento de estabilización en el punto (0,0,0.8) metros con orientación -1.57 rad.

9.1.2. Experimento de estabilización con perturbaciones externas

El siguiente experimento realizado se basa en el mismo punto de consigna pero cuando se encuentra estable en dicho punto se perturba de forma manual, volviendo al punto de estabilización y realizando de nuevo otra perturbación. Con este experimento se demuestra su reacción a las perturbaciones exteriores. La posición inicial al entrar en modo automático ha sido (-0.73,-0.37,0.66) metros con una orientación de -1.62 radianes y el resultado del experimento puede verse en la Figura 9.3 y en la Figura 9.4.

Como puede apreciarse en estas figuras las perturbaciones externas han sido bastante grandes y, a pesar de ello, ha vuelto al punto de estabilización en un breve tiempo de corrección.

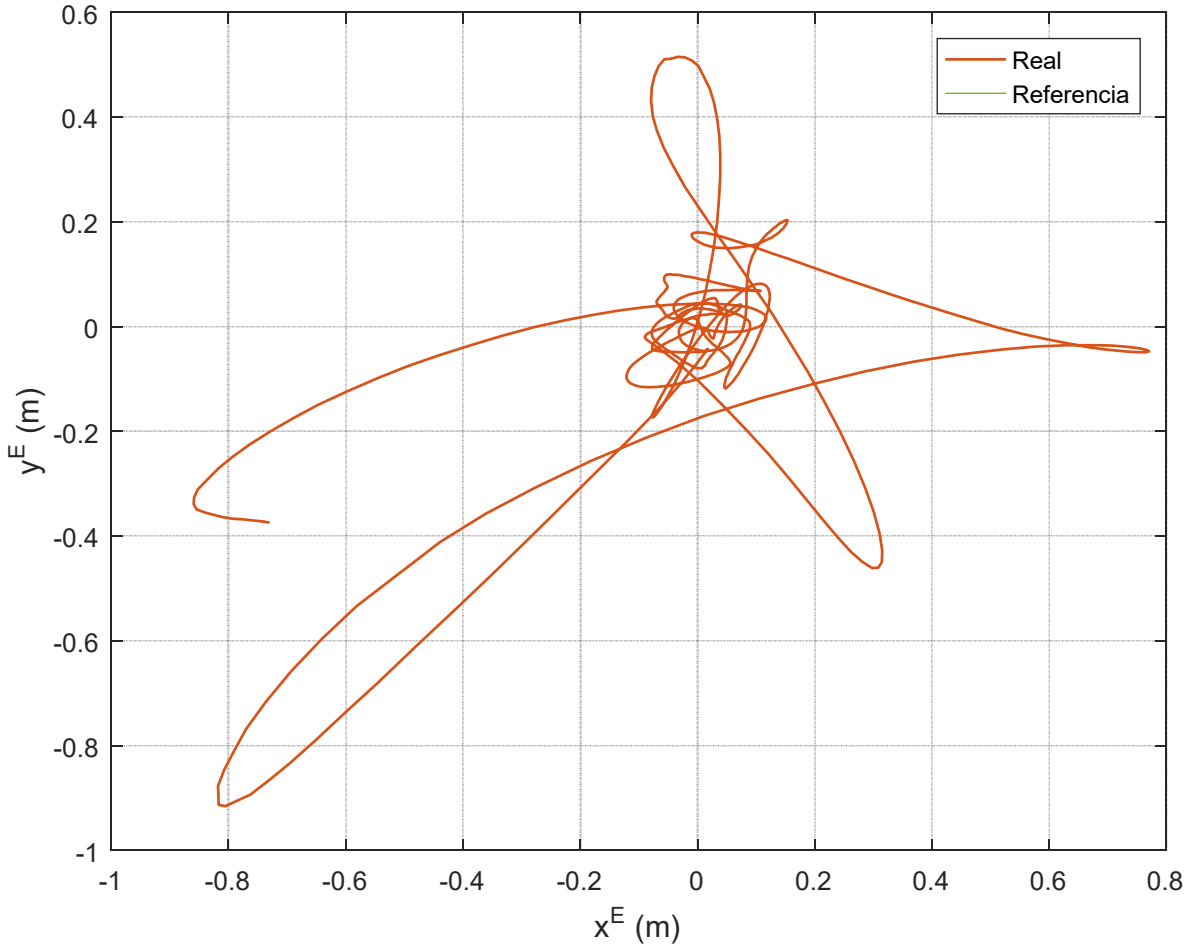


Figura 9.3: Gráfica XY del experimento de estabilización con perturbaciones externas.

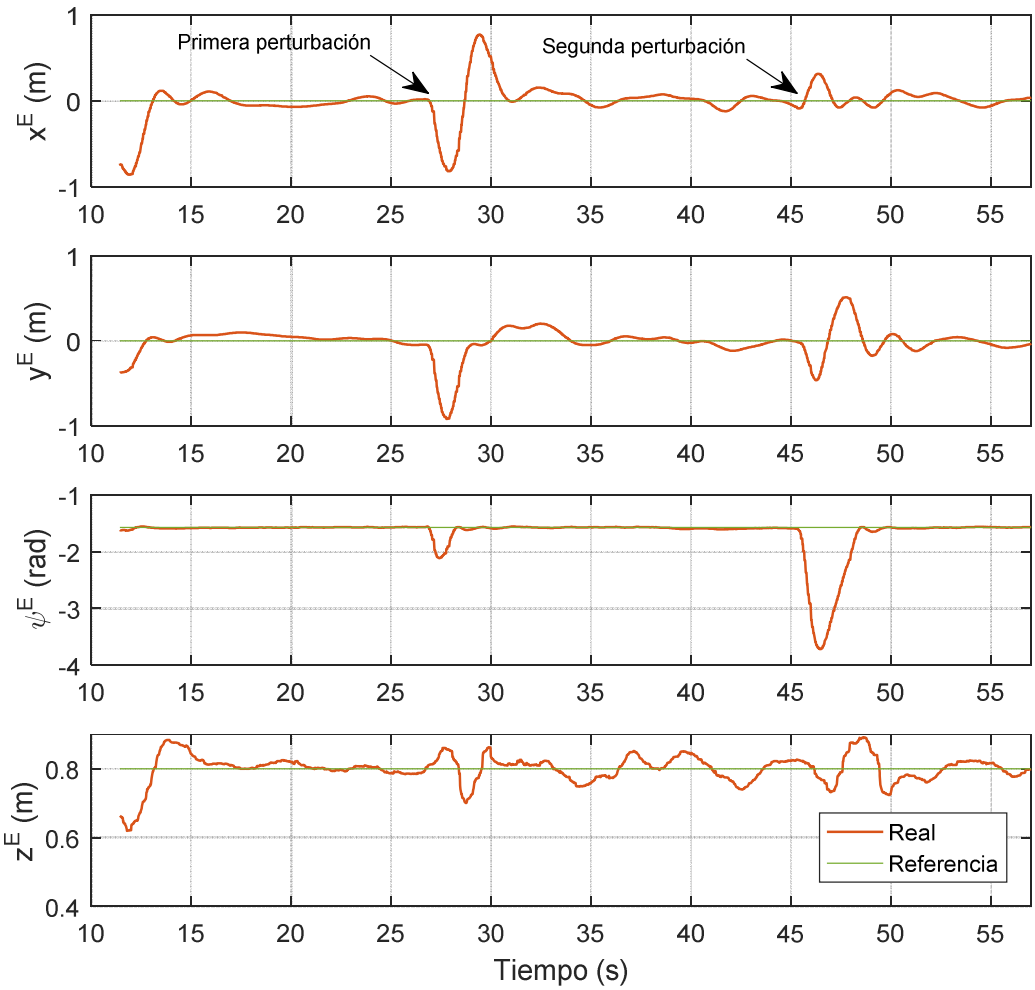


Figura 9.4: Respuesta temporal del experimento de estabilización con perturbaciones externas.

9.2. Navegación a través de Waypoints

Este tipo de experimento trata de conseguir que el cuadricóptero pase por diferentes puntos preestablecidos con anterioridad. La forma de llegar a cada punto es indiferente y cuando el cuadricóptero llega a dicho punto de un punto a otro y mantenerse en él durante un tiempo establecido antes de ir hacia el siguiente. Cada punto está definido por su posición en el espacio y su orientación.

En este experimento se ha definido un tiempo de permanencia en cada punto de 10 segundos desde el momento en que el software considera que ha llegado a él. Esta consideración se basa en que el cuadricóptero debe encontrarse dentro del radio de 10 centímetros en el plano XY, no sobrepasar los 3° de error en la orientación y no sobrepasar los 5 centímetros de error en altura en el mismo instante de tiempo. Hay que tener en cuenta que desde el primer instante en que se determina que ha llegado al punto se empieza a realizar la temporización y no se detiene aunque el cuadricóptero se haya salido de alguna de las tolerancias indicadas antes.

Cuando el cuadricóptero termina el tiempo de permanencia del último punto definido, vuelve al primero y así sucesivamente.

9.2.1. Experimento sin rotación ψ

En este primer experimento de navegación a través de Waypoints se ha considerado que la orientación sea la misma en todos los puntos. Con esto, los puntos por los que debe pasar el cuadricóptero son los que se indican en la Tabla 9.3.

	x^E (m)	y^E (m)	ψ^E (rad)	z^E (m)
Punto 1	0	0	-1.57	0.8
Punto 2	-0.5	-0.4	-1.57	1
Punto 3	-0.5	0.4	-1.57	0.7
Punto 4	0.5	0.4	-1.57	1.1
Punto 5	0.5	-0.4	-1.57	0.95

Tabla 9.3: Puntos definidos del primer experimento de navegación a través de Waypoints.

La posición inicial al entrar en modo automático ha sido (-1.22,-0.24,0.63) metros con una orientación de -1.54 radianes y el resultado del experimento puede verse en la Figura 9.5 y en la Figura 9.6.

En estas figuras se aprecia que en la fase de permanencia en cada punto se tiene un error algo mayor que en los experimentos de estabilización y esto puede deberse al aire generado por las hélices del cuadricóptero que, al estar en movimiento, choca contra las paredes y otros objetos cercanos y rebota hacia él haciendo que el tiempo de estabilización sea necesariamente mayor. A pesar de ello, el error con respecto al punto de permanencia es aceptable y la transición entre puntos la realiza de una manera bastante recta y se asemeja con suficiencia a la simulación.

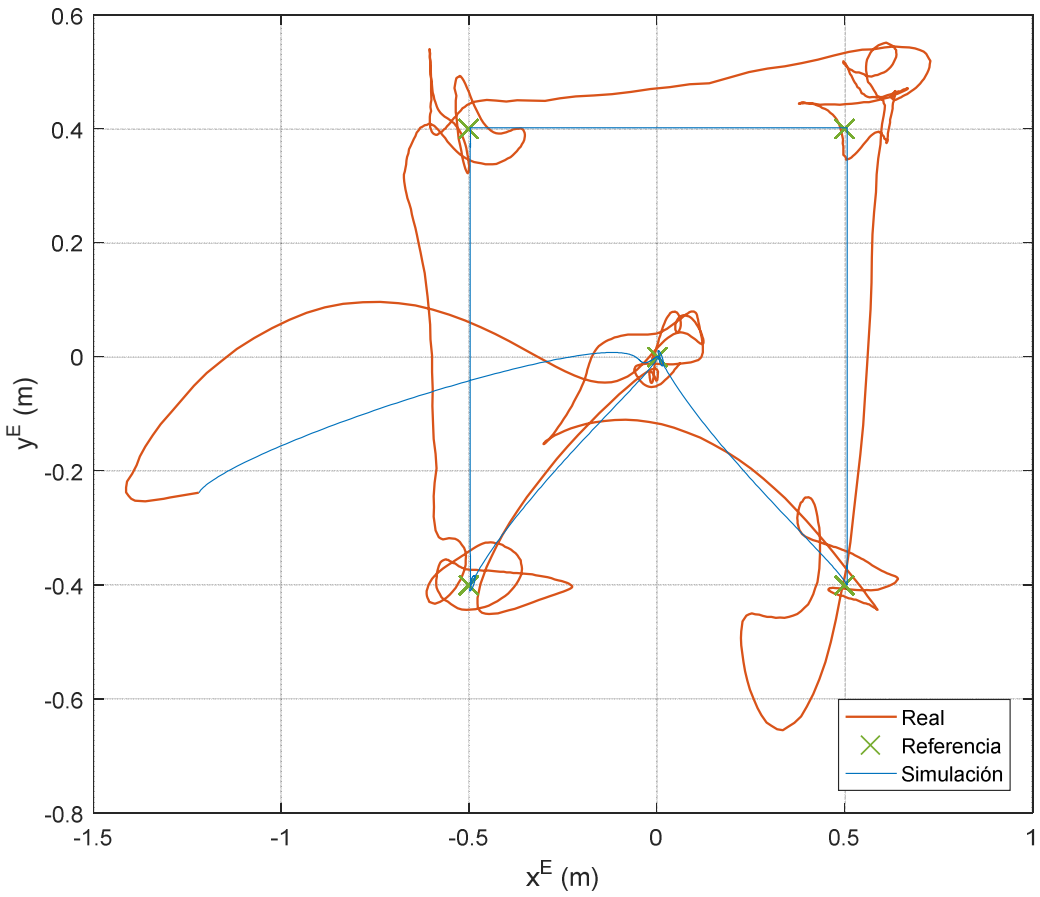


Figura 9.5: Gráfica XY del primer experimento de navegación a través de Waypoints.

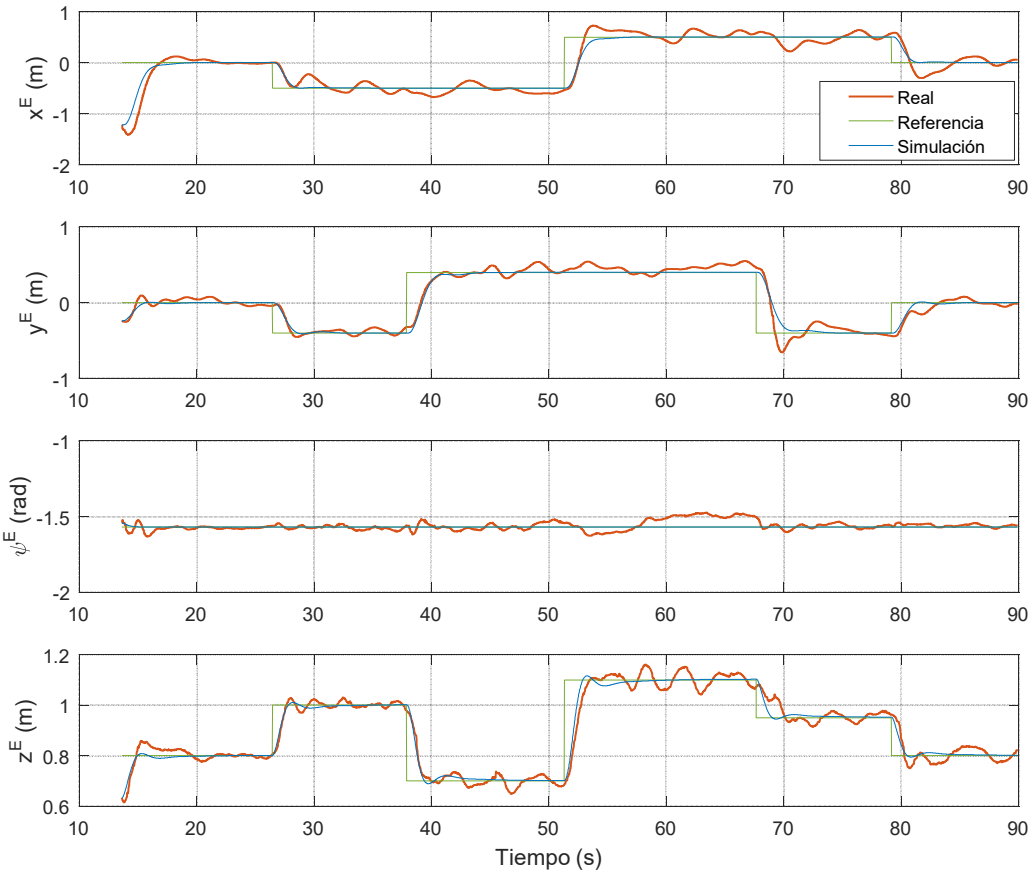


Figura 9.6: Respuesta temporal del primer experimento de navegación a través de Waypoints.

9.2.2. Experimento con rotación ψ

A continuación se realiza un experimento con otros puntos de navegación en los que la orientación es diferente de uno de ellos al siguiente. Los puntos se indican en la Tabla 9.4.

	x^E (m)	y^E (m)	ψ^E (rad)	z^E (m)
Punto 1	0	0	-1.57	0.8
Punto 2	-0.5	0	0	1
Punto 3	0	0.4	-1.57	0.7
Punto 4	0.5	0	-3.14	1.1
Punto 5	0	-0.4	-2.355	0.95

Tabla 9.4: Puntos definidos del segundo experimento de navegación a través de Waypoints.

La posición inicial al entrar en modo automático ha sido (-1.1,0.15,0.62) metros con una orientación de -1.49 radianes y el resultado del experimento puede verse en la Figura 9.7 y en la Figura 9.8.

En estas figuras puede apreciarse la existencia de grandes errores en la transición desde cada punto de navegación hasta el siguiente así como la tardanza en estabilizarse en ellos. Este comportamiento se debe a una dinámica del cuadricóptero que no ha sido considerada en el modelo y que no es más que la variación del ángulo de inclinación ϕ del Roll, y por tanto una variación en la velocidad lineal v_y^B, cuando se realiza un rotación del Yaw. El Roll, por tanto, tiende a inclinarse hacia el sentido de giro del Yaw teniendo en cuenta que conforme mayor es la velocidad de rotación mayor es el efecto en dicha inclinación.

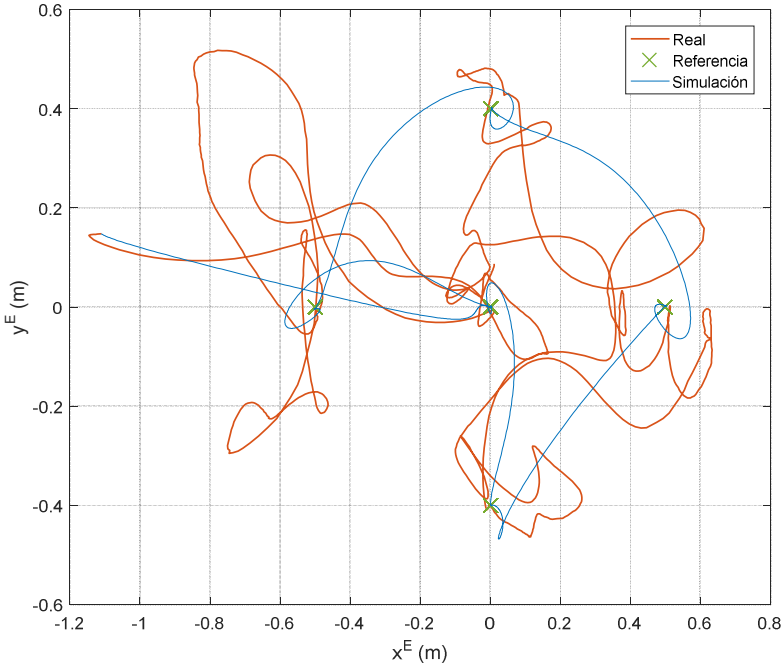


Figura 9.7: Gráfica XY del segundo experimento de navegación a través de Waypoints.

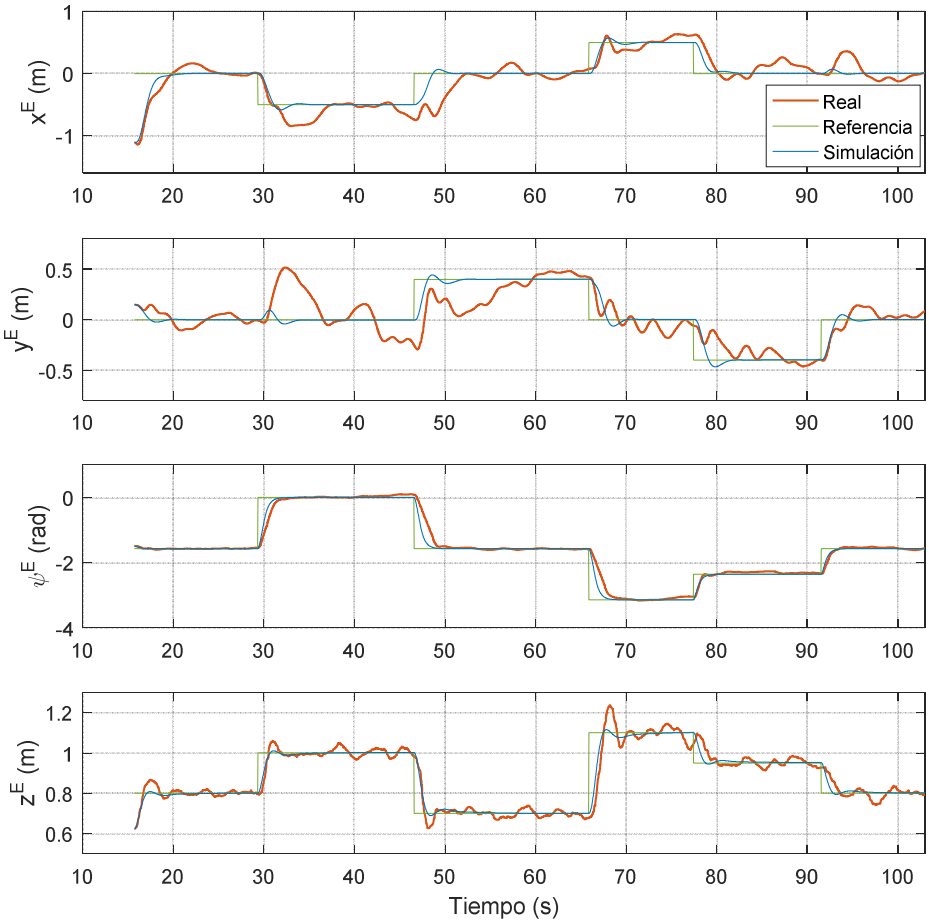


Figura 9.8: Respuesta temporal del segundo experimento de navegación a través de Waypoints.

De esta forma y dada la rápida dinámica de rotación del Yaw, el cuadricóptero se desvía demasiado de la trayectoria esperada del modelo y el controlador no puede manejar bien este comportamiento no modelado, pudiendo únicamente corregir la posición una vez el cuadricóptero ha llegado a la orientación definida por el punto de navegación.

9.3. Seguimiento de trayectorias (tracking)

En los experimentos anteriores se ha visto que el cuadricóptero se trasladaba de un punto a otro sin importar cómo realizaba dicha transición. En este apartado se van a ver los resultados de experimentos en los que el cuadricóptero debe ir de un punto a otro siguiendo una trayectoria definida en el tiempo, que habitualmente se conoce como tracking. Los primeros experimentos de este apartado están dedicados a trayectorias rectas, definidas en función del tiempo, que unen varios Waypoints, mientras que los siguientes están dedicados a trayectorias paramétricas definidas en función del tiempo.

9.3.1. Seguimiento de trayectorias rectas a través de Waypoints

La navegación definida en el apartado 9.2 puede ampliarse para que la transición del cuadricóptero de un punto a otro se realice siguiendo la línea recta que los une. Así se define el tiempo que debe tardar el cuadricóptero en realizar dicha transición de un punto a otro. Este tiempo comienza a contar cuando finaliza el tiempo de permanencia en un punto y durante dicha transición las ecuaciones de las posiciones y la orientación quedan como se indica en el conjunto de ecuaciones (9.1).

$$\begin{cases} x = x_n + (x_{n+1} - x_n)\left(\dfrac{t}{t_{trans}}\right) & , \quad 0 \le t < t_{trans} \\ x = x_{n+1} & , \quad t \ge t_{trans} \end{cases}$$
$$\begin{cases} y = y_n + (y_{n+1} - y_n)\left(\dfrac{t}{t_{trans}}\right) & , \quad 0 \le t < t_{trans} \\ y = y_{n+1} & , \quad t \ge t_{trans} \end{cases}$$
$$\begin{cases} z = z_n + (z_{n+1} - z_n)\left(\dfrac{t}{t_{trans}}\right) & , \quad 0 \le t < t_{trans} \\ z = z_{n+1} & , \quad t \ge t_{trans} \end{cases} \qquad (9.1)$$
$$\begin{cases} \psi = \psi_n + (\psi_{n+1} - \psi_n)\left(\dfrac{t}{t_{trans}}\right) & , \quad 0 \le t < t_{trans} \\ \psi = \psi_{n+1} & , \quad t \ge t_{trans} \end{cases}$$

En dicho conjunto de ecuaciones t_{trans} es el tiempo, en segundos, que debe tardar el cuadricóptero en hacer la transición del punto n al punto $n+1$, y t es el tiempo que ha transcurrido desde el comienzo de la transición al nuevo punto.

9.3.1.1. Experimento sin rotación ψ

En este primer experimento de navegación a través de Waypoints con seguimiento de las líneas rectas que los unen se ha considerado que la orientación sea la misma en todos los puntos y que de un punto a otro se tarden 2 segundos. Además se ha rebajado el tiempo de permanencia en cada punto a 5 segundos. Con esto, los puntos por los que debe pasar el cuadricóptero son los que se indican en la Tabla 9.5.

	x^E (m)	y^E (m)	ψ^E (rad)	z^E (m)	t_{trans} (s)
Punto 1	0	0	-1.57	0.8	2
Punto 2	-0.5	0	-1.57	1	2
Punto 3	0	0.4	-1.57	0.7	2
Punto 4	0.5	0	-1.57	1.1	2
Punto 5	0	-0.4	-1.57	0.95	2

Tabla 9.5: Puntos definidos del primer experimento de navegación a través de Waypoints con seguimiento de rectas.

La posición inicial al entrar en modo automático ha sido (-0.23,-0.03,0.67) metros con una orientación de -1.54 radianes y el resultado del experimento puede verse en la Figura 9.9 y en la Figura 9.10.

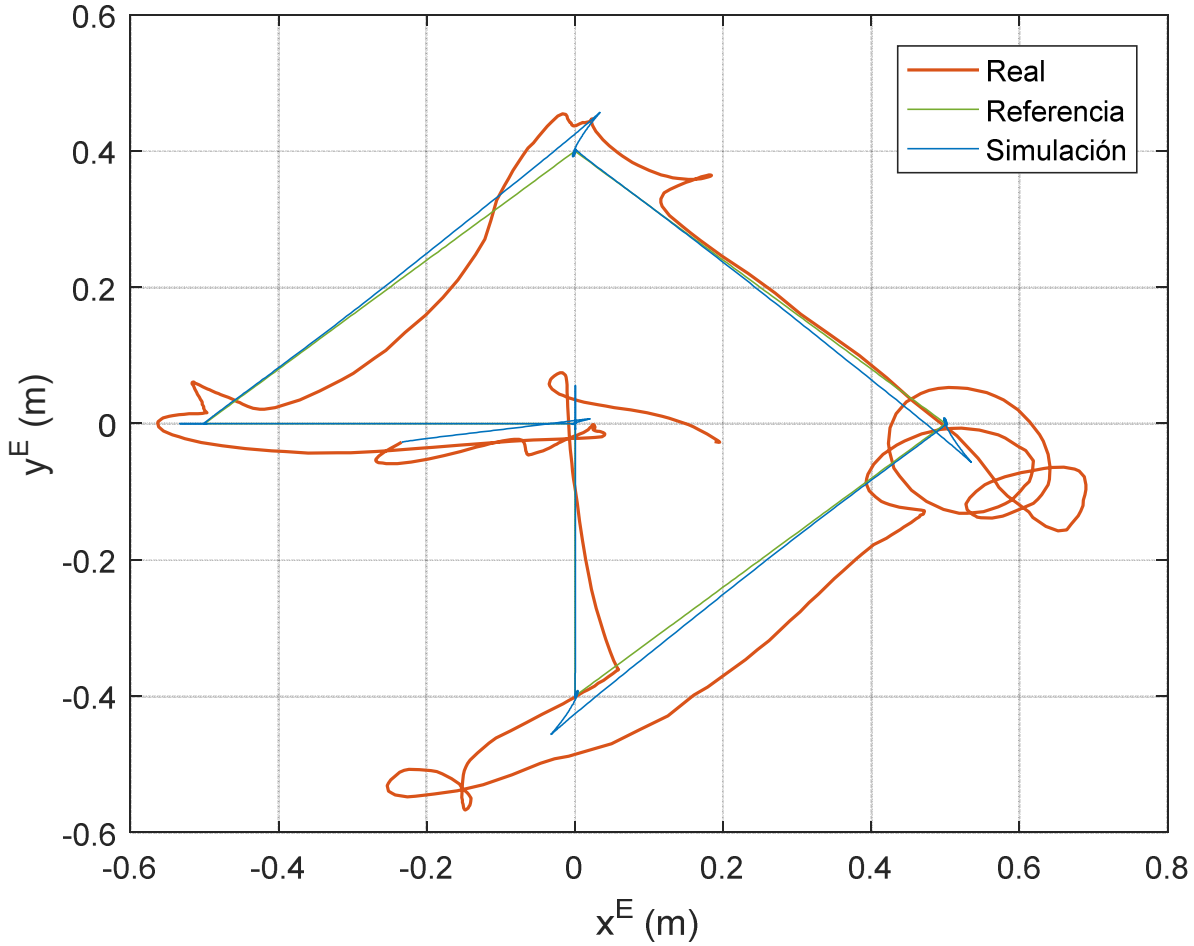


Figura 9.9: Gráfica XY del primer experimento de navegación a través de Waypoints con seguimiento de rectas.

En estas figuras se aprecia que la fase de transición de un punto a otro es, en general, mucho más suave y ajustada que en el experimento donde no se realiza el seguimiento de las líneas rectas que los unen. Teniendo en cuenta el tiempo desde que el cuadricóptero llega al primer punto hasta el final del experimento, se calcula el error máximo entre la respuesta real y la referencia, así como la raíz de su error cuadrático medio (RMSE), que se muestran en la Tabla 9.6.

	x^E (m)	y^E (m)	ψ^E (rad)	z^E (m)
Error máximo	0.2536	0.1670	0.0826	0.2398
RMSE	0.0964	0.0741	0.0268	0.0574

Tabla 9.6: Medidas de error del primer experimento de navegación a través de Waypoints con seguimiento de rectas.

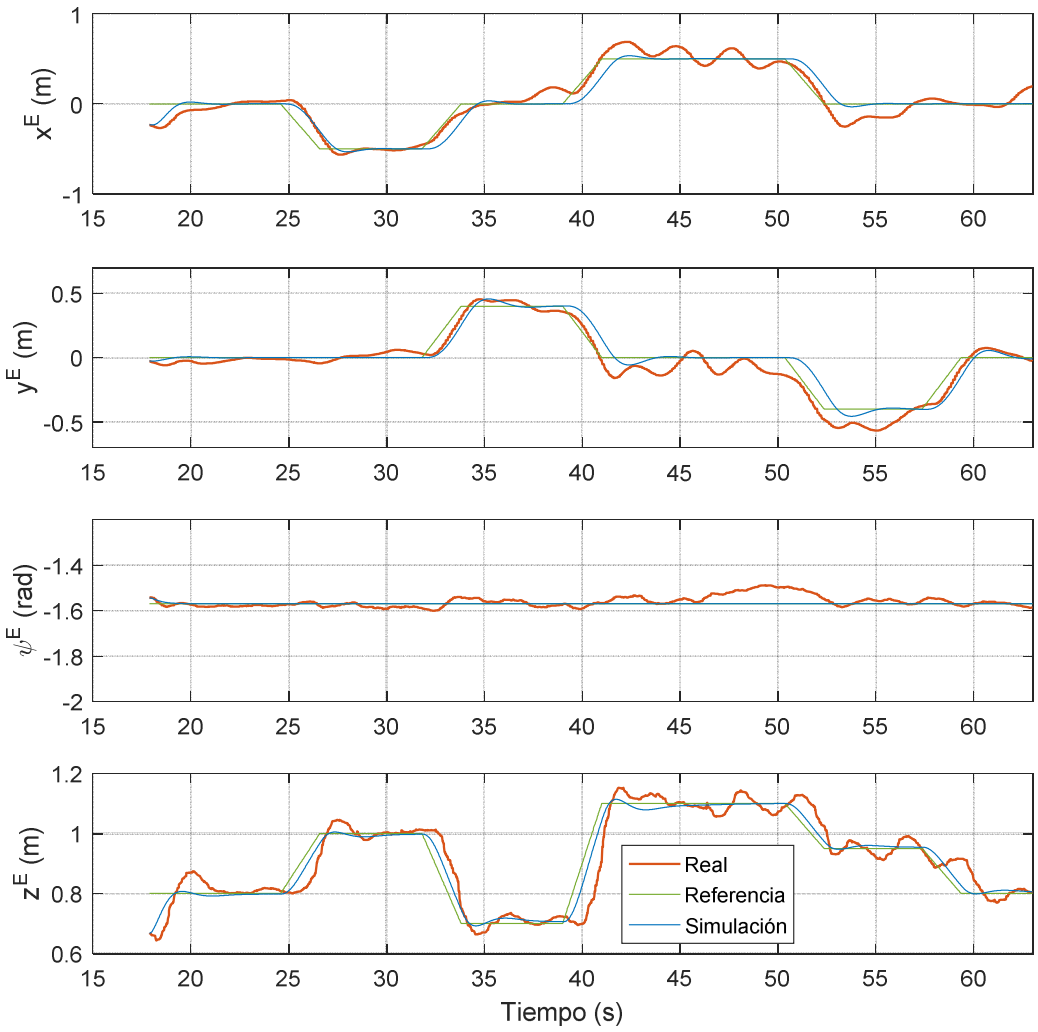


Figura 9.10: Respuesta temporal del primer experimento de navegación a través de Waypoints con seguimiento de rectas.

9.3.1.2. Experimento con rotación ψ

A continuación se realiza un segundo experimento de navegación a través de Waypoints con seguimiento de las líneas rectas que unen los puntos pero con diferentes orientaciones de un punto al siguiente. El tiempo de permanencia en cada punto se mantiene en 5 segundos pero se ha incrementado el tiempo de transición de un punto al siguiente a 4 segundos. Con esto, el conjunto de puntos de este experimento se muestra en la Tabla 9.7.

	x^E (m)	y^E (m)	ψ^E (rad)	z^E (m)	t_{trans} (s)
Punto 1	0.4	0.3	-1.57	0.7	4
Punto 2	0.4	-0.3	-3.14	1	4
Punto 3	-0.4	-0.3	1.57	1.1	4
Punto 4	-0.4	0.3	0	0.95	4

Tabla 9.7: Puntos definidos del segundo experimento de navegación a través de Waypoints con seguimiento de rectas.

La posición inicial al entrar en modo automático ha sido (-0.15,0.24,0.69) metros con una orientación de -1.60 radianes y el resultado del experimento puede verse en la Figura 9.11 y en la Figura 9.12.

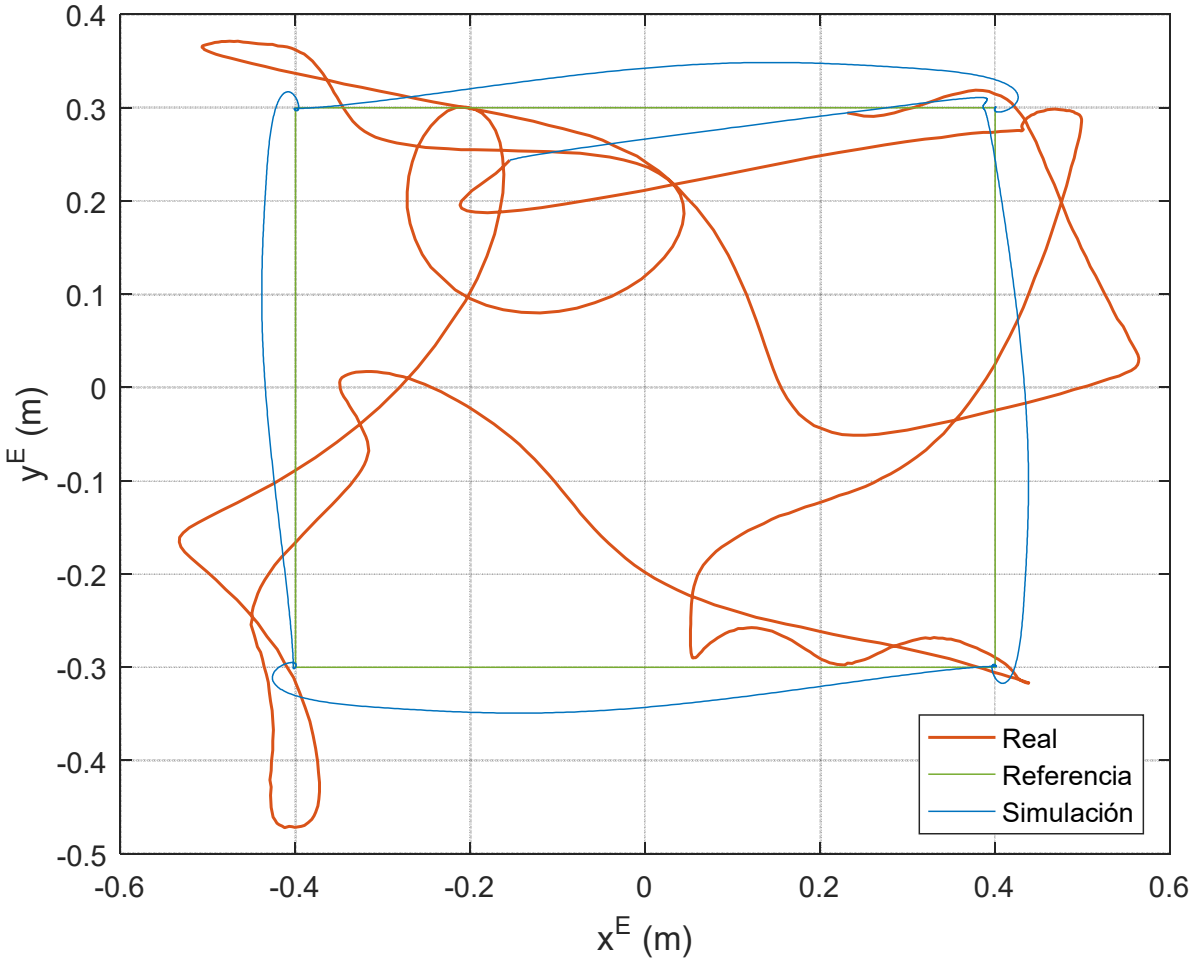


Figura 9.11: Gráfica XY del segundo experimento de navegación a través de Waypoints con seguimiento de rectas.

En estas figuras se aprecia el mismo efecto que en los resultados del apartado 9.2.2 producido por la dinámica no modelada que hace variar el ángulo de inclinación ϕ del Roll, y por tanto una variación en la velocidad lineal v_y^B, cuando se realiza un rotación del Yaw. Una vez que el cuadricóptero ha dejado de realizar la rotación ψ comienza a corregir el desvío producido para llegar al punto destino. Teniendo en cuenta el tiempo desde que el cuadricóptero llega al primer punto hasta el final del experimento, se calcula el error máximo entre la respuesta real y la referencia, así como la raíz de su error cuadrático medio (RMSE), que se muestran en la Tabla 9.8.

	x^E (m)	y^E (m)	ψ^E (rad)	z^E (m)
Error máximo	0.4443	0.3512	0.2097	0.0926
RMSE	0.1593	0.1406	0.0915	0.0397

Tabla 9.8: Medidas de error del segundo experimento de navegación a través de Waypoints con seguimiento de rectas.

Como puede observarse el error es mucho mayor que en el experimento anterior debido a dicho efecto.

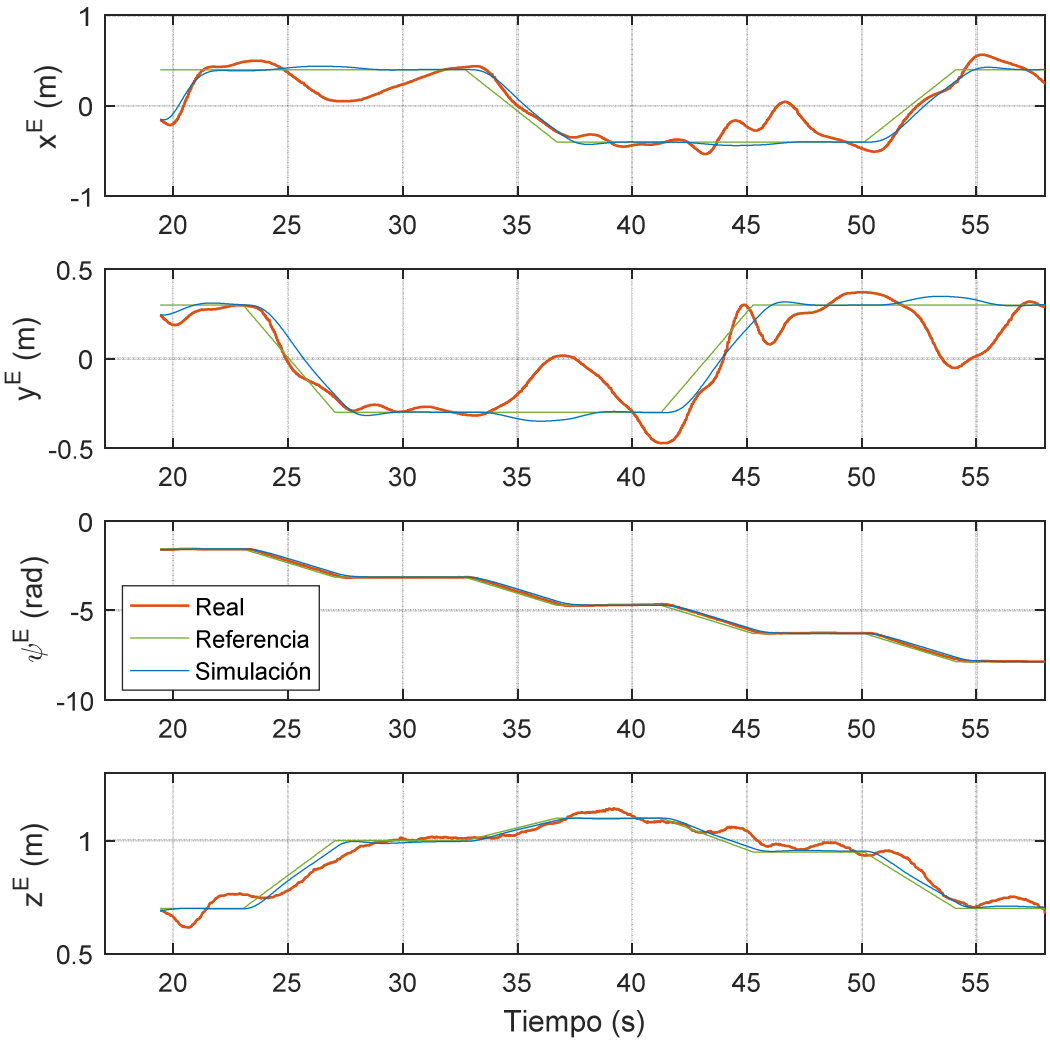


Figura 9.12: Respuesta temporal del experimento de navegación a través de Waypoints con seguimiento de rectas.

9.3.2. Seguimiento de trayectorias paramétricas

El seguimiento de trayectorias del apartado anterior se refiere a líneas rectas, pero sería adecuado comprobar el comportamiento del cuadricóptero cuando se le intenta hacer seguir otro tipo de trayectorias, elípticas como es el caso de los dos primeros experimentos que vienen a continuación, o en forma de ocho como el último experimento de este apartado.

9.3.2.1. Experimento de seguimiento de una trayectoria elíptica sin rotación ψ

La elipse considerada en este experimento tiene su centro en las coordenadas XY (0,0), su amplitud en x es de 0.5 metros mientras que su amplitud en y es de 0.4 metros. La orientación ψ se mantiene fija a un valor de -1.57 radianes mientras que la altura varía entre 0.7 y 1.1 metros. La variación temporal se muestra en el conjunto de ecuaciones (9.2), teniendo en cuenta que un ciclo se completa en 5π segundos.

$$\begin{aligned} x(t) &= 0.5\cos(0.4t) \\ y(t) &= 0.4\sin(0.4t) \\ z(t) &= 0.9 + 0.2\cos(0.4t) \\ \psi(t) &= -1.57 \end{aligned} \tag{9.2}$$

La posición inicial al entrar en modo automático ha sido (-0.08,0.24,0.68) metros con una orientación de -1.58 radianes y el resultado del experimento puede verse en la Figura 9.13 y en la Figura 9.14.

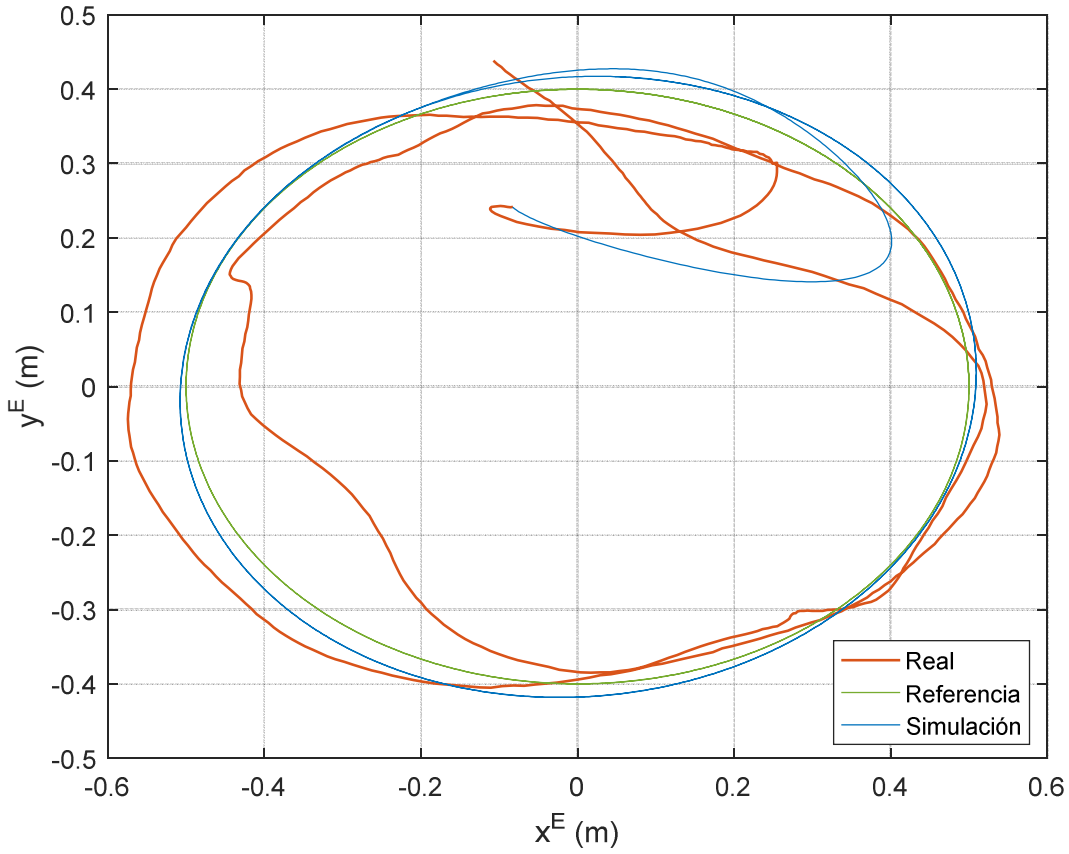


Figura 9.13: Gráfica XY del experimento de seguimiento de trayectoria elíptica.

En estas figuras se aprecia que el cuadricóptero sigue la trayectoria de forma aceptable pero teniendo zonas en las que se aleja bastante, sobre todo en la componente x. El problema de estos desvíos puede deberse al reducido espacio en el que se realizan los experimentos y a la selección de la velocidad para seguir la trayectoria.

Con una trayectoria definida tan pequeña con la velocidad de este experimento el cuadricóptero tiende a pasarse debido a que es una velocidad algo mayor de la que dicho cuadricóptero debería llevar para realizar bien la elipse. Sin embargo si se intenta realizar con una menor velocidad (cada ciclo dura más tiempo), el cuadricóptero también se desvía mucho ya que a velocidades menores está más expuesto a perturbaciones externas.

Si este experimento se pudiera realizar con una elipse del doble de tamaño se podría llevar a cabo con la misma velocidad y con menor error relativo que el de este experimento. Teniendo en cuenta el tiempo desde que el cuadricóptero llega inicialmente a la trayectoria hasta el final del experimento, se calcula el error máximo entre la respuesta real y la referencia, así como la raíz de su error cuadrático medio (RMSE), que se muestran en la Tabla 9.9.

	x^E (m)	y^E (m)	ψ^E (rad)	z^E (m)
Error máximo	0.2341	0.1325	0.0388	0.1138
RMSE	0.0769	0.0559	0.0163	0.0707

Tabla 9.9: Medidas de error del experimento de seguimiento de trayectoria elíptica.

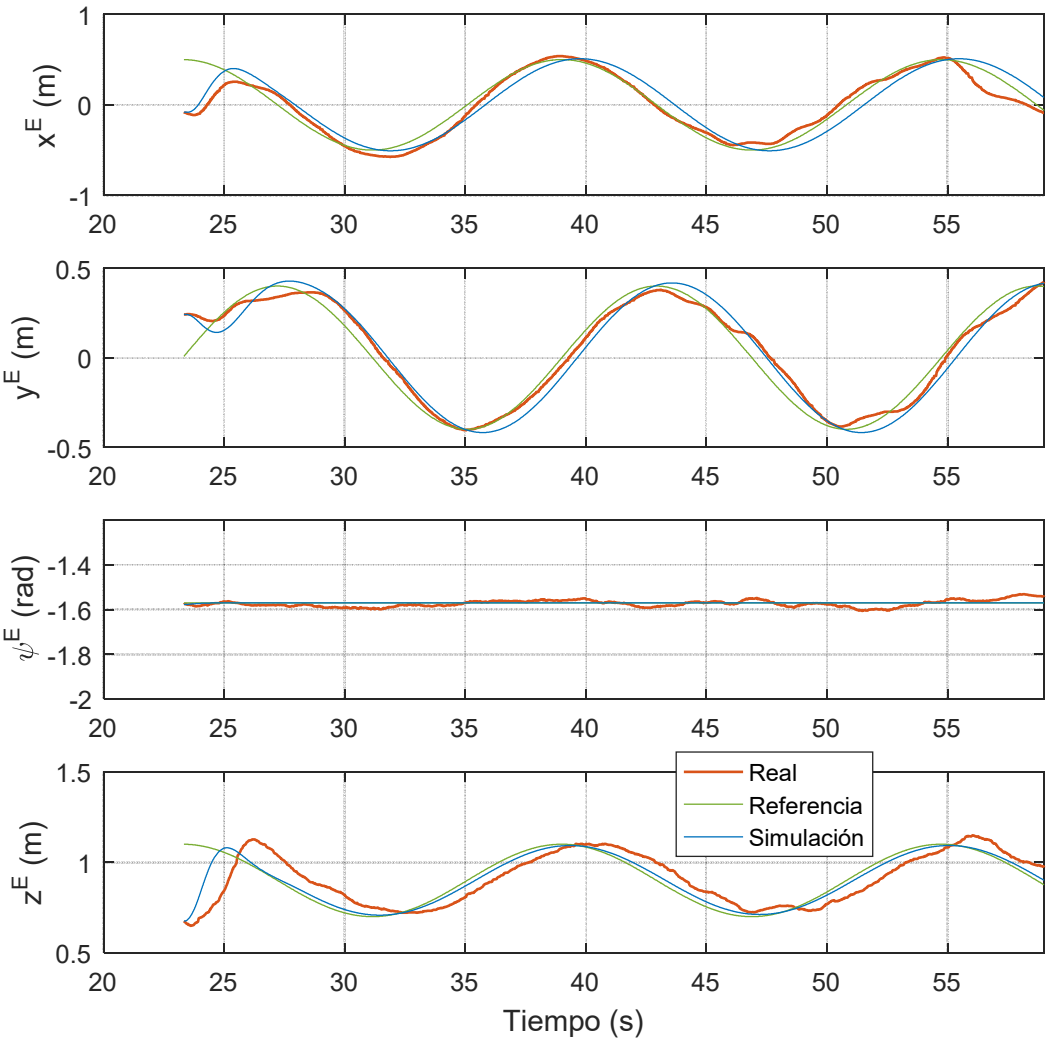


Figura 9.14: Respuesta temporal del experimento de seguimiento de trayectoria elíptica.

9.3.2.2. Experimento de seguimiento de una trayectoria elíptica con rotación ψ

La elipse considerada en este experimento tiene la misma forma que la del experimento anterior pero su ángulo de orientación varía de forma progresiva para mantener al cuadricóptero siempre orientado hacia el centro de la elipse. La variación temporal de esta elipse se muestra en el conjunto de ecuaciones (9.3), teniendo en cuenta que un ciclo se completa en 5π segundos.

$$\begin{aligned} x(t) &= 0.5\cos(0.4t) \\ y(t) &= 0.4\sin(0.4t) \\ z(t) &= 0.9 + 0.2\cos(0.4t) \\ \psi(t) &= -2\pi + 0.4t \end{aligned} \tag{9.3}$$

La posición inicial al entrar en modo automático ha sido (0.80,0.15,1.08) metros con una orientación de -2.47 radianes y el resultado del experimento puede verse en la Figura 9.15 y en la Figura 9.16.

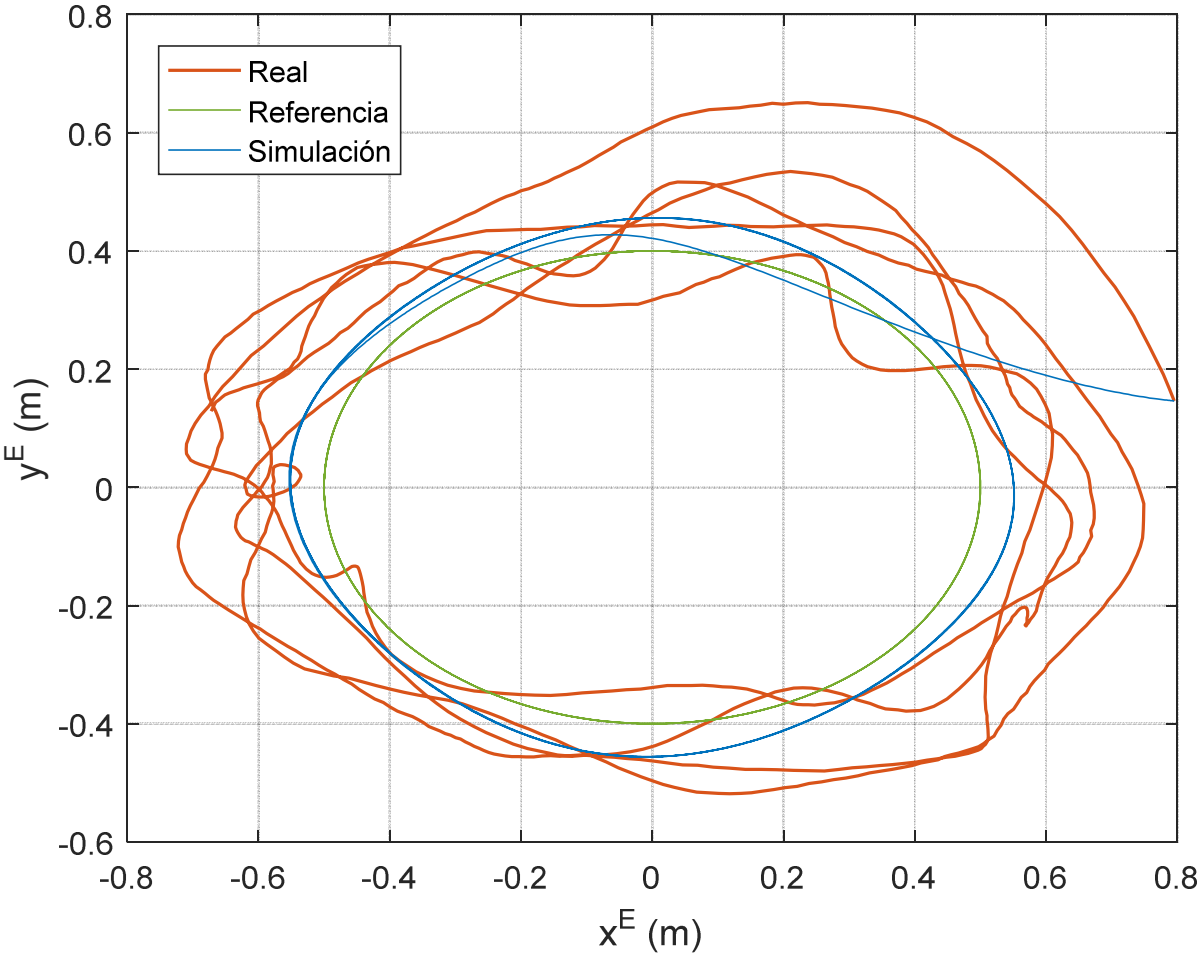


Figura 9.15: Gráfica XY del experimento de seguimiento de trayectoria elíptica con orientación hacia el centro.

En estas figuras se aprecia que el cuadricóptero sigue la trayectoria de forma aceptable. Sin embargo se aprecia un retraso bastante acusado en la respuesta temporal de la altura con respecto a la referencia. Esto puede deberse a que, al haber una continua rotación de ψ y de las inclinaciones θ y ϕ, se pierde parte de la componente vertical del empuje debido a las velocidades de rotación que los motores deben tener para conseguir dichos ángulos, y por ello el cuadricóptero pierde altura costándole más tiempo llegar a la referencia deseada.

Teniendo en cuenta el tiempo desde que el cuadricóptero llega inicialmente a la trayectoria hasta el final del experimento, se calcula el error máximo entre la respuesta real y la referencia, así como la raíz de su error cuadrático medio (RMSE), que se muestran en la Tabla 9.10.

	x^E (m)	y^E (m)	ψ^E (rad)	z^E (m)
Error máximo	0.3724	0.2338	0.3589	0.1487
RMSE	0.1521	0.0942	0.2635	0.0765

Tabla 9.10: Medidas de error del experimento de seguimiento de trayectoria elíptica con orientación hacia el centro.

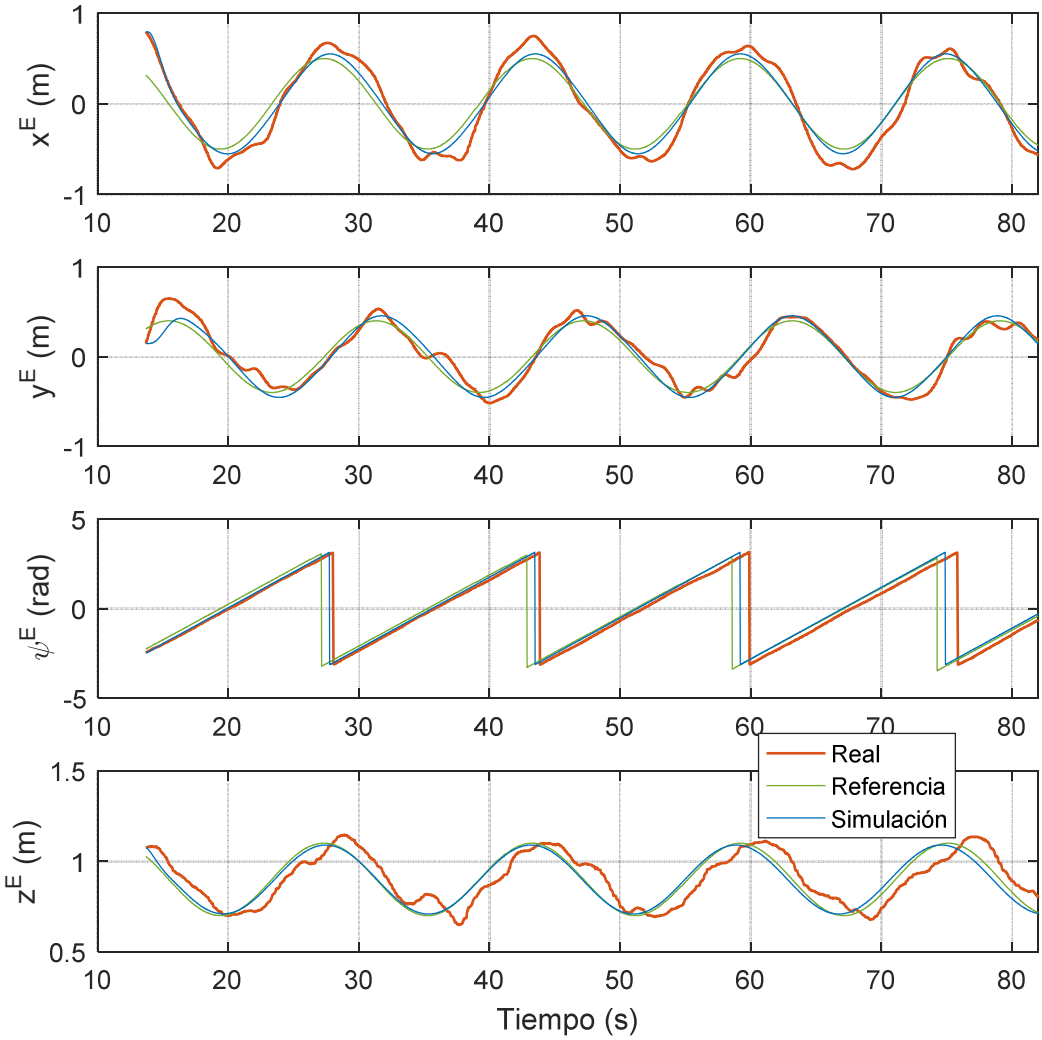


Figura 9.16: Respuesta temporal del experimento de seguimiento de trayectoria elíptica con orientación hacia el centro.

9.3.2.3. Experimento de seguimiento de una trayectoria en forma de ocho

En este experimento se va a realizar el seguimiento de una trayectoria definida por la lemniscata de Gerono. Es una trayectoria de tipo circular con forma de ocho algo especial. La variación temporal de la lemniscata de Gerono puede verse en el conjunto de ecuaciones (9.4).

$$\begin{aligned} x(t) &= 0.5\cos(0.4t) \\ y(t) &= 0.4\sin(0.8t) \\ z(t) &= 0.9 + 0.2\cos(0.4t) \\ \psi(t) &= -1.57 \end{aligned} \tag{9.4}$$

Esta trayectoria tiene su centro en las coordenadas XY (0,0), su amplitud en x es de 0.5 metros mientras que su amplitud en y es de 0.4 metros. La orientación ψ se mantiene fija a un valor de -1.57 radianes mientras que la altura varía entre 0.7 y 1.1 metros. Cada ciclo se completa en 5π segundos.

La posición inicial al entrar en modo automático ha sido (-0.29,-0.50,0.64) metros con una orientación de -1.55 radianes y el resultado del experimento puede verse en la Figura 9.17 y en la Figura 9.18.

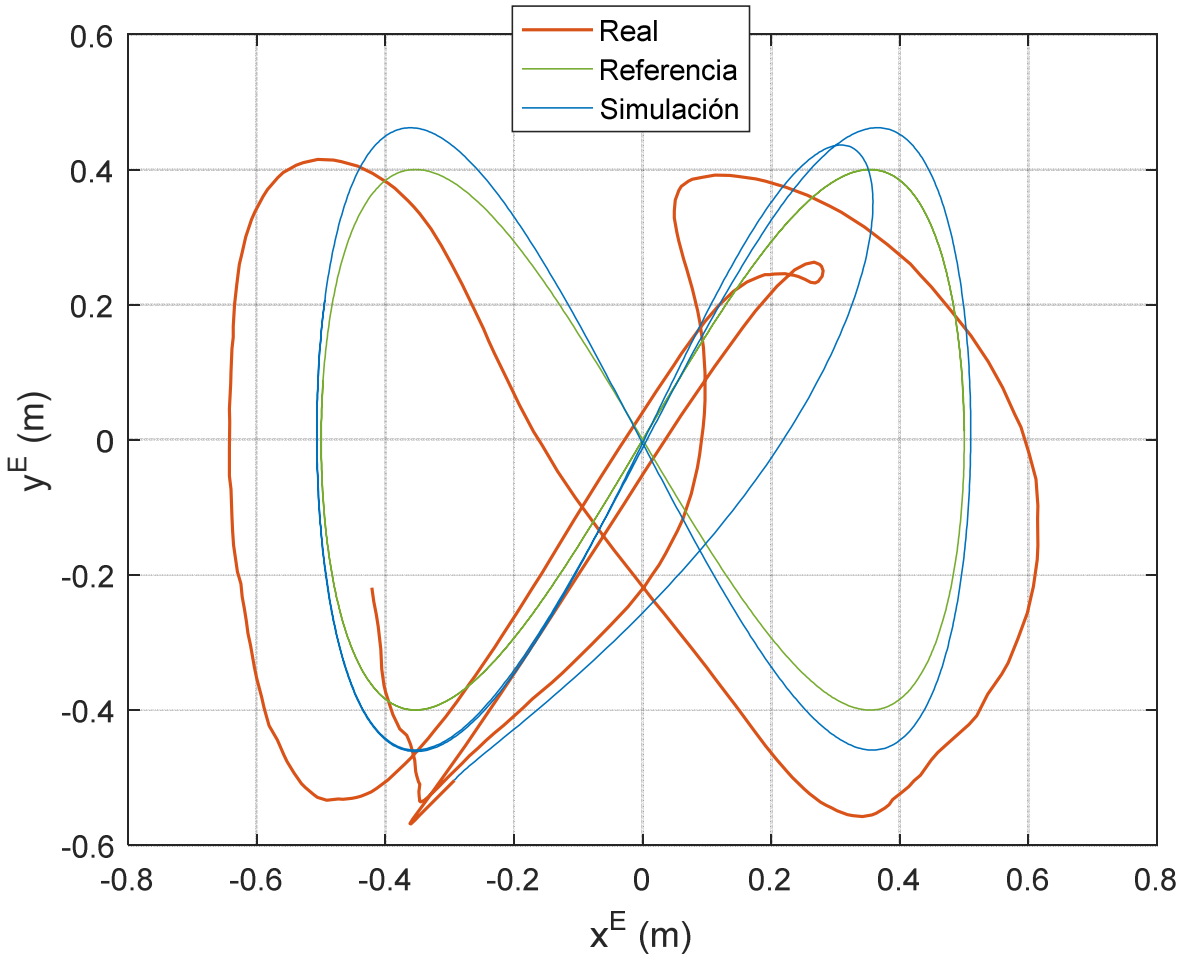


Figura 9.17: Gráfica XY del experimento de seguimiento de trayectoria de la lemniscata de Gerono.

En estas figuras se aprecia que el cuadricóptero sigue la trayectoria de forma aceptable, con márgenes de error similares al resto de experimentos de seguimiento de trayectorias circulares realizados. En este sentido se aprecia el mismo problema que en la trayectoria de la elipse sin rotación de ψ donde las trayectorias de amplitud tan pequeñas ponen de manifiesto el error de control que pasaría desapercibido en trayectorias más amplias.

Teniendo en cuenta el tiempo desde que el cuadricóptero llega inicialmente a la trayectoria hasta el final del experimento, se calcula el error máximo entre la respuesta real y la referencia, así como la raíz de su error cuadrático medio (RMSE), que se muestran en la Tabla 9.11.

	x^E (m)	y^E (m)	ψ^E (rad)	z^E (m)
Error máximo	0.2361	0.3285	0.0498	0.1761
RMSE	0.1220	0.1572	0.0239	0.0826

Tabla 9.11: Medidas de error del experimento de seguimiento de trayectoria de la lemniscata de Gerono.

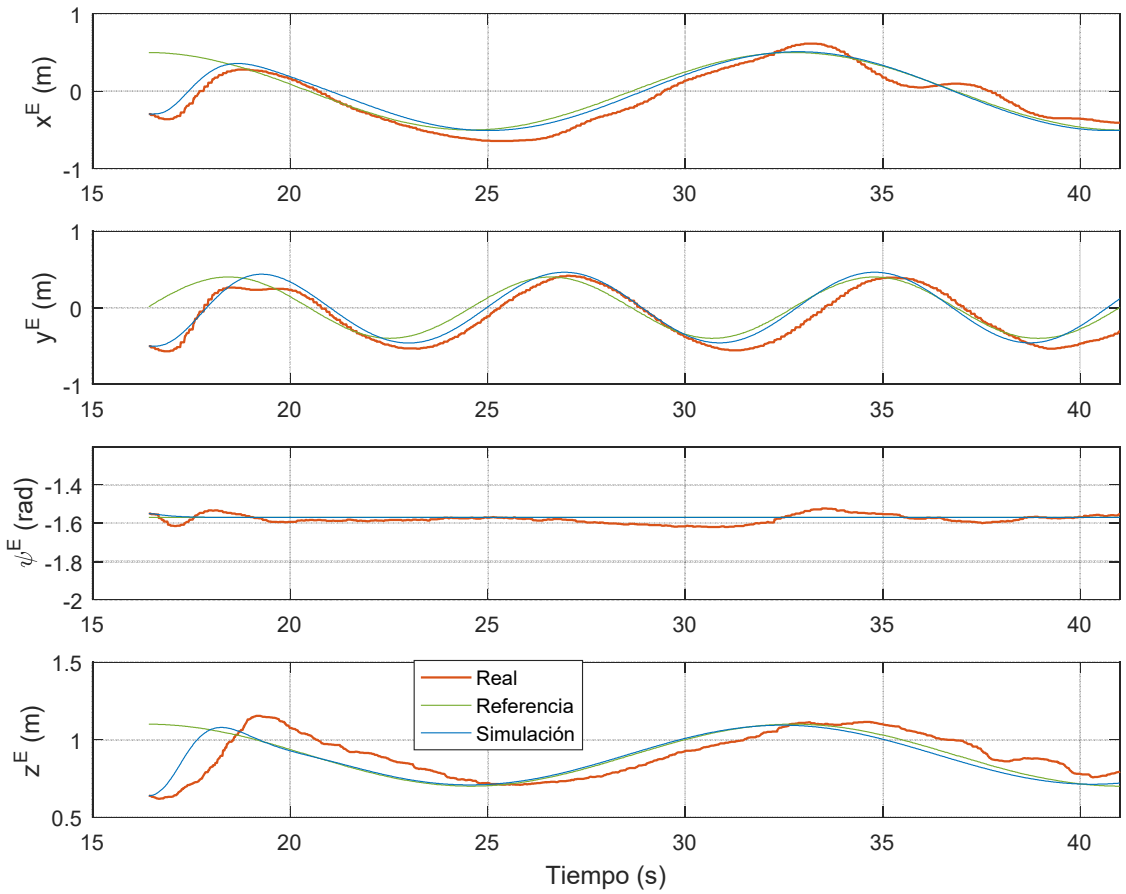


Figura 9.18: Respuesta temporal del experimento de seguimiento de trayectoria de la lemniscata de Gerono.

10. CONCLUSIONES

Los resultados presentados en esta memoria han permitido verificar que se ha cumplido con los objetivos del presente proyecto. Se ha realizado un estudio del cuadricóptero Parrot AR.Drone 2.0, derivando un modelo cinemático y dinámico general, y se ha obtenido un modelo simplificado para su control. Se ha diseñado un controlador PID que consigue hacer un control autónomo del cuadricóptero con un margen de error aceptable creando, a la vez, un software que puede utilizarse como plataforma de control con el que poder ensayar nuevas estrategias de control o de navegación autónoma con este cuadricóptero. Con los resultados incluidos en esta memoria se demuestra el buen funcionamiento de esta plataforma de control.

Aunque en un principio el lugar de experimentación parecía tener el suficiente espacio como para realizar buenos experimentos de control autónomo del AR.Drone 2.0, tras realizar diversos experimentos se ha visto que la pequeña área de observación del sistema de visión limita las trayectorias que se pueden pedir seguir al cuadricóptero.

Esta limitación del espacio también viene impuesta por la configuración del triángulo de leds del cuadricóptero que ayuda al sistema de visión a determinar su posición y orientación. La distancia del centro del lado de dicho triángulo corto al led más alejado es de algo más de 20 cm. En determinadas posiciones y orientaciones del cuadricóptero la cámara no puede llegar a captar dicho led perdiendo durante un tiempo la posición y orientación reales. Los experimentos han tenido que definirse teniendo en cuenta estas limitaciones de espacio y los posibles desvíos del cuadricóptero, y aunque se ha podido probar que el Filtro de Kalman estima bastante bien estas variables cuando el sistema de visión no las puede determinar durante un corto espacio de tiempo, en algunos experimentos se han perdido las medidas del sistema de visión durante largo tiempo haciendo muy difícil el control en el seguimiento de trayectorias de mayor amplitud.

Otro de los problemas identificados tiene que ver con los obstáculos existentes junto al área de vuelo. Paredes, mesas y sillas se encuentran muy cerca de dicha área haciendo que el aire generado por las hélices del cuadricóptero choque con ellas y retorne hacia él provocando alteraciones en su movimiento y posición.

El controlador PID ha sido obtenido en base a un modelo identificado en el marco de referencia B del cuadricóptero, obviando las no linealidades que son introducidas en el sistema debido a las transformaciones entre el marco de referencia B y el marco de referencia E, que es donde se verifica que las posiciones y las orientaciones son las correctas. Aunque ajustando las ganancias del controlador PID diseñado se consigue una mejor respuesta en el control autónomo, se plantea como trabajo futuro seguir un enfoque no lineal en el diseño del controlador para que se ajuste lo máximo posible al comportamiento del sistema real.

Durante la realización y análisis de los experimentos se han observado comportamientos que no fueron identificados en el modelo simplificado. Uno de estos efectos es el que hace que el ángulo de inclinación ϕ del Roll cambie, conllevando así mismo una variación en la velocidad lineal v_y^B, cuando se realiza una rotación del Yaw. De este modo el cuadricóptero tiende a cambiar su ángulo de Roll inclinándose hacia el sentido de giro del Yaw teniendo en cuenta que cuanta mayor velocidad de rotación mayor es el efecto en dicha inclinación.

Por ello, y dada la rápida dinámica de rotación del Yaw, el cuadricóptero se desvía demasiado de la trayectoria esperada del modelo y el controlador no puede manejar bien este comportamiento no modelado, pudiendo únicamente corregir la posición una vez el cuadricóptero ha llegado a la orientación definida por el algoritmo de navegación. Una posible corrección sencilla de este comportamiento, que se plantea como trabajo futuro, es diseñar un algoritmo de control que no permita la acción del controlador del Roll mientras el error en el Yaw sea mayor de un determinado valor límite.

Otro de los efectos observados es la pérdida de altura cuando el cuadricóptero está rotando de Yaw (variando el ángulo ψ) a la vez que se está desplazando en x e y (hay inclinaciones de los ángulos θ y ϕ). Cada uno de los motores debe girar a una determinada velocidad para conseguir estas inclinaciones a la vez, causando la pérdida de parte de la componente vertical del empuje. Esto conlleva la pérdida de altura del cuadricóptero costándole más tiempo llegar a la referencia deseada.

Estos comportamientos observados durante la realización de este proyecto provocan una clara desviación de las referencias que proporcionan los algoritmos de navegación, y es más acusada en el caso del primer comportamiento de los aquí indicados haciendo prácticamente imposible el seguimiento de una línea recta entre dos puntos cuando la orientación entre ellos es sustancialmente diferente.

Como trabajo futuro se plantea la solución ideal a estos comportamientos, que no es más que la de seguir una estrategia multivariable tanto en la identificación del modelo como en el diseño del controlador que compense los efectos que en el presente proyecto no han sido modelados.

LISTADO DE REFERENCIAS Y BIBLIOGRAFÍA

1. Drone militar Searcher II. [En línea] http://www.israeli-weapons.com/weapons/aircraft/uav/searcher2/Searcher2.html.

2. The Drone Racing League. [En línea] http://thedroneracingleague.com/.

3. Surveillance Drones - MicroDrones. [En línea] https://www.microdrones.com/en/applications/areas-of-application/monitoring/.

4. The Drone Co. - Professional Drone Filming. [En línea] https://thedrone.co/.

5. Fox Sports - Drones used in Sports Events. [En línea] http://www.foxsports.com/presspass/latestnews/2015/03/16/fox-sports-aggressivelypursuesdronecamerausageatliveevents.

6. ConservationDrones. [En línea] https://conservationdrones.org/.

7. Amazon Prime Air. [En línea] https://www.amazon.com/b?node=8037720011.

8. ElPais - Drones transporte de desfibriladores. [En línea] http://www.elpais.com.uy/vida-actual/presentan-holanda-primer-prototipo-drone.html.

9. MIT Technology Review - Drones for Agriculture. [En línea] https://www.technologyreview.com/s/526491/agricultural-drones/.

10. GISLounge - Drones for GIS. [En línea] https://www.gislounge.com/use-drones-gis/.

11. Nature - Drones in Science. [En línea] http://www.nature.com/news/drones-in-science-fly-and-bring-me-data-1.13161.

12. Parrot AR.Drone 2.0 Power Edition. [En línea] https://www.parrot.com/es/en/drones/parrot-ardrone-20-power-edition#share-and-impress-your-friends-with-your-hd-films.

13. **Bouabdallah, Samir.** Design and Control of Quadrotors with application to autonomous flying. [PhD Thesis]. Lausanne, Switzerland : École Polytechnique Fédérale de Lausanne, Febrero de 2007.

14. **Bresciani, Tommaso.** Modelling, Identification and Control of a Quadrotor Helicopter. [M.S. Thesis]. Lund, Sweden : Lund University, Octubre de 2008.

15. *PID vs LQ Control Techniques Applied to an Indoor Micro Quadrotor.* **Bouabdallah, Samir, Noth, André y Siegwart, Roland.** s.l. : Swiss Federal Institute of Technology, 2004.

16. *Direct Adaptative Tracking Control of Quadrotor Aerial Vehicles.* **Morel, Y. y Leonessa, A.** 2006, Florida Conference on Recent Advances in Robotics.

17. *Backstepping control for a quadrotor helicopter.* **Madani, T. y Benallegue, A.** 2006, Proceedings of 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems.

18. *AR-Drone as a Platform for Robotic Research and Education.* **Krajnik, Tomas, y otros.** Junio de 2011, Communications in Computer and Information Science. DOI: 10.1007/978-3-642-21975-7_16.

19. **Min Ahn, Young.** Autonomous Navigation and Localization of a Quadrotor Mini-UAV by Landmarks in Indoor Environments. *University of Illinios.* [Master Thesis]. Urbana (Illinois) : s.n., 2011.

20. *Camera-Based Navigation of a Low-Cost Quadrocopter.* **Engel, Jakob, Sturm, Jurgen y Cremers, Daniel.** 2012.

21. *Estimation and Control of the 3D Position of a Quadrotor in Indoor Environments.* **Vago Santana, Lucas, Sarcinelli-Filho, Mario y Carelli, Ricardo.** 2013.

22. *A Trajectory Tracking and 3D Positioning Controller for the AR.Drone Quadrotor.* **Vago Santana, Lucas, y otros.** 2014.

23. **Quebe, Stephen C.** Modeling, Parameter Estimation, and Navigation of Indoor Quadrotor Robots. *All Theses and Dissertations.* 2013. Paper 3565.

24. **Murray, Richard M., Li, Zexiang y Sastry, S. Shankar.** *A Mathematical Introduction to Robot Manipulation.* s.l. : CRC PRess, 1994. ISBN 9780849379819.

25. **Leishman, Gordon J.** *Principles of Helicopter Aerodynamics.* 2nd edition. London : Cambridge University Press, 2006.

26. **Johnson, Wayne.** *Helicopter Theory.* New York : Dover Publications, 1980. ISBN 9780486682303.

27. *Influence of Aerodynamics and Proximity Effects in Quadrotor Flight.* **Powers, Caitlin, y otros.** 2013, Experimental Robotics, Vol. 88.

28. *The Navigation and Control technology inside the AR.Drone micro UAV.* **Bristeau, Pierre-Jean, y otros.** Milan : International Federation of Automatic Control, 2011.

29. **Piskorski, Stephane, y otros.** *AR.Drone Developer Guide SDK 2.0.1.* s.l. : Parrot, 2012.

30. **Piskorski, Stephane, Brulez, Nicolas y Eline, Pierre.** *AR.Drone Developer Guide SDK 1.8.* s.l. : Parrot, 2011.

31. **Endres, Thomas.** Repositorio Github del proyecto ARDrone-Control-.NET. [En línea] https://github.com/shtejv/ARDrone-Control-.NET/.

32. **Zaliva, Vadim.** Repositorio Github del proyecto Javadrone. [En línea] https://github.com/codeminders/javadrone.

33. **Escobar Sanabria, David.** Matlab File Exchange para el AR Drone Simulink Development Kit. [En línea] https://es.mathworks.com/matlabcentral/fileexchange/43719-ar-drone-simulink-development-kit-v1-1.

34. **Monajjemi, Mani.** Repositorio Github del proyecto ardrone_autonomy. [En línea] https://github.com/AutonomyLab/ardrone_autonomy.

35. **Engel, Jakob, Sturm, Jürgen y Cremers, Daniel.** Repositorio Github del proyecto tum_ardrone. [En línea] https://github.com/tum-vision/tum_ardrone.

36. **Mogenson, Michael.** The AR Drone LabVIEW Toolkit: A Software Framework for the Control of Low-Cost Quadrotor Aerial Robots. s.l. : TUFTS University, Mayo de 2012.

37. **Mogenson, Michale.** Sitio oficial del proyecto AR Drone Toolkit para LabView - LVH en National Instruments. [En línea] http://sine.ni.com/nips/cds/view/p/lang/es/nid/211837.

38. **Mogenson, Michael.** Sitio de descarga de la versión 0.1.0.36-BETA del AR Drone Toolkit para LabVIEW - LVH. [En línea] https://decibel.ni.com/content/docs/DOC-35691.

39. **Chaos García, Dictino.** Control no lineal de vehículos marinos subactuados no-holonómicos. [Tesis Doctoral]. Madrid : Universidad Nacional de Educación a Distancia, 2010.

40. **Monteagudo Duro, Nicolás.** Plataforma de control para un vehículo aéreo no tripulado: Desarrollo hardware y software. *Universidad Nacional de Educación a Distancia.* [Proyecto Fin de Carrera]. 2013.

41. **Sun, Yue.** Modeling, Identification and Control of a Quad-Rotor Drone Using Low-Resolution Sensing. *University of Illinois.* [Master Thesis]. Urbana (Illinios) : s.n., 2012.

42. **Martin, Gerard.** Modelling and Control of the Parrot AR.Drone. *UNSW Canberra.* Canberra : s.n., 2012.

43. *Monocular Vision-Based Autonomous Navigation System on a Toy Quadcopter in Unknow Environments.* **Huang, Rui, Tan, Ping y M. Chen, Ben.** 2015, Unmanned Aircraft Systems. DOI: 10.1109/ICUAS.2015.7152419 .

44. **Yifang Tang, Sarah.** Vision-Based Control for Autonomous Quadrotor UAVs. *Princeton University.* 2013.

45. **Czerwinski, Fabian.** Matlab FileExchange - Allan variance 3.0. [En línea] https://es.mathworks.com/matlabcentral/fileexchange/26659-allan-v3-0.

46. *Survey on Various Noises and Techniques for Denoising the Color Image.* **Farooque, Mohd Awais y S. Rohankar, Jayant.** Issue 11, Noviembre de 2013, International Journal of Application or Innovation in Engineering and Management, Vol. 2. ISSN 2319 - 4847.

47. **S. Flenniken IV, Warren.** Modeling Inertial Measurement Units and Analyzing the Effects of their Errors in Navigation Applications. [Master Thesis]. Alabama : s.n., Diciembre de 2005.

48. **Thrun, Sebastian, Burgard, Wolfram y Fox, Dieter.** *Probabilistic Robots.* ISBN-13: 978-0262201629. s.l. : The MIT Press, 2005.

49. **Bar-Shalom, Yaakov, Li, X.-Rong y Kirubarajan, Thiagalingam.** *Estimation with Applications to Tracking and Navigation.* s.l. : John Wiley & Sons, 2001. ISBN 0-471-41655-X.

50. **Reid, Ian.** *Lecture Notes in Estimation.* s.l. : University of Adelaide, 2001.

51. **R. Labbe Jr, Roger.** Kalman and Bayesian Filters in Python. [En línea] Diciembre de 2015. https://github.com/rlabbe/Kalman-and-Bayesian-Filters-in-Python.

52. *Methods for Estimating State and Measurement Noise Covariance Matrices: Aspects and Comparison.* **Dunik, Jindrich, Simandl, Miroslav y Straka, Ondrej.** s.l. : University of West Bohemia, Enero de 2009.

53. **Astrom, Karl J. y Hagglund, Tore.** *PID Controllers.* 2nd. s.l. : Instrument Society of America, 1994.

54. **Ogata, Katsuhiko.** *Modern Control Engineering.* 5th. s.l. : Prentice Hall, 2010.

55. *A Survey of Modelling and Identification of Quadrotor Robot.* **Zhang, Xiaodong, y otros.** Volume 2014, Article ID 320526, s.l. : Hindawi Publishing Corporation, 2014, Vol. Abstract and Applied Analysis.

56. **Cai, Guowei, Chen, Ben M. y Lee, Tong Heng.** Coordinate Systems and Transformations. *Unmanned Rotorcraft Systems.* s.l. : Springer, 2011, 2.

57. *An Algebraic Approach to Lens Distortion by Line Rectification.* **Álvarez, L., Gómez, L. y Sendra, J. R.** 2009, Journal of Mathematical Imaging and Vision, Vol. 35(1), págs. 1573-7683.

58. **Hartley, P. R. y Zisserman, A.** *Multiple view geometry in computer vision.* (2 ed.). s.l. : Cambridge University Press, 2003.

59. **Astrom, Karl Johan y Murray, Richard M.** *Feedback Systems.* v2.10e. s.l. : Princeton University Press, 2011.

60. Wikipedia UAV. [En línea] https://en.wikipedia.org/wiki/Unmanned_aerial_vehicle.

61. Intelligent UAVs. [En línea] http://www.iuavs.com/pages/aplicaciones_y_usos.

LISTADO DE SIGLAS, ABREVIATURAS Y ACRÓNIMOS

ACK	Acuse de recibo
ADC	Convertidor Analógico-Digital
ARW	Variación Aleatoria del Ángulo de un giróscopo
ASCII	Código Estándar Estadounidense para el Intercambio de Información
AT	Comando, en cadena de texto, que se utiliza en las comunicaciones con el AR.Drone
AVR	Familia de microcontroladores RISC del fabricante Atmel
BS	Parámetro Bias Stability de un giróscopo
CCD	Dispositivo de Carga Acoplada
CMOS	Semiconductor Complementario de Óxido Metálico
CR	Carácter de retorno de carro
DDR2	Versión 2 de las memorias RAM de Tasa Doble de Datos
DHCP	Protocolo de Configuración Dinámica de Equipos
DSP	Procesador Digital de Señal
FGV	Variable Global Funcional
FPS	Imágenes por segundo
FTP	Protocolo de Transferencia de Archivos
GPS	Sistema de Posicionamiento Global
H264	Norma que define un codec de video de alta compresión
HD	Alta Definición
HID	Dispositivo de Interfaz Humana
ID	Identificación
IEEE	Instituto de Ingenieros Eléctricos y Electrónicos
IMU	Unidad de Medida Inercial
iOS	Sistema Operativo iPhone
IP	Protocolo de Internet
KF	Filtro de Kalman
LED	Diodo Emisor de Luz
LiPo	Polímero de Litio
LQR	Regulador Cuadrático Lineal
MAC	Control de Acceso al Medio
MIPS	Millones de Instrucciones por Segundo
PC	Ordenador personal
P	Controlador Proporcional

PI	Controlador Proporcional-Integral
PID	Control Proporcional-Integral-Derivativo
PWM	Modulación del Ancho de Pulso
QVGA	Quarter Video Graphics Array (Tarjeta gráfica de resolución 320x240)
RAM	Memoria de acceso aleatorio
RISC	Computador con Conjunto de Instrucciones Reducidas
RMSE	Raíz del Error Cuadrático Medio
ROS	Robot Operating System
RT	Tiempo Real
SDK	Kit de Desarrollo de Software
TCP	Protocolo de Control de Transmisión
UART2	Versión 2 del Transmisor-Receptor Asíncrono Universal
UAV	Vehículo Aéreo no Tripulado
UDP	Protocolo de Datagrama de Usuario
UNED	Universidad Nacional de Educación a Distancia
USB	Bus Serie Universal
VI	Virtual Instrument
VRW	Variación Aleatoria de la Velocidad de un acelerómetro
Wi-Fi	Conexión inalámbrica

ANEXO A. SISTEMA DE VISIÓN

En este anexo se describe someramente el sistema de visión utilizado para determinar la posición y orientación del cuadricóptero, y que resultó del trabajo realizado en [39], teniendo en cuenta las posteriores mejoras en los algoritmos del sistema de visión. Si se desea ampliar la información respecto a lo aquí indicado puede consultarse dicho trabajo.

A.1. Captura de la imagen

Para realizar una captura de la imagen de la cámara que pueda utilizarse para extraer puntos de luz que estén por encima de un umbral de luminosidad, es necesario configurar dos parámetros de la cámara, *Shutter* y *Gain*. El parámetro *Shutter* establece el tiempo de exposición de la cámara mientras que el parámetro *Gain* establece la ganancia hardware de la cámara.

Configurando correctamente estos dos parámetros se consiguen detectar sólo aquellos puntos de luz de interés, que en el caso del presente proyecto son los que forman la baliza en forma de triángulo.

La captura de imágenes se realiza con una velocidad de 18.7 fps, con lo que se obtiene una imagen cada 53.47 ms aproximadamente.

A.2. Extracción de puntos de la imagen capturada

Realizando un análisis de partículas sobre la imagen capturada se determina la existencia de zonas válidas cuya luminosidad está por encima de un umbral configurado, así como la posición con respecto a la imagen de la parte central, o centroide, de cada una de dichas zonas.

Las zonas válidas, y en consecuencia los puntos válidos, deben ser tres para poder determinar posteriormente la posición y orientación del cuadricóptero. Si se detectasen menos de tres o más de tres no se podría determinar y se produciría un error en la extracción de puntos, bien por pérdida o bien por exceso.

A.3. Correcciones a la imagen capturada

A.3.1. Distorsión

La óptica de la cámara utilizada en el sistema de visión introduce una distorsión en la imagen con respecto a un modelo de cámara perfecta "*pinhole*". A este tipo de distorsión se le denomina de barril y provoca que las líneas rectas paralelas en el espacio objeto se vean en la imagen como líneas curvas que se encuentran más separadas en el centro y más juntas en las esquinas.

Este efecto se debe a que la óptica de la cámara tiene una mayor potencia óptica (mayor aumento local) en los puntos que se encuentran cerca del eje focal (por este motivo la imagen se ve más grande en el centro) y menor potencia cerca de los bordes (de tal modo que las esquinas se ven más cerca de lo que realmente están).

Para corregir esta distorsión se realiza una calibración que consigue el valor de un parámetro k_1 que minimice las distancias entre unas líneas rectas determinadas en el espacio objeto que deben ser paralelas y las líneas rectas correspondientes que deben ser paralelas en el espacio imagen.

A.3.2. Perspectiva

Una vez corregida la distorsión es necesario transformar las posiciones de los puntos de la imagen expresados en píxeles a posiciones del plano del suelo expresado en metros.

Para ello hay que tener en cuenta tres factores. En primer lugar hay que considerar la trasformación de perspectiva que resulta de proyectar el punto objeto en el plano focal de la cámara. En segundo lugar hay que considerar la transformación de escala de píxeles a metros, y por último hay que tener en cuenta la rotación y

traslación entre las coordenadas en el sistema de referencia de la cámara y el sistema de referencia paralelo al suelo tal y como se muestra en la Figura A.1.

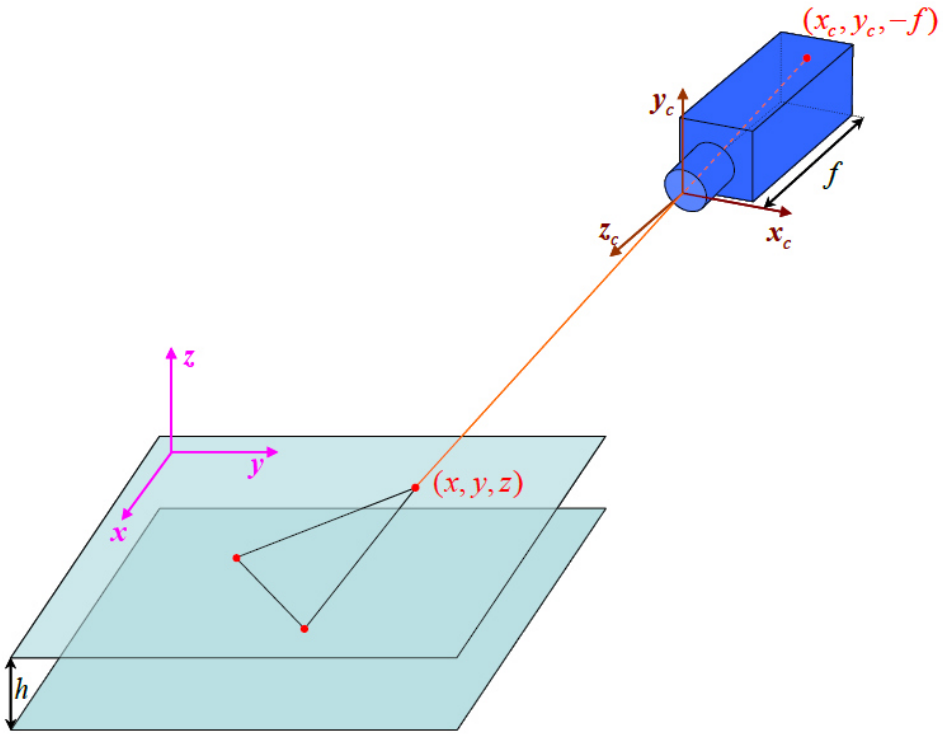


Figura A.1: Transformación general de perspectiva.

Finalmente la transformación general de perspectiva puede parametrizarse por medio de ocho parámetros $c_1..c_8$ según la ecuación (A.1).

$$x_c = \frac{x_h}{z} = \frac{c_1 x + c_2 y + c_3}{c_7 x + c_8 y + 1}$$
$$y_c = \frac{y_h}{z} = \frac{c_4 x + c_5 y + c_6}{c_7 x + c_8 y + 1} \qquad \text{(A.1)}$$

Los ocho parámetros $c_1..c_8$ se obtienen mediante un procedimiento de calibración considerando la altura h del triángulo de leds cuando el cuadricóptero se encuentra en reposo en el suelo. Con estos parámetros se obtiene la posición x, y sobre el plano que se encuentra a una altura h del suelo y que es paralelo a éste.

Durante el vuelo, sin embargo, la altura del cuadricóptero puede variar mucho y es inviable obtener los parámetros $c_1..c_8$ para todas las diferentes alturas h que puede tomar. Por ello, a la posición x, y obtenida se le realiza el ajuste proporcional que se muestra en la ecuación (A.2), donde h_{cam} es la altura de la cámara con respecto al suelo, h_{calib} es la altura del triángulo de leds en el momento de la calibración y h es la altura actual de vuelo del cuadricóptero proporcionada por sus sensores.

$$x_h = \left(\frac{h_{cam} - h_{calib} - h}{h_{cam} - h_{calib}}\right) x$$
$$y_h = \left(\frac{h_{cam} - h_{calib} - h}{h_{cam} - h_{calib}}\right) y \qquad \text{(A.2)}$$

A.4. Cálculo de la posición y orientación

La posición y orientación del cuadricóptero se determina mediante un análisis morfológico de las posiciones de las balizas detectadas mediante el algoritmo de visión. Como se comenta en el apartado 5.2.1, éstas forman un triángulo isósceles similar al de la Figura A.2 cuyos lados son conocidos.

El procedimiento de cálculo de la posición y orientación comienza localizando el punto p_c, calculando posteriormente el punto auxiliar x_{int} y finalmente determinando la posición del centro del cuadricóptero x_{cm} y la orientación θ.

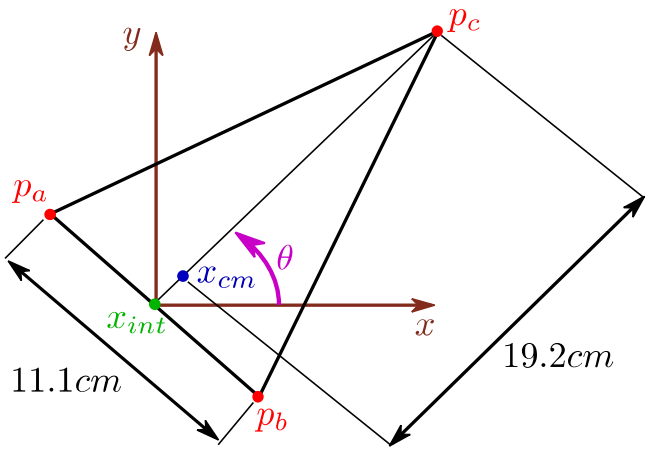


Figura A.2: Cálculo de la posición y la orientación.

Las ecuaciones que determinan, por tanto, la posición y orientación del cuadricóptero se muestran en (A.3), donde α es una constante que indica la relación entre la distancia del punto p_{cm} al punto p_c con respecto a la distancia del punto p_{int} al punto p_c, y cuyo valor puede verse en la ecuación (A.4).

$$\begin{aligned} x_{cm} &= (1-\alpha)x_c + \alpha x_{int} \\ y_{cm} &= (1-\alpha)y_c + \alpha y_{int} \\ \theta &= \tan^{-1}\left(\frac{y_c - y_{int}}{x_c - x_{int}}\right) \end{aligned} \tag{A.3}$$

$$\alpha = \frac{19.2}{\sqrt{20.2^2 - (11.1/2)^2}} = \frac{19.2}{\sqrt{377.24}} \approx 0.05089 \tag{A.4}$$

A.5. Cálculo de las velocidades

Una vez calculada la posición del vehículo es necesario estimar las velocidades del mismo. Dicha estimación se realiza por diferencias finitas en base al periodo de muestreo real T tal y como se muestra en las ecuaciones de (A.5).

$$\begin{aligned} T &= t - t_{ant} \\ v_x &= \frac{x - x_{ant}}{T} \\ v_y &= \frac{y - y_{ant}}{T} \\ \omega &= \frac{\tan^{-1}(\sin(\theta - \theta_{ant})/\cos(\theta - \theta_{ant}))}{T} \end{aligned} \tag{A.5}$$

Donde t_{ant} es el instante de captura de la última imagen válida, x_{ant}, y_{ant} y θ_{ant} la posición y orientación estimada en el último cálculo correcto de la posición.

En el cálculo de la velocidad angular se ha utilizado la función $\tan^{-1}(\sin(\theta - \theta_{ant})/\cos(\theta - \theta_{ant}))$ para evitar errores en el cálculo debido a los saltos de módulo 2π.

A.6. Procedimiento de calibración

El procedimiento utilizado para corregir la distorsión sólo tiene que realizarse una única vez, ya que el parámetro k_1 sólo depende del sistema cámara-óptica que permanece fijo. Por este motivo no es necesario automatizar dicho proceso.

No obstante, los parámetros de perspectiva de la cámara sí que pueden variar sustancialmente ente dos experimentos. Esto es debido a que dichos parámetros dependen de la colocación de la cámara respecto del suelo y debido a la holgura presente en el soporte de la cámara y a otros efectos es necesario recalibrar los parámetros $c_1..c_8$ con relativa frecuencia.

El procedimiento de calibración consiste en la colocación manual de la baliza en cuatro puntos distintos del suelo de forma sucesiva, en $p_1 = [0,0]^T$, $p_2 = [0,1]^T$, $p_3 = [1,0]^T$ y $p_4 = [1,1]^T$, determinando posteriormente sus posiciones en el espacio imagen p_{c1}, p_{c2}, p_{c3} y p_{c4}.

Sustituyendo las coordenadas de cada uno de los puntos anteriores en las ecuaciones (A.1), se obtiene el sistema lineal de ocho ecuaciones con ocho incógnitas que se muestra en la ecuación (A.6).

$$\begin{aligned} c_3 &= x_{c1} \\ c_6 &= y_{c1} \\ (c_1 + c_3)/(c_7 + 1) &= x_{c2} \\ (c_4 + c_6)/(c_7 + 1) &= y_{c2} \\ (c_2 + c_3)/(c_8 + 1) &= x_{c3} \\ (c_5 + c_6)/(c_8 + 1) &= y_{c3} \\ (c_1 + c_2 + c_3)/(c_7 + c_8 + 1) &= x_{c4} \\ (c_4 + c_5 + c_6)/(c_7 + c_8 + 1) &= y_{c4} \end{aligned} \qquad \text{(A.6)}$$

La solución de este sistema de ecuaciones permite calcular de forma directa los valores de los parámetros $c_1..c_8$.

ANEXO B. DESARROLLO EN LABVIEW

A continuación se va a describir de forma somera los bloques de LabVIEW que componen el software desarrollado para la plataforma de control del cuadricóptero AR.Drone 2.0. Si se desea ampliar el conocimiento sobre la implementación exacta de ellos, puede consultarse el código fuente entregado.

B.1. Descripción de los bloques del sistema principal

B.1.1. Bloque Principal

Principal.vi



Este bloque contiene la base del sistema principal del software, y se muestra una parte de su funcionamiento interno en la Figura B.1, donde se aprecian los 4 bucles de ejecución independientes de los que se compone. Dentro del bucle principal se pueden ver las acciones que se realizan cuando el sistema se encuentra en el estado "Marcha", que es la parte más interesante del código fuente de este bloque.

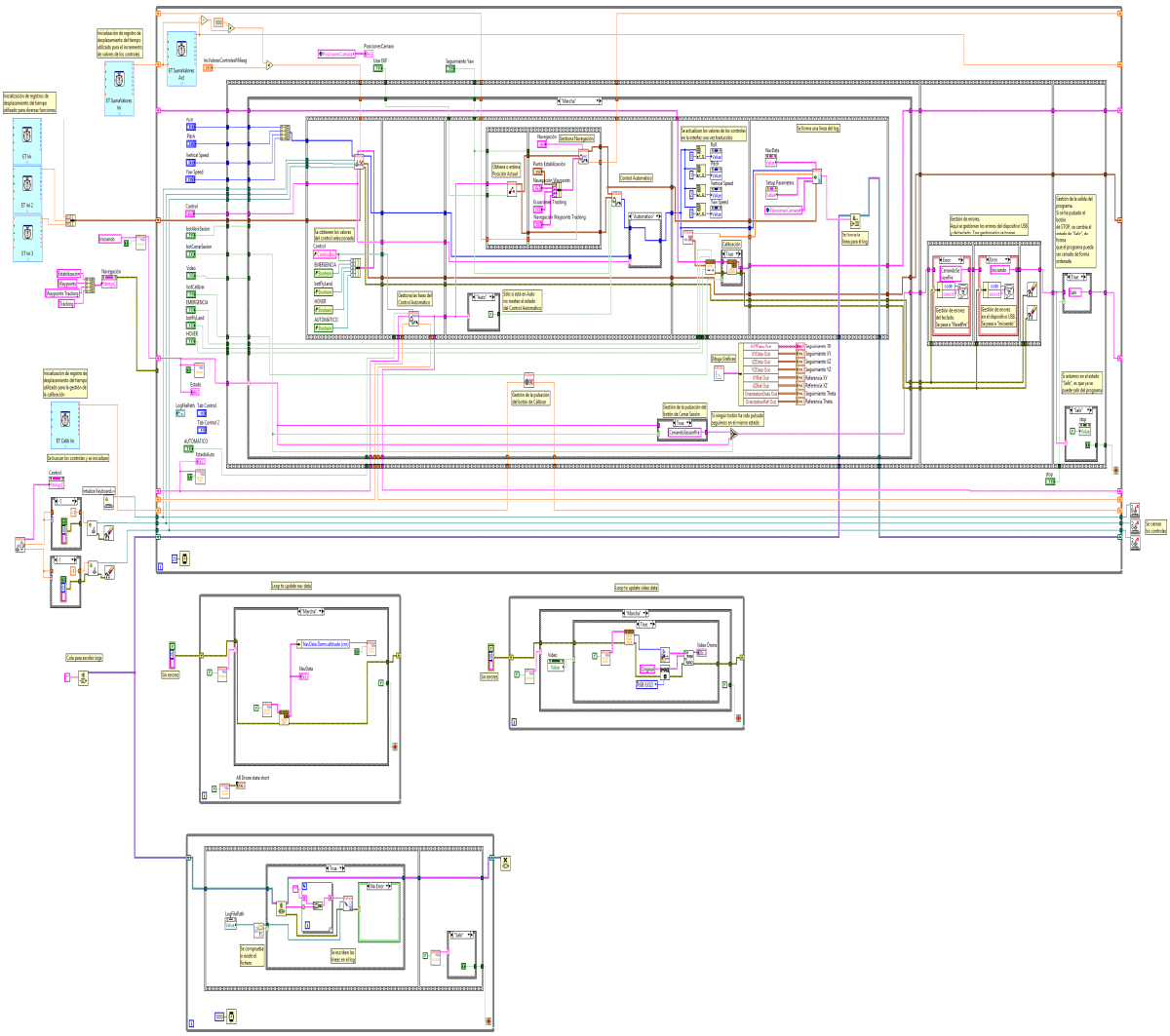
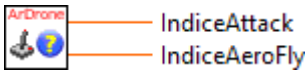


Figura B.1: Parte del funcionamiento interno del bloque Principal.

B.1.2. Bloques de control manual

BuscaJoysticks.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.2, realiza la búsqueda de los joysticks implementados en este proyecto, el Attack3 de Logitech y la emisora AeroFly de BSD Racing, para saber si se encuentran conectados al PC. Si es así devuelve el índice de dispositivo encontrado para cada uno de ellos.

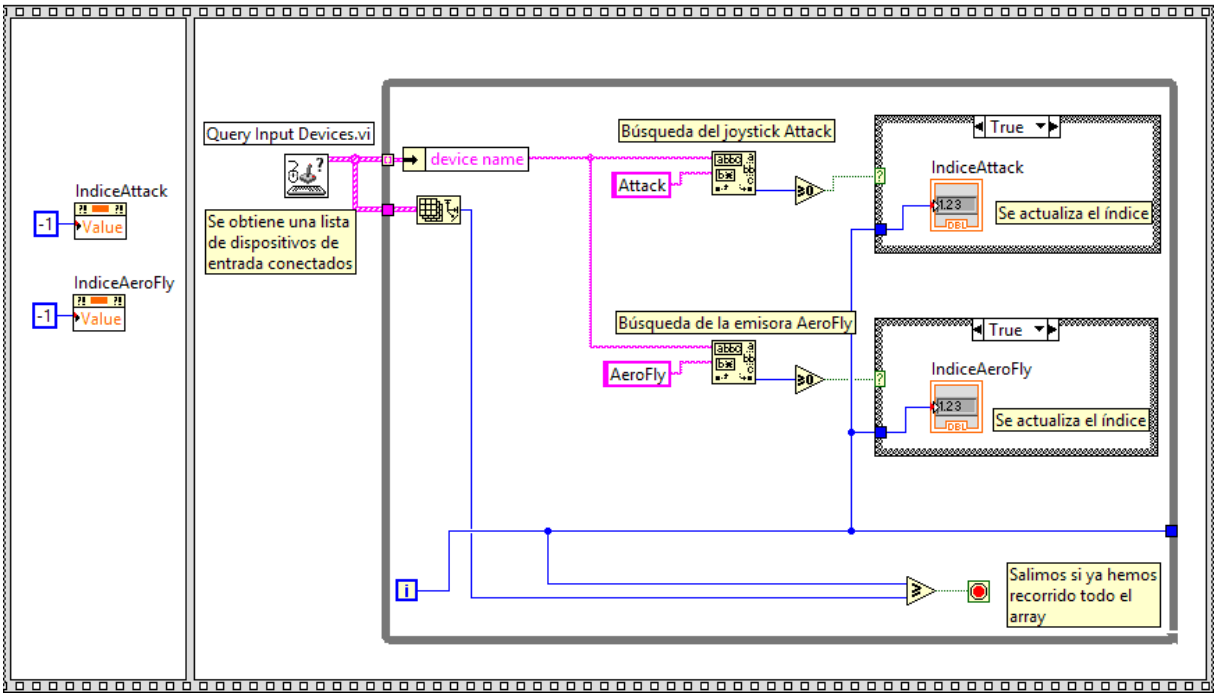
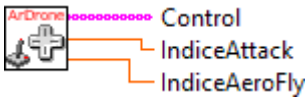


Figura B.2: Parte del funcionamiento interno del bloque BuscaJoysticks.

EstablecerControles.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.3, devuelve un array de los controles manuales encontrados junto con el control Automático. Se añade el teclado por defecto, los controles asociados al Joystick Attack 3 de Logitech si se encuentra conectado y el control de la emisora AeroFly de BSD Racing si se encuentra conectada.

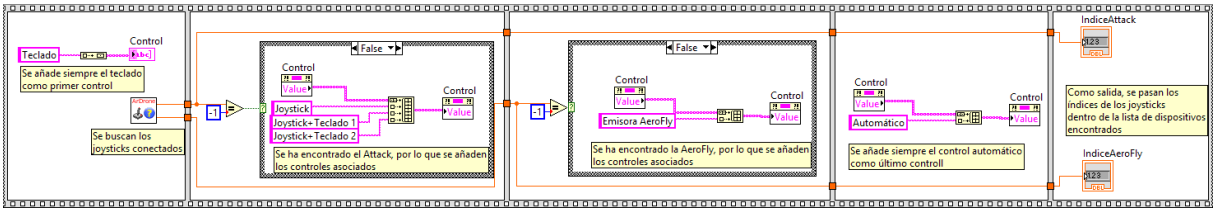
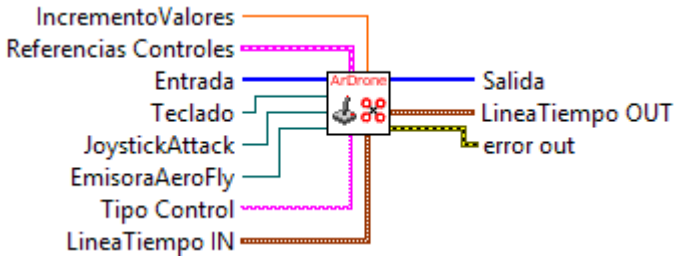


Figura B.3: Parte del funcionamiento interno del bloque EstablecerControles.

ControlesValores.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.4, realiza el sondeo del control manual seleccionado actualmente para determinar los valores de las acciones de control (multiplicadas por 10000) que el usuario ha determinado a través de pulsaciones de teclas o de pulsaciones de botones y movimientos de la palanca del joystick.

En este bloque también se gestiona la transición del modo "Automático" al modo "Manual". Existe una tecla que si se pulsa durante el modo "Automático" hace que este bloque realice el cambio al modo "Manual", estableciendo el teclado como control manual. Igualmente si se pulsa un botón determinado del joystick hace que este bloque realice el cambio al modo "Manual" estableciendo el "Joystick+Teclado 1" como control manual.

En este bloque se utilizan diversas líneas de tiempo para gestionar correctamente la pulsación de botones y teclas, de modo que no se cambie algún switch del programa principal si no ha transcurrido un tiempo desde la última vez que se cambió.

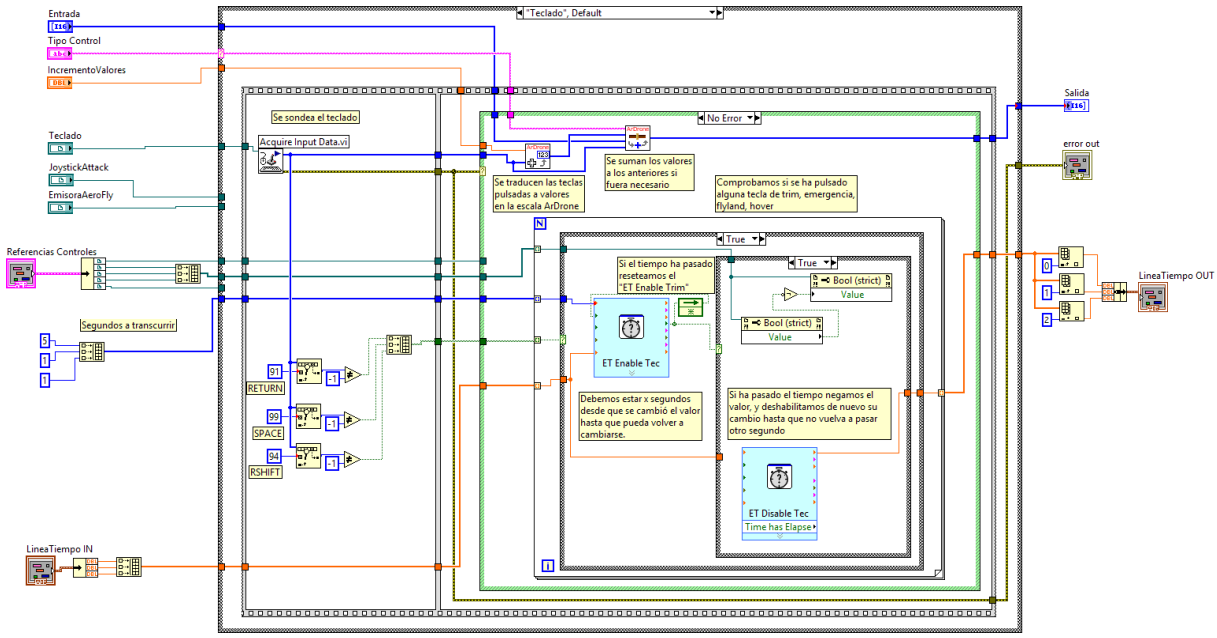
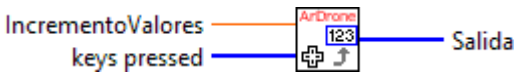


Figura B.4: Parte del funcionamiento interno del bloque ControlesValores.

TraduceTeclas.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.5, realiza la interpretación de las teclas pulsadas, incrementando o decrementando los valores de la acción de control en función de ellas.

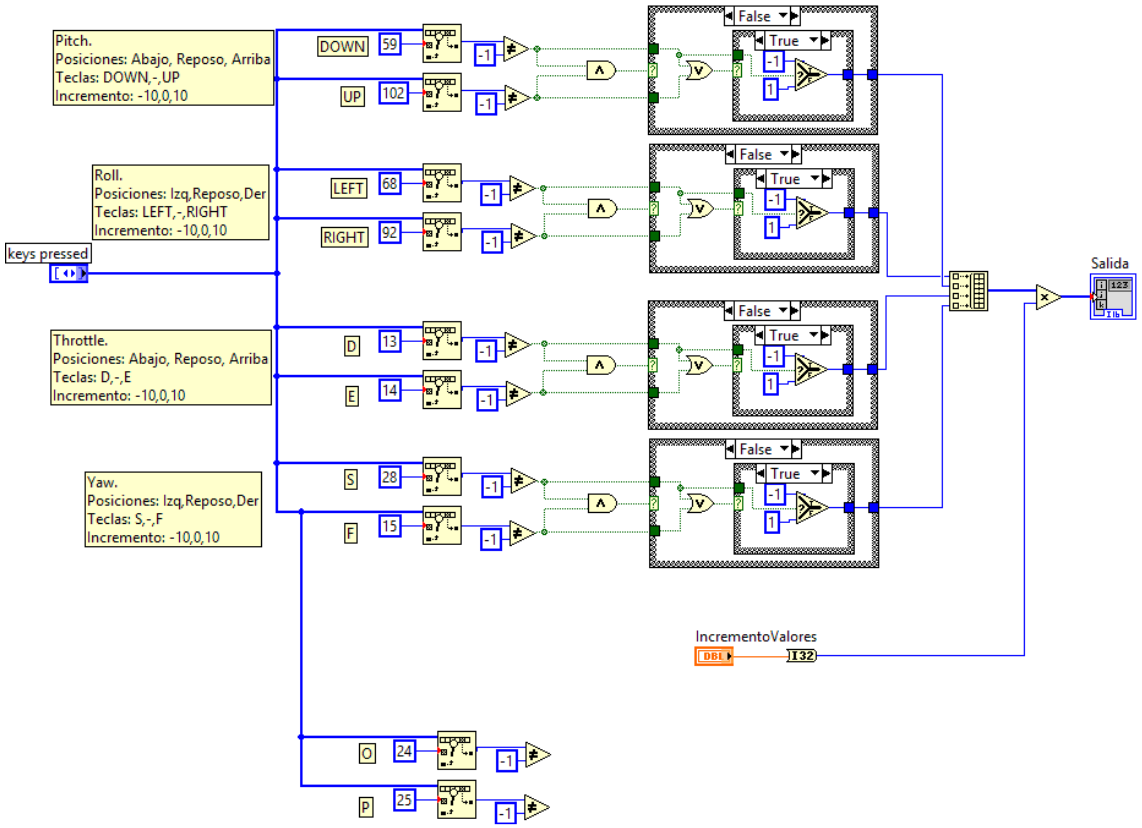
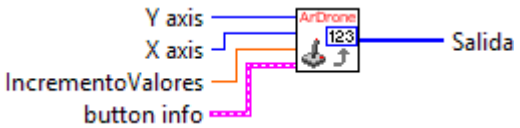


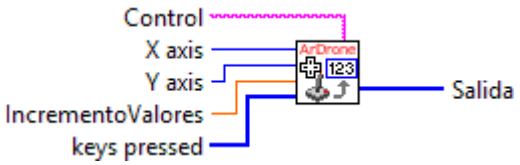
Figura B.5: Parte del funcionamiento interno del bloque TraduceTeclas.

TraduceJoystickAttack.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.6, realiza la interpretación de los botones pulsados del Joystick Attack3 de Logitech, incrementando o decrementando los valores de la acción de control en función de ellos, así como la interpretación de los movimientos realizados en los ejes del joystick para escalar su valor entre -10000 y 10000.

TraduceJoystickAttackTeclas.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.7, posibilita el uso combinado del joystick con el teclado para el control manual. De esta forma se realiza la interpretación de los botones pulsados del Joystick Attack3 de Logitech, incrementando o decrementando los valores de la acción de control en función de ellos, así como la interpretación de los movimientos realizados en los ejes del joystick para escalar su valor entre -10000 y 10000. E igualmente se realiza la interpretación de las teclas pulsadas, incrementando o decrementando los valores de la acción de control en función de ellas. Existen dos combinaciones posibles en el uso del joystick con el teclado.

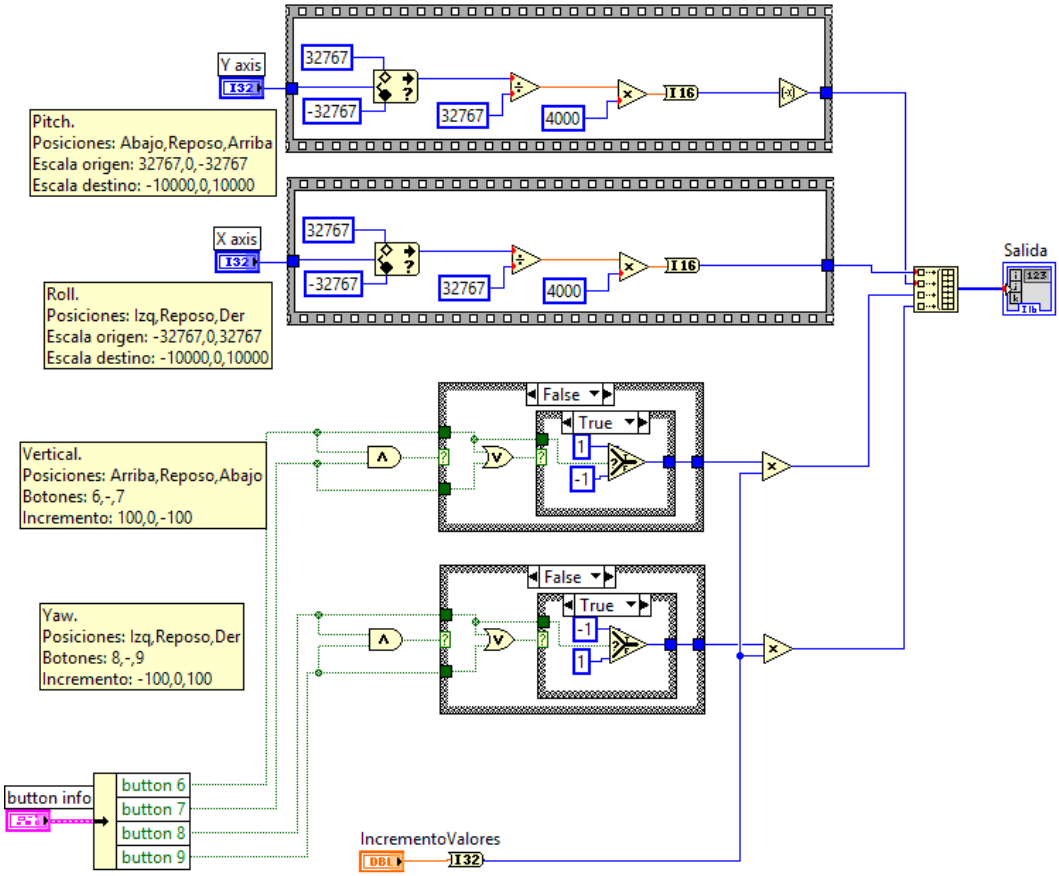


Figura B.6: Parte del funcionamiento interno del bloque TraduceJoystickAttack.

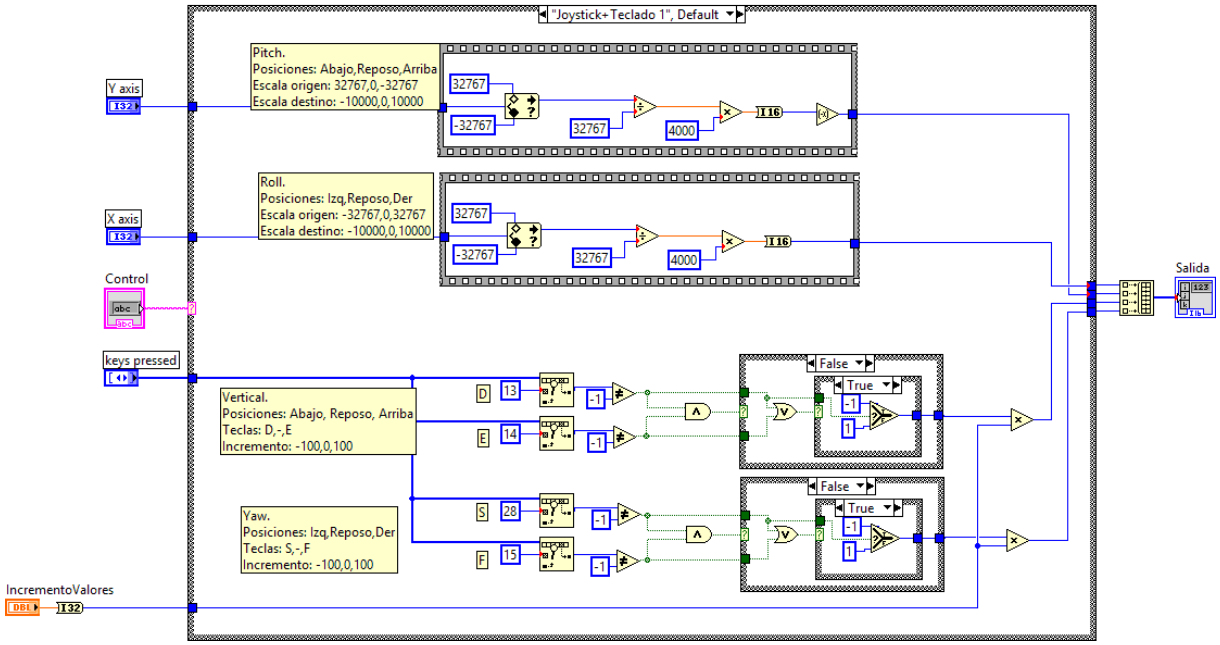
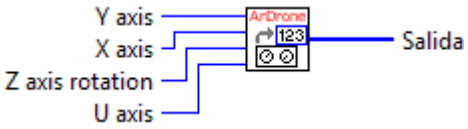


Figura B.7: Parte del funcionamiento interno del bloque TraduceJoystickAttackTeclas.

TraduceEmisoraAerofly.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.8, realiza la interpretación de los movimientos realizados en los ejes del joystick para escalar su valor entre -10000 y 10000. El driver HID de Windows, al conectar este joystick, a veces interpreta los ejes de una forma y otras veces de otra, por lo que se han configurados las dos posibles combinaciones de ejes y es seleccionable de forma interna desde este bloque.

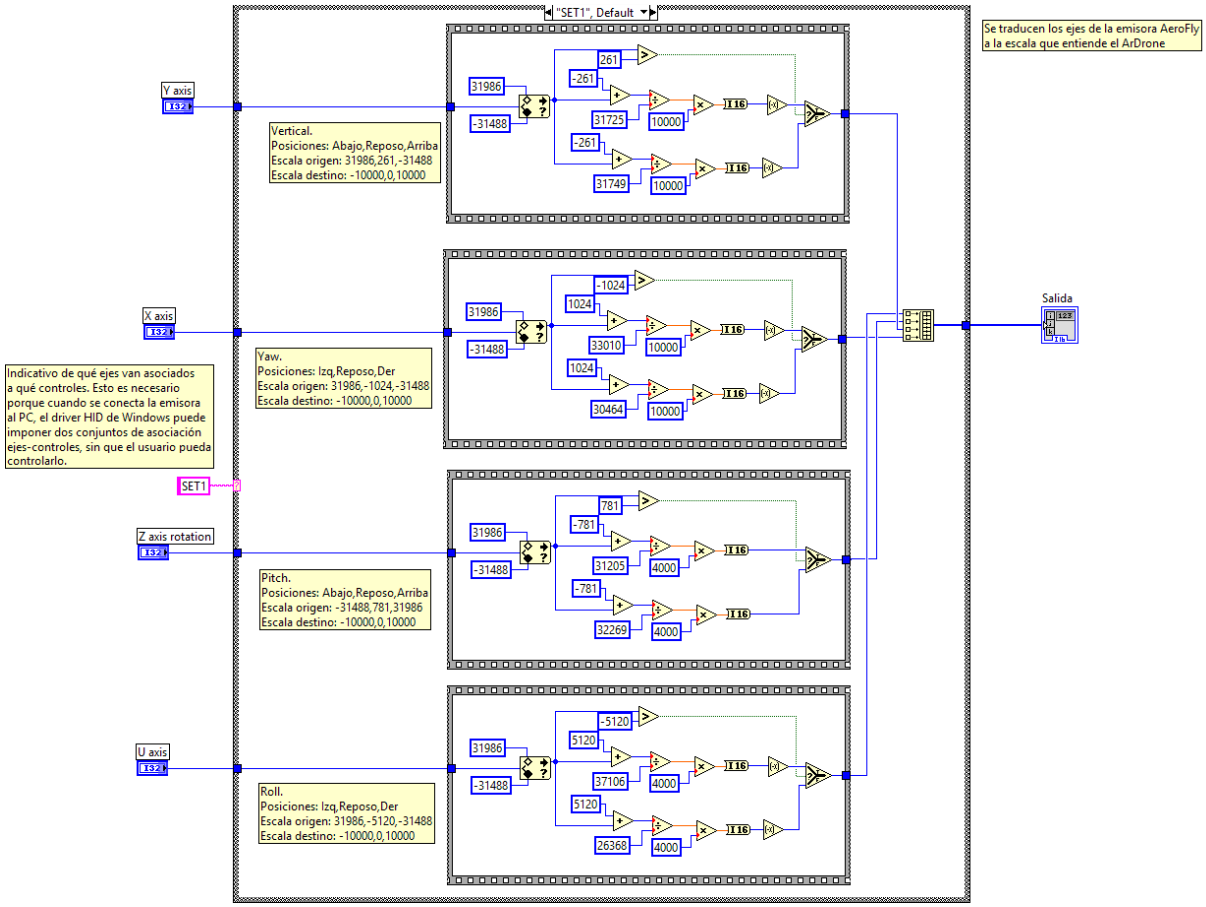
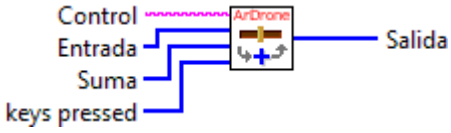


Figura B.8: Parte del funcionamiento interno del bloque TraduceEmisoraAerofly.

SumarValoresVuelo.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.9, realiza la suma de los valores de las acciones de control (multiplicadas por 10000) con los incrementos determinados por pulsación de teclas o botones. Además realiza la limitación a los valores máximos.

Por otra parte, este bloque contiene la funcionalidad para generar escalones únicamente pulsando una tecla y resulta útil cuando se necesitan realizar experimentos de identificación.

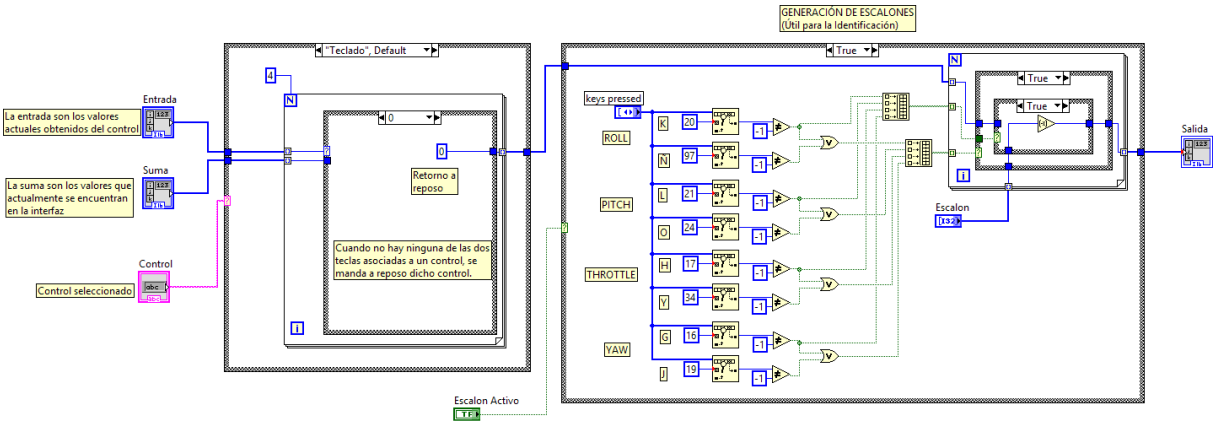


Figura B.9: Parte del funcionamiento interno del bloque SumarValoresVuelo.

B.1.3. Bloques de control automático

GestionaFasesAuto.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.10, realiza la gestión del cambio del estado del modo "Automático" cambiando los valores de ciertas variables globales relacionadas con el control autónomo.

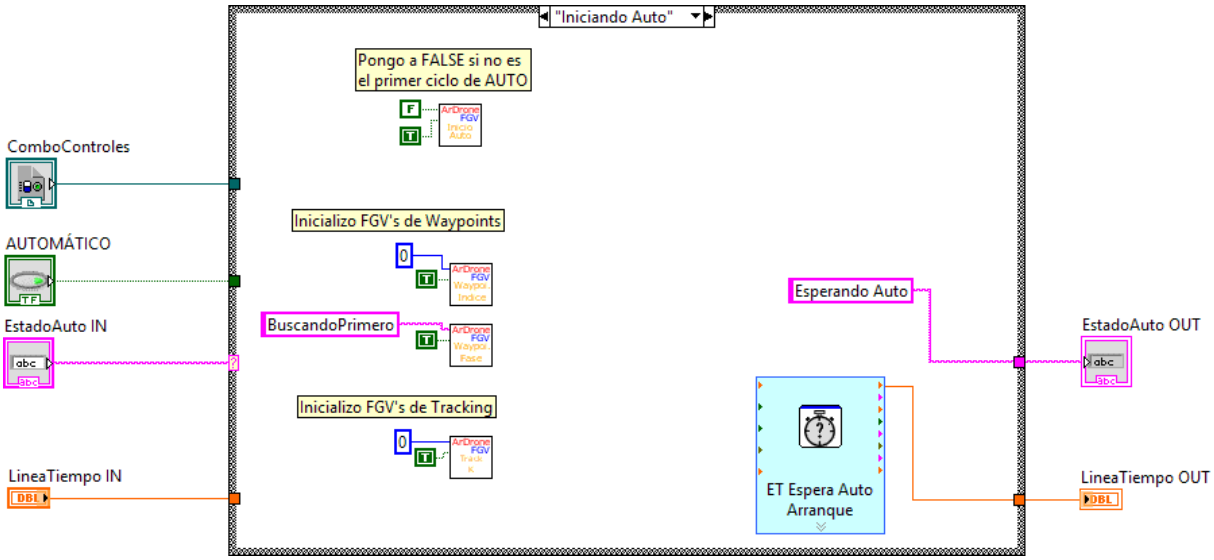
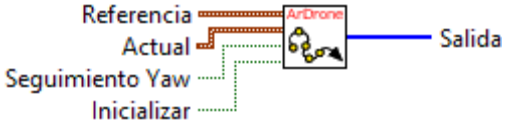


Figura B.10: Parte del funcionamiento interno del bloque GestionaFasesAuto.

ControlAutomatico.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.11, realiza el cálculo del error entre la referencia y la posición real o estimada, hace la transformación de dichos errores del marco de

referencia E al marco de referencia B, y los utiliza como entrada para los controladores que se encuentran implementados en este bloque. La salida del bloque es la acción de control resultado de los controladores pero multiplicada por 10000. Este bloque dispone de una entrada de inicialización que resetea los estados de los controladores implementados en él.

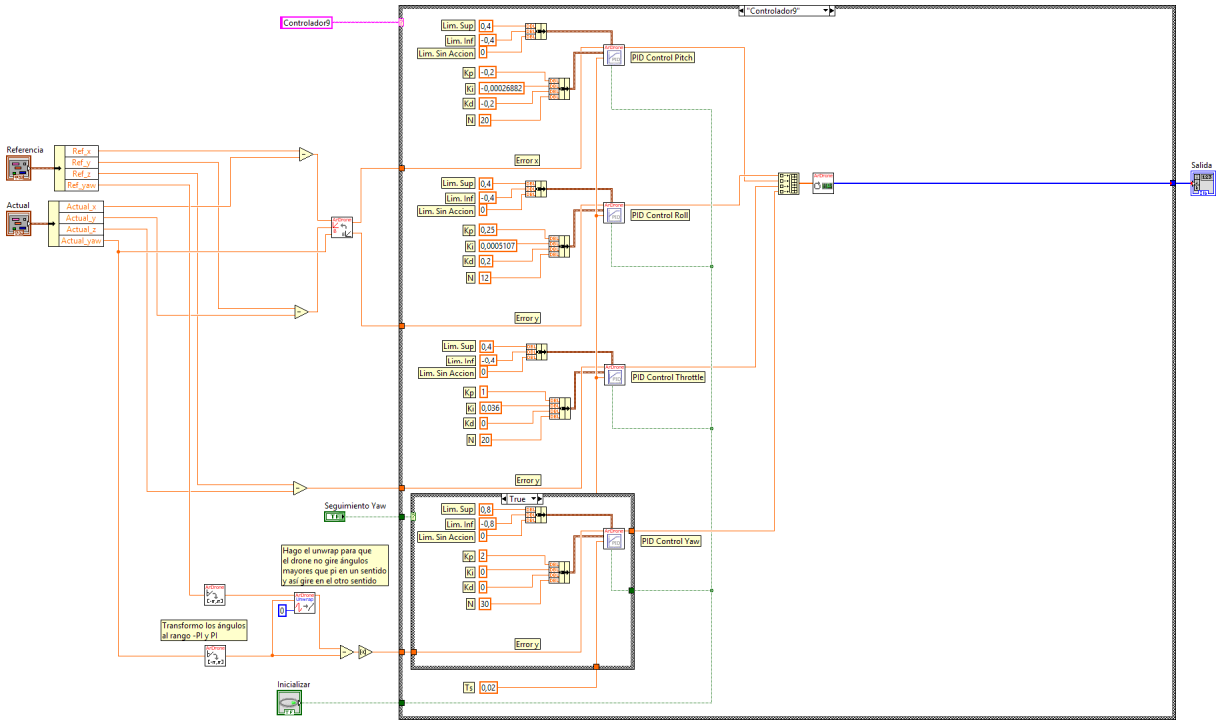
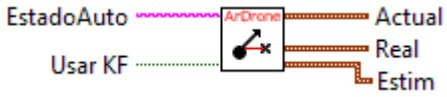


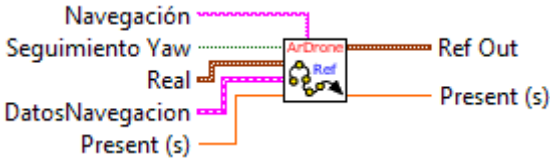
Figura B.11: Parte del funcionamiento interno del bloque ControlAutomatico.

ObtenerPosicionActual.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.12, proporciona la posición y orientación obtenida directamente de los sensores en su salida "Real", mientras que en su salida "Estim" proporciona la posición y orientación tras ejecutar el Filtro de Kalman. La salida "Actual" es igual a la salida "Real" o a la salida "Estim" en función de si está activado el switch "Usar KF". Este bloque también actualiza la variable global de Medidas que es utilizada en otras partes del programa y que puede utilizarse en los controladores.

ObtenerReferenciaNavegacion.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.13, obtiene la referencia de posición y orientación que el cuadricóptero debe seguir en el instante actual. Para ello utiliza los parámetros configurados en la pestaña de "Navegación" de la pantalla principal en función del tipo de navegación seleccionado. La posición Real se utiliza para determinar si el Ar.Drone 2.0 ha llegado al punto de referencia destino, mientras que la línea de tiempo se destina para diferentes funcionalidades como la permanencia en un

Waypoint o el tiempo t a utilizar en las ecuaciones de seguimiento de trayectorias. En este sentido, se han utilizado los bloques de parseo de fórmulas que dispone LabView para interpretar las ecuaciones.

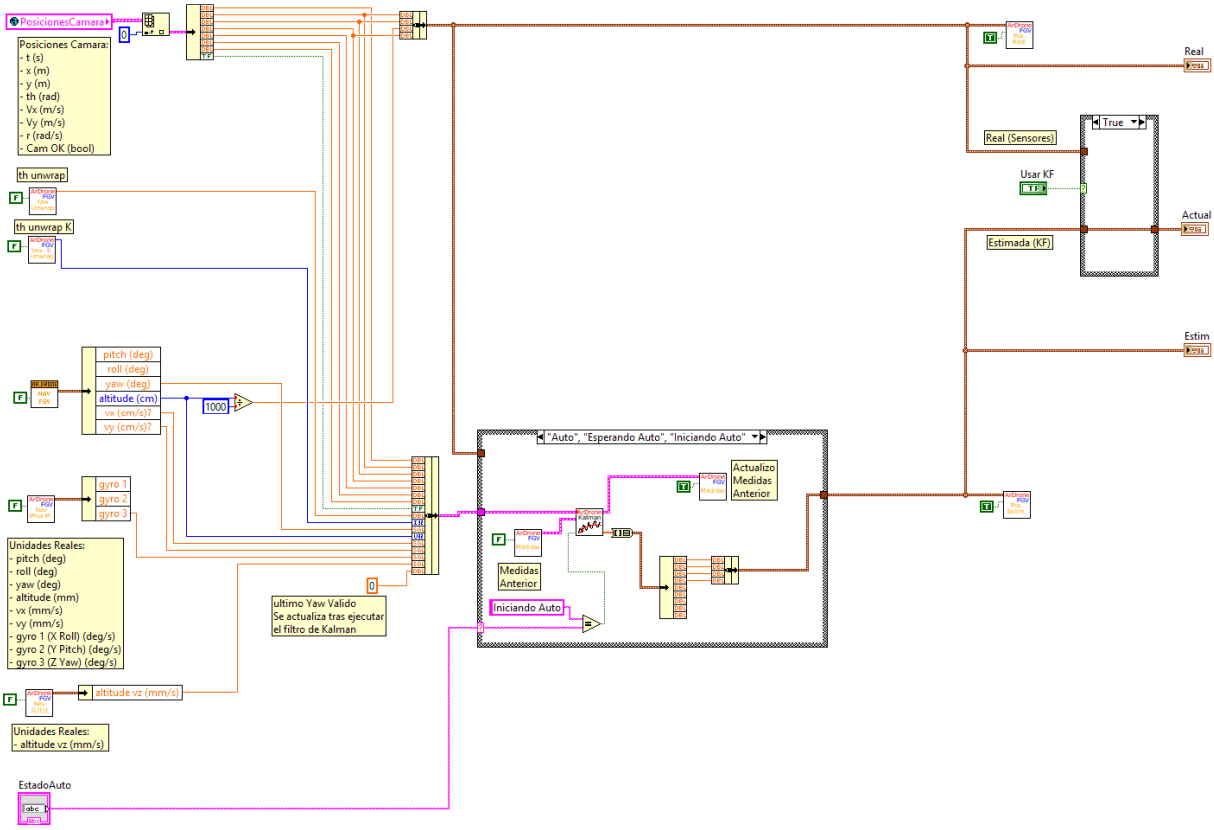


Figura B.12: Parte del funcionamiento interno del bloque ObtenerPosicionActual.

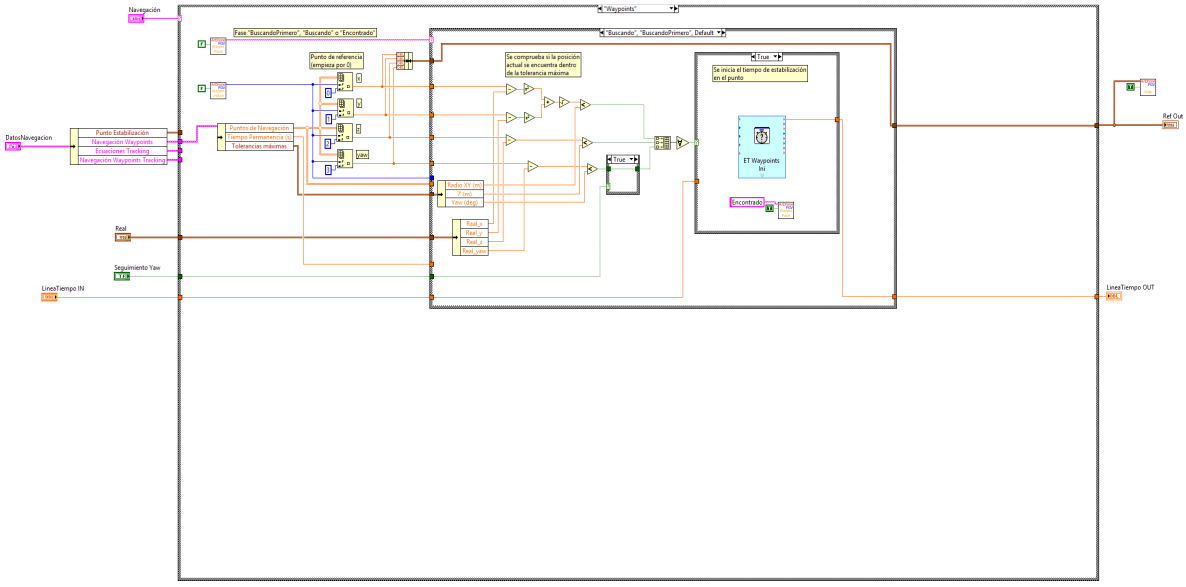
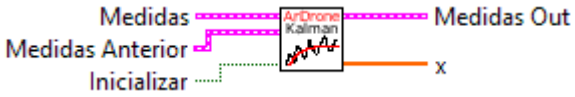


Figura B.13: Parte del funcionamiento interno del bloque ObtenerReferenciaNavegacion.

FiltroKalman.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.14, realiza la ejecución de una iteración del Filtro de Kalman. A través de los datos necesarios obtenidos de las medidas, y de las matrices y vectores constantes necesarios, calcula gracias a un algoritmo MathScript la estimación del estado del AR.Drone 2.0. Este bloque dispone de una entrada de inicialización para resetear el estado del Filtro de Kalman.

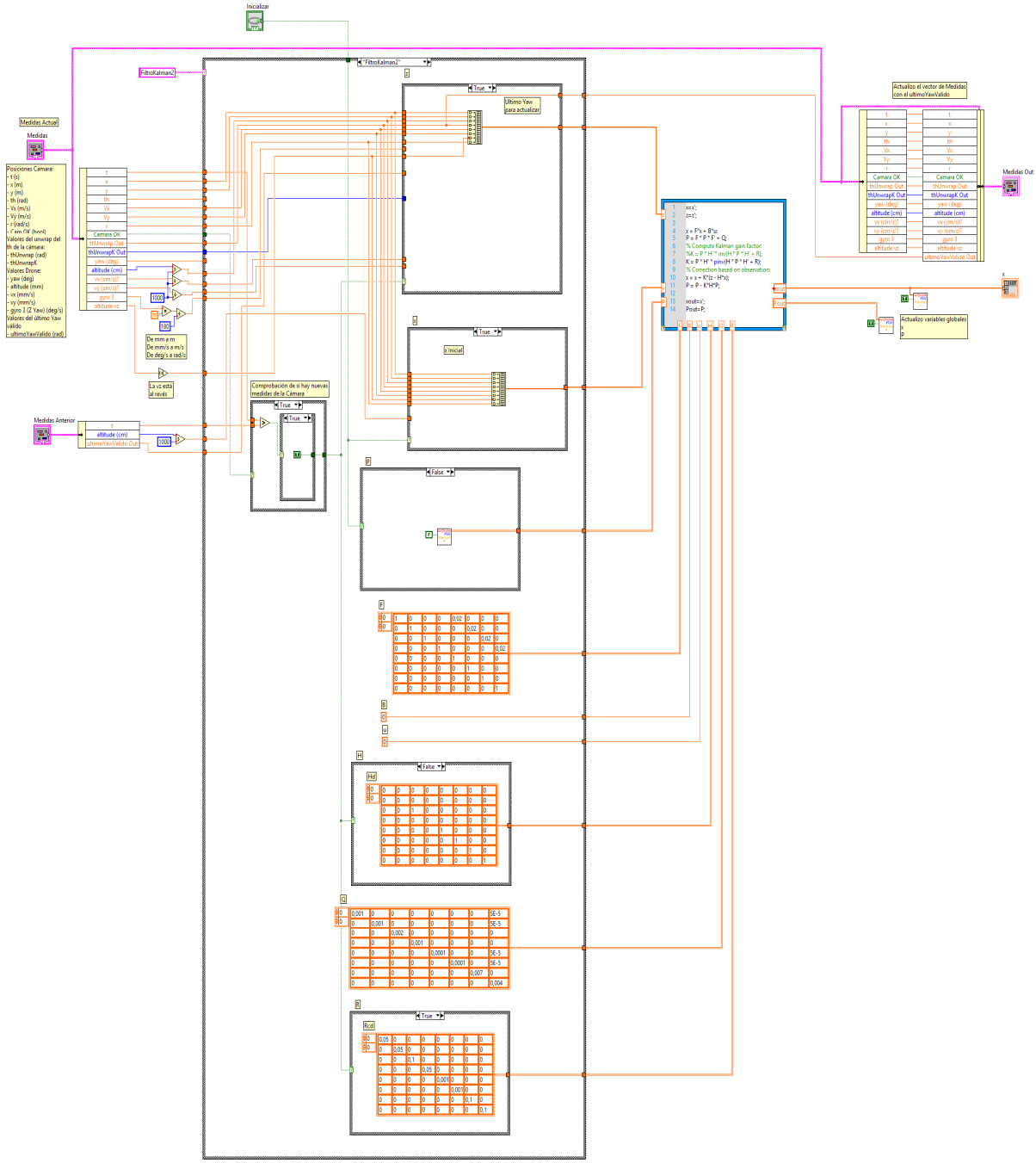
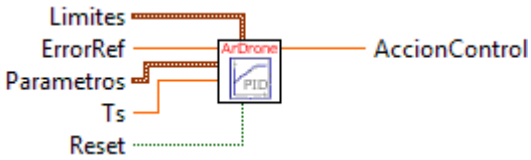


Figura B.14: Parte del funcionamiento interno del bloque FiltroKalman.

PIDBloque.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.15, hace las funciones de controlador PID discreto parametrizable que puede utilizarse en el control automático. Este bloque mantiene sus variables de estado independientes de otras instancias de este mismo bloque. Dispone de entradas para establecer los parámetros de las ganancias y de la constante N del filtro derivativo, así como de limitación superior e inferior de la acción de control, limitación de la acción de control en caso de ser menor que un determinado valor y el tiempo de muestreo. Así mismo dispone de una entrada para inicializar su estado interno. En este bloque se utilizan funcionalidades del módulo *Control Design and Simulation Module* de LabView para establecer, de forma parametrizada, las funciones de transferencia de la parte Integral y de la parte Derivativa.

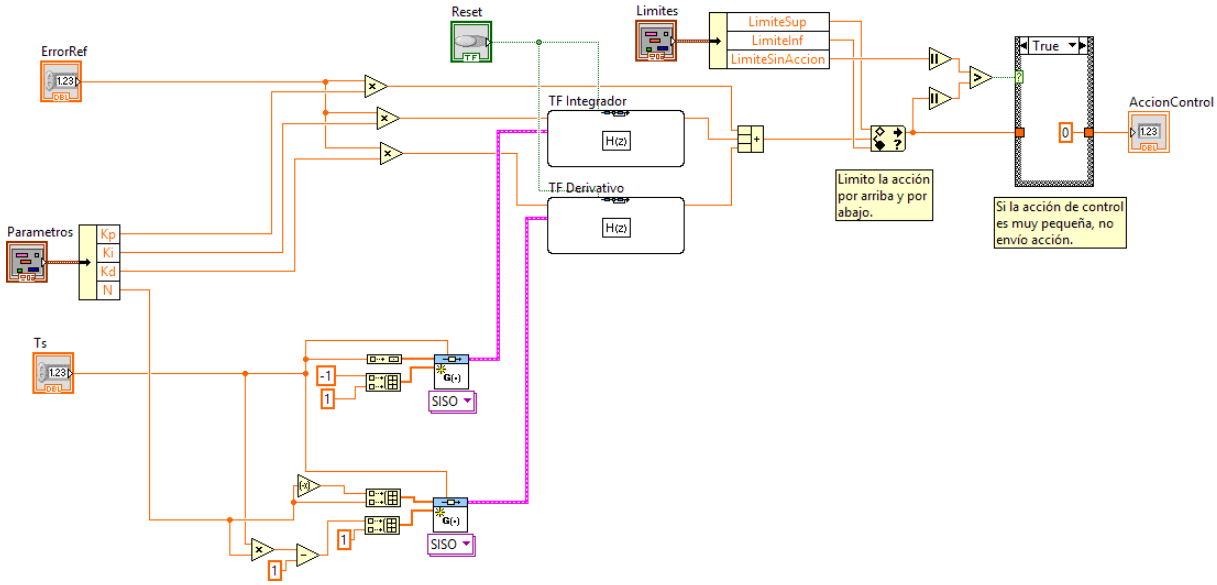


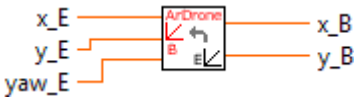
Figura B.15: Parte del funcionamiento interno del bloque PIDBloque.

TransformaBE.vi



Este bloque, del que se muestra su funcionamiento interno en la Figura B.16, realiza la transformación de la posición o la velocidad lineal del marco de referencia B al marco de referencia E.

TransformaEB.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.17, realiza la transformación de la posición o la velocidad lineal del marco de referencia E al marco de referencia B.

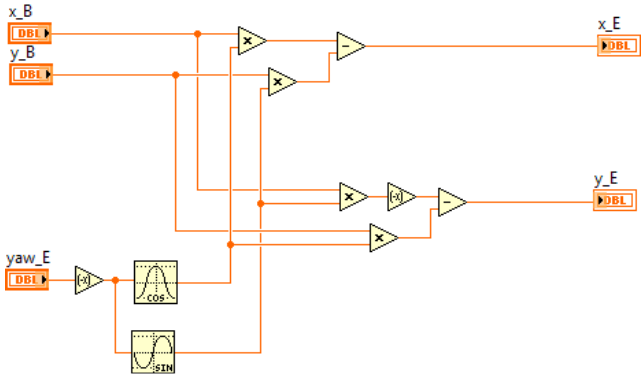


Figura B.16: Funcionamiento interno del bloque TransformaBE.

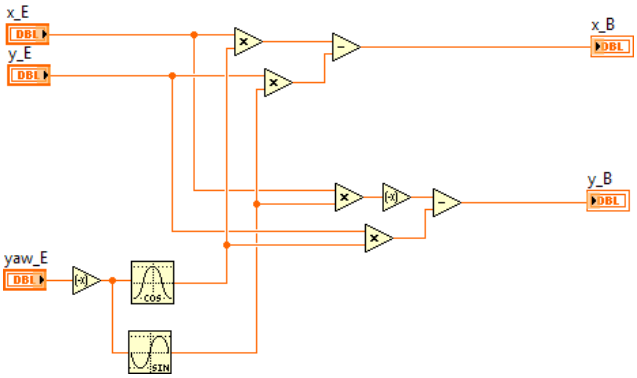
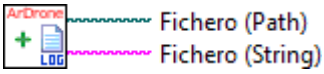


Figura B.17: Funcionamiento interno del bloque TransformaEB.

B.1.4. Bloques de logs

CrearLog.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.18, realiza la creación del fichero de log cuyo nombre contiene la fecha y hora con minutos y segundos, añadiéndole la primera línea que con los títulos de los diferentes campos separados por tabuladores.

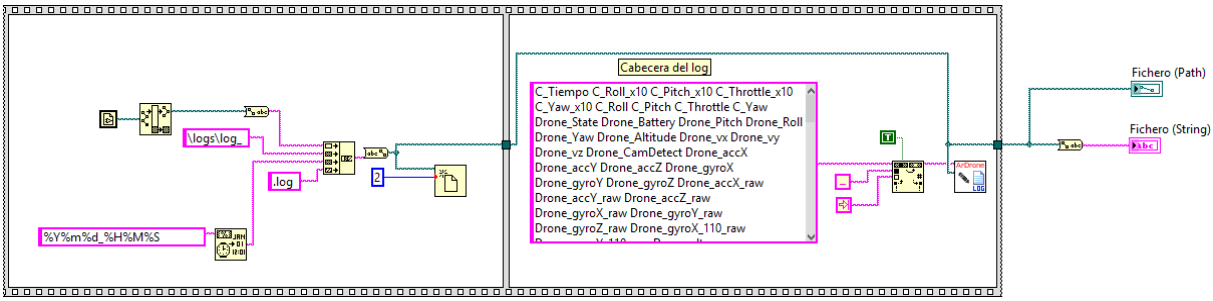
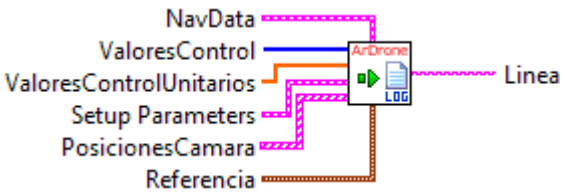


Figura B.18: Parte del funcionamiento interno del bloque CrearLog.

FormarLineaLog.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.19, crear una línea con todos los datos a registrar en el log en el instante de tiempo actual.

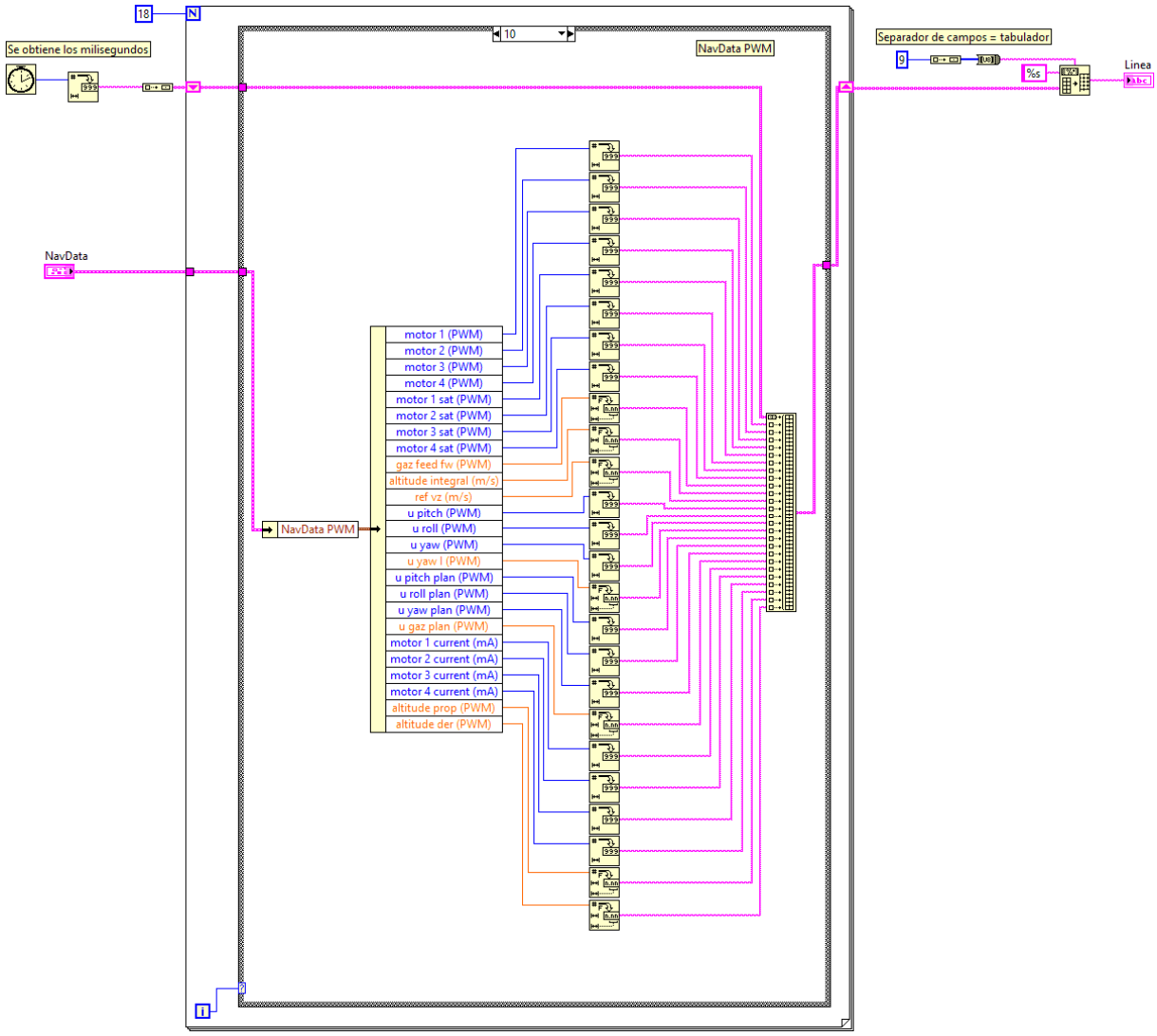


Figura B.19: Parte del funcionamiento interno del bloque FormarLineaLog.

EscribirLog.vi



Este bloque, del que se muestra su funcionamiento interno en la Figura B.20, realiza la escritura de un conjunto de líneas justo al final del contenido del fichero de log.

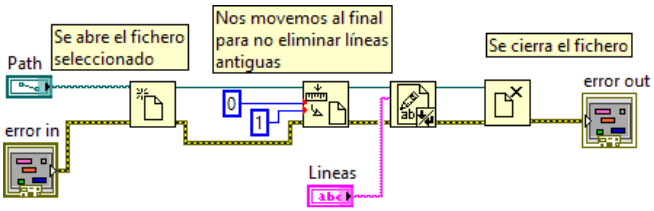


Figura B.20: Funcionamiento interno del bloque EscribirLog.

SeleccionarLogFileDialog.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.21, no se utiliza actualmente en este proyecto pero se ha dejado por si fuera de utilidad en un futuro. Con este boque se selecciona el nombre y el directorio para el fichero de log.

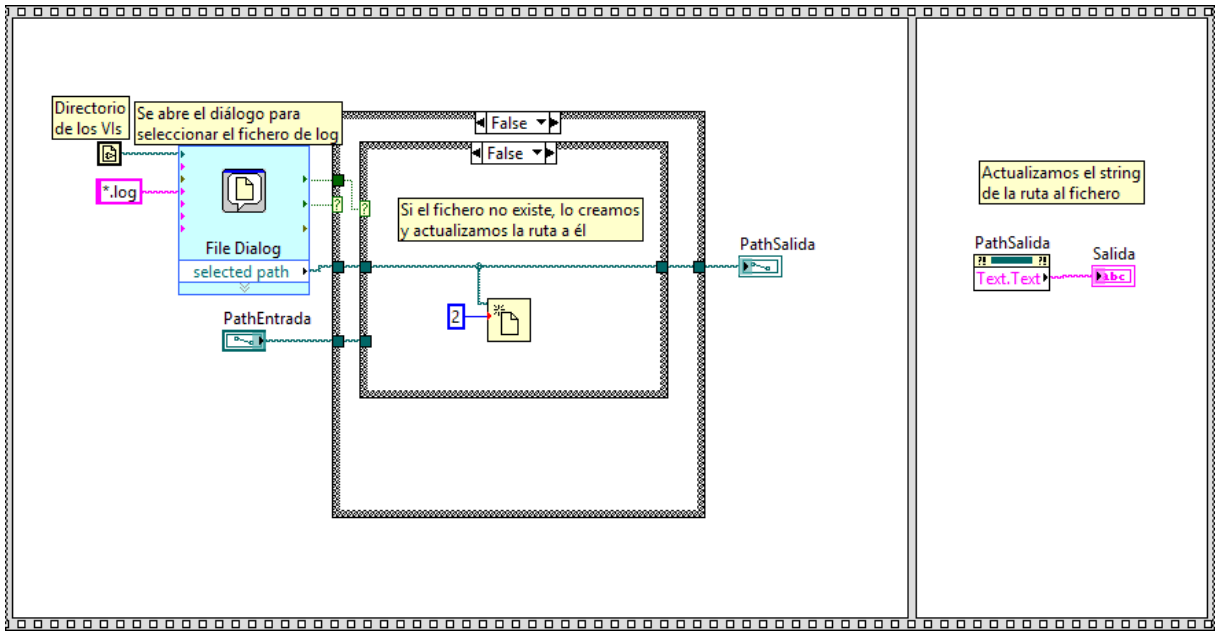


Figura B.21: Parte del funcionamiento interno del bloque SeleccionarFileDialog.

B.1.5. Bloques de pruebas

PruebaControles.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.22, permite realizar pruebas de los diferentes controles conectados al PC. Con él se puede verificar que las teclas pulsadas y los ejes movidos son interpretados correctamente. Igualmente es útil cuando se necesita incorporar un nuevo joystick para el control manual. Su implementación interna es, en esencia, similar a la implementación en el bloque Principal de la interpretación de los controles.

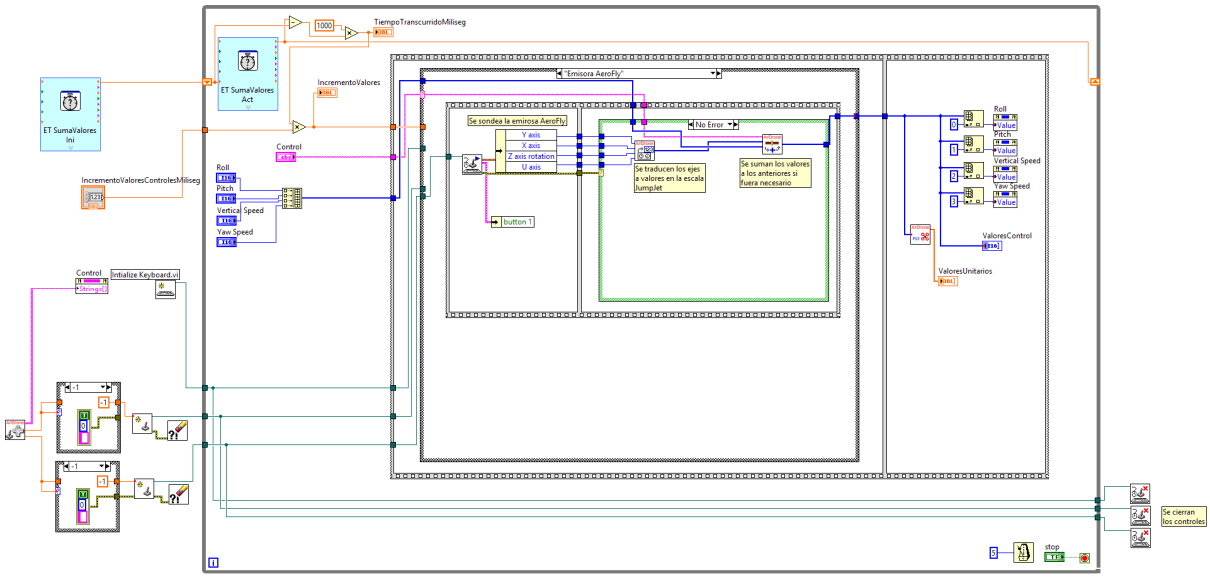


Figura B.22: Parte del funcionamiento interno del bloque PruebaControles.

PruebaTeclado.vi



Este bloque, del que se muestra su funcionamiento interno en la Figura B.23, permite realizar pruebas del teclado con el objetivo de determinar los valores internos de las teclas pulsadas, por si fuera necesario implementar nuevas funcionalidades que usen el teclado.

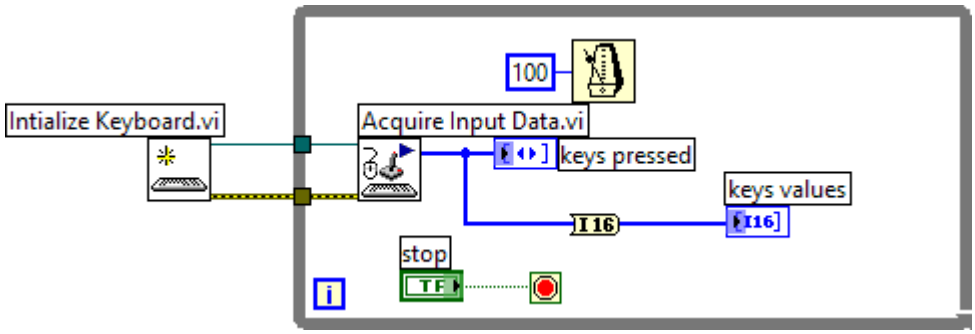


Figura B.23: Funcionamiento interno del bloque PruebaTeclado.

PruebaJoystick.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.24, permite realizar pruebas de los joysticks conectados en el PC con el objetivo de determinar los números internos de botón cuando se pulsa uno de ellos en el joystick, así como para conocer los valores máximos y mínimos de cada eje. Es útil cuando se necesita incorporar un nuevo joystick para el control manual.

En el funcionamiento interno de este bloque se obtienen las referencias de todos los joysticks conectados al PC, y cuando se selecciona uno de ellos y se comienza a probar, el bloque reacciona a cualquier movimiento o pulsación en el joystick y muestra por pantalla estas reacciones.

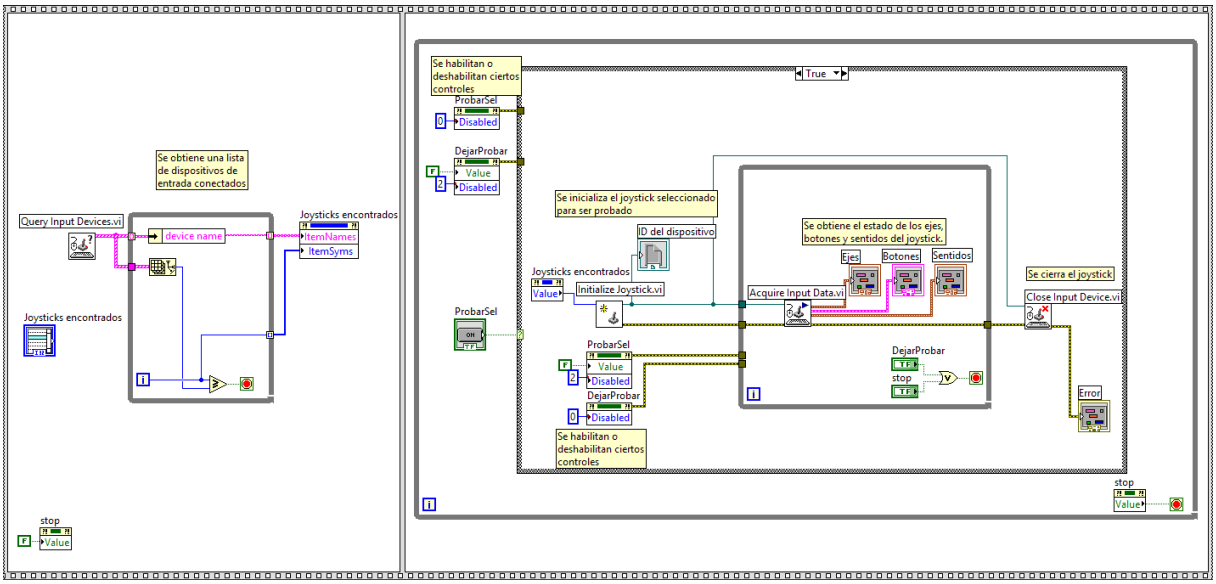


Figura B.24: Parte del funcionamiento interno del bloque PruebaJoystick.

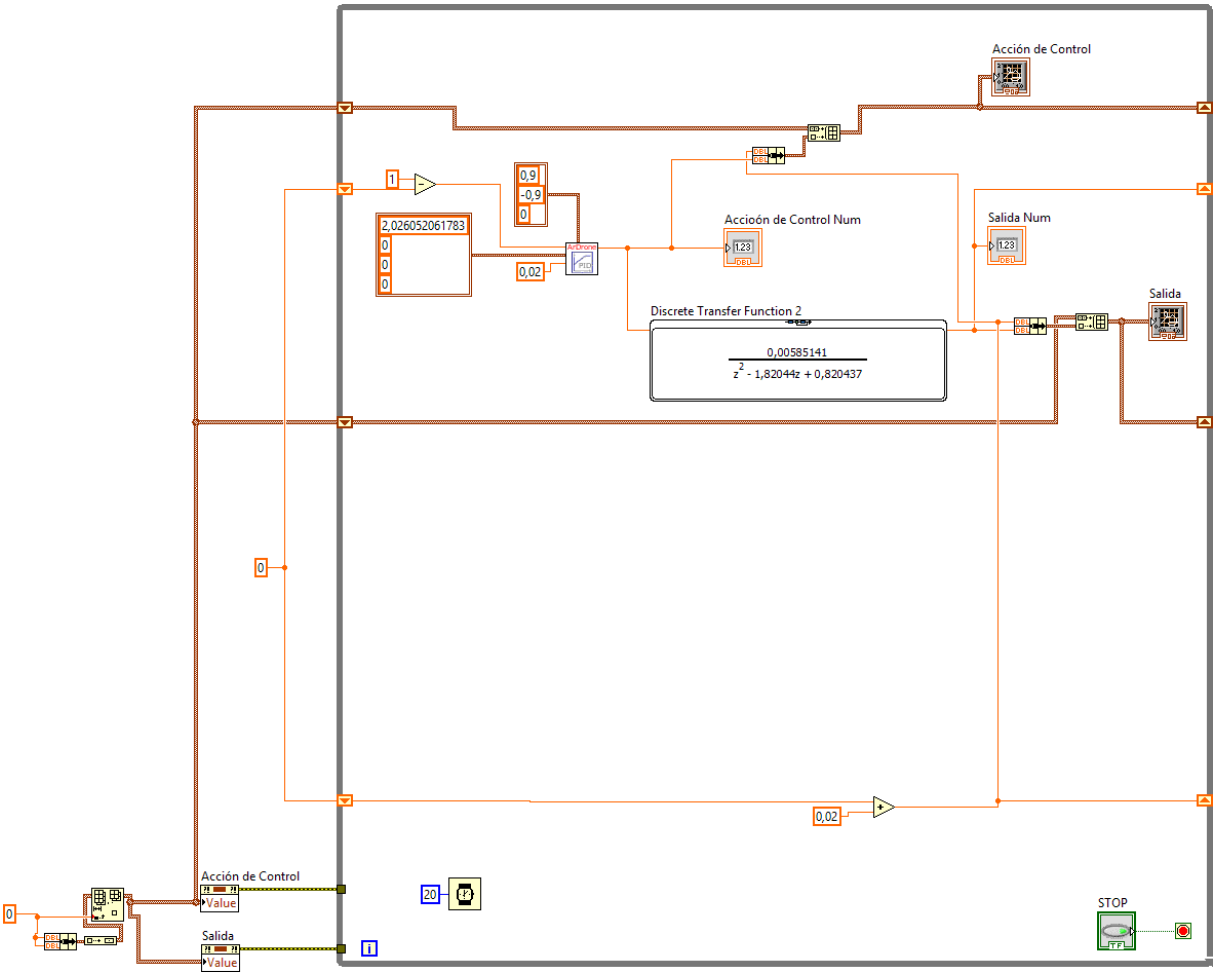


Figura B.25: Funcionamiento interno del bloque PruebaPID.

PruebaPID.vi

Este bloque, del que se muestra su funcionamiento interno en la Figura B.25, permite realizar pruebas de un controlador PID implementado en su diagrama de bloques y verificar que la respuesta temporal y la acción de control son las esperadas.

Como se puede ver, se compone básicamente de un modelo implementado como una función de transferencia discreta y de un controlador PID parametrizado.

PruebaPIDMulti.vi

Este bloque, del que se muestra su funcionamiento interno en la Figura B.26, permite realizar pruebas de varios controladores PID implementados en su diagrama de bloques y verificar que las respuestas temporales y las acciones de control son las esperadas.

Como se puede ver, internamente está formado por una parte donde se encuentra definido el modelo del cuadricóptero y otra parte en la que se encuentra definido el controlador para dicho modelo. Pueden probarse estrategias básicas de control o, como se ve en la figura, estrategias más complejas (multivariable).

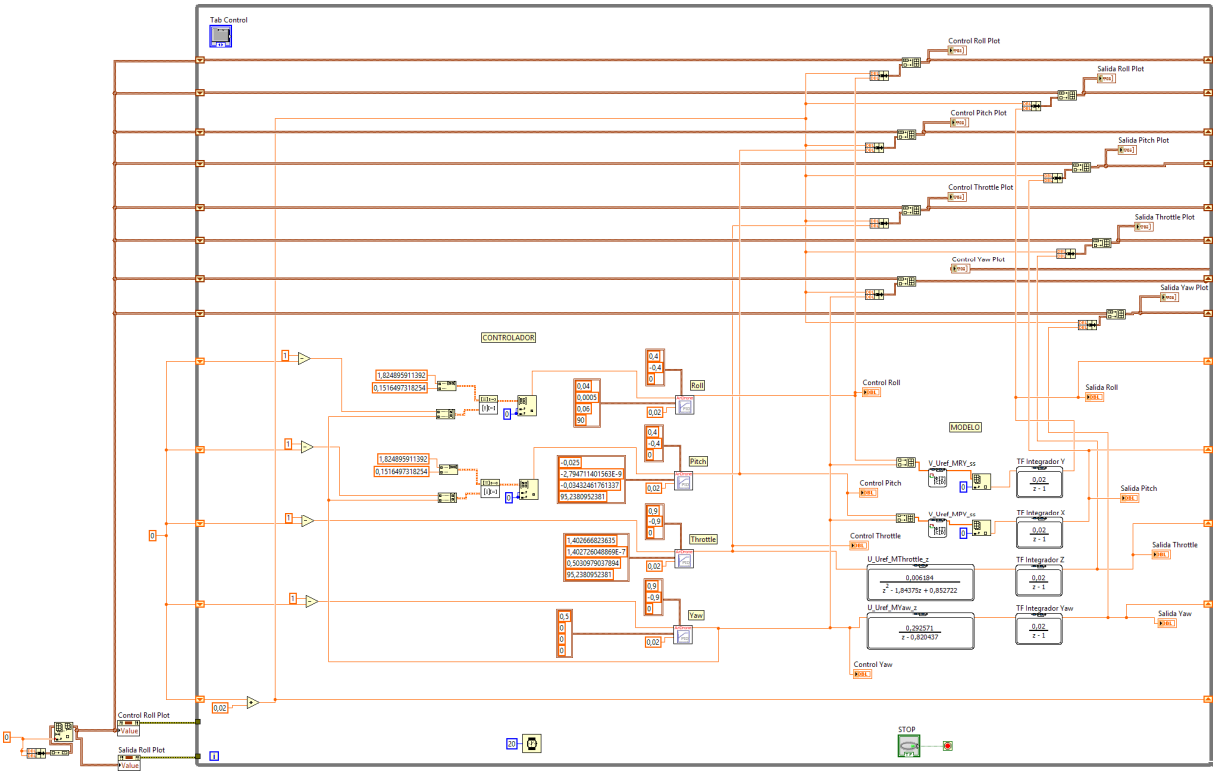


Figura B.26: Panel frontal del bloque PruebaPIDMulti.

PruebaTrayectorias.vi

Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.27, permite realizar pruebas de trayectorias definidas por ecuaciones en función de t y de k, tal y como se encuentran en la pestaña de

navegación de "Tracking". De esta forma se verifica su buen funcionamiento de forma previa a su incorporación en un control autónomo real.

Su implementación interna es, en esencia, similar a la del modo de navegación "Tracking" del sistema principal de la plataforma.

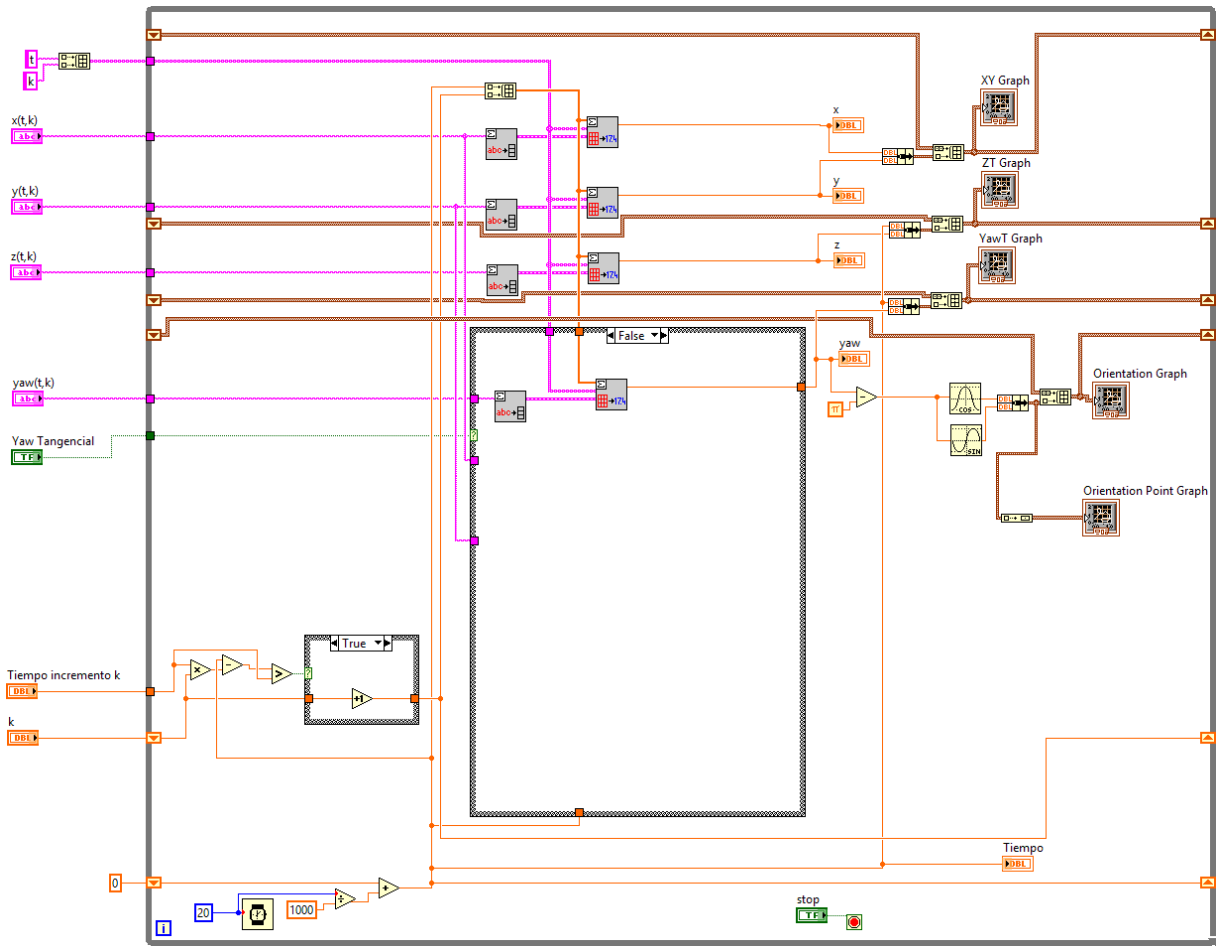


Figura B.27: Parte del funcionamiento interno del bloque PruebaTrayectorias.

B.1.6. Bloques auxiliares

AdecuarSentidos.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.28, realiza la adecuación de los valores de las acciones de control utilizadas en el programa (multiplicadas por 10000) a los valores de las acciones de control tal y como los espera el AR.Drone 2.0. También se adecuan los sentidos con la misma finalidad.

AdecuarSentidosInverso.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.29, realiza la adecuación de los valores de las acciones de control tal y como los interpreta el AR.Drone a los valores utilizados en el programa (multiplicados por 10000). También se adecuan los sentidos con la misma finalidad.

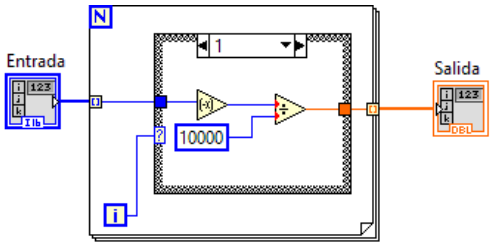


Figura B.28: Parte del funcionamiento interno del bloque AdecuarSentidos.

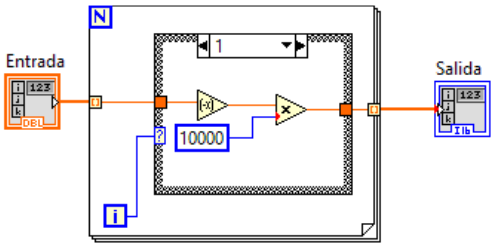
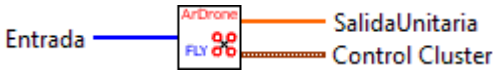


Figura B.29: Parte del funcionamiento interno del bloque AdecuarSentidosInverso.

DatosEnvioVuelo.vi



Este bloque, del que se muestra su funcionamiento interno en la Figura B.30, adecúa los valores de las acciones de control y los agrupa en la forma de cluster que necesita el bloque de vuelo del AR.Drone 2.0.

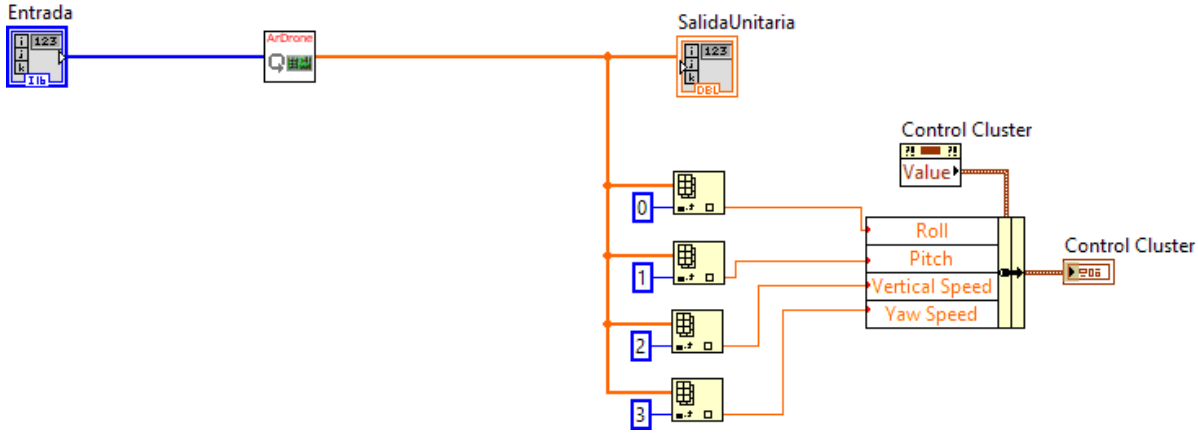


Figura B.30: Funcionamiento interno del bloque DatosEnvioVuelo.

DibujaGraficas.vi



Este bloque, del que se muestra su funcionamiento interno en la Figura B.31, obtiene los datos del instante actual y los prepara en las diferentes estructuras que las gráficas necesitan para mostrar el seguimiento de la posición y la orientación.

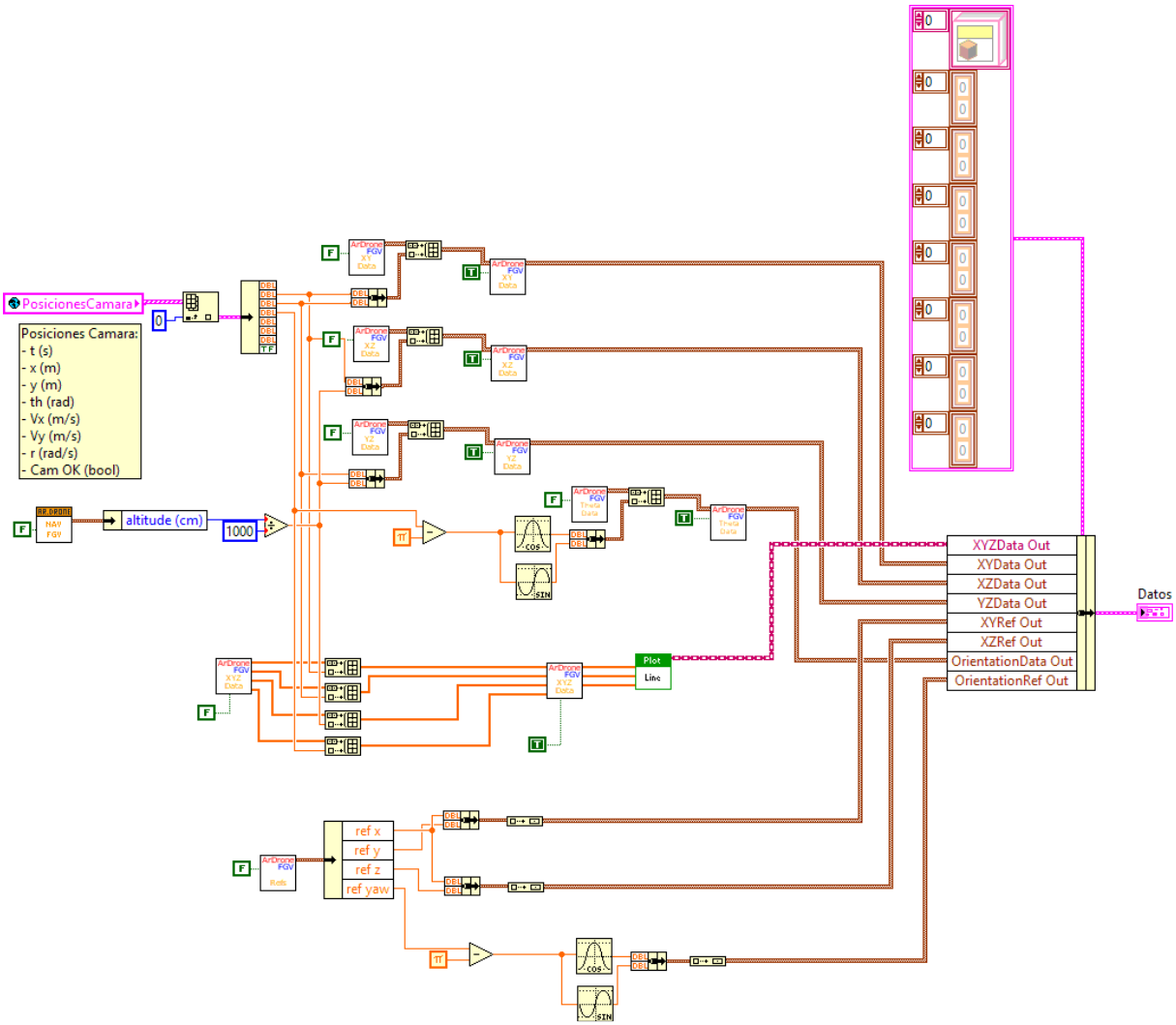


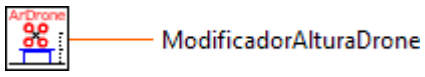
Figura B.31: Funcionamiento interno del bloque DibujaGraficas.

GestionarCalibracion.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.32, realiza la gestión de la pulsación del botón "CALIBRA" y que hace una calibración del magnetómetro del AR.Drone 2.0. Este bloque simplemente limita el tiempo que transcurre desde que se pulsa dicho botón hasta que puede volverse a pulsar. De esta forma se envía la orden de calibración de forma correcta.

ModificadorAlturaDroneCamara.vi



Este bloque, del que se muestra su funcionamiento interno en la Figura B.33, establece el modificador de altura que se utiliza en un bloque del sistema de visión para determinar la posición real según la perspectiva de la cámara. Esto es porque la posición x e y que determina la cámara está influenciada por el plano h en el que se encuentra el triángulo de leds.

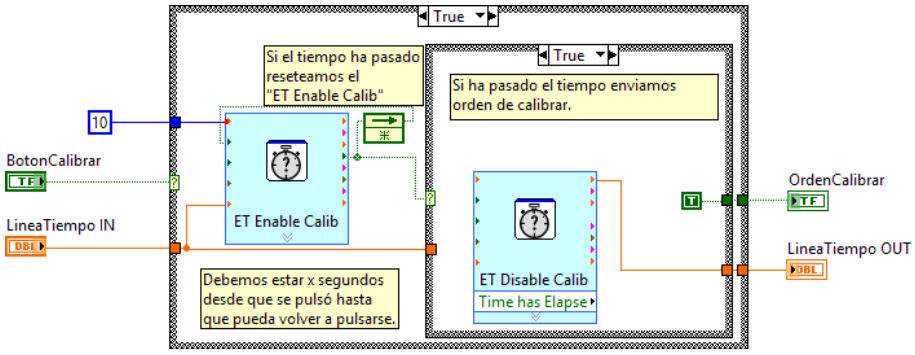


Figura B.32: Parte del funcionamiento interno del bloque GestionarCalibracion.

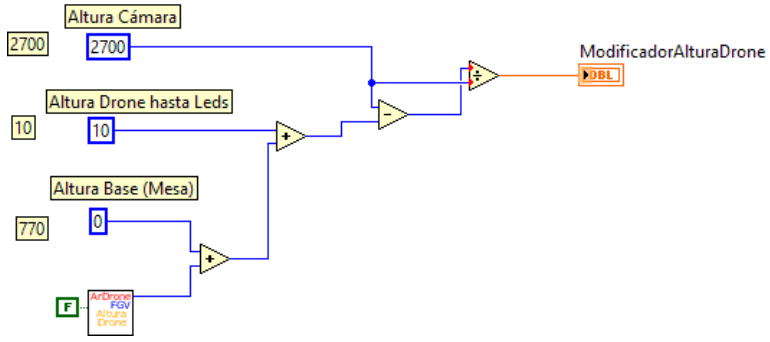


Figura B.33: Funcionamiento interno del bloque ModificadorAlturaDroneCamara.

TransformaAnguloRangoPI.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.34, realiza la transformación de un ángulo al rango $[-\pi, \pi]$ sumando o restando de forma iterativa 2π hasta que se encuentre en dicho rango.

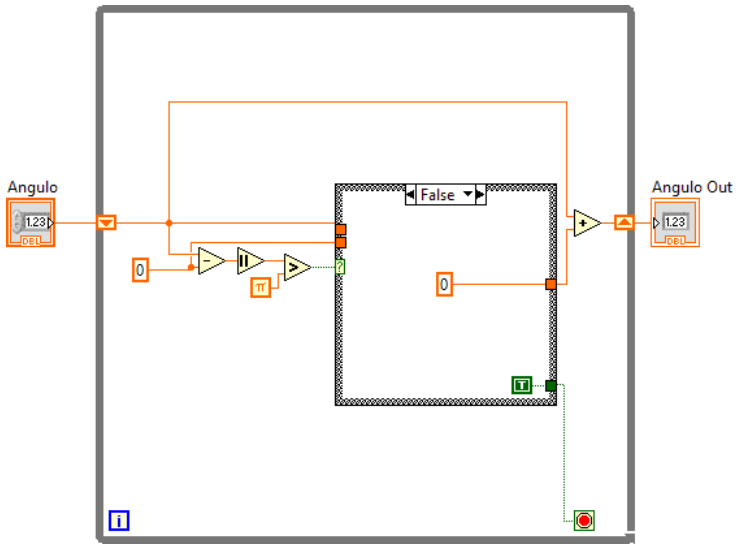


Figura B.34: Parte del funcionamiento interno del bloque TransformaAnguloRangoPI.

Unwrap.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.35, cambia el valor de un ángulo para evitar discontinuidades dado que la orientación $-\pi$ es equivalente a la orientación π.

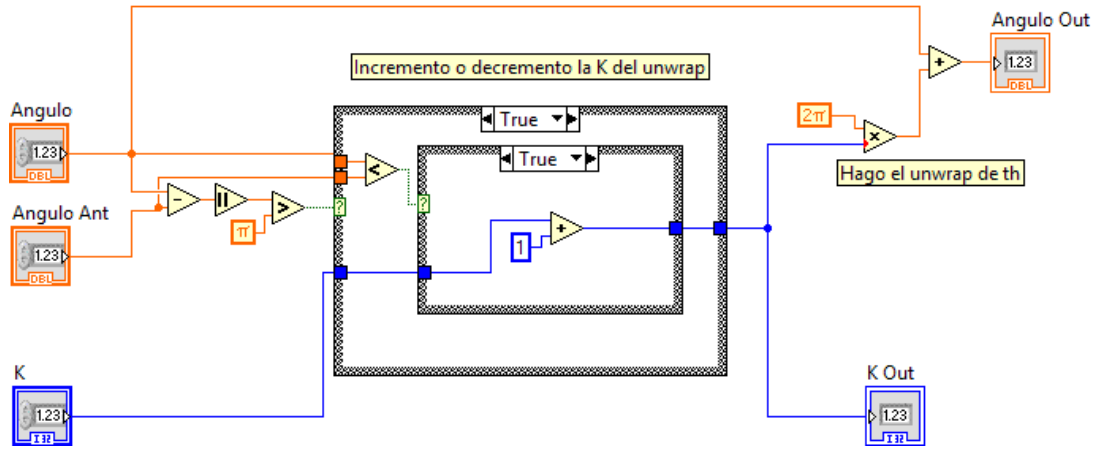


Figura B.35: Parte del funcionamiento interno del bloque Unwrap.

B.2. Descripción de los bloques del Sistema de Visión

El software del sistema de visión se encuentra desarrollado en LabVIEW utilizando las funcionalidades que proporciona el módulo *NI Vision Development Module* y ha sido creado en base a lo indicado en el ANEXO A. Para adecuarlo al presente proyecto se han hecho unas pequeñas modificaciones que se describen a continuación en cada uno de los bloques que haya sido modificado.

Camara.vi



Este bloque, del que se muestra su panel frontal en la Figura B.36, es el bloque principal del sistema de visión. En él se gestiona la adquisición de imágenes de la cámara, se utilizan el resto de bloques del sistema para determinar la posición y velocidad del cuadricóptero, se gestionan los errores en la detección y se actualiza la variable global que contiene todos estos datos.

Con respecto al bloque original se han realizado varias modificaciones con el objetivo de adecuarlo al presente proyecto. La primera modificación fue la de incluir el modificador de la altura del AR.Drone 2.0 en el cálculo de la posición debido a que ésta depende de la altura del plano h en el que se encuentre el triángulo de leds.

La segunda modificación fue la de actualizar dos variables globales de fuera del sistema de visión en el momento en el que la cámara determina una nueva orientación. Estas variables consisten en el unwrap del ángulo th (ψ) determinado por la cámara, almacenando dicho unwrap y su variable k, y son utilizadas en el Filtro de Kalman.

La tercera modificación consistió en la inicialización de ciertas variables antes de comenzar la ejecución de este bloque.

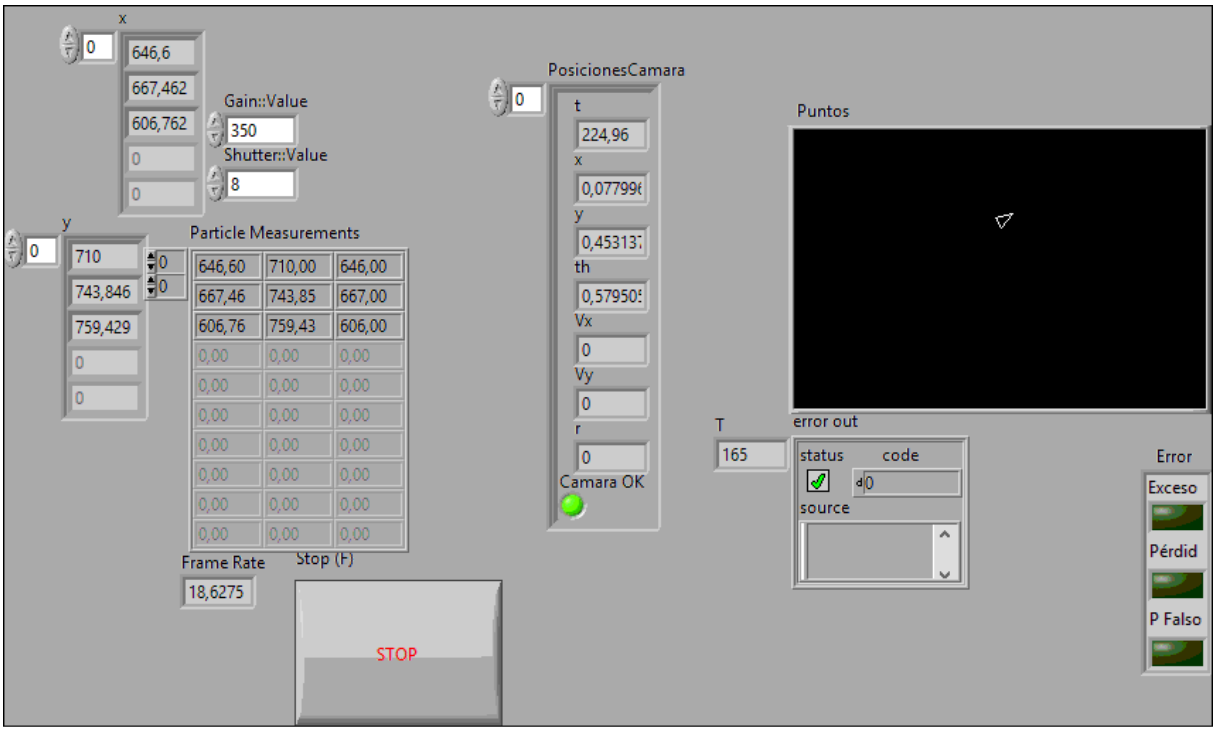
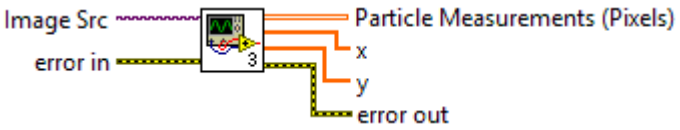


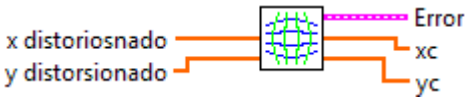
Figura B.36: Panel frontal del bloque Camara.

Puntos.vi



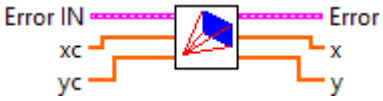
Este bloque determina, según lo indicado en el apartado A.2, la posición en el espacio imagen de los puntos de luz encontrados mediante funcionalidades de análisis de partículas que ofrece el módulo *NI Vision Development Module*.

Distorsión.vi



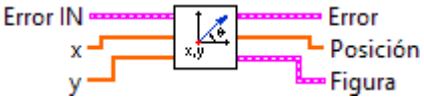
Este bloque determina, según lo indicado en el apartado A.3.1, la corrección a los puntos extraídos de la imagen debido a la distorsión en barril existente en ella.

Perspectiva.vi



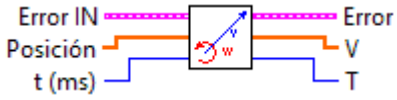
Este bloque determina, según lo indicado en el apartado A.3.2, la corrección a los puntos extraídos de la imagen debido a la perspectiva existente en ella.

Posición.vi



Este bloque determina, según lo indicado en el apartado A.4, la posición y la orientación en el marco de referencia E. Este bloque ha sido modificado con respecto al original para indicar las diferentes tolerancias en la forma del triángulo de leds y para indicar la distancia desde el led más alejado del triángulo hasta el centro del cuadricóptero.

Velocidad.vi



Este bloque determina, según lo indicado en el apartado A.5, las velocidades lineales y angulares en el marco de referencia E.

Calibracion.vi



Este bloque, del que se muestra su panel frontal en la Figura B.37, permite realizar el proceso de calibración de la cámara según lo indicado en el apartado A.6.

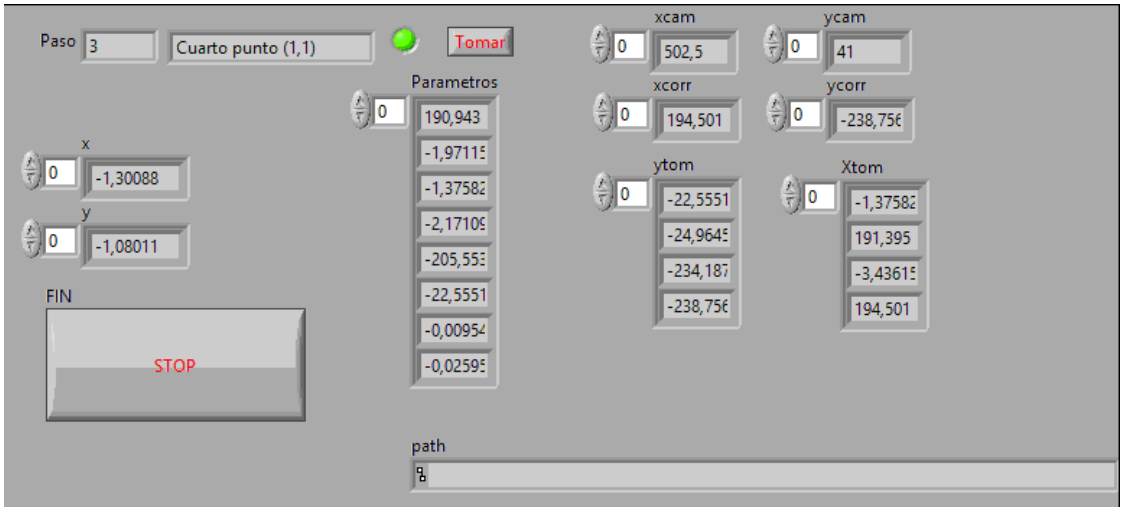


Figura B.37: Panel frontal del bloque Calibracion.

B.3. Modificaciones realizadas a la librería AR.Drone Toolkit

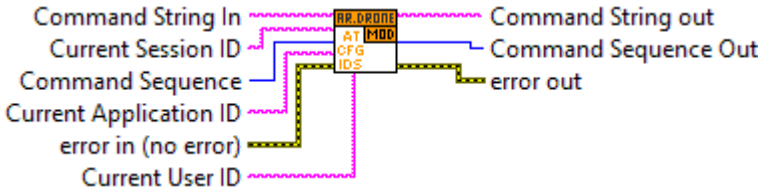
En el desarrollo del presente proyecto se ha utilizado la librería AR Drone Toolkit para LabVIEW - LVH que proporciona las funciones necesarias para comunicarse con el AR.Drone 2.0 y así poder configurarlo, manejarlo y obtener datos de navegación sin necesidad de implementar toda esta funcionalidad.

Desafortunadamente la librería contiene diferentes errores en lo relativo a la inicialización y configuración del cuadricóptero que ha sido necesario subsanar mediante la modificación de algunos de sus bloques y la creación de otros nuevos. Además, ha sido necesario crear nuevos bloques para la obtención de datos de navegación ya que la librería sólo tenía implementada la obtención de los datos de navegación básicos.

A continuación se detallan de forma breve los bloques nuevos y modificados, así como las causas.

B.3.1. Bloques modificados debido a incorrecta inicialización y configuración

ATCONFIG_IDS_modif.vi



Este bloque, del que se muestra su funcionamiento interno en la Figura B.38, ha sido modificado con el objetivo de manejar de forma correcta la multiconfiguración. Los identificadores de sesión, usuario y aplicación deben ir entre comillas y en el bloque original no estaban puestos así.

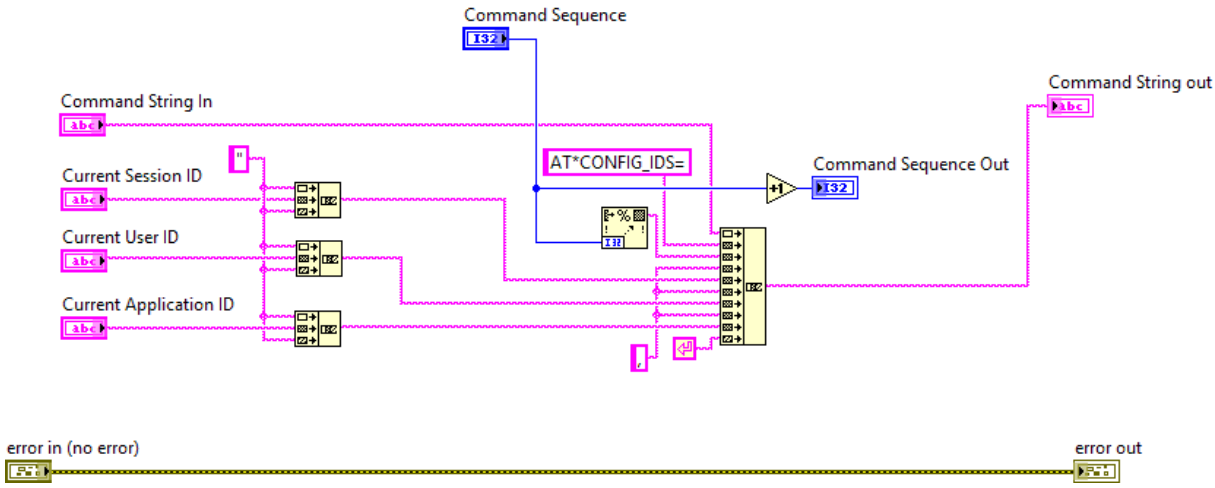
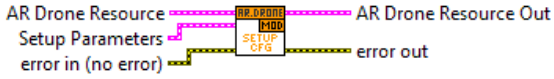


Figura B.38: Parte del funcionamiento interno del bloque ATCONFIG_IDS_modif.

Initialize Configuration Settings_modif_Multiconfig.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.39, se ha modificado para realizar una correcta configuración del AR.Drone 2.0 teniendo en cuenta la multiconfiguración y los tiempos de espera para que el AR.Drone 2.0 pueda reconocer los comandos. En el bloque original, cada configuración era enviada sin esperar el tiempo suficiente y esto resultaba, a veces, en comportamientos anómalos del cuadricóptero por no haber almacenado diversas configuraciones de forma correcta. En este nuevo bloque se ha incluido el envío inicial de los comandos ATPMODE y ATMISC que no están documentados pero sí que son los primeros en enviarse desde el SDK. Además, se ha incluido el envío de un comando ATCONFIG_IDS cuando resulta necesario en cada momento de la configuración.

El envío de la configuración se ha arreglado para seguir esta secuencia:

1. Se cambia el AR.Drone a configuración por defecto (Session = Application = Profile = 00000000).
2. Se borran los archivos de multiconfiguración para que no ocupe espacio en el sistema de archivos del AR.Drone 2.0.
3. Se crea y se cambia a Session_ID aleatoria para el UNEDrone.
4. Se cambia a Application_ID de UNEDrone.
5. Se cambia a Profile_ID de UNEDrone.

6. Se envían los valores de configuración aplicándose a la multiconfiguración de UNEDrone.

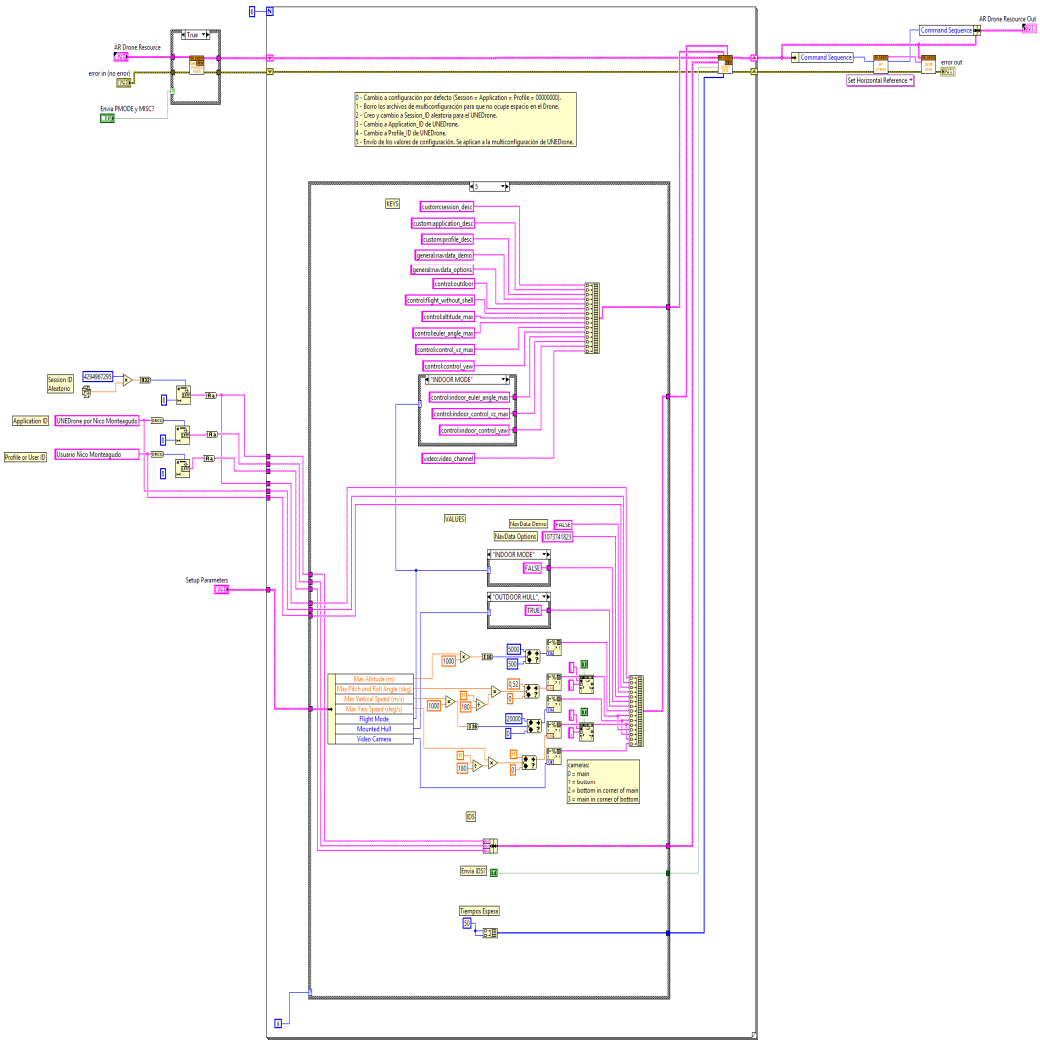
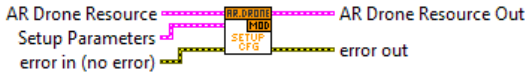


Figura B.39: Parte del funcionamiento interno del bloque Initialize Configuration Settings_modif_Multiconfig.

Initialize Configuration Settings_modif_No_Multiconfig.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.40, se ha modificado para realizar una correcta configuración del AR.Drone 2.0 sin tener en cuenta la multiconfiguración, pero teniendo en cuenta los tiempos de espera para que el AR.Drone 2.0 pueda reconocer los comandos. En el bloque original, cada configuración era enviada sin esperar el tiempo suficiente y esto resultaba, a veces, en comportamientos anómalos del cuadricóptero por no haber almacenado diversas configuraciones de forma correcta. En este nuevo bloque se ha incluido el envío inicial de los comandos ATPMODE y ATMISC que no están documentados pero sí que son los primeros en enviarse desde el SDK. Además, se ha incluido el envío de un comando ATCONFIG_IDS cuando resulta necesario en cada momento de la configuración.

El envío de la configuración se ha arreglado para seguir esta secuencia:

1. Se cambia el AR.Drone a configuración por defecto (Session = Application = Profile = 00000000).
2. Se envían los valores de configuración aplicándose a la configuración por defecto.

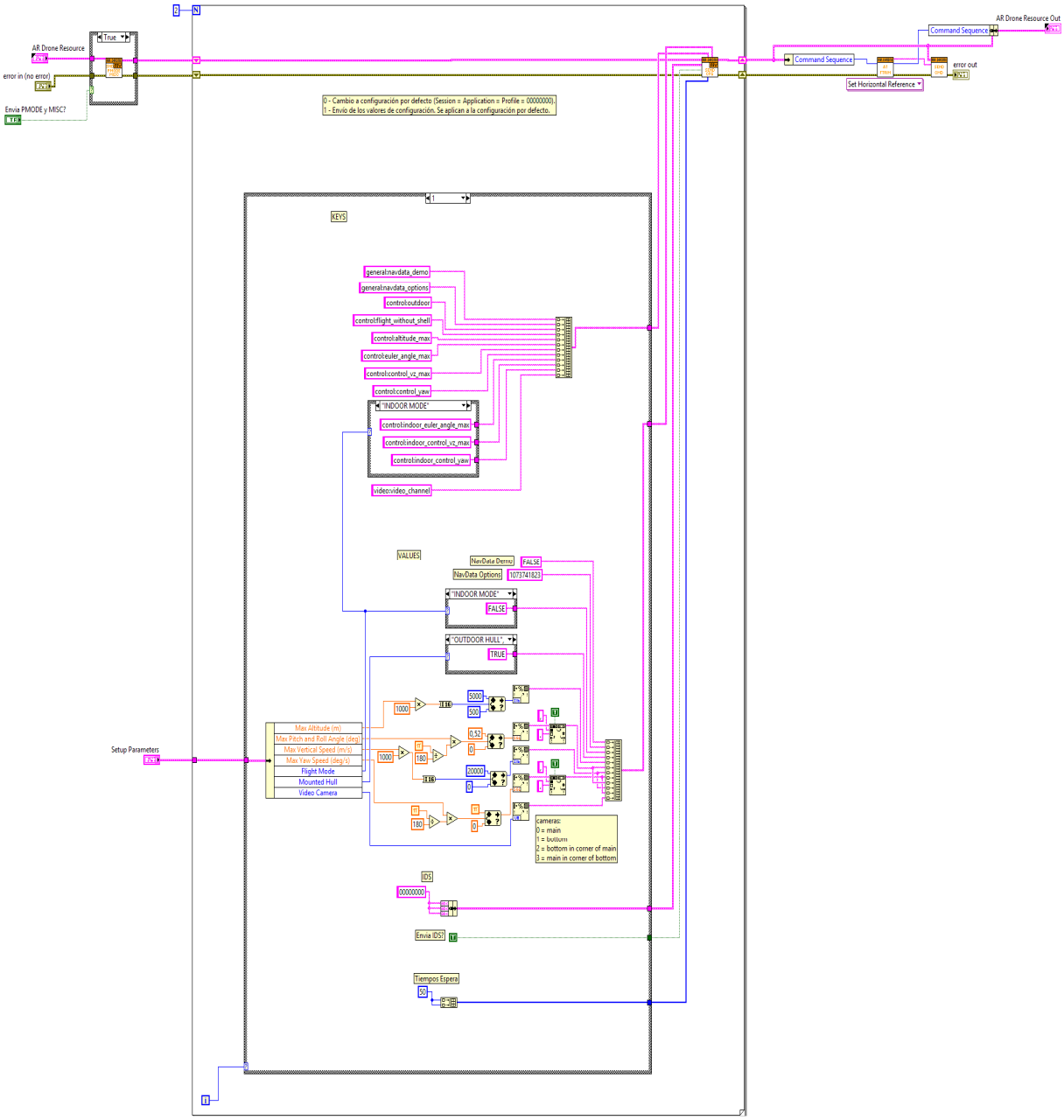


Figura B.40: Parte del funcionamiento interno del bloque Initialize Configuration Settings_modif_No_Multiconfig.

Open_modif.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.41, ha sido modificado para incluir las versiones modificadas del bloque de inicialización de la configuración del AR.Drone 2.0 y usarlas en función de si se desea usar la multiconfiguración o no.

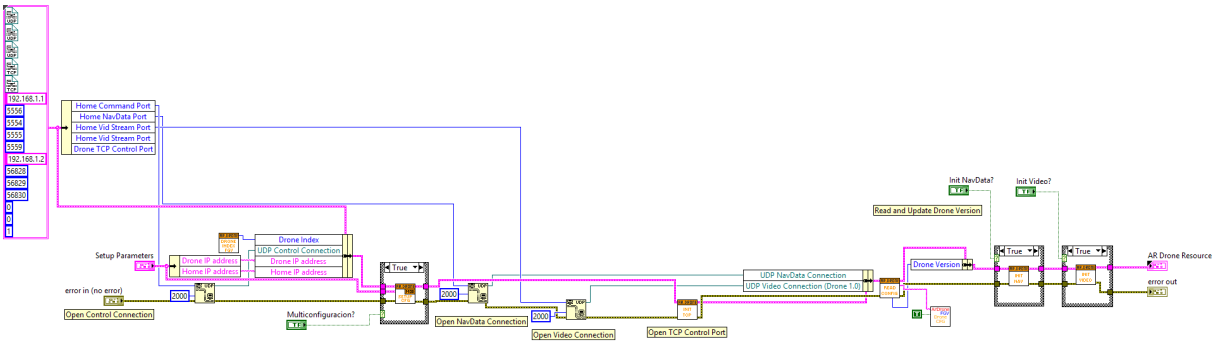
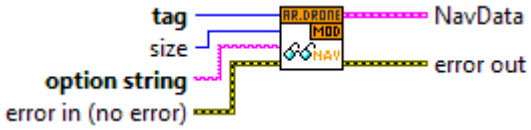


Figura B.41: Parte del funcionamiento interno del bloque Open_modif.

B.3.2. Bloques modificados debido a la inclusión de nuevos NavData

OptionParser_modif.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.42, se ha modificado para añadir las diferentes opciones de parseado de los nuevos NavData implementados. En este mismo bloque se actualiza directamente las variables globales de cada NavData.

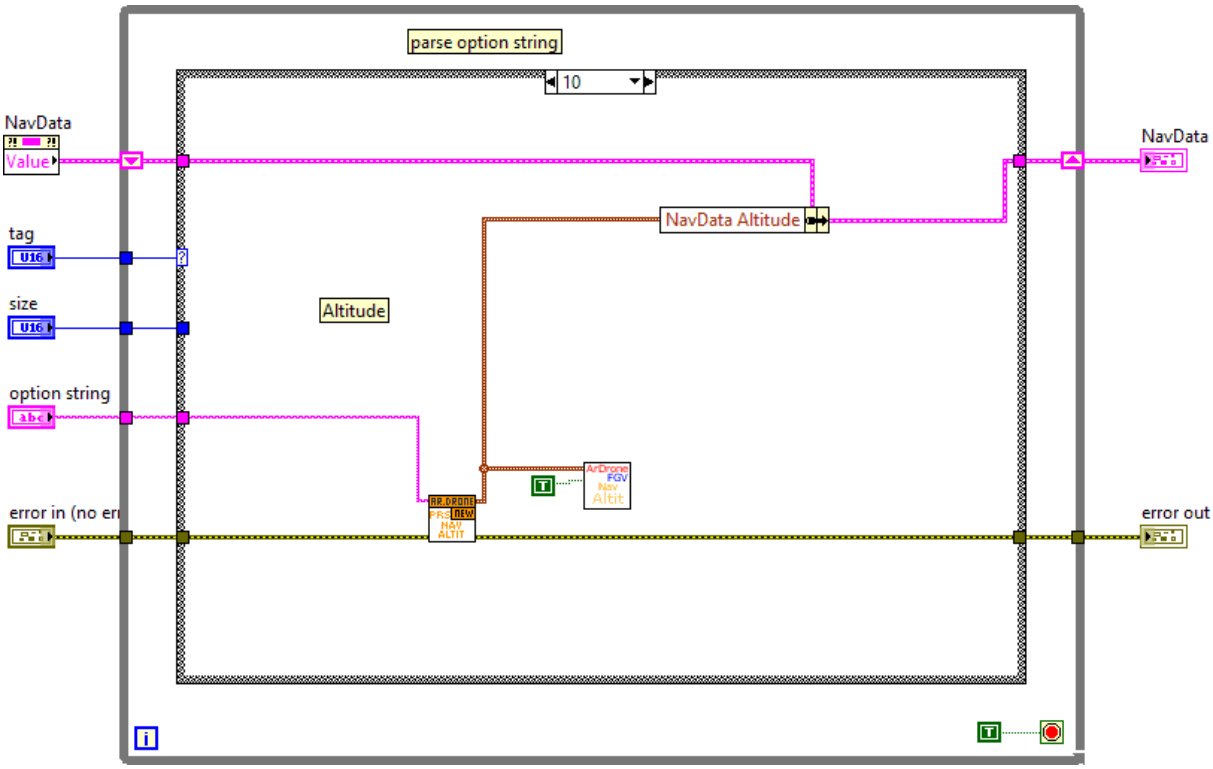


Figura B.42: Parte del funcionamiento interno del bloque OptionParser_modif.

ParseNavHeader_modif.vi



Este bloque, del que se muestra su funcionamiento interno en la Figura B.43, ha sido modificado para poder actualizar la variable global del estado del AR.Drone 2.0 de forma directa.

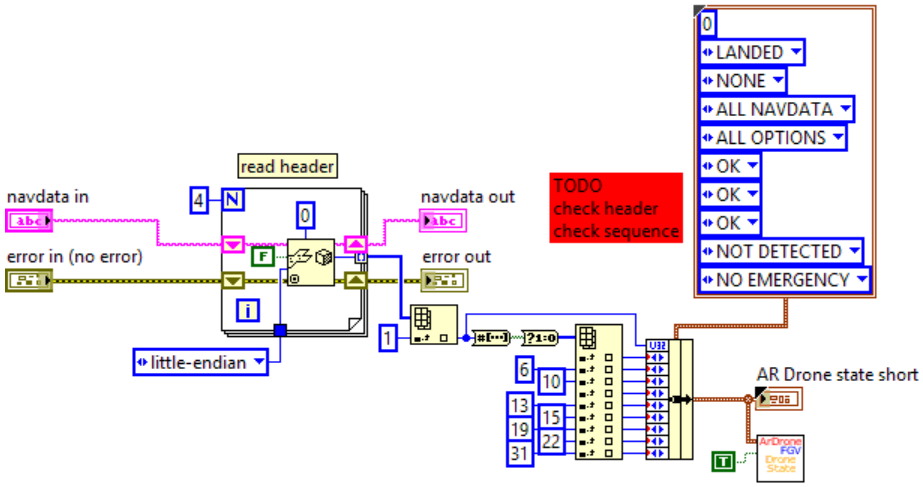


Figura B.43: Parte del funcionamiento interno del bloque ParseNavHeader_modif.

ParseNavData_modif.vi



Este bloque, del que se muestra su funcionamiento interno en la Figura B.44, ha sido modificado para incluir los bloques modificados del parseador de la cabecera de los NavData y del parseador de opciones de NavData.

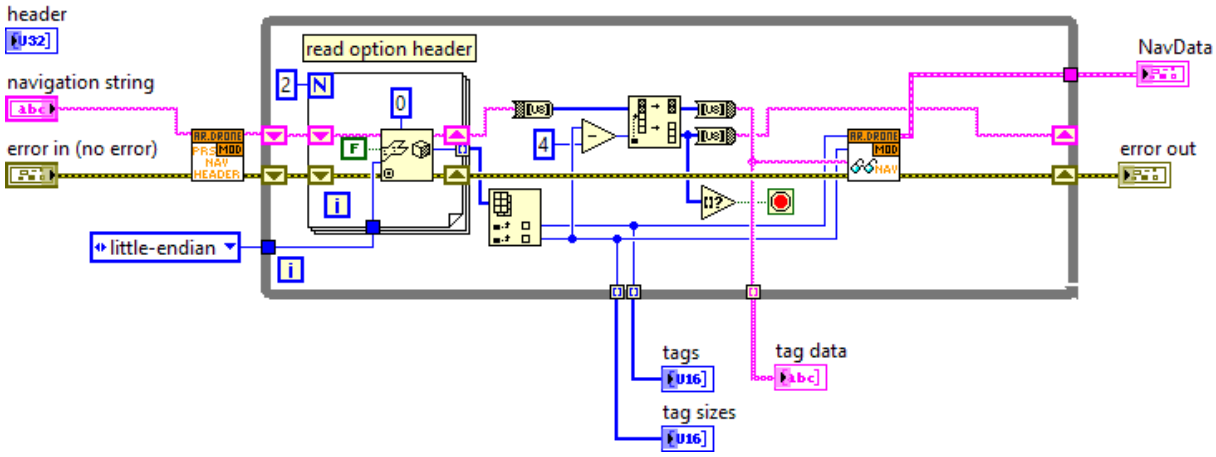
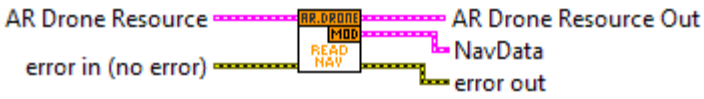


Figura B.44: Parte del funcionamiento interno del bloque ParseNavData_modif.

Read NavData_modif.vi



Este bloque, del que se muestra su funcionamiento interno en la Figura B.45, ha sido modificado para incluir el bloque modificado del parseador de datos de navegación NavData.

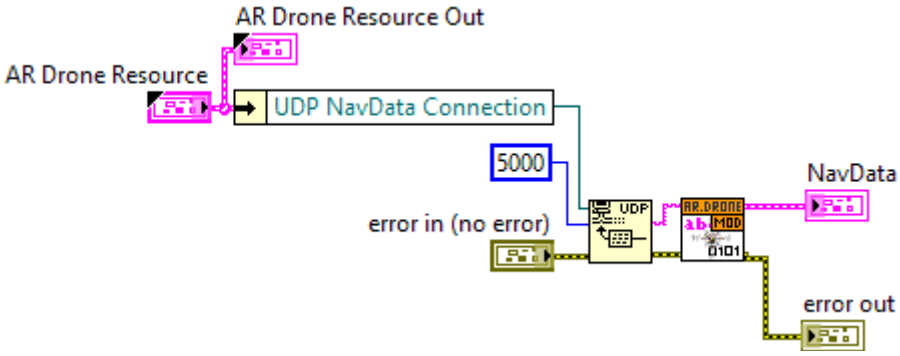


Figura B.45: Funcionamiento interno del bloque Read NavData_modif.

B.3.3. Bloques nuevos debido a incorrecta inicialización y configuración

ATMISC.vi



Este bloque, del que se muestra su funcionamiento interno en la Figura B.46, crea el comando AT*MISC. Este comando no está documentado en el SDK pero es utilizado por los programas que dicho SDK proporciona durante la inicialización de la configuración del AR.Drone 2.0.

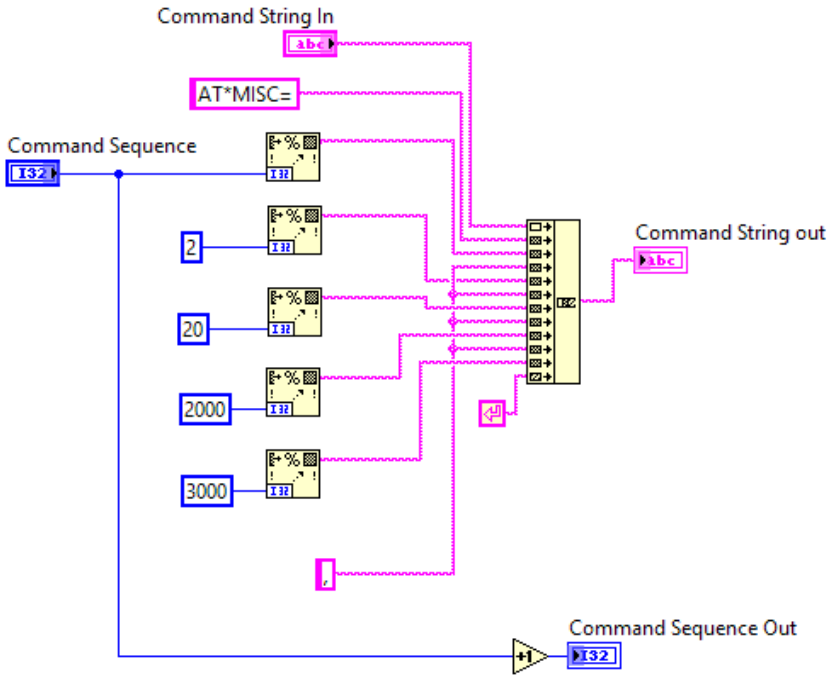


Figura B.46: Funcionamiento interno del bloque ATMISC.

ATPMODE.vi



Este bloque, del que se muestra su funcionamiento interno en la Figura B.47, crea el comando AT*PMODE. Este comando no está documentado en el SDK pero es utilizado por los programas que dicho SDK proporciona durante la inicialización de la configuración del AR.Drone 2.0.

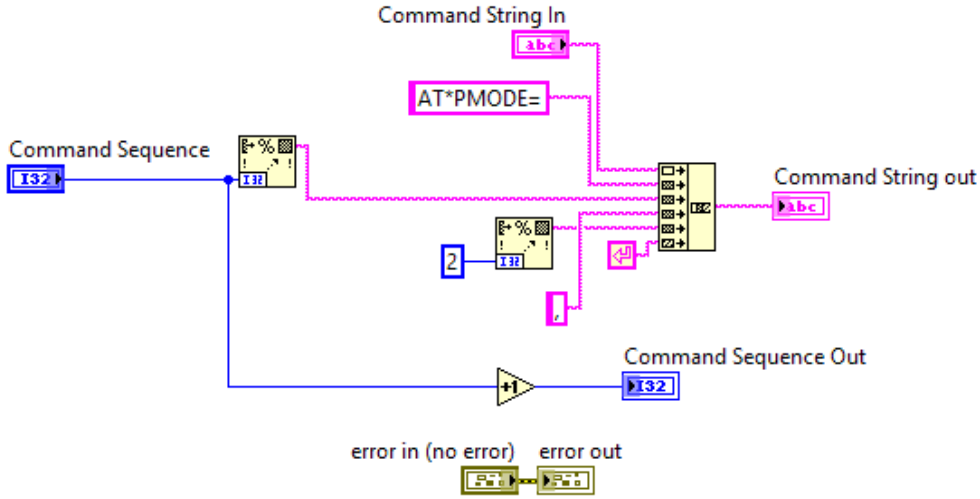


Figura B.47: Funcionamiento interno del bloque ATPMODE.

Envia_PMODE_MISC.vi



Este bloque, del que se muestra su funcionamiento interno en la Figura B.48, realiza el envío de los comandos AT*PMODE y AT*MISC de la forma correcta con los tiempos de espera necesarios.

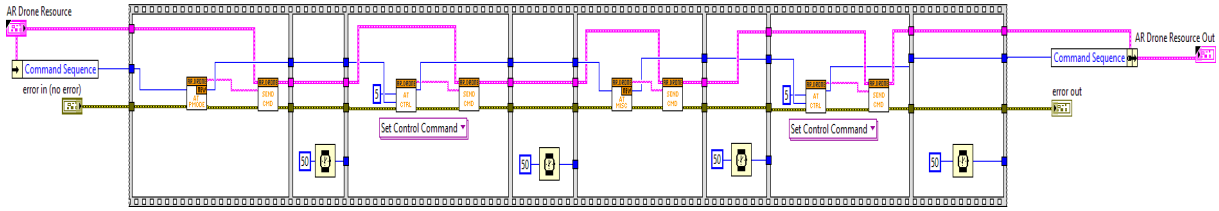
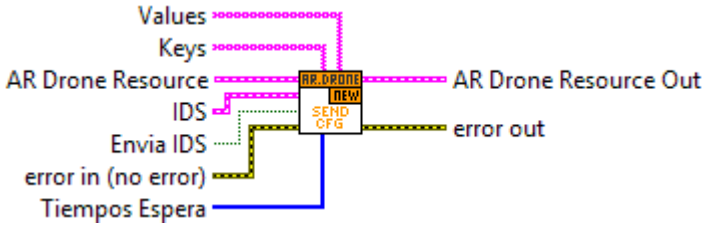


Figura B.48: Funcionamiento interno del bloque Envia_PMODE_MISC.

Envia_Config.vi



Este bloque, del que se muestra parte de su funcionamiento interno en la Figura B.49, realiza el envío de una configuración particular al AR.Drone 2.0 respetando los tiempos de espera necesarios y enviando el comando AT*CONFIG_IDS si es necesario.

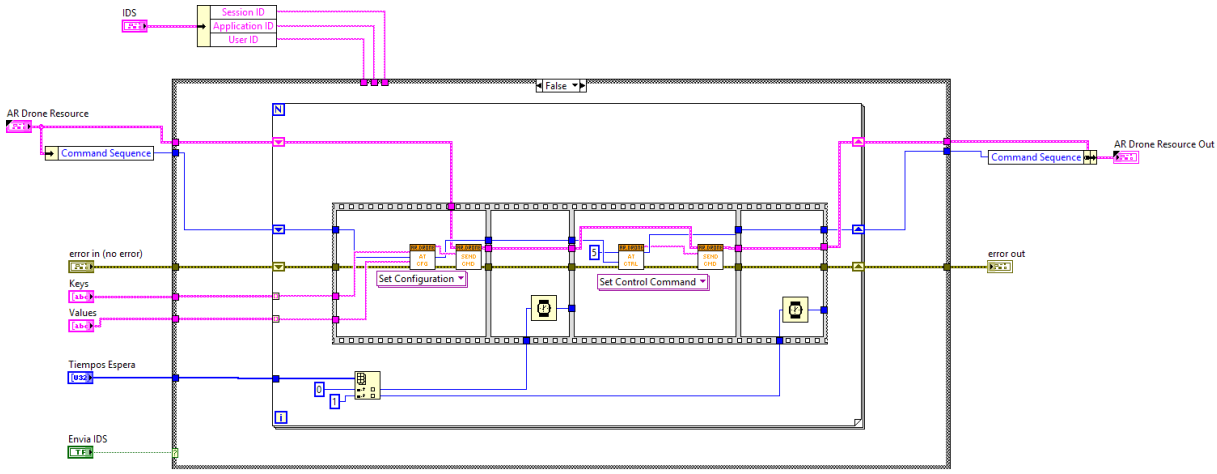


Figura B.49: Parte del funcionamiento interno del bloque Envia_Config.

B.3.4. Bloques nuevos debido a la inclusión de funcionalidad de calibración

ATCALIB.vi



Este bloque, del que se muestra su funcionamiento interno en la Figura B.50, crea el comando AT*CALIB utilizado en la calibración del magnetómetro del AR.Drone 2.0.

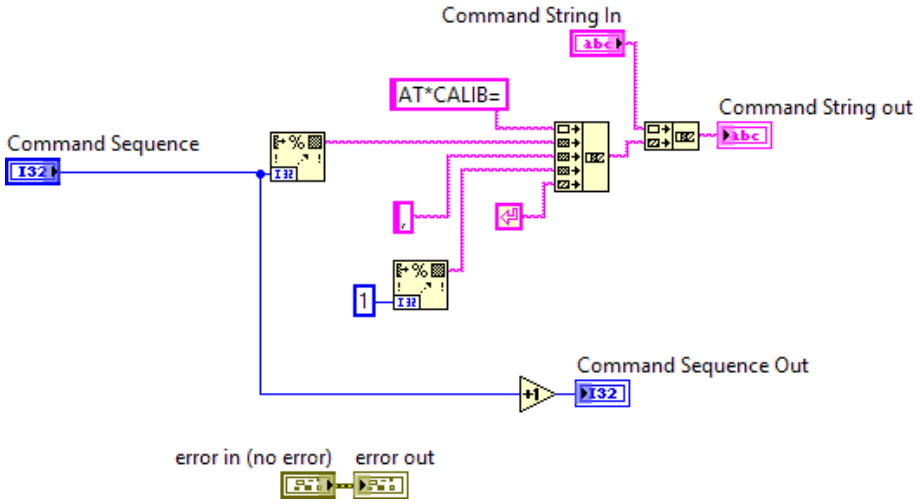


Figura B.50: Funcionamiento interno del bloque ATCALIB.

Calib Drone.vi



Este bloque, del que se muestra su funcionamiento interno en la Figura B.51, realiza el envío del comando AT*CALIB.

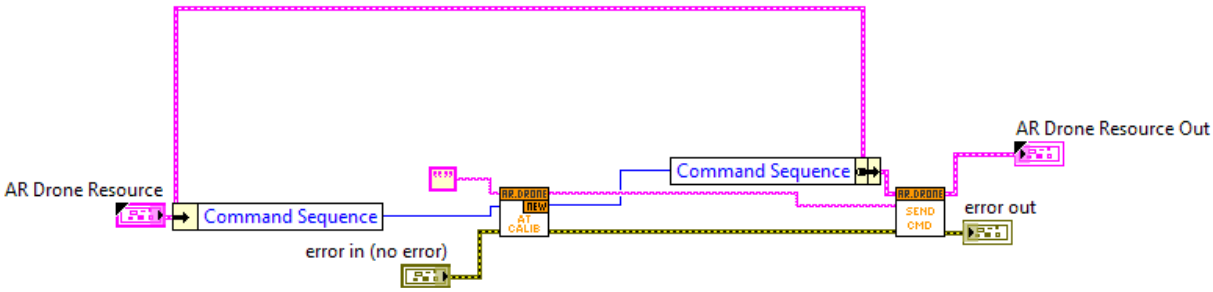


Figura B.51: Funcionamiento interno del bloque Calib Drone.

B.3.5. Bloques nuevos debido a la inclusión de nuevos NavData

ParseOption(Altitude).vi



Este bloque realiza el parseado de los datos del NavData Altitude.

ParseOption(EulerAngles).vi



Este bloque realiza el parseado de los datos del NavData Euler Angles.

ParseOption(GyrosOffsets).vi



Este bloque realiza el parseado de los datos del NavData Gyros Offsets.

ParseOption(PhysMeasures).vi



Este bloque realiza el parseado de los datos del NavData Phys Measures.

ParseOption(PWM).vi



Este bloque realiza el parseado de los datos del NavData PWM.

ParseOption(RawMeasures).vi



Este bloque realiza el parseado de los datos del NavData Raw Measures.

ParseOption(RC References).vi



Este bloque realiza el parseado de los datos del NavData RC References.

ParseOption(References).vi



Este bloque realiza el parseado de los datos del NavData References.

ParseOption(Trims).vi



Este bloque realiza el parseado de los datos del NavData Trims.

B.4. Descripción de los bloques de variables globales

En este proyecto se ha decidido usar el enfoque denominado FGV (Functional Global Variable) para el uso de variables globales que puedan compartirse entre diferentes bloques del programa sin necesidad de pasarlas como parámetros.

Como excepción se encuentra la variable global del sistema de visión que proporciona los valores de posición y velocidades del instante actual, y que se ha preferido no modificar para intentar mantener lo más aislado posible los bloques de dicho sistema por si en algún momento se libera una nueva versión, con el fin de mantener la compatibilidad.

A continuación se van a describir de forma muy breve estas variables.

B.4.1. Variables globales del sistema de visión

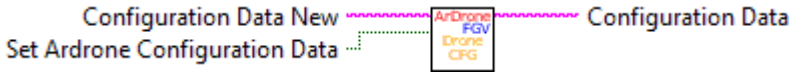
Posiciones_Camara.vi



Contiene los datos de la posición y velocidad determinados por el sistema de visión en el instante actual.

B.4.2. Variables globales de la librería AR Drone Toolkit

FGV_Ardrone_Configuration.vi



Contiene los datos de la configuración del AR.Drone 2.0.

FGV_Ardrone_Resource.vi



Contiene el recurso, o referencia, de la sesión abierta con el AR.Drone 2.0 para ser utilizado en los bloques de la librería AR Drone Toolkit.

FGV_Ardrone_State_Short.vi



Contiene los datos del estado del AR.Drone 2.0.

B.4.3. Variables globales generales

FGV_Estado.vi



Contiene el estado actual del programa.

FGV_StopGlobal.vi



Contiene la señal de Stop Global utilizada para detener la ejecución del sistema de visión cuando se ha cerrado la sesión en el programa principal.

FGV_AlturaDrone.vi



Contiene la altura que proporciona el AR.Drone 2.0. Esta variable global es utilizada por el modificador de altura necesario para el cálculo de las posiciones en la perspectiva de la cámara.

B.4.4. Variables globales del control automático

FGV_EstadoAuto.vi



Contiene el estado del modo automático en el que se encuentra el programa actualmente.

FGV_InicioAuto.vi



Contiene la indicación de si es el primer instante en el que se ha activado el modo automático.

FGV_Kalman_P.vi



Contiene la matriz P resultado de la ejecución de una iteración del Filtro de Kalman.

FGV_Kalman_x.vi



Contiene el vector x de estados resultado de la ejecución de una iteración del Filtro de Kalman.

FGV_Medidas.vi



Contiene el conjunto de medidas en el instante actual necesarias para realizar el control automático y/o estimar el estado del cuadricóptero.

FGV_Posicion_Estimada.vi



Contiene la posición del instante actual estimada por el Filtro de Kalman.

FGV_Posicion_Real.vi



Contiene la posición del instante actual determinada por el sistema de visión.

FGV_Referencia.vi



Contiene la referencia actual que el cuadricóptero debe seguir en su navegación autónoma.

FGV_thAnt.vi



Contiene la medida del ángulo de orientación determinado por el sistema de visión del instante justamente anterior al actual.

FGV_thUnwrap.vi



Contiene el unwrap de la medida del ángulo de orientación determinado por el sistema de visión.

FGV_thUnwrapK.vi



Contiene la variable k del unwrap de la medida del ángulo de orientación determinado por el sistema de visión.

FGV_TrackingK.vi



Contiene la variable k que se utiliza en las ecuaciones del seguimiento de trayectorias.

FGV_Waypoints_Fase.vi



Contiene la fase en la que se encuentra el cuadricóptero mientras está realizándose una navegación a través de Waypoints.

FGV_Waypoints_Indice_Punto.vi



Contiene el índice del Waypoint que el cuadricóptero está utilizando como referencia actual durante su navegación autónoma a través de Waypoints.

B.4.5. Variables globales de los datos de navegación

FGV_NavData_Altitude.vi



Contiene los datos de navegación actuales del NavData Altitude.

FGV_NavData_Euler_Angles.vi



Contiene los datos de navegación actuales del NavData Euler Angles.

FGV_NavData_Gyros_Offsets.vi



Contiene los datos de navegación actuales del NavData Gyros Offsets.

FGV_NavData_Phys_Measures.vi



Contiene los datos de navegación actuales del NavData Phys Measures.

FGV_NavData_PWM.vi



Contiene los datos de navegación actuales del NavData PWM.

FGV_NavData_Raw_Measures.vi



Contiene los datos de navegación actuales del NavData Raw Measures.

FGV_NavData_RC_References.vi



Contiene los datos de navegación actuales del NavData RC References.

FGV_NavData_References.vi



Contiene los datos de navegación actuales del NavData References.

FGV_NavData_Trims.vi



Contiene los datos de navegación actuales del NavData Trims.

B.4.6. Variables globales de los datos de las gráficas de seguimiento

FGV_OrientationData.vi



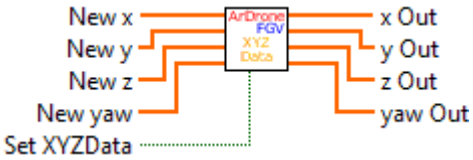
Contiene toda la serie temporal de datos que se muestran en la gráfica de seguimiento del ángulo de orientación del cuadricóptero.

FGV_XYData.vi



Contiene toda la serie temporal de datos que se muestran en la gráfica de seguimiento XY de la posición del cuadricóptero.

FGV_XYZData.vi



Contiene toda la serie temporal de datos que se muestran en la gráfica de seguimiento en 3 dimensiones de la posición y altura del cuadricóptero.

FGV_XZData.vi



Contiene toda la serie temporal de datos que se muestran en la gráfica de seguimiento XZ de la componente X de la posición y la altura del cuadricóptero.

FGV_YZData.vi



Contiene toda la serie temporal de datos que se muestran en la gráfica de seguimiento YZ de la componente Y de la posición y la altura del cuadricóptero.

ANEXO C. MANUAL DE USO BÁSICO DEL SOFTWARE

C.1. Requisitos previos

El PC en el que se ha probado el correcto funcionamiento del software dispone de un procesador Intel Core i3-2100 de 2 núcleos y 4 subprocesos a 3.10 Ghz y con 4 GB de memoria principal, en el que hay instalado como sistema operativo Microsoft Windows 7 Professional de 64 bits con Service Pack 1. Se recomienda que se use un PC con características similares para utilizar este software.

El PC deberá disponer, además, de una tarjeta PCI Matrox Meteor2-1394, o similar, para la adquisición de imágenes debido a que la cámara del sistema de visión tiene su conexión de tipo FireWire. Así mismo el PC deberá disponer de capacidades de conexión inalámbricas o disponer de un dispositivo externo USB que permita conectarse a redes Wi-Fi y comunicarse a través de ellas.

En cuanto al entorno de ejecución se deberá disponer de National Instruments LabVIEW 2013 con los módulos Vision Development Module, Control Design and Simulation Module y MathScript RT Module instalados. Además deberá instalarse la versión v0.1.0.36 de la librería *AR Drone Toolkit para LabVIEW – LVH* a través del gestor de paquetes VI Package Manager.

C.2. Instalación del software

El software no requiere de instalación ya que funciona perfectamente ejecutándolo desde el directorio donde se encuentra el código fuente.

El código fuente consiste en el proyecto de LabVIEW denominado *UNEDrone,* de una serie de ficheros de instrumentos virtuales VI que conforman el núcleo del proyecto y que se encuentran en el directorio raíz del proyecto junto al archivo *UNEDrone.lvproj,* así como de los siguientes subdirectorios:

- LVHModif: Dentro de este directorio se encuentran los ficheros VI añadidos o modificados de la librería *AR Drone Toolkit para LabVIEW – LVH.*
- Vision II: Dentro de este directorio se encuentran los ficheros VI del sistema de visión junto con sus ficheros de configuración.
- VGlobales: Dentro de este directorio se encuentran los ficheros VI de las variables globales que pueden utilizarse en cualquier bloque del proyecto.
- logs: Dentro de este directorio se guardan los ficheros de log que registran los datos de cada sesión abierta con el AR.Drone 2.0.

Para ejecutar el software simplemente se debe abrir el fichero de proyecto *UNEDrone.lvproj* dentro de LabVIEW, presentándose en pantalla la estructura del proyecto pudiendo seleccionar cualquier de sus VIs para su ejecución. La pantalla principal del software, que puede verse en la Figura C.1, se encuentra en *Principal.vi*.

C.3. Configuración de los controles manuales

La plataforma de software creada para controlar el cuadricóptero AR.Drone 2.0 ha sido diseñada para permitir su control manual, y para ello se da la posibilidad de utilizar el teclado o el joystick Attack 3 de Logitech. En la Figura C.2 se aprecia la ubicación de los botones y ejes de este joystick utilizados en el software.

El software está preparado para añadir nuevos joysticks creando VIs específicos para ellos.

Existen varias configuraciones de control que usan un tipo de control u otro o la combinación de ambos de ellos. A continuación se describen, teniendo en cuenta que por facilidad de manejo es preferible usar la configuración “Joystick + Teclado 1”.

Cuando el cuadricóptero se encuentra en vuelo autónomo se debe volver al modo manual pulsando la tecla Alt izquierda o el botón 4 del Joystick Attack 3, independientemente del control que estuviera configurado previamente al inicio del modo automático.

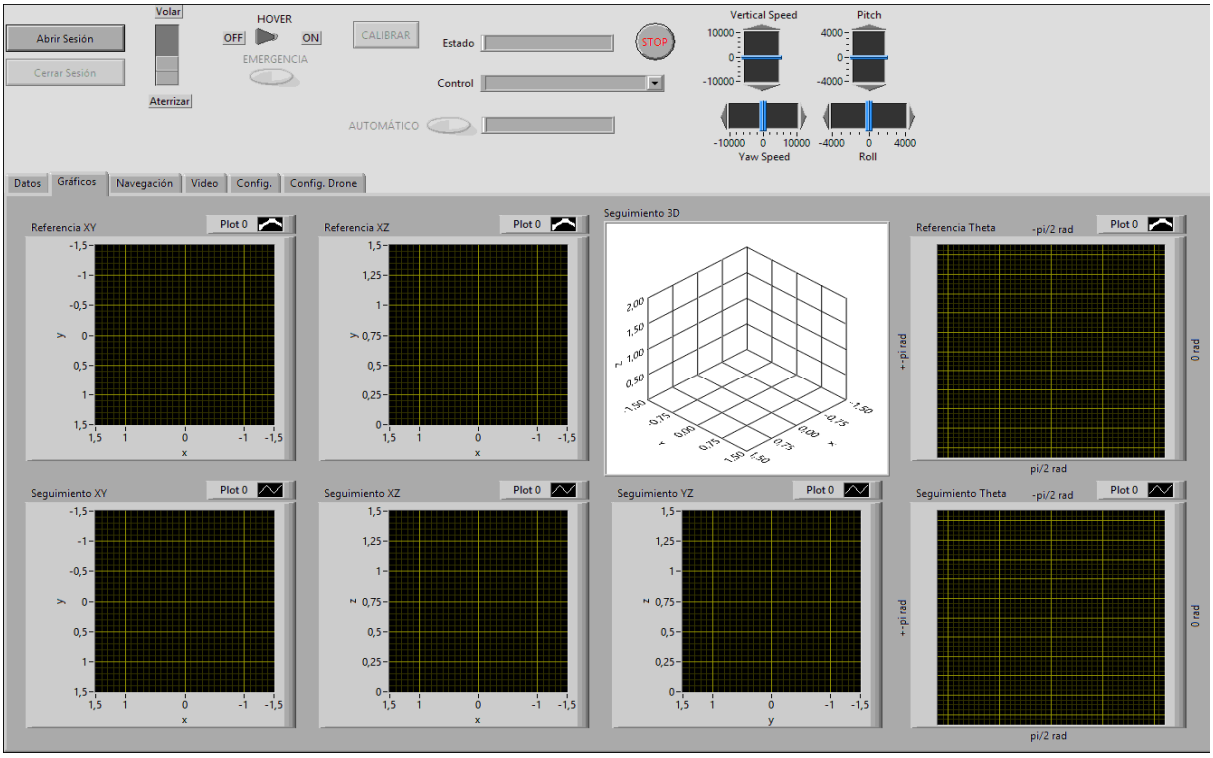


Figura C.1: Pantalla principal de la plataforma software.



Figura C.2: Ubicación de botones y ejes utilizados del joystick Attack 3 de Logitech.

C.3.1. Control manual "Teclado"

Las entradas de control de movimiento se consiguen mediante la pulsación de teclas. Cuando no hay teclas pulsadas la entrada de control es nula, y cuando se mantiene una tecla pulsada se va incrementando la entrada de control en un valor configurado cada 20 ms. Son:

- Entrada de control del Pitch hacia adelante: Tecla de flecha hacia arriba.
- Entrada de control del Pitch hacia atrás: Tecla de flecha hacia abajo.
- Entrada de control del Roll hacia la izquierda: Tecla de flecha hacia izquierda.
- Entrada de control del Roll hacia la derecha: Tecla de flecha hacia derecha.
- Entrada de control del Throttle hacia arriba: Tecla E.
- Entrada de control del Throttle hacia abajo: Tecla D.
- Entrada de control del Yaw en sentido inverso a las agujas del reloj: Tecla S.
- Entrada de control del Yaw en sentido de las agujas del reloj: Tecla F.

Otras acciones de la plataforma y que pueden realizarse a través de esta configuración de control son:

- Despegar/Aterrizar: Barra espaciadora.
- Hover: Tecla de mayúsculas derecha.
- Emergencia: Tecla Intro.

C.3.2. Control manual "Joystick"

Las entradas de control de movimiento se consiguen mediante la pulsación de botones y el movimiento de ejes del joystick. En relación a los botones, cuando no se encuentran pulsados la entrada de control es nula, y cuando se mantiene un botón pulsado se va incrementando la entrada de control en un valor configurado cada 20 ms. Son:

- Entrada de control del Pitch hacia adelante: Palanca del joystick hacia adelante (Eje Y).
- Entrada de control del Pitch hacia atrás: Palanca del joystick hacia atrás (Eje Y).
- Entrada de control del Roll hacia la izquierda: Palanca del joystick hacia izquierda (Eje X).
- Entrada de control del Roll hacia la derecha: Palanca del joystick hacia derecha (Eje X).
- Entrada de control del Throttle hacia arriba: Botón 6.
- Entrada de control del Throttle hacia abajo: Botón 7.
- Entrada de control del Yaw en sentido inverso a las agujas del reloj: Botón 8.
- Entrada de control del Yaw en sentido de las agujas del reloj: Botón 9.

Otras acciones de la plataforma y que pueden realizarse a través de esta configuración de control son:

- Despegar/Aterrizar: Botón 1.
- Hover: Botón 2.
- Emergencia: Botón 3.

C.3.3. Control manual "Joystick + Teclado 1"

Las entradas de control de movimiento se consiguen mediante la pulsación de teclas del teclado y el movimiento de ejes del joystick. En relación a las teclas, cuando no hay teclas pulsadas la entrada de control es

nula, y cuando se mantiene una tecla pulsada se va incrementando la entrada de control en un valor configurado cada 20 ms. Son:

- Entrada de control del Pitch hacia adelante: Palanca del joystick hacia adelante (Eje Y).
- Entrada de control del Pitch hacia atrás: Palanca del joystick hacia atrás (Eje Y).
- Entrada de control del Roll hacia la izquierda: Palanca del joystick hacia izquierda (Eje X).
- Entrada de control del Roll hacia la derecha: Palanca del joystick hacia derecha (Eje X).
- Entrada de control del Throttle hacia arriba: Tecla E.
- Entrada de control del Throttle hacia abajo: Tecla D.
- Entrada de control del Yaw en sentido inverso a las agujas del reloj: Tecla S.
- Entrada de control del Yaw en sentido de las agujas del reloj: Tecla F.

Otras acciones de la plataforma y que pueden realizarse a través de esta configuración de control son:

- Despegar/Aterrizar: Botón 1.
- Hover: Botón 2.
- Emergencia: Botón 3.

C.3.4. Control manual "Joystick + Teclado 2"

Las entradas de control de movimiento se consiguen mediante la pulsación de teclas del teclado y el movimiento de ejes del joystick. En relación a las teclas, cuando no hay teclas pulsadas la entrada de control es nula, y cuando se mantiene una tecla pulsada se va incrementando la entrada de control en un valor configurado cada 20 ms. Son:

- Entrada de control del Pitch hacia adelante: Tecla de flecha hacia arriba.
- Entrada de control del Pitch hacia atrás: Tecla de flecha hacia abajo.
- Entrada de control del Roll hacia la izquierda: Tecla de flecha hacia izquierda.
- Entrada de control del Roll hacia la derecha: Tecla de flecha hacia derecha.
- Entrada de control del Throttle hacia arriba: Palanca del joystick hacia adelante (Eje Y).
- Entrada de control del Throttle hacia abajo: Palanca del joystick hacia atrás (Eje Y).
- Entrada de control del Yaw en sentido inverso a las agujas del reloj: Palanca del joystick hacia izquierda (Eje X).
- Entrada de control del Yaw en sentido de las agujas del reloj: Palanca del joystick hacia derecha (Eje X).

Otras acciones de la plataforma y que pueden realizarse a través de esta configuración de control son:

- Despegar/Aterrizar: Botón 1.
- Hover: Botón 2.
- Emergencia: Botón 3.

C.4. Arranque y apertura de sesión con la cámara y con el AR.Drone 2.0

De forma previa a la apertura del proyecto *UNEDrone* en LabVIEW es necesario verificar que el joystick esté conectado al PC (si se necesitase usarlo), que la cámara esté conectada al PC, que el AR.Drone 2.0 y su triángulo de leds están conectado a su alimentación, y que el PC está conectado vía Wi-Fi a la red creada por el AR.Drone 2.0.

Una vez se ha abierto el proyecto *UNEDrone* en LabVIEW sólo se necesita acceder al panel frontal del VI Principal y ejecutarlo. En ese momento se pueden revisar y/o cambiar las configuraciones de la base de la plataforma, del sistema de visión y del AR.Drone 2.0. Para cambiar las del sistema de visión es necesario acceder al panel frontal del VI de la cámara.

De esta forma ya se está en disposición de iniciar sesión pulsando el botón “Abrir Sesión”. Esta acción conecta el software con el AR.Drone 2.0 enviándole los parámetros de configuración adecuados, conecta el software con la cámara enviándole los parámetros de configuración adecuados y prepara el software para el control del cuadricóptero. Una vez el software ha realizado todas estas acciones cambia su estado a “Marcha”, y es en este momento cuando el usuario debe revisar en la pestaña “Datos” que la cámara y el AR.Drone 2.0 están funcionando bien porque los datos se están actualizando. Si no fuera así habría que cerrar sesión y solucionar el problema en cuestión.

Hecha esta verificación se está en disposición de hacer volar el cuadricóptero y manejarlo de forma manual o automática.

C.5. Uso de funcionalidades propias del AR.Drone 2.0

La plataforma software permite utilizar funcionalidades que el AR.Drone 2.0 incorpora en su firmware. La primera de ellas es la del despegue autónomo que se utiliza desde la plataforma para hacer que el cuadricóptero se ponga en vuelo. Para ello es necesario haber abierto una sesión y poner el switch de "Volar/Aterrizar" al modo "Volar" o pulsar la tecla o botón correspondiente del control manual seleccionado. Una vez el cuadricóptero se encuentra en vuelo puede manejarse de forma manual o activar el modo automático que proporciona el software.

Durante el vuelo puede cambiarse dicho switch al modo "Aterrizar", bien mediante el ratón o mediante la pulsación de la tecla o botón correspondiente del control manual seleccionado, para que el AR.Drone 2.0 realice el aterrizaje de forma autónoma.

Mientras el cuadricóptero se encuentra en vuelo puede activarse el modo "Hover" con el ratón o con la tecla o botón correspondiente del control manual seleccionado. Al activar este modo el AR.Drone 2.0 intenta mantenerse estable durante todo el tiempo en la misma posición, orientación y altura en la que se encontraba al activarlo, gracias al control interno implementado en su firmware.

Precisamente cuando el cuadricóptero se encuentra en modo "Hover" es el único estado en el que se puede realizar la calibración de su magnetómetro. Para ello se pulsa el botón "CALIBRAR" y el cuadricóptero realiza una vuelta sobre sí mismo consiguiendo así calibrarlo.

Por último, el botón de "Emergencia" se utiliza cuando se necesita apagar los motores del AR.Drone 2.0 y no puede usarse la funcionalidad de aterrizaje autónomo por cualquier circunstancia excepcional. Esta característica puede activarse también mediante la tecla o botón correspondiente del control manual seleccionado. Hay otro uso de este botón y que es importante, ya que cuando el cuadricóptero tiene un choque y se ponen en rojo los 4 leds de sus motores, no se puede despegar de nuevo hasta que no se haya quitado dicho estado de "Emergencia" mediante su pulsación.

C.6. Funcionamiento del modo automático

Cuando se ha abierto una sesión y se ha realizado el despegue, se puede manejar el cuadricóptero mediante el control manual seleccionado o activar el modo automático para que siga las referencias parametrizadas en el software, momento en el cual se desactivan los controles manuales impidiendo que el usuario lo maneje manualmente.

Antes de entrar en el modo automático se deben haber parametrizado las referencias que el cuadricóptero debe seguir de forma autónoma. Es en la pestaña de "Navegación" donde se indican los valores de estas referencias, pudiendo elegir entre la estabilización en un punto, la navegación a través de Waypoints, la

navegación a través de Waypoints con seguimiento (tracking) de las rectas que los unen, y el seguimiento (tracking) de trayectorias paramétricas.

Una vez parametrizadas las referencias debe seleccionarse en el desplegable el tipo de control autónomo que se desea realizar en cuanto se active dicho modo.

Igualmente puede seleccionarse si se desea realizar el seguimiento del Yaw, es decir, si se actúa o no en la entrada de control del Yaw, y si se desea utilizar el Filtro de Kalman para estimar el estado del cuadricóptero para mejorar el control autónomo.

Cuando el cuadricóptero se encuentra en vuelo autónomo se debe volver al modo manual pulsando la tecla Alt izquierda o el botón 4 del Joystick Attack 3.

Hay que tener en cuenta que tras activar el modo automático transcurre 1 segundo hasta que el cuadricóptero comienza a seguir las referencias parametrizadas. Este tiempo de espera se ha incluido para que el Filtro de Kalman pueda realizar suficientes iteraciones como para que su ganancia converja más rápidamente. Independientemente de que se utilice o no dicho filtro el software siempre realiza dicho tiempo de espera.

C.7. Parámetros de configuración

En la pestaña "Config" del panel frontal del bloque Principal puede configurarse el parámetro *IncValoresControlesMiliseg* que establece el valor del incremento (multiplicado por 10000) de las acciones de control que se envían al AR.Drone 2.0. Cuando se mantiene pulsada una tecla o un botón del joystick se incrementa o decrementa la acción de control en una cantidad igual al valor aquí configurado por cada 20 ms que transcurran.

Por su parte, en la pestaña "Config. Drone" se puede ver la configuración actual del AR.Drone y puede realizarse la modificación de algunos de sus parámetros básicos: Direcciones IP, Modo de vuelo Interior/Exterior, Vídeo de qué cámara enviar, altura máxima de vuelo, así como los valores máximos de inclinación de Pitch y Roll, de velocidad vertical y de Yaw.

En relación a los valores máximos de inclinación de Pitch y Roll, de velocidad vertical y de velocidad de Yaw, se recomienda no cambiarlos debido a que la identificación del modelo y el control autónomo han sido derivados a través de ellos, mostrados en la Tabla C.1. Si se desea modificar estos valores máximos sería necesario realizar una nueva identificación y un nuevo diseño del controlador.

Configuración	Valor
Máxima inclinación de Pitch	15 °
Máxima inclinación de Roll	15 °
Máxima velocidad vertical de Throttle	0.7 m/s
Máxima velocidad angular de Yaw	100 °/s

Tabla C.1: Valores máximos configurados de las entradas de control en el AR.Drone 2.0.

C.8. Ficheros de log

Cada vez que se abre una sesión se crea un fichero de log cuyo nombre contiene la fecha, hora, minuto y segundo de su creación dentro del directorio "logs" que se encuentra en el raíz del proyecto de LabVIEW. Durante todo el tiempo que permanece abierta se registra una línea cada 20 ms que contiene los datos más relevantes obtenidos del sistema de visión, del AR.Drone 2.0 o del propio software.

Cuando se cierra la sesión finaliza la escritura de este fichero y puede ser analizado con herramientas adecuadas para ello. En la Tabla C.2 se muestra la lista de datos que se almacenan en cada línea, separados por tabulaciones y en el mismo orden. La columna "Unidades Matlab" de esta tabla son las unidades a las que se convierten los datos del fichero de log cuando se utiliza el script *funImportar_Log_v5.m,* que acompaña al resto del código fuente de este proyecto, y que permite importar fácilmente los datos del log a Matlab.

Este script puede utilizarse de forma autónoma o a través de dos scripts auxiliares *Importar_Log_v5.m* e *Importar_Logs_Directorio_v5.m.* El primero de ellos abre un cuadro de diálogo para seleccionar el fichero a importar, mientras que el segundo realiza la importación de todos los ficheros *.log* de un directorio determinado.

Dato	Fuente	Descripción	Unidades	Unidades Matlab
C_Tiempo	Control	Tiempo transcurrido	ms	s
C_Roll_x10	Control	Entrada de control Roll (multiplicada por 10000)		
C_Pitch_x10	Control	Entrada de control Pitch (multiplicada por 10000)		
C_Throttle_x10	Control	Entrada de control Throttle (multiplicada por 10000)		
C_Yaw_x10	Control	Entrada de control Yaw (multiplicada por 10000)		
C_Roll	Control	Entrada de control Roll		
C_Pitch	Control	Entrada de control Pitch		
C_Throttle	Control	Entrada de control Throttle		
C_Yaw	Control	Entrada de control Yaw		
Drone_State	Drone	Dato del NavData Demo		
Drone_Battery	Drone	Dato del NavData Demo	%	%
Drone_Pitch	Drone	Dato del NavData Demo	°	rad
Drone_Roll	Drone	Dato del NavData Demo	°	rad
Drone_Yaw	Drone	Dato del NavData Demo	°	rad
Drone_Altitude	Drone	Dato del NavData Demo	mm	m
Drone_vx	Drone	Dato del NavData Demo	mm/s	m/s
Drone_vy	Drone	Dato del NavData Demo	mm/s	m/s
Drone_vz	Drone	Dato del NavData Demo	mm/s	m/s
Drone_CamDetect	Drone	Dato del NavData Demo		
Drone_accX	Drone	Dato del NavData Phys Measures	g	m/s^2
Drone_accY	Drone	Dato del NavData Phys Measures	g	m/s^2
Drone_accZ	Drone	Dato del NavData Phys Measures	g	m/s^2
Drone_gyroX	Drone	Dato del NavData Phys Measures	°/s	rad/s
Drone_gyroY	Drone	Dato del NavData Phys Measures	°/s	rad/s
Drone_gyroZ	Drone	Dato del NavData Phys Measures	°/s	rad/s
Drone_accX_raw	Drone	Dato del NavData Raw Measures	LSB	LSB
Drone_accY_raw	Drone	Dato del NavData Raw Measures	LSB	LSB
Drone_accZ_raw	Drone	Dato del NavData Raw Measures	LSB	LSB
Drone_gyroX_raw	Drone	Dato del NavData Raw Measures	LSB	LSB
Drone_gyroY_raw	Drone	Dato del NavData Raw Measures	LSB	LSB
Drone_gyroZ_raw	Drone	Dato del NavData Raw Measures	LSB	LSB
Drone_gyroX_110_raw	Drone	Dato del NavData Raw Measures	LSB	LSB
Drone_gyroY_110_raw	Drone	Dato del NavData Raw Measures	LSB	LSB
Drone_alt_raw	Drone	Dato del NavData Raw Measures	mm	m
Drone_gradient	Drone	Dato del NavData Raw Measures	LSB	LSB
Drone_gyroX_offset	Drone	Dato del NavData Gyros Offsets	°/s	rad/s
Drone_gyroY_offset	Drone	Dato del NavData Gyros Offsets	°/s	rad/s
Drone_gyroZ_offset	Drone	Dato del NavData Gyros Offsets	°/s	rad/s
Drone_theta_euler	Drone	Dato del NavData Euler Angles	°	rad
Drone_phi_euler	Drone	Dato del NavData Euler Angles	°	rad

Dato	Fuente	Descripción	Unidades	Unidades Matlab
Drone_yaw_trim	Drone	Dato del NavData Trims	°/s	rad/s
Drone_theta_euler_trim	Drone	Dato del NavData Trims	°	rad
Drone_phi_euler_trim	Drone	Dato del NavData Trims	°	rad
Drone_ref_theta	Drone	Dato del NavData References	m°	rad
Drone_ref_phi	Drone	Dato del NavData References	m°	rad
Drone_ref_theta_internal	Drone	Dato del NavData References	m°	rad
Drone_ref_phi_internal	Drone	Dato del NavData References	m°	rad
Drone_ref_pitch	Drone	Dato del NavData References	m°	rad
Drone_ref_roll	Drone	Dato del NavData References	m°	rad
Drone_ref_yaw	Drone	Dato del NavData References	m°/s	rad/s
Drone_ref_psi	Drone	Dato del NavData References	m°	rad
Drone_ref_vx	Drone	Dato del NavData References	m/s	m/s
Drone_ref_vy	Drone	Dato del NavData References	m/s	m/s
Drone_theta_modele	Drone	Dato del NavData References	rad	rad
Drone_phi_modele	Drone	Dato del NavData References	rad	rad
Drone_k_v_x	Drone	Dato del NavData References		
Drone_k_v_y	Drone	Dato del NavData References		
Drone_k_mode	Drone	Dato del NavData References		
Drone_time_ui	Drone	Dato del NavData References		
Drone_theta_ui	Drone	Dato del NavData References		
Drone_phi_ui	Drone	Dato del NavData References		
Drone_psi_ui	Drone	Dato del NavData References		
Drone_psi_accuracy	Drone	Dato del NavData References		
Drone_seq_ui	Drone	Dato del NavData References		
Drone_rc_ref_roll	Drone	Dato del NavData RC References		
Drone_rc_ref_pitch	Drone	Dato del NavData RC References		
Drone_rc_ref_yaw	Drone	Dato del NavData RC References		
Drone_rc_ref_gaz	Drone	Dato del NavData RC References		
Drone_rc_ref_ag	Drone	Dato del NavData RC References		
Drone_motor_1	Drone	Dato del NavData PWM	PWM	PWM
Drone_motor_2	Drone	Dato del NavData PWM	PWM	PWM
Drone_motor_3	Drone	Dato del NavData PWM	PWM	PWM
Drone_motor_4	Drone	Dato del NavData PWM	PWM	PWM
Drone_motor_1_sat	Drone	Dato del NavData PWM	PWM	PWM
Drone_motor_2_sat	Drone	Dato del NavData PWM	PWM	PWM
Drone_motor_3_sat	Drone	Dato del NavData PWM	PWM	PWM
Drone_motor_4_sat	Drone	Dato del NavData PWM	PWM	PWM
Drone_gaz_feed_forward	Drone	Dato del NavData PWM	PWM	PWM
Drone_altitude_integral	Drone	Dato del NavData PWM	m/s	m/s
Drone_ref_vz	Drone	Dato del NavData PWM	m/s	m/s
Drone_u_pitch	Drone	Dato del NavData PWM	PWM	PWM
Drone_u_roll	Drone	Dato del NavData PWM	PWM	PWM
Drone_u_yaw	Drone	Dato del NavData PWM	PWM	PWM
Drone_u_yaw_I	Drone	Dato del NavData PWM	PWM	PWM
Drone_u_pitch_plan	Drone	Dato del NavData PWM	PWM	PWM
Drone_u_roll_plan	Drone	Dato del NavData PWM	PWM	PWM
Drone_u_yaw_plan	Drone	Dato del NavData PWM	PWM	PWM
Drone_u_gaz_plan	Drone	Dato del NavData PWM	PWM	PWM

Dato	Fuente	Descripción	Unidades	Unidades Matlab
Drone_motor_1_current	Drone	Dato del NavData PWM	mA	A
Drone_motor_2_current	Drone	Dato del NavData PWM	mA	A
Drone_motor_3_current	Drone	Dato del NavData PWM	mA	A
Drone_motor_4_current	Drone	Dato del NavData PWM	mA	A
Drone_altitude_prop	Drone	Dato del NavData PWM	PWM	PWM
Drone_altitude_der	Drone	Dato del NavData PWM	PWM	PWM
Drone_Altitude_Vision	Drone	Dato del NavData Altitude	mm	m
Drone_Altitude_vz	Drone	Dato del NavData Altitude	mm/s	m/s
Drone_Altitude_ref	Drone	Dato del NavData Altitude	mm	m
Drone_Altitude_raw	Drone	Dato del NavData Altitude	mm	m
Drone_Obs_accZ	Drone	Dato del NavData Altitude	m/s^2	m/s^2
Drone_Obs_Altitude_US	Drone	Dato del NavData Altitude	m	m
Drone_Estim_Obs_Altitude	Drone	Dato del NavData Altitude	m	m
Drone_Estim_Obs_vz	Drone	Dato del NavData Altitude	m/s	m/s
Drone_Estim_Obs_accZ_Bias	Drone	Dato del NavData Altitude	m/s^2	m/s^2
Drone_Obs_State	Drone	Dato del NavData Altitude		
Cam_t	Cámara	Tiempo transcurrido de la cámara	s	s
Cam_x	Cámara	Posición x dada por la cámara	m	m
Cam_y	Cámara	Posición y dada por la cámara	m	m
Cam_th	Cámara	Orientación de Yaw dada por la cámara	rad	rad
Cam_vx	Cámara	Velocidad en x dada por la cámara	m/s	m/s
Cam_vy	Cámara	Velocidad en y dada por la cámara	m/s	m/s
Cam_vz	Cámara	Velocidad de Yaw dada por la cámara	m/s	m/s
Cam_ok	Cámara	Indicador de que la cámara ha obtenido correctamente una posición	STRING	
Setup_Max_Altitude	Setup	Configuración del AR.Drone 2.0. Altura máxima de vuelo	m	m
Setup_Max_Pitch_and_Roll_Angle	Setup	Configuración del AR.Drone 2.0. Inclinación máxima de Pitch y Roll	°	rad
Setup_Max_Vertical_Speed	Setup	Configuración del AR.Drone 2.0. Máxima velocidad vertical	m/s	m/s
Setup_Max_Yaw_Speed	Setup	Configuración del AR.Drone 2.0. Máxima velocidad de Yaw	°/s	rad/s
Estado_Auto	Auto	Estado del modo Automático (0: Manual, 1: Iniciando Auto, 2: Esperando Auto, 3: Auto)		
Ref_x	Auto	Posición x de la referencia a seguir	m	m
Ref_y	Auto	Posición y de la referencia a seguir	m	m
Ref_z	Auto	Altura z de la referencia a seguir	m	m
Ref_yaw	Auto	Orientación del Yaw de la referencia a seguir	rad	rad
Real_x	Auto	Posición x real determinada	m	m
Real_y	Auto	Posición y real determinada	m	m
Real_z	Auto	Altura z real determinada	m	m
Real_yaw	Auto	Orientación del Yaw real determinada	rad	rad

Dato	Fuente	Descripción	Unidades	Unidades Matlab
Kalman_x	Auto	Estimación de la posición x por el KF	m	m
Kalman_y	Auto	Estimación de la posición y por el KF	m	m
Kalman_z	Auto	Estimación de la altura z por el KF	m	m
Kalman_yaw	Auto	Estimación de la orientación del Yaw por el KF	rad	rad
Kalman_vx	Auto	Estimación de la velocidad en x por el KF	m/s	m/s
Kalman_vy	Auto	Estimación de la velocidad en y por el KF	m/s	m/s
Kalman_vz	Auto	Estimación de la velocidad vertical por el KF	m/s	m/s
Kalman_vyaw	Auto	Estimación de la velocidad del Yaw por el KF	rad/s	rad/s
thUnwrap	Auto	Unwrap del ángulo de Yaw de la cámara	rad	rad
thUnwrapK	Auto	Valor de K del unwrap del ángulo de Yaw de la cámara	rad	rad
ultimoYawValido	Auto	Valor del último Yaw válido para usarlo en el KF	rad	rad

Tabla C.2: Datos almacenados en cada línea del fichero de log.

ANEXO D. PARÁMETROS DE CONFIGURACIÓN, ESTADO Y NAVEGACIÓN DEL AR.DRONE 2.0

D.1. Parámetros de configuración

En la Tabla D.1 se describen los posibles parámetros de configuración del AR.Drone 2.0 indicando si son únicamente de lectura (R) o de lectura y escritura (W). Estos parámetros se dividen en las siguientes categorías:

- General: Parámetros generales del AR.Drone 2.0.
- Control: Parámetros relacionados con el control del AR.Drone 2.0.
- Network: Parámetros relacionados con la red.
- Pic: Parámetros relacionados con la placa de navegación.
- Video: Parámetros relacionados con el video que proporciona el AR.Drone 2.0 a través de sus cámaras.
- Leds: Parámetros relacionados con las animaciones predefinidas que pueden realizarse con los leds.
- Detect: Parámetros relacionados con la detección de etiquetas a través de las cámaras.
- Syslog: Parámetros relacionados con el registro del sistema.
- Userbox: Parámetros relacionados con el almacenamiento de datos de navegación.
- GPS: Parámetros relacionados con el etiquetado de datos GPS recibidos desde el dispositivo de control remoto.
- Custom: Parámetros relacionados con el soporte a la multiconfiguración.

Parámetro	RW	Descripción
general:num_version_config	R	Versión del subsistema de configuración.
general:num_version_mb	R	Versión de la placa base.
general:num_version_soft	R	Versión del firmware.
general:drone_serial	R	Número de serie.
general:soft_build_date	R	Fecha de compilación del firmware.
general:motor1_soft	R	Versión del software de la placa del motor 1.
general:motor1_hard	R	Versión del hardware de la placa del motor 1.
general:motor1_supplier	R	Proveedor del motor 1.
general:motor2_soft	R	Versión del software de la placa del motor 2.
general:motor2_hard	R	Versión del hardware de la placa del motor 2.
general:motor2_supplier	R	Proveedor del motor 2.
general:motor3_soft	R	Versión del software de la placa del motor 3.
general:motor3_hard	R	Versión del hardware de la placa del motor 3.
general:motor3_supplier	R	Proveedor del motor 3.
general:motor4_soft	R	Versión del software de la placa del motor 4.
general:motor4_hard	R	Versión del hardware de la placa del motor 4.
general:motor4_supplier	R	Proveedor del motor 4.
general:ardrone_name	W	Nombre del AR.Drone
general:flying_time	R	Número de segundos de vuelo en la vida del AR.Drone.
general:navdata_demo	W	Indica si se envía una pequeña selección de datos de navegación o si se envían todos.
general:navdata_options	W	Esta configuración permite enviar otros datos de navegación cuando se está en el modo navdata_demo.

Parámetro	RW	Descripción
general:com_watchdog	W	Tiempo en segundos que el AR.Drone puede estar sin recibir algún comando del cliente. Si se sobrepasa este tiempo sin recibir comando alguno, el AR.Drone se queda en modo hovering.
general:video_enable	W	No utilizado. Debe estar a TRUE.
general:vision_enable	W	No utilizado. Debe estar a TRUE.
general:vbat_min	R	Mínimo nivel de batería antes de apagar el AR.Drone.
control:accs_offset	R	Offsets de los acelerómetros del AR.Drone.
control:accs_gains	R	Ganancias de los acelerómetros del AR.Drone.
control:gyros_offset	R	Offsets de los giróscopos del AR.Drone.
control:gyros_gains	R	Ganancias de los giróscopos del AR.Drone.
control:gyros110_offset	R	No indicado en la guía de desarrollo.
control:gyros110_gains	R	No indicado en la guía de desarrollo.
control:magneto_offset	R	No indicado en la guía de desarrollo.
control:magneto_radius	R	No indicado en la guía de desarrollo.
control:gyro_offset_thr_x	R	No indicado en la guía de desarrollo.
control:gyro_offset_thr_y	R	No indicado en la guía de desarrollo.
control:gyro_offset_thr_z	R	No indicado en la guía de desarrollo.
control:pwm_ref_gyros	R	No indicado en la guía de desarrollo.
control:osctun_value	R	No indicado en la guía de desarrollo.
control:osctun_test	R	No indicado en la guía de desarrollo.
control:altitude_max	W	Altura máxima de vuelo del AR.Drone.
control:altitude_min	W	Altura mínima de vuelo del AR.Drone.
control:control_level	W	Esta configuración permite describir cómo el AR.Drone interpretará los comandos AT*PCMD y AT*PCMD_MAG del usuario en relación al modo Combinado de Yaw. En este modo el AR.Drone rota sobre su eje Z cuando el usuario envía un comando de Roll, de modo que el cuadricóptero trata de orientar su parte frontal como si de un avión se tratase.
control:euler_angle_max	W	Máximo ángulo de inclinación en radianes, para Pitch y Roll.
control:control_iphone_tilt	W	Configuración para el iPhone.
control:control_vz_max	W	Máxima velocidad vertical en milímetros por segundo.
control:control_yaw	W	Máxima velocidad de rotación de Yaw en radianes por segundo.
control:outdoor	W	Indica si se está volando en el exterior o en el interior.
control:flight_without_shell	W	Indica si se está usando el casco de interior o de exterior.
control:autonomous_flight	W	No utilizado. Debe estar en FALSE.
control:manual_trim	R	Activo si se están utilizando trimados manuales. No utilizado.
control:indoor_euler_angle_max	W	Equivalente a control:euler_angle_max. Se utiliza cuando se está volando en el interior.
control:indoor_control_vz_max	W	Equivalente a control:control_vz_max. Se utiliza cuando se está volando en el interior.
control:indoor_control_yaw	W	Equivalente a control:control_yaw. Se utiliza cuando se está volando en el interior.
control:outdoor_euler_angle_max	W	Equivalente a control:euler_angle_max. Se utiliza cuando se está volando en el exterior.

Parámetro	RW	Descripción
control:outdoor_control_vz_max	W	Equivalente a control:control_vz_max. Se utiliza cuando se está volando en el exterior.
control:outdoor_control_yaw	W	Equivalente a control:control_yaw. Se utiliza cuando se está volando en el exterior.
control:flying_mode	W	Diferentes modos de vuelo utilizados en congresos. Para un uso normal debe estar a 0.
control:hovering_range	W	Configuración para un modo de vuelo especial utilizado en congresos.
control:flight_anim	W	Se utiliza para hacer que el AR.Drone realice animaciones predefinidas.
network:ssid_single_player	W	El SSID de la red Wi-Fi del AR.Drone.
network:ssid_multi_player	W	No utilizado.
network:wifi_mode	W	Modo de la red Wi-Fi. Access Point, Ad-Hoc o Station.
network:wifi_rate	W	Configuración de depuración que no debe ser modificada.
network:owner_mac	W	Dirección MAC emparejada con el AR.Drone.
pic:ultrasound_freq	W	Frecuencia de las medidas de los ultrasonidos para la altura. Sólo dos: 22.22 y 25 Hz.
pic:ultrasound_watchdog	W	Configuración de depuración que no debe ser modificada.
pic:pic_version	W	Versión del software de la placa de navegación.
video:camif_fps	R	FPS actual del interfaz de video.
video:codec_fps	W	FPS actual del codec de video. Máximo 30.
video:camif_buffers	R	Profundidad del buffer para el interfaz de video.
video:num_trackers	R	Número de puntos de seguimiento para la estimación de la velocidad.
video:video_codec	W	Codec de video del AR.Drone.
video:video_slices	W	Configuración de depuración que no debe ser modificada.
video:video_live_socket	W	Configuración de depuración que no debe ser modificada.
video:video_storage_space	R	Tamaño del buffer de grabación Wi-Fi de video.
video:bitrate	W	Bitrate de la transmisión de video en kilobits por segundo.
video:max_bitrate	W	Bitrate máximo que el dispositivo puede decodificar.
video:bitrate_ctrl_mode	W	Habilita o deshabilita el control automático del Bitrate. Sólo puede modificarse en Multiconfiguración.
video:bitrate_storage	W	Bitrate del flujo de grabación, tanto para grabación Wi-Fi como USB.
video:video_channel	W	Canal de video que será enviado al cliente. Puede ser la cámara horizontal o la vertical.
video:video_on_usb	W	Habilita la grabación del video en un dispositivo USB.
video:video_file_index	W	Número del último video grabado en el dispositivo USB.
leds:leds_anim	W	Se utiliza para activar las animaciones predefinidas de los leds.
detect:enemy_colors	W	Color de los cascos de los enemigos que se quieren detectar.
detect:groundstripe_colors	W	No utilizada.
detect:enemy_without_shell	W	Habilita o deshabilita la detección de cascos de exterior o interior.
detect:detect_type	W	Selecciona la etiqueta de detección activa.
detect:detections_select_h	W	Para la detección de etiquetas con la cámara horizontal.
detect:detections_select_v_hsync	W	Para la detección de etiquetas con la cámara vertical.
detect:detections_select_v	W	Para la detección de etiquetas con la cámara vertical.
syslog:output	-	No indicado en la guía de desarrollo.

Parámetro	RW	Descripción
syslog:max_size	-	No indicado en la guía de desarrollo.
syslog:nb_files	-	No indicado en la guía de desarrollo.
userbox:userbox_cmd	W	Permite el almacenamiento de datos de navegación.
gps:latitude	W	Latitud GPS enviada por el cliente únicamente para etiquetado de medios y grabación de datos de navegación.
gps:longitude	W	Longitud GPS enviada por el cliente únicamente para etiquetado de medios y grabación de datos de navegación.
gps:altitude	W	Altitud GPS enviada por el cliente únicamente para etiquetado de medios y grabación de datos de navegación.
custom:application_id	W	Establece el identificador de la aplicación actual. Si ya existe, cambia a la configuración relacionada con esta aplicación.
custom:application_desc	W	Establece la descripción de la aplicación actual.
custom:profile_id	W	Establece el identificador del usuario actual. Si ya existe, cambia a la configuración relacionada con este usuario.
custom:profile_desc	W	Establece la descripción del usuario actual.
custom:session_id	W	Establece el identificador de la sesión actual. Si ya existe, cambia a la configuración relacionada con esta sesión.
custom:session_desc	W	Establece la descripción de la sesión actual.

Tabla D.1: Parámetros de configuración del AR.Drone 2.0.

D.2. Información de estado

En la Tabla D.2 se describe el significado de los bits de estado del AR.Drone 2.0 que éste incluye en la información que envía al dispositivo de control remoto.

Nombre	Bit	Valores
Vuelo	0	(0) aterrizado, (1) en vuelo
Video	1	(0) deshabilitado, (1) habilitado
Visión	2	(0) deshabilitada, (1) habilitada
Algoritmo de control	3	(0) control de ángulos de Euler, (1) control de velocidades angulares
Algoritmo de control de altura	4	(0) deshabilitado, (1) habilitado
Realimentación del usuario	5	Estado del botón de Inicio
ACK de comandos de control	6	(0) ninguno, (1) recibido uno
Cámara habilitada	7	(0) sí, (1) no
Movimiento habilitado	8	(0) no, (1) sí
Dispositivo USB	9	(0) no preparado, (1) sí preparado
Demo de Navdata	10	(0) todos los Navdata, (1) sólo Navdata Demo
Inicialización de Navdata	11	(0) opciones de Navdata enviadas, (1) ninguna opción de Navdata enviada
Estado de los motores	12	(0) correcto, (1) incorrecto
Comunicación perdida	13	(0) sin problemas, (1) problema en la comunicación
	14	No se utiliza.
Nivel de batería bajo	15	(0) no, (1) demasiado bajo
Aterrizaje de emergencia por el usuario	16	(0) desactivado, (1) activado
Temporizador transcurrido	17	(0) no ha transcurrido, (1) ha transcurrido
Estado de calibración del magnetómetro	18	(0) no necesita calibración, (1) necesita calibración

Nombre	Bit	Valores
Ángulos	19	(0) dentro del rango, (1) fuera del rango
Viento	20	(0) correcto, (1) demasiado viento
Sensor de ultrasonidos	21	(0) correcto, (1) incorrecto
Detección de apagado del sistema	22	(0) no detectado, (1) detectado
Versión de la placa de navegación	23	(0) incorrecta, (1) correcta
Hilo de comandos AT	24	(0) no funcionando, (1) funcionando
Hilo de Navdata	25	(0) no funcionando, (1) funcionando
Hilo de Video	26	(0) no funcionando, (1) funcionando
Hilo de Adquisición	27	(0) no funcionando, (1) funcionando
Watchdog de Control	28	(0) bien planificado, (1) retraso en la ejecución
Watchdog de ADC	29	(0) correcto, (1) retraso en la UART2
Watchdog de Comunicación	30	(0) correcto, (1) problema de comunicación
Aterrizaje de emergencia	31	(0) no hay emergencia, (1) hay emergencia

Tabla D.2: Información de estado del AR.Drone 2.0.

D.3. Datos de navegación

La mayoría de opciones de Navdata no se encuentran documentadas y es necesario buscar en el código fuente del SDK para encontrar el significado de las variables que contienen. Aún así, de muchas de ellas no es sencillo interpretar su significado o las unidades en las que se expresan, por lo que este apartado va a describir, de la mejor manera posible, los datos de navegación teniendo en cuenta las limitaciones existentes.

En la Tabla D.3 se muestran los datos de navegación del *Navdata Demo* que proporciona los valores de las variables de navegación básicas.

Variable	Unidades	Descripción
ctrl_state	-	Estado de vuelo (Aterrizado, Volando, Hovering, etc)
vbat_flying_percentage	%	Porcentaje del voltaje de la batería
theta	m°	Ángulo de Pitch
phi	m°	Ángulo de Roll
psi	m°	Ángulo de Yaw
altitude	mm	Altura
vx	mm/s	Velocidad en el eje X (Pitch)
vy	mm/s	Velocidad en el eje Y (Roll)
vz	mm/s	Velocidad en el eje Z (Throttle)
num_frames	-	No se utiliza.
detection_camera_rot	-	No se utiliza.
detection_camera_trans	-	No se utiliza.
detection_tag_index	-	No se utiliza.
detection_camera_type	-	Tipo de etiqueta buscada en la detección de enemigos.
drone_camera_rot	-	No se utiliza.
drone_camera_trans	-	No se utiliza.

Tabla D.3: Datos de navegación de la opción Navdata Demo del AR.Drone 2.0.

En la Tabla D.4 se muestran los datos de navegación del *Navdata Cks* que proporciona la suma de comprobación de todas las opciones incluidas en la estructura Navdata.

Variable	Unidades	Descripción
cks	-	Checksum de todas las opciones incluidas en la estructura de NavData.

Tabla D.4: Datos de navegación de la opción Navdata Cks del AR.Drone 2.0.

En la Tabla D.5 se muestran los datos de navegación del *Navdata Time* proporciona el tiempo transcurrido desde que se encendió el AR.Drone 2.0.

Variable	Unidades	Descripción
time	s	Tiempo transcurrido

Tabla D.5: Datos de navegación de la opción Navdata Time del AR.Drone 2.0.

En la Tabla D.6 se muestran los datos de navegación del *Navdata Raw Measures* proporciona medidas en bruto de algunos de sus sensores.

Variable	Unidades	Descripción
raw_accs	-	Medidas filtradas de los acelerómetros
raw_gyros	-	Medidas filtradas de los giróscopos
raw_gyros_110	-	Medidas filtradas de los giróscopos X/Y de 110º
vbat_raw	mV	Voltaje de la batería
us_debut_echo	-	Variable relacionada con los ultrasonidos
us_fin_echo	-	Variable relacionada con los ultrasonidos
us_association_echo	-	Variable relacionada con los ultrasonidos
us_distance_echo	-	Variable relacionada con los ultrasonidos
us_courbe_temps	-	Variable relacionada con los ultrasonidos
us_courbe_valeur	-	Variable relacionada con los ultrasonidos
us_courbe_ref	-	Variable relacionada con los ultrasonidos
flag_echo_ini	-	Variable relacionada con los ultrasonidos
frame_number	-	Variable relacionada con los ultrasonidos
nb_echo	-	Variable relacionada con los ultrasonidos
sum_echo	-	Variable relacionada con los ultrasonidos
alt_temp_raw	mm	Medida de la altura en bruto.
gradient	-	Variable relacionada con los ultrasonidos

Tabla D.6: Datos de navegación de la opción Navdata Raw Measures del AR.Drone 2.0.

En la Tabla D.7 se muestran los datos de navegación del *Navdata Pressure Raw* que proporciona medidas en bruto de su sensor de presión.

Variable	Unidades	Descripción
up	-	Variable relacionada con el sensor de presión
ut	-	Variable relacionada con el sensor de presión
Temperature_meas	°	Medida de temperatura del sensor de presión
Pression_meas	Pa	Medida de presión del sensor de presión

Tabla D.7: Datos de navegación de la opción Navdata Pressure Raw del AR.Drone 2.0.

En la Tabla D.8 se muestran los datos de navegación del *Navdata Magneto* que proporciona valores del magnetómetro.

Variable	Unidades	Descripción
mx	-	Variable interna del magnetómetro
my	-	Variable interna del magnetómetro
mz	-	Variable interna del magnetómetro

Variable	Unidades	Descripción
magneto_raw	mG	Medida en bruto de los campos magnéticos en x, y, z
magneto_rectified	mG	Medida rectificada de los campos magnéticos en x, y, z
magneto_offset	mG	Offset de la medida de los campos magnéticos en x, y, z
heading_unwrapped	°	Orientación obtenida del magnetómetro
heading_gyro_unwrapped	°	Orientación obtenida de los giróscopos
heading_fusion_unwrapped	m°	Orientación obtenida fusionando las orientaciones del magnetómetro y de los giróscopos
magneto_calibration_ok	BOOLEAN	Calibración correcta del magnetómetro
magneto_state	-	Estado del magnetómetro
magneto_radius	mG	Radio del magnetómetro
error_mean	mG	Desviación estándar del error en la medida del magnetómetro
error_var	mG^2	Varianza del error en la medida del magnetómetro

Tabla D.8: Datos de navegación de la opción Navdata Magneto del AR.Drone 2.0.

En la Tabla D.9 se muestran los datos de navegación del *Navdata Wind Speed* que proporciona valores de variables relacionadas con la medida del viento.

Variable	Unidades	Descripción
wind_speed	m/s	Velocidad del viento estimada
wind_angle	°	Ángulo del viento estimado en el marco Norte-Este
wind_compensation_theta	rad	Ángulo de compensación al Pitch provocado por el viento
wind_compensation_phi	rad	Ángulo de compensación al Roll provocado por el viento
state_x1	-	Variable relacionada con la medición del viento
state_x2	-	Variable relacionada con la medición del viento
state_x3	-	Variable relacionada con la medición del viento
state_x4	-	Variable relacionada con la medición del viento
state_x5	-	Variable relacionada con la medición del viento
state_x6	-	Variable relacionada con la medición del viento
magneto_debug1	-	Variable relacionada con el magnetómetro
magneto_debug2	-	Variable relacionada con el magnetómetro
magneto_debug3	-	Variable relacionada con el magnetómetro

Tabla D.9: Datos de navegación de la opción Navdata Wind Speed del AR.Drone 2.0.

En la Tabla D.10 se muestran los datos de navegación del *Navdata Kalman Pressure* que proporciona valores de variables relacionadas con la determinación de la altura.

Variable	Unidades	Descripción
offset_pressure	Pa	Offset de la medida de presión
est_z	mm	Altura estimada a través de la presión
est_zdot	m/s	Velocidad estimada en el eje Z a través de la presión
est_bias_PWM	m/s^2	Variable relacionada con la determinación de la altura a través de la presión
est_biais_pression	Pa	Tendencia estimada de la presión
offset_US	m	Variable relacionada con la determinación de la altura a través del sensor de ultrasonidos
prediction_US	mm	Variable relacionada con la determinación de la altura a través del sensor de ultrasonidos
cov_alt	m	Covarianza de la medida de altura

Variable	Unidades	Descripción
cov_PWM	-	Covarianza de un valor relacionado con el PWM
cov_vitesse	m/s	Covarianza de la medida de la velocidad en el eje Z
bool_effet_sol	-	Variable relacionada con el efecto suelo
somme_inno	mm	
flag_rejet_US	-	Variable relacionada con el sensor de ultrasonidos
u_multisinus	-	
gaz_altitude	-	
Flag_multisinus	-	
Flag_multisinus_debut	-	

Tabla D.10: Datos de navegación de la opción Navdata Kalman Pressure del AR.Drone 2.0.

En la Tabla D.11 se muestran los datos de navegación del *Navdata Phys Measures* que proporciona valores de las medidas físicas.

Variable	Unidades	Descripción
accs_temp	°K	Temperatura del acelerómetro
gyro_temp	°K	Temperatura del giróscopo
phys_accs	mg	Medida física de la aceleración de los ejes x, y, z (g=9.8 m/s^2)
phys_gyros	°/s	Medida física de la velocidad de giro de los ejes x, y, z
alim3V3	-	Variable relacionada con la alimentación del giróscopo
vrefEpson	-	Variable relacionada con la alimentación del giróscopo
vrefIDG	-	Variable relacionada con la alimentación del giróscopo

Tabla D.11: Datos de navegación de la opción Navdata Phys Measures del AR.Drone 2.0.

En la Tabla D.12 se muestran los datos de navegación del *Navdata Gyros Offsets* que proporciona los valores de los offsets del giróscopo.

Variable	Unidades	Descripción
offset_g	°/s	Offsets de la velocidad de giro de los ejes x, y, z

Tabla D.12: Datos de navegación de la opción Navdata Gyros Offsets del AR.Drone 2.0.

En la Tabla D.13 se muestran los datos de navegación del *Navdata Euler Angles* que proporciona los valores de dos ángulos de Euler.

Variable	Unidades	Descripción
theta_a	m°	Ángulo de Euler del Pitch
phi_a	m°	Ángulo de Euler del Roll

Tabla D.13: Datos de navegación de la opción Navdata Euler Angles del AR.Drone 2.0.

En la Tabla D.14 se muestran los datos de navegación del *Navdata References* que proporciona valores de variables relacionadas con las referencias del control interno.

Variable	Unidades	Descripción
ref_theta	m°	Referencia para el control interno del ángulo de Euler del Pitch
ref_phi	m°	Referencia para el control interno del ángulo de Euler del Roll
ref_theta_I	m°	Referencia interna para el control interno del ángulo de Euler del Pitch

Variable	Unidades	Descripción
ref_phi_I	m°	Referencia interna para el control interno del ángulo de Euler del Roll
ref_pitch	m°	Referencia para el control interno del Pitch
ref_roll	m°	Referencia para el control interno del Roll
ref_yaw	m°/s	Referencia para el control interno de la velocidad de Yaw
ref_psi	m°	Referencia para el control interno del Yaw
vx_ref	mm/s	Referencia para el control interno de la velocidad lineal en el eje X
vy_ref	mm/s	Referencia para el control interno de la velocidad lineal en el eje Y
theta_mod	rad	Referencia para el control interno según el modelo del ángulo de Euler del Pitch
phi_mod	rad	Referencia para el control interno según el modelo del ángulo de Euler del Roll
k_v_x	-	Ganancia del control interno en relación a la velocidad lineal en el eje X
k_v_y	-	Ganancia del control interno en relación a la velocidad lineal en el eje Y
k_mode	-	Modo de la ganancia del control interno en relación a las velocidades lineales del eje X e Y
ui_time	-	
ui_theta	-	
ui_phi	-	
ui_psi	-	
ui_psi_accuracy	-	
ui_seq	-	

Tabla D.14: Datos de navegación de la opción Navdata References del AR.Drone 2.0.

En la Tabla D.15 se muestran los datos de navegación del *Navdata Trims* que proporciona valores del trimado de ángulos de Euler.

Variable	Unidades	Descripción
angular_rates_trim_r	m°/s	Trimado de la velocidad del ángulo de Euler del Yaw
euler_angles_trim_theta	m°	Trimado del ángulo de Euler del Pitch
euler_angles_trim_phi	m°	Trimado del ángulo de Euler del Roll

Tabla D.15: Datos de navegación de la opción Navdata Trims del AR.Drone 2.0.

En la Tabla D.16 se muestran los datos de navegación del *Navdata RC References* que proporciona valores de variables relacionadas con las referencias del control externo.

Variable	Unidades	Descripción
rc_ref_pitch	-	Referencia de Pitch del control remoto
rc_ref_roll	-	Referencia de Roll del control remoto
rc_ref_yaw	-	Referencia de Yaw Speed del control remoto
rc_ref_gaz	-	Referencia de Throttle del control remoto
rc_ref_ag	-	

Tabla D.16: Datos de navegación de la opción Navdata RC References del AR.Drone 2.0.

En la Tabla D.17 se muestran los datos de navegación del *Navdata PWM* que proporciona valores de variables relacionadas con el lazo de control interno asociado a los motores y con el lazo de control interno de la altura.

Variable	Unidades	Descripción
motor1	-	Señal PWM enviada al motor 1
motor2	-	Señal PWM enviada al motor 2
motor3	-	Señal PWM enviada al motor 3
motor4	-	Señal PWM enviada al motor 4
sat_motor1	-	Saturación de la señal PWM enviada al motor 1
sat_motor2	-	Saturación de la señal PWM enviada al motor 2
sat_motor3	-	Saturación de la señal PWM enviada al motor 3
sat_motor4	-	Saturación de la señal PWM enviada al motor 4
gaz_feed_forward	-	FeedForward del Throttle
gaz_altitude	-	PWM derivado del Throttle
altitude_integral	-	Variable relacionada con el PID del control de altura
vz_ref	mm/s	Referencia para el control interno de la velocidad lineal en el eje Z
u_pitch	-	
u_roll	-	
u_yaw	-	
yaw_u_I	-	
u_pitch_planif	-	
u_roll_planif	-	
u_yaw_planif	-	
u_gaz_planif	-	
current_motor1	mA	Intensidad que circula por el motor 1
current_motor2	mA	Intensidad que circula por el motor 2
current_motor3	mA	Intensidad que circula por el motor 3
current_motor4	mA	Intensidad que circula por el motor 4
altitude_prop	-	Variable relacionada con el PID del control de altura
altitude_der	-	Variable relacionada con el PID del control de altura

Tabla D.17: Datos de navegación de la opción Navdata PWM del AR.Drone 2.0.

En la Tabla D.18 se muestran los datos de navegación del *Navdata Altitude* que proporciona valores de variables relacionadas con la medida de la altura.

Variable	Unidades	Descripción
altitude_vision	mm	Altura según el módulo de visión
altitude_vz	mm/s	Velocidad en el eje Z
altitude_ref	mm	Referencia de la altura
altitude_raw	mm	Medida de la altura en bruto
obs_accZ	m/s^2	Medida de la aceleración en el eje Z proporcionada por el observador
obs_alt	m	Medida de la altura proporcionada por el observador
obs_x	-	Estimación de la altura (m), la velocidad en el eje Z (m/s) y la aceleración en el eje Z (m/s^2) proporcionada por el observador
obs_state	-	Estado del observador
est_vb	-	
est_state	-	Estado del observador de vuelo

Tabla D.18: Datos de navegación de la opción Navdata Altitude del AR.Drone 2.0.

En la Tabla D.19 se muestran los datos de navegación del *Navdata Vision Raw* que proporciona valores en bruto de variables del módulo de visión.

Variable	Unidades	Descripción
vision_tx_raw	-	Variable relacionada con las medidas en bruto del módulo de visión
vision_ty_raw	-	Variable relacionada con las medidas en bruto del módulo de visión
vision_tz_raw	-	Variable relacionada con las medidas en bruto del módulo de visión

Tabla D.19: Datos de navegación de la opción Navdata Vision Raw del AR.Drone 2.0.

En la Tabla D.20 se muestran los datos de navegación del *Navdata Vision* que proporciona valores de variables del módulo de visión.

Variable	Unidades	Descripción
vision_state	-	Estado del módulo de visión
vision_misc	-	Variable relacionada con el módulo de visión
vision_phi_trim	rad	Trimado del ángulo de Euler del Roll a través del módulo de visión
vision_phi_ref_prop	rad	Referencia del ángulo de Euler del Roll dentro del módulo de visión
vision_theta_trim	rad	Trimado del ángulo de Euler del Pitch a través del módulo de visión
vision_theta_ref_prop	rad	Referencia del ángulo de Euler del Pitch dentro del módulo de visión
new_raw_picture	-	Variable relacionada con una captura de imagen del módulo de visión
theta_capture	-	Variable relacionada con una captura de imagen del módulo de visión
phi_capture	-	Variable relacionada con una captura de imagen del módulo de visión
psi_capture	-	Variable relacionada con una captura de imagen del módulo de visión
altitude_capture	-	Variable relacionada con una captura de imagen del módulo de visión
time_capture	s	Tiempo en el que se ha realizado una captura de imagen del módulo de visión
body_v	mm/s	Velocidades lineales en el eje X, Y, Z determinadas por el módulo de visión
delta_phi	-	Variable relacionada con el módulo de visión
delta_theta	-	Variable relacionada con el módulo de visión
delta_psi	-	Variable relacionada con el módulo de visión
gold_defined	-	
gold_reset	-	
gold_x	-	
gold_y	-	

Tabla D.20: Datos de navegación de la opción Navdata Vision del AR.Drone 2.0.

En la Tabla D.21 se muestran los datos de navegación del *Navdata Vision Perf* que proporciona valores de variables relacionadas con el rendimiento del módulo de visión.

Variable	Unidades	Descripción
time_szo	ms	Tiempo transcurrido al hacer alguna operación el módulo de visión

Variable	Unidades	Descripción
time_corners	ms	Tiempo transcurrido en la obtención de esquinas
time_compute	ms	Tiempo transcurrido en el cálculo
time_tracking	ms	Tiempo transcurrido en el seguimiento de los puntos
time_trans	ms	Tiempo transcurrido al hacer alguna operación el módulo de visión
time_update	ms	Tiempo transcurrido al hacer alguna operación el módulo de visión
time_custom	ms	Tiempo transcurrido al hacer alguna operación el módulo de visión

Tabla D.21: Datos de navegación de la opción Navdata Vision Perf del AR.Drone 2.0.

En la Tabla D.22 se muestran los datos de navegación del *Navdata Trackers Send* que proporciona valores de variables relacionadas con el seguimiento de puntos del módulo de visión.

Variable	Unidades	Descripción
locked	-	Variable relacionada con el seguimiento de puntos del módulo de visión
point	-	Puntos en seguimiento del módulo de visión

Tabla D.22: Datos de navegación de la opción Navdata Trackers Send del AR.Drone 2.0.

En la Tabla D.23 se muestran los datos de navegación del *Navdata Vision Detect* que proporciona valores de variables relacionadas con la detección de etiquetas por parte del módulo de visión.

Variable	Unidades	Descripción
nb_detected	-	Indicador de si se ha detectado una nueva etiquetas
type	-	Tipo de etiqueta detectada
xc	mm	Coordenada X de la posición de la etiqueta detectada
yc	mm	Coordenada Y de la posición de la etiqueta detectada
width	mm	Anchura de la etiqueta detectada
height	mm	Altura de la etiqueta detectada
dist	mm	Distancia a la etiqueta detectada
orientation_angle	°	Orientación hacia la etiqueta detectada
rotation	-	Variable relacionada con la cámara que ha detectado la etiqueta
translation	-	Variable relacionada con la cámara que ha detectado la etiqueta
camera_source	-	Cámara que ha detectado la etiqueta

Tabla D.23: Datos de navegación de la opción Navdata Vision Detect del AR.Drone 2.0.

En la Tabla D.24 se muestran los datos de navegación del *Navdata Wind Of*.

Variable	Unidades	Descripción
of_dx	-	
of_dy	-	

Tabla D.24: Datos de navegación de la opción Navdata Vision Of del AR.Drone 2.0.

En la Tabla D.25 se muestran los datos de navegación del *Navdata Watchdog* que proporciona valores de variables relacionadas con el watchdog del AR.Drone.

Variable	Unidades	Descripción
watchdog	-	Variable relacionada con el Watchdog

Tabla D.25: Datos de navegación de la opción Navdata Watchdog del AR.Drone 2.0.

En la Tabla D.26 se muestran los datos de navegación del *Navdata ADC Data Frame* que proporciona valores de variables relacionadas con el convertidor analógico digital.

Variable	Unidades	Descripción
version	-	
data_frame	-	

Tabla D.26: Datos de navegación de la opción Navdata ADC Data Frame del AR.Drone 2.0.

En la Tabla D.27 se muestran los datos de navegación del *Navdata Video Stream* que proporciona valores de variables de flujo de video.

Variable	Unidades	Descripción
quant	-	Referencia del cuantificador utilizada para codificar los fotogramas.
frame_size	bytes	Tamaño del fotograma
frame_number	-	Índice del fotograma
atcmd_ref_seq	-	Referencia al número secuencial de los comandos AT
atcmd_mean_ref_gap	ms	Desviación estándar entre dos comandos AT*REF consecutivos
atcmd_var_ref_gap	$(ms)^2$	Varianza entre dos comandos AT*REF consecutivos
atcmd_ref_quality	-	Estimación de la calidad del enlace de comunicación de los comandos AT
out_bitrate	-	Rendimiento de salida medido del puerto TCP de video
desired_bitrate	-	Tamaño del último fotograma generado por el codificador de video
data1	-	Dato temporal relacionado con el flujo de video
data2	-	Dato temporal relacionado con el flujo de video
data3	-	Dato temporal relacionado con el flujo de video
data4	-	Dato temporal relacionado con el flujo de video
data5	-	Dato temporal relacionado con el flujo de video
tcp_queue_level	-	Variable relacionada con el uso de colas en el flujo de video
fifo_queue_level	-	Variable relacionada con el uso de colas en el flujo de video

Tabla D.27: Datos de navegación de la opción Navdata Video Stream del AR.Drone 2.0.

En la Tabla D.28 se muestran los datos de navegación del *Navdata HDVideo Stream* que proporciona valores de variables de flujo de video en HD.

Variable	Unidades	Descripción
hdvideo_state	-	Estado del video HD
storage_fifo_nb_packets	-	Variable relacionada con el uso de colas en el flujo de video HD
storage_fifo_size	-	Variable relacionada con el uso de colas en el flujo de video HD
usbkey_size	kbytes	Tamaño del dispositivo de grabación USB
usbkey_freespace	kbytes	Tamaño disponible del dispositivo de grabación USB

Variable	Unidades	Descripción
frame_number	-	Número del fotograma inicial de la codificación del video HD
usbkey_remaining_time	s	Tiempo de grabación disponible del dispositivo de grabación USB

Tabla D.28: Datos de navegación de la opción Navdata HDVideo Stream del AR.Drone 2.0.

En la Tabla D.29 se muestran los datos de navegación del *Navdata Games* que proporciona valores de variables relacionadas con los juegos predefinidos del AR.Drone 2.0.

Variable	Unidades	Descripción
double_tap_counter	-	Variable relacionada con los juegos predefinidos
finish_line_counter	-	Variable relacionada con los juegos predefinidos

Tabla D.29: Datos de navegación de la opción Navdata Games del AR.Drone 2.0.

En la Tabla D.30 se muestran los datos de navegación del *Navdata Wifi* que proporciona valores de variables relacionadas con la calidad del enlace de comunicación Wi-Fi.

Variable	Unidades	Descripción
link_quality	-	Variable relacionada con la calidad del enlace de comunicación Wi-Fi

Tabla D.30: Datos de navegación de la opción Navdata Wifi del AR.Drone 2.0.