

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

TRABAJO FIN DE GRADO

DISEÑO Y CONSTRUCCIÓN DE UN COHETE CON ATERRIZAJE PROPULSIVO

Autor: Héctor Sánchez Valencia
Director: Juan Luis Zamora Macho

Madrid
Julio de 2026

Declaración de originalidad

Declaro bajo mi responsabilidad que el Proyecto presentado con el título **Diseño y construcción de un cohete con aterrizaje propulsivo** en la ETS de Ingeniería – ICAI de la Universidad Pontificia Comillas en el curso académico **2025/2026** es de mi autoría y no ha sido presentado con anterioridad a otros efectos. El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada de otros documentos está debidamente referenciada.

Uso de Inteligencia Artificial[1]

Declaro bajo mi responsabilidad que (indicar la opción correcta):

☐ No he utilizado Inteligencia Artificial en la elaboración del presente documento.

☒ He utilizado Inteligencia Artificial en la elaboración del presente documento y/o del Anexo B siempre en las condiciones permitidas por la Universidad Pontificia Comillas, es decir, aplicando el Nivel 2 de la Escala de Evaluación de Perkins et al. (2024): "*La IA puede utilizarse para actividades previas a la tarea, como la lluvia de ideas, la descripción y la investigación inicial. Este nivel se centra en el uso de la IA para la planificación, las síntesis y la generación de ideas, pero las evaluaciones deben hacer hincapié en la capacidad de desarrollar y refinar estas ideas de forma independiente*". En concreto, las Inteligencia Artificial ha sido empleada para:

(indicar aquí el uso concreto que se ha hecho de la Inteligencia Artificial)
La IA me ha asistido en diferentes tareas del TFG como en búsqueda de componentes y comparación entre ellos o búsqueda documental, pero la planificación, supervisión y ejecución de las principales tareas del proyecto son mías.


Firmado (alumno): Héctor Sánchez Valencia
Fecha: 14/07/2026

Autorización para la entrega del Proyecto

El Director del Proyecto	El co-Director del Proyecto (si aplica)
	
Fdo: Juan Luis Zamora Macho	Fdo:
Fecha: 15/07/2026	Fecha:

[1] Esta declaración se refiere al uso de la Inteligencia Artificial generativa para realizar los documentos del Proyecto (Anexo B y Memoria). No aplica a Proyectos donde, por su naturaleza, deban emplear inteligencia artificial como parte de los mismos (aplicación de técnicas de aprendizaje automático, redes neuronales, análisis de datos…)

Resumen

Este Trabajo Fin de Grado estudia el aterrizaje retropropulsivo de un cohete mediante control vectorial de empuje (TVC). Se toma como referencia el cohete de competición del IRT 2026, documentado por el ICAI Rocket Team (IRT26), y se sustituye conceptualmente su recuperación convencional por una arquitectura propia con tobera orientable, aerofrenos, estimación de estado y una segunda carga sólida de descenso. El objetivo es disponer de una plataforma reproducible con la que analizar la dinámica, el control y la futura integración física antes de realizar ensayos de vuelo.

Se ha desarrollado un simulador no lineal de seis grados de libertad en MATLAB y una arquitectura equivalente en Simulink. El estado utiliza posición NED, velocidad en ejes cuerpo, cuaternión $[q_0, q_1, q_2, q_3]$ y velocidad angular. El modelo incorpora masa e inercia variables, atmósfera, viento, aerodinámica, actuadores no ideales y una curva equivalente al motor Cesaroni Pro38 I212. El control estabiliza la actitud a partir del estado estimado y decide la ignición irreversible de la carga de descenso, pues un motor sólido no admite regulación continua. La planta se apoya en la documentación de OpenRocket (NISK13) y en la formulación TVC de dos Santos y Oliveira (SANT24); el estimador sigue el esquema recursivo de predicción y corrección del filtro de Kalman (KALM60). También se ha preparado un banco con Raspberry Pi 3B+, sensores I2C, servos PWM y una PCB de aviónica, manteniendo la ignición deshabilitada.

En la simulación de referencia, con paso de $0{,}01\,s$ y modo de aterrizaje, el contacto se produjo en $31{,}91\,s$, con una altitud máxima de $559{,}05\,m$, velocidad máxima de $114{,}40\,m/s$ y velocidad vertical final de $-0{,}85\,m/s$. El EKF obtuvo un error RMS de altitud de $0{,}047\,m$. En el banco Raspberry se registraron 821 muestras durante $30{,}00\,s$; el PWM permaneció en los valores seguros de $1500\,\mu s$ para TVC y $500\,\mu s$ para el aerofreno.

Los resultados validan la coherencia de la cadena de simulación, control, estimación y adquisición de banco. Este trabajo sienta una base técnica sólida y verificable para la futura construcción e integración de un cohete capaz de realizar un aterrizaje propulsivo.

Referencias citadas:

- (IRT26) ICAI Rocket Team, reporte técnico final SURTAM 2026.
- (NISK13) S. Niskanen, documentación técnica de OpenRocket.
- (SANT24) P. dos Santos y P. Oliveira, arquitectura integrada de navegación y control.
- (KALM60) R. E. Kalman, formulación recursiva de filtrado y predicción.

Palabras clave: cohete, aterrizaje retropropulsivo, TVC, MATLAB, Simulink, dinámica 6DOF, Raspberry Pi, EKF, aerofrenos.

Abstract

This Bachelor's Thesis studies the retropropulsive landing of a rocket using thrust vector control (TVC). The IRT 2026 competition rocket, documented by the ICAI Rocket Team (IRT26), is used as the reference vehicle, while its conventional recovery is conceptually replaced by a project-specific architecture with a gimballed nozzle, airbrakes, state estimation and a second solid landing charge. The objective is to provide a reproducible platform for analysing dynamics, control and future physical integration before flight testing.

A nonlinear six-degree-of-freedom simulator was developed in MATLAB and transferred to an equivalent Simulink architecture. The state contains NED position, body-frame velocity, quaternion $[q_0, q_1, q_2, q_3]$ and angular velocity. The model includes variable mass and inertia, atmosphere, wind, aerodynamics, non-ideal actuators and an equivalent Cesaroni Pro38 I212 thrust curve. The controller stabilizes attitude from the estimated state and decides the irreversible ignition of the landing charge, since a solid motor cannot provide continuously throttled thrust. The plant uses the OpenRocket technical documentation (NISK13) and the TVC formulation by dos Santos and Oliveira (SANT24); the estimator follows the recursive prediction–correction scheme introduced by Kalman (KALM60). A Raspberry Pi 3B+ bench with I2C sensors, PWM servos and an avionics PCB was also prepared, with ignition disabled.

In the reference simulation, using a $0.01\,s$ step and landing mode, ground contact occurred at $31.91\,s$. Maximum altitude was $559.05\,m$, maximum speed was $114.40\,m/s$, and final vertical speed was $-0.85\,m/s$. The EKF achieved an altitude RMS error of $0.047\,m$. The Raspberry bench recorded 821 samples over $30.00\,s$, while PWM remained at the safe values of $1500\,\mu s$ for TVC and $500\,\mu s$ for the airbrake.

These results validate the consistency of the simulation, control, estimation and bench-acquisition chain. This work establishes a solid and verifiable technical foundation for the future construction and integration of a rocket capable of performing a propulsive landing.

References cited:

- (IRT26) ICAI Rocket Team, final SURTAM 2026 technical report.
- (NISK13) S. Niskanen, OpenRocket technical documentation.
- (SANT24) P. dos Santos and P. Oliveira, integrated navigation and control architecture.
- (KALM60) R. E. Kalman, recursive filtering and prediction formulation.

Keywords: rocket, retropropulsive landing, TVC, MATLAB, Simulink, 6DOF dynamics, Raspberry Pi, EKF, airbrakes.

Dedico este trabajo a mis padres, a mis abuelos
y a toda mi familia, por haber creído siempre en mí
y por acompañarme durante toda la carrera.

Non est ad astra mollis e terris via.
No hay camino fácil de la tierra a las estrellas.
SÉNECA, HERCULES FURENS, V. 437 (SENE87)

We choose to go to the Moon in this decade
and do the other things, not because they are easy,
but because they are hard.
Elegimos ir a la Luna en esta década
y hacer las demás cosas, no porque sean fáciles,
sino porque son difíciles.
JOHN F. KENNEDY, RICE UNIVERSITY, 1962 (KENN62)

Agradecimientos

En primer lugar, quiero agradecer a todo el ICAI Rocket Team su trabajo y su apoyo durante el desarrollo de este proyecto. Sin el equipo, este TFG no habría sido posible. En particular, quiero dar las gracias a Mario Mora, Belén Gonzalo, Inés Thepaut, Agnès Ribera y Carolina Pueche por el diseño y la construcción del modelo de cohete que sirve de base para este trabajo.

En segundo lugar, quiero agradecer a mi director del TFG, Juan Luís Zamora, su ayuda y orientación para sacar adelante este proyecto. También quiero dar las gracias a Daniel Muñoz Frías por su ayuda con la PCB y a Romano Giannetti por su apoyo con la célula de carga.

Por último, doy las gracias a Valeria Ortiz por ayudarme a hacer alguna prueba y por escucharme divagar tantas veces sobre mis avances del TFG.

Índice general

1. Física del vuelo del cohete **13**
1.1. Modelo de referencia en OpenRocket . . . 13
1.2. Modelo dinámico implementado en MATLAB . . . 14
1.3. Fuerzas que actúan sobre el cohete . . . 15
1.3.1. Empuje y control vectorial . . . 15
1.3.2. Gravedad . . . 17
1.3.3. Aerodinámica . . . 17
1.4. Momentos y estabilidad . . . 18
1.5. Masa variable e inercias . . . 19
1.6. Viento, atmósfera y condiciones de lanzamiento . . . 19
1.7. Relación con el control y el aterrizaje retropropulsivo . . . 21

2. Hardware **23**
2.1. Selección de componentes . . . 23
2.1.1. Requisitos generales del sistema . . . 23
2.1.2. Unidad de control de vuelo . . . 23
2.1.3. Sensores . . . 24
2.1.3.1. Unidad de medición inercial (IMU) . . . 24
2.1.3.2. Barómetro . . . 24
2.1.3.3. Magnetómetro . . . 25
2.1.4. Actuadores . . . 25
2.1.4.1. Servomotores . . . 25
2.1.4.2. Señales PWM de actuación . . . 26
2.1.4.3. Aerofreno . . . 27
2.1.5. Tobera y sistema de propulsión . . . 28
2.1.6. Almacenamiento y telemetria de banco en la Raspberry Pi . . . 30
2.1.7. Sistema de alimentación . . . 30
2.1.8. Resumen de componentes . . . 31
2.2. Subsistema de célula de carga . . . 33
2.2.1. Función dentro del proyecto . . . 33
2.2.2. Requisitos de diseño . . . 33
2.2.3. Sistema de medida . . . 33
2.2.4. Integración mecánica . . . 36
2.2.5. Calibración y tratamiento de señal . . . 37
2.3. Desarrollo de la PCB . . . 37
2.3.1. Objetivo y alcance . . . 37
2.3.2. Requisitos de diseño . . . 37
2.3.3. Arquitectura eléctrica . . . 38
2.3.4. Configuracion de buses de sensores . . . 38
2.3.5. Interfaz con actuadores TVC . . . 40

2.3.6. Criterios de layout . . . 40
2.3.7. Verificación previa a fabricación . . . 40
2.3.8. Validación funcional inicial . . . 40

3. Diseño mecánico del cohete **41**
3.1. Arquitectura mecánica . . . 41
3.2. Criterios de diseño . . . 42
3.3. Estructuras principales del cohete . . . 42
3.4. Aerofrenos y elementos desplegables . . . 43
3.5. Estabilidad pasiva y modificaciones realizadas . . . 44
3.6. Tobera y sistema TVC . . . 45
3.6.1. Función de la tobera . . . 46
3.6.2. Criterios de diseño de la tobera . . . 47
3.6.3. Límites de actuación . . . 47
3.6.4. Interfaz con simulación y control . . . 48

4. Programa de MATLAB **49**
4.1. Objetivo y alcance del programa . . . 49
4.2. Estructura del software . . . 50
4.3. Vector de estado y convenciones . . . 50
4.4. Modelo dinámico 6DOF . . . 51
4.5. Fuerzas y momentos implementados . . . 52
4.5.1. Empuje con TVC . . . 52
4.5.2. Gravedad . . . 53
4.5.3. Viento y velocidad relativa . . . 53
4.6. Atmósfera ISA . . . 54
4.7. Modelo aerodinámico . . . 55
4.7.1. Arrastre . . . 55
4.7.2. Fuerza normal y momento aerodinamico . . . 56
4.7.3. Amortiguamiento aerodinámico . . . 57
4.8. Propiedades másicas variables . . . 57
4.9. Integración temporal y eventos . . . 58
4.10. Perfiles de control en lazo abierto . . . 59
4.11. Autopiloto TVC en lazo cerrado . . . 60
4.11.1. Medida geometrica de inclinacion . . . 60
4.11.2. Filtrado de sensores . . . 61
4.11.3. Ley PD con autoridad TVC . . . 61
4.11.4. Dinámica de servo, retardo y PWM . . . 61
4.12. GUI de simulación . . . 62
4.13. Modelo Simulink . . . 62
4.13.1. Modelo de sensores y EKF con ruido simulado . . . 64
4.14. Programa de adquisicion de datos y banco Raspberry Pi . . . 66
4.15. Rama de control LQR . . . 67
4.16. Validación disponible . . . 68
4.17. Pseudocódigo del flujo principal . . . 69

5. Resultados **71**
5.1. Simulación de vuelo . . . 71
5.2. Control . . . 72
5.3. Célula de carga . . . 73
5.4. Ordenador de vuelo y sensores . . . 74

6. Conclusiones **77**
6.1. Conclusiones técnicas . . . 77
6.2. Limitaciones . . . 77
6.3. Trabajo futuro . . . 78

Bibliografía **79**

A. Alineación con los Objetivos de Desarrollo Sostenible **83**
A.1. Planteamiento original recogido en el Anexo B . . . 83
A.1.1. Objetivo 12: Producción y consumo responsables . . . 83
A.1.2. Objetivo 9: Industria, innovación e infraestructura . . . 83
A.2. Cumplimiento específico en el trabajo realizado . . . 84
A.2.1. Contribución al ODS 12 . . . 84
A.2.2. Contribución al ODS 9 . . . 84
A.3. Alcance de la contribución . . . 85

B. Programa MATLAB y Simulink **87**
B.1. Criterio de documentacion del codigo . . . 87
B.2. Mapa funcional del software . . . 87
B.3. Flujo principal de simulacion MATLAB . . . 88
B.4. Planta 6DOF y convenciones de estado . . . 88
B.5. Mando, actuadores y limites . . . 89
B.6. Modelo Simulink-first . . . 89
B.6.1. Bloques MATLAB Function del modelo . . . 91
B.6.2. GUI Simulink-first . . . 91
B.7. Programa de adquisicion Raspberry . . . 91
B.8. Validacion asociada . . . 92

C. Reporte técnico SURTAM 2026 del ICAI Rocket Team **93**

D. Enlaces a hojas de características **135**

E. Planos **137**
E.1. Esquemático de la caja de electrónica . . . 138
E.2. Plano eléctrico de la PCB de aviónica con Raspberry Pi 3B+ . . . 139
E.3. Layout de la PCB de aviónica con buses I2C/SPI seleccionables . . . 140
E.4. Vista lateral del cohete de competición del IRT 2026 en OpenRocket . . . 141
E.5. Esquema LTspice del circuito final de acondicionamiento y adquisición del puente de galgas . . . 142

Índice de figuras

1. Vista lateral del cohete de competición del IRT 2026 en OpenRocket, usada como referencia geometrica y de estabilidad. Fuente: ICAI Rocket Team (IRT26). 14
2. Curva de empuje del motor Cesaroni $364I212\text{-}14A$ generada a partir de los puntos RASP de fabricante publicados por ThrustCurve.org (THRU06). 17
3. Raspberry Pi 3 Model B+ seleccionada como computador de vuelo del prototipo. Fuente: Raspberry Pi Ltd. (RASP26a). 24
4. Esquema de tobera orientable con TVC y fuerzas aerodinamicas asociadas. Fuente: dos Santos y Oliveira (SANT24). 29
5. Mecanismo TVC tomado como referencia de diseno con servos y soporte impreso. Fuente: BPS.Space (BPSS26). 29
6. Placa de expansión Geekworm X728 para alimentación con celdas 18650. Fuente: Geekworm (GEEK26). 30
7. Celdas JESSPOW 18650 *button top* de $3{,}7\,V$ y $3300\,mAh$ empleadas como referencia de batería. Fuente: JESSPOW (JESS26). 31
8. Esquema LTspice del circuito final de acondicionamiento y adquisición del puente de galgas, con AD620 y etapas ADTL082. 35
9. Barrido transitorio de la salida del circuito final de adquisición. Cada línea corresponde a un valor distinto de desequilibrio del puente. 35
10. Relación simulada entre R_{var} y la tensión de salida V_{out} del circuito de adquisición con AD620. 36
11. PCB fabricada con la topologia nominal I^{2}C seleccionada: todos los sensores quedan agrupados en I^{2}C y las rutas SPI no se montan. 39
12. PCB equipada con los enlaces de configuracion, manteniendo la posibilidad de seleccionar fisicamente distintas topologias de bus durante las pruebas. 39
13. Vista de conjunto del montaje mecanico previsto para el cohete. Fuente: ICAI Rocket Team (IRT26). 43
14. Detalle del compartimento interno y de los puntos de sujecion mecanica de la electronica. Fuente: ICAI Rocket Team (IRT26). 43
15. Mecanismo TVC tomado como referencia mecánica para la tobera orientable que se empleará en el prototipo. Fuente: BPS.Space (BPSS26). 46
16. Esquema de TVC donde se identifican la deflexión del vector de empuje y la distancia del gimbal al centro de gravedad, usada como referencia del brazo efectivo l_{TVC}. Fuente: dos Santos y Oliveira (SANT24). 47
17. Resultado de simulación de vuelo obtenido desde el programa de control. El eje X es el tiempo $[s]$; azul: altitud AGL $[m]$; naranja: velocidad vertical $[m/s]$; verde: aceleración vertical $[m/s^2]$; líneas punteadas: eventos de vuelo. 72
18. Resultado de control obtenido desde la GUI Simulink-first. Eje X: tiempo $[s]$. Línea discontinua negra: mando calculado `u_cmd`. Líneas continuas: mando aplicado `u_sat`; azul en empuje/aerofreno, verde en pitch TVC y naranja en yaw TVC. . . . 73

19. Montaje de laboratorio del sistema de adquisición de la célula de carga, con prototipo en placa de pruebas y osciloscopio para observar la señal acondicionada. 74
20. Banco de integración de la PCB de aviónica con Raspberry Pi 3B+ y GUI de telemetría en el ordenador. 75
21. Telemetría real recibida desde la Raspberry Pi 3B+ por Ethernet. Eje X: tiempo $[s]$. Panel 1: roll azul y pitch naranja $[deg]$. Panel 2: a_x azul, a_y verde, a_z naranja y $|a|$ gris $[m/s^2]$. Panel 3: altitud azul claro $[m]$ y $|B|$ rosa $[\mu T]$. Panel 4: PWM pitch azul, yaw verde y aerofreno naranja $[\mu s]$. 76

Índice de tablas

1. Datos de hoja de datos del servo TVC seleccionado 26
2. Canales PWM actuales para el banco Raspberry 27
3. Valores actuales de integración del aerofreno 28
4. Requisitos actualizados del subsistema de célula de carga 33
5. Parámetros mecánicos 41
6. Valores mecánicos reportados del cohete de competición del IRT 2026 42
7. Contrato mecánico y aerodinámico inicial de aerofrenos 44
8. Parámetros del modelo TVC 48
9. Estructura completa del software MATLAB/Simulink 50
10. Mapa funcional del codigo documentado 87
11. Enlaces a las fuentes de componentes empleadas en el diseño 135

Índice de extractos de código

1. Pseudocódigo del perfil TVC hold+ramp 60
2. Pseudocódigo general del simulador MATLAB 69
3. Esqueleto funcional de una simulacion MATLAB 88
4. Pseudocodigo de `state_derivatives` 88
5. Operacion central de la planta 6DOF 89
6. Fuerza y momento de empuje con TVC 89
7. Pseudocodigo del modelo Simulink-first generado 90
8. Funcionamiento de la GUI Simulink-first 91
9. Pseudocodigo del programa de adquisicion Raspberry–PC 92

Acrónimos

6DOF	Seis grados de libertad (*six degrees of freedom*)
AGL	Altura sobre el nivel del terreno (*Above Ground Level*)
AV	Aviónica
AyD	Aerodinámica y Diseño
BS	Business Strategy
CG	Centro de gravedad
CP	Centro de presiones
EKF	Filtro de Kalman extendido (*Extended Kalman Filter*)
HIL	Hardware-in-the-Loop
I2C	Inter-Integrated Circuit
ICAI	Instituto Católico de Artes e Industrias
IMU	Unidad de medida inercial (*Inertial Measurement Unit*)
IRT	ICAI Rocket Team
ISA	Atmósfera estándar internacional (*International Standard Atmosphere*)
LQR	Regulador cuadrático lineal (*Linear Quadratic Regulator*)
MSL	Nivel medio del mar (*Mean Sea Level*)
NED	North-East-Down
NW	Noroeste (*Northwest*)
PCB	Placa de circuito impreso (*Printed Circuit Board*)
TFG	Trabajo Fin de Grado
PRP	Propulsión
PWM	Modulación por ancho de pulso (*Pulse Width Modulation*)
RASP	Rocket Altitude Simulation Program; formato de fichero de curvas de empuje usado por simuladores de cohetes
REC	Recovery
SFC	Control por variables de estado (*State Feedback Control*)
SI	Sistema Internacional de Unidades
TVC	Control vectorial de empuje (*Thrust Vector Control*)
WNW	Oeste-noroeste (*West-Northwest*)

Símbolos

α	Primera letra del alfabeto griego
β	Segunda letra del alfabeto griego

Capítulo 1

Física del vuelo del cohete

El comportamiento del cohete viene dado por la combinación de fuerzas propulsivas, aerodinámicas y gravitatorias. Este capítulo fija todas las hipótesis físicas usadas por el modelo.

1.1. Modelo de referencia en OpenRocket

OpenRocket (NISK13) es un software gratuito que se utiliza en el mundo de la cohetería amateur para diseñar y simular vuelos de cohetes. En este proyecto se usa como referencia principal para comprobar, en primer lugar, la geometría, la masa, las aletas y el motor antes de pasar al modelo propio en MATLAB y Simulink y, tras ello, se emplea su documentación técnica para crear un simulador en MATLAB sobre el que diseñar el control.

El modelo de referencia se ha desarrollado en OpenRocket a partir del cohete de competición del IRT 2026, documentado en el reporte técnico final del ICAI Rocket Team para la competición SURTAM 2026 (IRT26). En este TFG se toma como base la estructura y la propulsión, pero se introducen los cambios necesarios para estudiar el aterrizaje retropropulsivo con TVC. Por ese motivo, no se importan las soluciones de recuperación, aviónica ni CanSat descritas en el informe de competición; esos subsistemas se sustituyen por la arquitectura propia del proyecto.

Según el reporte final (IRT26), la configuración base tiene una longitud total de $1{,}150\,m$, un diámetro máximo de $0{,}084\,m$, una masa de $2{,}339\,kg$ sin motor y una masa completa de $2{,}814\,kg$. El centro de gravedad de referencia se sitúa a $0{,}682\,m$ desde la punta de la ojiva y el centro de presiones a $0{,}818\,m$, lo que da un margen de estabilidad de $1{,}61$ calibres. Los tramos resistentes usan tubo Quantum, que es PVC ligero de densidad $1{,}07\,\mathrm{g/cm^3}$ frente a los $1{,}39\,\mathrm{g/cm^3}$ del PVC convencional; la funda exterior es de cartón de $0{,}696\,m$, la ojiva de ABS de $0{,}20\,m$, el tren inferior de ABS de $0{,}254\,m$, los conectores de PETG y las aletas de contrachapado de abedul. El radio exterior registrado es de $0{,}042\,m$ y las piezas impresas restantes en ABS o PLA se limitan a elementos auxiliares, no al cuerpo resistente.

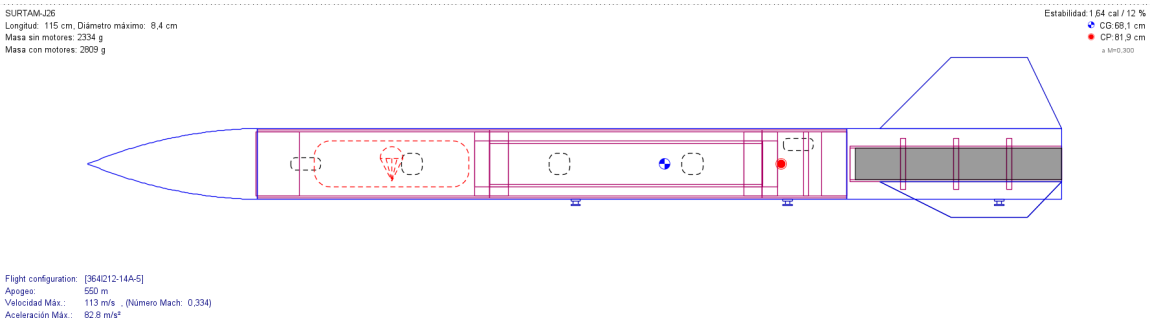


Figura 1. Vista lateral del cohete de competición del IRT 2026 en OpenRocket, usada como referencia geometrica y de estabilidad. Fuente: ICAI Rocket Team (IRT26).

El modelo incorpora tres aletas de forma libre en la zona posterior, fabricadas en contrachapado de abedul, con espesor de $0{,}006\,m$, pestaña de anclaje y puntos de contorno definidos en coordenadas locales. En el tramo inferior, las aletas se insertan a través de ranuras del fuselaje y se unen al tubo interno del motor, que queda centrado mediante tres anillos. Esta geometría desplaza hacia atrás el centro de presiones y aumenta el margen de estabilidad pasiva durante el ascenso. Dicho de otra forma, las aletas hacen que el cohete tienda a alinearse con la corriente de aire en vez de girarse sin control (IRT26).

El entorno operativo de competición recogido en el reporte final (IRT26) se sitúa en Aerocinca, Alcolea de Cinca, con una cota de referencia de $213\,m$ sobre el nivel del mar y un techo autorizado de $1000\,m$. El informe identifica vientos dominantes de componente WNW/NW y fija como ventana preferente las horas de menor velocidad de viento; si las medidas reales lo exigen, la guía puede inclinarse hasta $20°$ respecto a la vertical para compensar deriva. La tabla final de parámetros de vuelo del informe da una velocidad de salida de guía de $19{,}8\,m/s$, apogeo de $521\,m$, velocidad máxima de $108\,m/s$, aceleración máxima de $79{,}3\,m/s^2$, tiempo a apogeo de $10{,}6\,s$, velocidad de despliegue de $6{,}43\,m/s$, tiempo total de vuelo de $89{,}8\,s$ y velocidad de impacto de $7{,}92\,m/s$. Estos valores describen el vuelo convencional con recuperación por paracaídas y sirven como referencia externa de envolvente y seguridad. La simulacion MATLAB/OpenRocket del TFG utilizada hasta ahora produce resultados propios para la configuración con motor Cesaroni `364I212-14A`: apogeo de $550{,}208\,m$, velocidad máxima de $113{,}056\,m/s$, número de Mach máximo de $0{,}334$, aceleración máxima de $82{,}837\,m/s^2$, tiempo a apogeo de $10{,}81\,s$ y velocidad de salida de guía de $20{,}08\,m/s$. Por tanto, si se quiere que el simulador reproduzca exactamente el reporte final, queda pendiente recalibrar la configuración y sus pruebas. Se ha utilizado una previsión de Open Meteo para el 11 de junio de 2026 en las coordenadas del campo y cota $213\,m$; el resumen diario da temperatura media de $20{,}483\,°C$, viento medio a $10\,m$ de $2{,}315\,m/s$, racha máxima de $9{,}600\,m/s$ y precipitación nula (OPEN26). Este archivo se usará para simular las perturbaciones de viento y rachas.

1.2. Modelo dinámico implementado en MATLAB

A partir de esa referencia se desarrolla un modelo propio en MATLAB. Se quiere construir un simulador que se pueda modificar, conectar con Simulink y usar después para diseñar controladores. Para ello, el cohete se representa como un sólido rígido de seis grados de libertad, con traslación y rotación en el espacio.

El vector de estado tiene trece variables agrupadas en cuatro bloques: posición en ejes NED $[p_N, p_E, p_D]$, velocidad lineal en ejes cuerpo $[u, v, w]$, cuaternión de actitud $[q_0, q_1, q_2, q_3]$ y velocidad angular en ejes cuerpo $[p, q, r]$. Esta elección evita las singularidades de los ángulos de

Euler y coincide con la formulación de navegación y control tomada como referencia (SANT24). En esta notación, q_0 es la componente escalar y q_1, q_2 y q_3 son las componentes vectoriales.

Como el vector completo es largo, se expresa transpuesto:

$$\mathbf{x}^T = [p_N,\ p_E,\ p_D,\ u,\ v,\ w,\ q_0,\ q_1,\ q_2,\ q_3,\ p,\ q,\ r] \tag{1}$$

La posición se obtiene transformando la velocidad del cohete desde ejes cuerpo hasta ejes NED. La actitud se integra mediante la ecuación diferencial del cuaternión:

$$\dot{q} = \frac{1}{2}\Omega(\omega)q \tag{2}$$

donde $q = [q_0, q_1, q_2, q_3]^T$ es el cuaternión de actitud y $\omega = [p, q, r]^T$ es la velocidad angular del cohete expresada en ejes cuerpo. Durante la integración se normaliza el cuaternión para evitar que los errores numéricos acumulados degraden la matriz de rotación.

1.3. Fuerzas que actúan sobre el cohete

Las fuerzas principales que actúan sobre el cohete son el empuje del motor, la aerodinámica y la gravedad. La forma de partida es la segunda ley de Newton escrita como masa por aceleración, es decir, masa por derivada inercial de la velocidad. En un sistema inercial I se escribe:

$$\sum F = ma_I = m\left.\frac{dv}{dt}\right|_I \tag{3}$$

Como las ecuaciones del simulador se escriben en ejes cuerpo, la velocidad v_b se expresa en el sistema solidario al cohete. La derivada inercial y la derivada vista desde ejes cuerpo se relacionan mediante:

$$\left.\frac{dv}{dt}\right|_I = \left.\frac{dv_b}{dt}\right|_b + \omega_b \times v_b \tag{4}$$

Al agrupar las fuerzas del cohete, usando la masa instantánea m y dejando el efecto de la masa expulsada recogido en la curva de empuje, se obtiene la ecuación usada en el modelo:

$$m\dot{v}_b = F_T + F_A + F_g - m\left(\omega_b \times v_b\right) \tag{5}$$

donde v_b es la velocidad en ejes cuerpo, F_T es el empuje, F_A es la fuerza aerodinámica y F_g es el peso transformado al sistema de ejes del cohete. El término $m(\omega_b \times v_b)$ aparece porque no se está derivando la velocidad en un sistema inercial, sino en un sistema solidario al vehículo.

1.3.1. Empuje y control vectorial

El empuje se aplica como una fuerza de módulo T. Si la tobera está alineada con el eje longitudinal, el empuje actúa prácticamente sobre el eje del cohete. Sin embargo, en este proyecto se considera un sistema TVC, por lo que la dirección del empuje puede desviarse en cabeceo y guiñada siguiendo la formulación vectorial adoptada para la planta (SANT24). En la interfaz actual del simulador se usan las deflexiones μ_p y μ_y, medidas en radianes:

$$F_T = \begin{bmatrix} T\cos\mu_p\cos\mu_y \\ -T\cos\mu_p\sin\mu_y \\ -T\sin\mu_p \end{bmatrix} \tag{6}$$

Esta fuerza genera también un momento de control por el brazo que existe entre la tobera y el centro de gravedad. Para ángulos pequeños, el momento principal puede aproximarse como:

$$M_{TVC} \simeq T\, l_{TVC}\, \delta \tag{7}$$

donde M_{TVC} es la magnitud del momento de control generado por el sistema TVC, expresado en $N\,m$; T es el empuje instantáneo del motor, en N; l_{TVC} es el brazo de palanca entre el punto de aplicación del empuje y el centro de gravedad, en m; y δ es la deflexión efectiva de la tobera, en radianes. La aproximación usa $\sin\delta \simeq \delta$, por lo que solo es válida para ángulos pequeños.

El motor de ascenso de referencia es un Cesaroni Pro38 3G Smoky Sam I212, designación $364I212\text{-}14A$. ThrustCurve.org recoge para este motor un diámetro de $38\,mm$, longitud de $244\,mm$, masa total de $0{,}475\,kg$, masa de propelente de $0{,}248\,kg$, empuje medio certificado de $212{,}7\,N$, empuje máximo de $255{,}2\,N$, impulso total de $364{,}0\,N\,s$ y tiempo de combustión de $1{,}7\,s$ (THRU06). La curva RASP de fabricante tiene un empuje medio de $214{,}6\,N$ e impulso de $364{,}3\,N\,s$, y se muestra en la Figura 2. Para el simulador de control, este motor se trata como una curva de empuje impuesta y no como un actuador regulable; mientras no se introduzca en código la curva tabulada completa, se usa un pulso rectangular equivalente que conserva el impulso total. La deflexión máxima de TVC se mantiene como parámetro de actuador separado y se expresa en radianes.

La carga en unidades de gravedad se obtiene a partir del sumatorio de todas las fuerzas que actuan sobre el cohete, normalizado por su peso. Por tanto, no depende solo del empuje, sino tambien del arrastre aerodinamico, de la fuerza normal aerodinamica cuando exista angulo de ataque y del peso:

$$n_g(t) = \frac{\left\|\sum \vec{F}(t)\right\|}{m(t)\, g_0} = \frac{\left\|\vec{F}_T(t) + \vec{F}_D(t) + \vec{F}_N(t) + \vec{F}_g(t)\right\|}{m(t)\, g_0} \tag{8}$$

En un ascenso vertical con la tobera alineada y tomando positivo hacia arriba, la aceleracion axial neta se puede escribir como:

$$a_z(t) = \frac{T(t) - D(t) - m(t)g_0}{m(t)} \tag{9}$$

Por tanto, la carga neta instantanea en g queda:

$$n_{g,z}(t) = \frac{T(t) - D(t) - m(t)g_0}{m(t)\, g_0} \tag{10}$$

El arrastre $D(t)$ no es constante: depende de la velocidad relativa, la densidad, el coeficiente aerodinamico y el area de referencia. De forma simplificada:

$$D(t) = \frac{1}{2}\rho(t)\, V_{rel}^2(t)\, C_D(t)\, A_{ref} \tag{11}$$

Con la masa completa de referencia $m = 2{,}814\,kg$, el empuje maximo $T_{max} = 255{,}2\,N$ y $g_0 = 9{,}80665\,m/s^2$, el peso inicial es $mg_0 = 27{,}60\,N$. En el instante inicial, antes de que la velocidad genere arrastre apreciable, $D(0) \simeq 0$ y la estimacion de carga neta maxima queda:

$$n_{g,z}(0) \simeq \frac{255{,}2 - 0 - 27{,}60}{27{,}60} = 8{,}25\,g \tag{12}$$

La relacion empuje/peso asociada es $T_{max}/(mg_0) = 9{,}25$. La diferencia entre ambos valores es el propio peso del cohete. Durante el vuelo real se emplea la expresion instantanea con $D(t)$ y $m(t)$, ya que el arrastre reduce la aceleracion neta y la masa cambia durante la combustion.

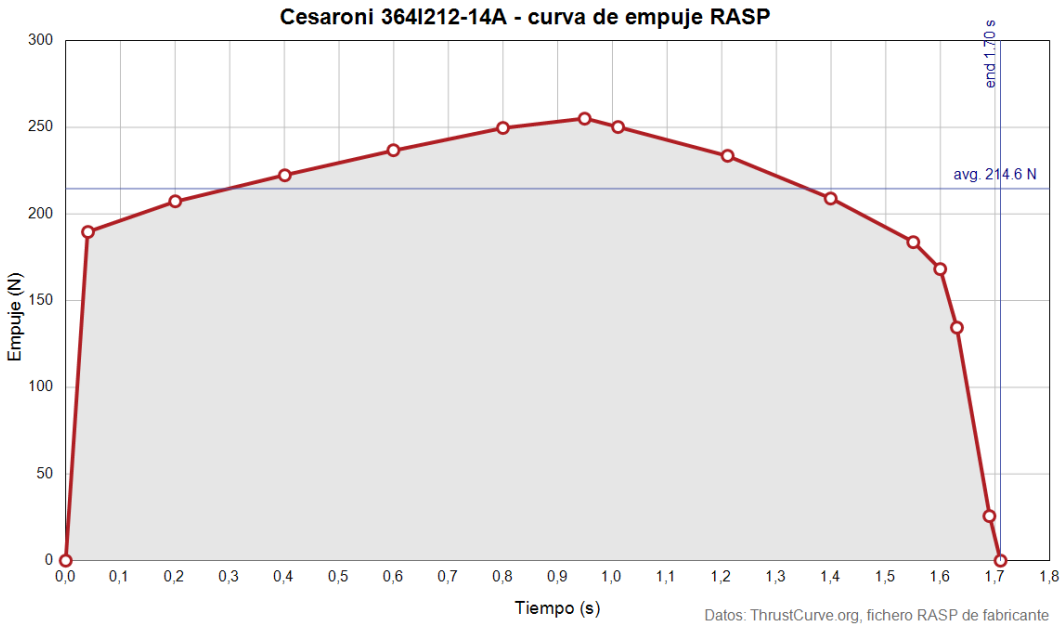


Figura 2. Curva de empuje del motor Cesaroni $364I212\text{-}14A$ generada a partir de los puntos RASP de fabricante publicados por ThrustCurve.org (THRU06).

1.3.2. Gravedad

La gravedad se modela como una aceleración constante $g_0 = 9{,}81\,m/s^2$ dirigida hacia abajo en NED. Para el rango de alturas del proyecto esta simplificación es razonable, ya que las variaciones de gravedad con la altura son pequeñas frente a otras incertidumbres más importantes, como el empuje real, el viento, la masa final o los coeficientes aerodinámicos.

En el simulador, el peso se define primero en ejes NED y después se transforma a ejes cuerpo con la matriz de rotación de actitud. Así se suma en el mismo sistema de ejes que el empuje y las fuerzas aerodinámicas.

1.3.3. Aerodinámica

La aerodinámica depende de la velocidad relativa respecto al aire, no de la velocidad respecto al suelo. Por eso el programa calcula V_{rel} restando la velocidad del viento a la velocidad del cohete antes de calcular la presión dinámica. La presión dinámica mide la intensidad del flujo que impacta contra el cohete: equivale a la energía cinética del aire por unidad de volumen y es el término que escala fuerzas como el arrastre y la fuerza normal, de acuerdo con la formulación aerodinámica usada como referencia (GREG70, NISK13). Se calcula como:

$$q_\infty = \frac{1}{2}\rho V_{rel}^2 \tag{13}$$

donde q_∞ es la presión dinámica, ρ es la densidad del aire y V_{rel} es el módulo de la velocidad del cohete respecto al aire.

El arrastre se aplica en sentido opuesto a la velocidad relativa y se calcula con un coeficiente C_D construido por componentes. El modelo contempla contribuciones de ojiva, cuerpo, base, aletas e interferencia, e incorpora una corrección con el número de Mach. Como el caso de OpenRocket alcanza $M = 0{,}34$, el vuelo de referencia se mantiene en régimen subsónico.

Además del arrastre axial, el simulador calcula fuerza normal por ángulo de ataque y momento aerodinámico debido a la separación entre centro de gravedad y centro de presiones. Para evitar

resultados poco realistas cuando el ángulo de ataque es grande, el modelo usa una dependencia no lineal con $\sin(\alpha)$ y una transición hacia un régimen de cuerpo atravesado o en tumbado cuando el cohete deja de volar casi alineado con el flujo.

1.4. Momentos y estabilidad

La variable que se integra en el modelo rotacional es la velocidad angular, por lo que el resultado que se necesita es la aceleración angular $\dot{\omega}_b$. Sin embargo, para un sólido rígido expresado en ejes cuerpo no conviene escribir directamente $\sum M = J\dot{\omega}_b$, porque faltaría el término giroscópico. La forma correcta de partida es la segunda ley de Newton–Euler aplicada al momento cinético. En un sistema inercial, el sumatorio de momentos respecto al centro de gravedad es:

$$\sum M = \left.\frac{dH}{dt}\right|_I, \qquad H = J\omega_b \tag{14}$$

La derivada inercial del momento cinético se relaciona con la derivada observada desde el sistema solidario al cohete mediante:

$$\left.\frac{dH}{dt}\right|_I = \left.\frac{dH}{dt}\right|_b + \omega_b \times H \tag{15}$$

Tomando los ejes cuerpo como ejes principales de inercia, la matriz de inercia queda diagonal. El vector de velocidad angular y el vector de momentos totales se escriben como:

$$J = \begin{bmatrix} J_x & 0 & 0 \\ 0 & J_y & 0 \\ 0 & 0 & J_z \end{bmatrix}, \qquad \omega_b = \begin{bmatrix} p \\ q \\ r \end{bmatrix}, \qquad M = M_T + M_A = \begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} \tag{16}$$

El producto vectorial asociado a la velocidad angular puede expresarse con la matriz antisimétrica:

$$[\omega_b]_\times = \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix}, \qquad \omega_b \times (J\omega_b) = [\omega_b]_\times J\omega_b \tag{17}$$

Sustituyendo $H = J\omega_b$ en el balance de momentos, y suponiendo que el tensor de inercia se expresa en ejes cuerpo principales, se obtiene:

$$M = J\dot{\omega}_b + \omega_b \times (J\omega_b) \tag{18}$$

Al despejar directamente la aceleración angular aparece la ecuación de Euler usada por el modelo:

$$\dot{\omega}_b = J^{-1}\left(M_T + M_A - \omega_b \times (J\omega_b)\right) \tag{19}$$

El desarrollo por componentes que implementa el simulador queda:

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \dfrac{M_x - (J_z - J_y)\,qr}{J_x} \\ \dfrac{M_y - (J_x - J_z)\,pr}{J_y} \\ \dfrac{M_z - (J_y - J_x)\,pq}{J_z} \end{bmatrix} \tag{20}$$

donde M_x, M_y y M_z son las componentes de la suma de momentos de TVC y aerodinámica en ejes cuerpo. El momento aerodinámico aparece porque la fuerza normal no actúa exactamente en el centro de gravedad, sino aproximadamente en el centro de presiones.

La estabilidad pasiva del cohete depende de la posición relativa entre el centro de gravedad y el centro de presiones. Para vuelo hacia delante, el cohete es estable si el centro de presiones queda por detrás del centro de gravedad. En ese caso, cuando aparece un ángulo de ataque, el momento aerodinámico tiende a devolver el cohete a la dirección de vuelo.

El margen de estabilidad se expresa normalmente en calibres:

$$SM = \frac{x_{CP} - x_{CG}}{D} \tag{21}$$

donde x_{CP} y x_{CG} se miden desde la punta de la ojiva y D es el diámetro del fuselaje. En el reporte final del cohete de competición del IRT 2026 (IRT26) se recogen $x_{CP} = 0{,}818\,m$, $x_{CG} = 0{,}682\,m$ y $D = 0{,}084\,m$, lo que da un margen aproximado de $1{,}61$ calibres para la configuración completa. Con estos valores, el modelo base es estable durante el ascenso; el modelo recalcula esa estabilidad al variar la masa con el tiempo. Además, al abrir el aerofreno se desplaza el centro de presiones para que, durante la bajada, el cohete siga siendo estable pasivamente.

1.5. Masa variable e inercias

Durante la combustión cambian simultáneamente la masa total, el centro de gravedad y las inercias. En el simulador se usa una primera aproximación en la que el propelente se consume de forma lineal durante el tiempo de quemado:

$$m(t) = m_{seco} + m_{prop,0}\left(1 - \frac{t}{t_b}\right) \tag{22}$$

para $0 \leq t \leq t_b$. Si $x_{CG,seco}$ es el centro de gravedad del cohete sin propelente consumible y $x_{CG,prop}$ es la posición equivalente del propelente restante, el centro de gravedad instantáneo se calcula como:

$$x_{CG}(t) = \frac{m_{seco}x_{CG,seco} + m_{prop}(t)x_{CG,prop}}{m_{seco} + m_{prop}(t)} \tag{23}$$

Esta aproximación no describe todavía la geometría real del grano o del motor, pero captura el efecto principal: al consumirse el propelente, cambia el brazo aerodinámico, cambia el brazo efectivo del TVC y cambian las inercias de cabeceo y guiñada.

1.6. Viento, atmósfera y condiciones de lanzamiento

OpenRocket registra viento, guía de lanzamiento, altitud y condiciones atmosféricas. En el simulador MATLAB la atmósfera se calcula con un modelo ISA simplificado (NOAA76) y el viento se introduce como una perturbación sobre la velocidad relativa respecto al aire. La atmósfera proporciona densidad, presión, temperatura y velocidad del sonido a partir de la altura. En la troposfera se usa:

$$
\begin{aligned}
T(h) &= T_0 + Lh, \\
P(h) &= P_0 \left(\frac{T(h)}{T_0} \right)^{-g_0/(RL)}, \\
\rho(h) &= \frac{P(h)}{RT(h)}
\end{aligned}
\tag{24}
$$

donde $T_0 = 288{,}15\,K$, $P_0 = 101325\,Pa$, $L = -0{,}0065\,K/m$, $R = 287{,}05\,J/(kg\,K)$ y $g_0 = 9{,}80665\,m/s^2$. Para el rango de vuelo considerado, esta aproximación es suficiente para obtener presión dinámica y número de Mach con el orden de magnitud correcto. La formulación procede de la atmósfera estándar estadounidense/ISA, particularizada a la troposfera con gradiente térmico lineal.

El viento se calcula en el marco local NED mediante la función `wind_model`. El vector base $v^n_{w,ref} = [V_N, V_E, V_D]^T$ procede de la configuración ambiental y se interpreta en m/s. Si el perfil de viento está desactivado, el modelo devuelve $v^n_w = 0$. Si está activado, se puede mantener viento constante, aplicar cizalladura vertical o sumar una ráfaga determinista. El caso constante es:

$$
v^n_w(h,t) = v^n_{w,ref} \tag{25}
$$

Para representar que el viento cambia con la altura, el simulador incluye un perfil logarítmico y una ley potencial (STUL88, PETE78). En forma resumida:

$$
v^n_w(h) = \frac{\ln(z/z_0)}{\ln(z_{ref}/z_0)}\, v^n_{w,ref} \tag{26}
$$

$$
v^n_w(h) = \left(\frac{\text{máx}(h, z_{ref})}{z_{ref}} \right)^{\alpha} v^n_{w,ref} \tag{27}
$$

En estas expresiones h es la altura AGL usada por la dinámica, z_0 es la rugosidad superficial, z_{ref} la altura de referencia del viento y α el exponente de la ley potencial. En la ley potencial se fuerza $h \geq z_{ref}$ para evitar que el viento se reduzca artificialmente cerca del suelo durante la fase de aterrizaje. Las ráfagas se simulan como una señal sinusoidal reproducible:

$$
v^n_w(t) = v^n_{w,ref} + A_g \sin(2\pi f_g t + \phi_g) \tag{28}
$$

donde A_g es un vector de amplitud en NED, f_g la frecuencia y ϕ_g la fase. La previsión de Open Meteo del 11 de junio de 2026 se usará para fijar valores representativos de viento medio y racha máxima en estos parámetros.

El viento no se suma como una fuerza directa sobre el cohete. Primero se transforma a ejes cuerpo con la actitud instantánea y después se resta de la velocidad del vehículo. La velocidad que entra en aerodinámica es:

$$
v^b_{air} = v^b - R_{n\to b} v^n_w \tag{29}
$$

Así, una racha o una cizalladura cambia la presión dinámica, el ángulo de ataque y el resbalamiento, y de ahí salen las variaciones de fuerzas y momentos aerodinámicos.

En la preparación del lanzamiento habrá que revisar los datos atmosféricos reales del día, especialmente viento, presión y temperatura, para ajustar la simulación a las condiciones existentes. Con esos datos, junto con la curva real de empuje, la masa final del cohete y los coeficientes aerodinámicos asociados a la geometría definitiva, el modelo podrá representar con mayor fidelidad el vuelo real.

1.7. Relación con el control y el aterrizaje retropropulsivo

La física anterior condiciona directamente el control. En ascenso, la estabilidad pasiva de las aletas ayuda a mantener el cohete alineado y reduce la carga del TVC. Sin embargo, durante el descenso o durante una maniobra de aterrizaje retropropulsivo el cohete puede trabajar con baja velocidad aerodinámica, ángulos de ataque altos o incluso vuelo de cola. En esta fase, los aerofrenos también aportan una contribución de estabilidad pasiva durante el descenso, al aumentar la superficie resistente y el amortiguamiento aerodinámico. En esas condiciones, las aletas dejan de ser suficientes y el control por empuje vectorial pasa a ser el actuador principal de actitud.

El modelo propio no se limita a reproducir una trayectoria balística como en una simulación convencional de OpenRocket. Incluye fuerzas y momentos en ejes cuerpo, masa variable, amortiguamiento aerodinámico, TVC y eventos de vuelo. OpenRocket sirve como referencia de diseño y estabilidad del cohete en ascenso; MATLAB y Simulink se reservan para estudiar la dinámica necesaria para control activo, retropropulsión e integración con hardware.

Capítulo 2

Hardware

En este capítulo se agrupan los componentes seleccionados para la célula de carga, para la PCB y demás sistemas, y cómo se han desarrollado.

Cuando se emplean ecuaciones de base procedentes de OpenRocket, se toma como referencia su documentacion tecnica (NISK13); los valores numericos se particularizan despues al cohete de competición del IRT 2026 y a los modelos y ensayos del proyecto.

2.1. Selección de componentes

La selección de componentes describe los criterios y las características de los principales componentes que conforman el sistema de control, instrumentación y alimentación del cohete. La seleccion usa cuatro criterios: bajo consumo, volumen y masa reducidos, conexion digital con la Raspberry Pi 3B+ y margen frente a vibraciones y aceleraciones de vuelo.

2.1.1. Requisitos generales del sistema

Los componentes se han seleccionado para operar con la misma alimentación, evitar añadir masa en forma de convertidores y ofrecer márgenes suficientes frente a transitorios eléctricos y grandes aceleraciones. Además, el sistema de sensores combina bajo peso, encapsulados compactos y buses de comunicación compatibles con la electrónica de vuelo para simplificar la integración.

2.1.2. Unidad de control de vuelo

La arquitectura de control embarcado se fija en esta fase alrededor de una **Raspberry Pi 3B+** (RASP26a). Se ha decidido utilizar esta plataforma porque el desarrollo del proyecto se realiza en MATLAB/Simulink y Raspberry Pi Blockset permite generar, desplegar y ejecutar en la placa el código asociado a un modelo Simulink (MATH26). De este modo se mantiene continuidad entre la simulación, el diseño del control y la futura ejecución embarcada. La Raspberry queda encargada de ejecutar los lazos de guiado y control con las frecuencias definidas en simulación, procesar las señales procedentes de los sensores de la PCB y generar las consignas PWM hacia los servos del sistema TVC. El banco utiliza Raspberry Pi OS de 64 bits, sistema oficial de la plataforma basado en Debian (RASP26c).

En resumen, se ha seleccionado la Raspberry Pi 3B+ porque puede ejecutar en la plataforma embarcada el código generado a partir de modelos Simulink. Aunque no sea un microcontrolador, en este proyecto se utilizará con un papel funcional similar: leer sensores, ejecutar la lógica de control y generar salidas PWM hacia los actuadores (MATH26).

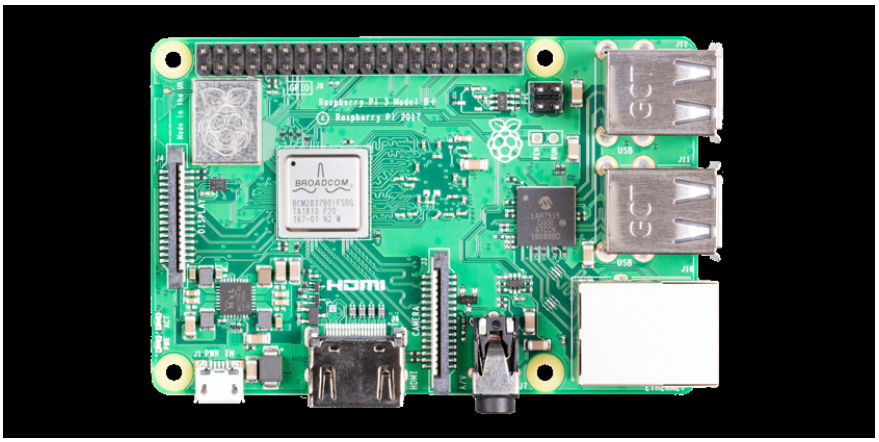


Figura 3. Raspberry Pi 3 Model B+ seleccionada como computador de vuelo del prototipo. Fuente: Raspberry Pi Ltd. (RASP26a).

2.1.3. Sensores

La arquitectura considerada para esta fase del proyecto se apoya en una IMU, un barómetro digital y un magnetómetro triaxial. Esta combinación mide velocidad angular, aceleración, altitud y rumbo con sensores compactos, interfaz digital y consumo reducido. Aunque los tres componentes admiten configuraciones con SPI o I^2C, la configuracion nominal de vuelo se fija en **I^2C** a partir de la PCB diseñada. Se han colocado resistencias de 0 Ω para poder configurar la PCB de tres formas distintas: una configuración con todos los sensores conectados por I^2C, otra con todos los sensores conectados por SPI, y una tercera configuración en la que el magnetómetro se conecta por I^2C a la IMU, mientras que la IMU se comunica por SPI con la Raspberry Pi.

2.1.3.1. Unidad de medición inercial (IMU)

Para la medida de aceleraciones y velocidades angulares se ha seleccionado la IMU **LSM6DSV32X** de STMicroelectronics (STMI26). Este circuito integra un acelerómetro triaxial con rango de hasta $\pm 32\,g$ y un giróscopo triaxial con rango de hasta $\pm 4000\,dps$. Se seleccionó porque permite medir los grados de libertad inerciales necesarios para estimar el movimiento del cohete: aceleraciones lineales y velocidades angulares en los tres ejes. Además, el rango de $\pm 32\,g$ ofrece margen frente al máximo esperado en la configuración actual del cohete, estimado en $8{,}25\,g$ en el cálculo de cargas. Dependiendo de los motores considerados durante el desarrollo del TFG, la aceleración podía llegar aproximadamente a $16\,g$; por ello se escogió un sensor con margen de seguridad adicional frente a IMUs habituales limitadas a $\pm 16\,g$.

Su encapsulado de $2{,}5 \times 3{,}0 \times 0{,}83\,mm$, su consumo reducido y las interfaces SPI/I^2C/I3C encajan con una PCB de aviónica compacta. El dispositivo incorpora también filtrado digital y funciones embebidas de procesado de movimiento, lo que desplaza parte del preprocesado de señales fuera del procesador principal antes de la etapa de estimación (STMI26).

2.1.3.2. Barómetro

Como sensor de presión estática se emplea el **MS5607-02BA03** de TE Connectivity (TECO26). Se trata de un barómetro digital orientado a aplicaciones altimétricas, con conversión de 24 bits, resolución de hasta 20 cm y rango de operación entre 10 y 2000 mbar. Con esos valores, el sensor cubre la estimación de altitud durante ascenso y descenso y complementa la información procedente de la IMU. Para que esta medida sea representativa de la presión atmosférica exterior, se ha previsto una abertura o toma de presión estática que comunique el

volumen donde se ubica el sensor con el exterior. Esta abertura permite igualar presiones y evita que el barómetro mida una presión interna encerrada, lo que mejora la precisión de la estimación de altitud.

El componente presenta tambien un consumo muy reducido, encapsulado compacto y compatibilidad con buses SPI e I^2C (TECO26). Su combinación de precisión, tamaño y simplicidad de interfaz lo convierte en una elección razonable para la electrónica embarcada.

2.1.3.3. Magnetómetro

Para la medida del campo magnético terrestre se ha seleccionado el **MMC5983MA** de MEMSIC (MEMS26). Este magnetómetro triaxial ofrece fondo de escala de $\pm 8\,G$, resolución de 18 bits y ruido típico de 0.4 mG RMS, con tasas de salida de hasta 1000 Hz. Estas características aportan una referencia de rumbo complementaria al resto de la cadena de sensado.

El dispositivo integra función *SET/RESET* para reducir errores de offset y magnetización residual, así como sensor de temperatura y buses SPI/I^2C (MEMS26). En la PCB, el magnetómetro se ha colocado en una zona sin planos de masa ni de tensión, y sin pistas que pasen por debajo del componente. Esta disposición busca reducir campos parásitos de alimentación o señal que puedan interferir en la medición del campo magnético terrestre. Aunque durante las fases más dinámicas del vuelo la medida magnética puede verse afectada por perturbaciones y corrientes próximas, el MMC5983MA resulta útil como referencia adicional en la cadena de calibración.

2.1.4. Actuadores

La tobera orientable del TVC se acciona mediante servomotores dedicados, encargados de generar las deflexiones de pitch y yaw, y el aerofreno incorpora un servomotor propio para gobernar su apertura. En ambos casos, el mando se ha definido con rango angular disponible, saturaciones mecánicas, velocidad máxima del actuador e interfaz PWM definida para la PCB.

2.1.4.1. Servomotores

Los servomotores asociados al sistema TVC se seleccionan para proporcionar la autoridad suficiente al alcanzar las deflexiones máximas contempladas en el modelo de control sin entrar en saturación sostenida. Esto obliga a considerar el par nominal, el ancho de banda, la rigidez mecánica del conjunto y el comportamiento frente a vibraciones.

Para el prototipo de TVC se selecciona el servomotor digital de alto par **Miuzei RC Servo 45KG 270deg**. Al no disponer de una hoja de caracteristicas independiente del fabricante, se toma como fuente la pagina comercial identificada por ASIN (MIUZ26). Esta ficha declara alimentacion entre $4{,}8$ y $8{,}4\,V$, mando PWM entre 500 y $2500\,\mu s$ con neutro en $1500\,\mu s$, recorrido de $270°$, cuenta con 25 dientes, par de bloqueo de $45\,kgf\,cm$ a $6{,}0\,V$ y $50\,kgf\,cm$ a $8{,}4\,V$, y velocidad sin carga de $0{,}13\,s/60°$ a $4{,}8\,V$ y $0{,}10\,s/60°$ a $8{,}4\,V$.

Magnitud	Valor hoja de datos	Uso en el modelo
Tension de alimentacion	$4{,}8$–$8{,}4\,V$	Dimensionado de alimentacion y ensayos
Par comercial	$45\,kgf\,cm$ a $6{,}0\,V$; $50\,kgf\,cm$ a $8{,}4\,V$	Margen inicial de autoridad mecanica
Velocidad sin carga	$0{,}13\,s/60°$ a $4{,}8\,V$; $0{,}10\,s/60°$ a $8{,}4\,V$	Limite inicial de velocidad del actuador
PWM	500–$2500\,\mu s$, neutro $1500\,\mu s$	Mapeo electrico de mando
Recorrido electrico	$270°$	Escala de PWM; la tobera satura aparte

Tabla 1. Datos de hoja de datos del servo TVC seleccionado

2.1.4.2. Señales PWM de actuación

El controlador no gobierna directamente el ancho de pulso. La variable fisica interna sigue siendo angular: $\delta_{tvc,pitch}$ y $\delta_{tvc,yaw}$ para la desviación de la tobera, y δ_{ab} para la apertura del aerofreno. La conversion a PWM se hace en la capa de hardware, justo antes de escribir en los pines de la Raspberry Pi, de modo que el modelo de control permanece expresado en radianes y el servo recibe pulsos en microsegundos.

Para el TVC se usa un mapeo lineal firmado alrededor del neutro del servo. Si δ_{max} es el limite mecanico de la tobera en el eje correspondiente, el pulso aplicado se calcula como:

$$PWM_{tvc} = 1500\,\mu s + \mathrm{sat}\left(\frac{\delta_{tvc}}{\delta_{max}}, -1, 1\right) \cdot 1000\,\mu s \tag{30}$$

Asi, una orden nula deja el servo en $1500\,\mu s$, una orden negativa se aproxima al extremo de $500\,\mu s$ y una orden positiva se aproxima al extremo de $2500\,\mu s$. En el aerofreno el mando es no firmado: $\delta_{ab} = 0$ representa cerrado y $\delta_{ab} = \delta_{ab,max}$ representa la apertura maxima. El mapeo usado es:

$$PWM_{ab} = 500\,\mu s + \mathrm{sat}\left(\frac{\delta_{ab}}{\delta_{ab,max}}, 0, 1\right) \cdot 2000\,\mu s \tag{31}$$

El reparto actual de senales para el banco con Raspberry Pi queda fijado en la tabla siguiente. Los tres canales se verifican con osciloscopio antes de conectar los servos, porque el ensayo de banco busca medir jitter, latencia y sentido real del movimiento antes de considerar el conjunto como valido para vuelo.

Canal	GPIO	Mando fisico	Mapeo actual
TVC pitch	GPIO12	$\delta_{tvc,pitch}$ [rad]	Mando firmado: $1500\,\mu s$ en neutro, saturado entre 500 y $2500\,\mu s$.
TVC yaw	GPIO13	$\delta_{tvc,yaw}$ [rad]	Mando firmado con el mismo criterio que pitch; el signo final se comprobara en banco.
Aerofreno	GPIO18	δ_{ab} [rad]	Mando no firmado: $500\,\mu s$ cerrado y $2500\,\mu s$ para apertura maxima.

Tabla 2. Canales PWM actuales para el banco Raspberry

2.1.4.3. Aerofreno

El aerofreno queda incluido como actuador real de la arquitectura. Su apertura se gobierna mediante un servomotor dedicado, independiente de los servomotores de la tobera, de forma que el sistema pueda modificar el arrastre aerodinámico durante las fases en que sea necesario. Además de aumentar el arrastre, su despliegue modifica la posición efectiva del centro de presiones durante el descenso. Este desplazamiento se aprovecha para conservar la estabilidad pasiva y favorecer que el cohete permanezca alineado con la corriente, aproximadamente vertical, mientras reduce velocidad. La integración adoptada mantiene el mando como un ángulo de apertura δ_{ab}, expresado en radianes, con saturación mecánica, mapeo PWM y límite de velocidad definidos por los valores disponibles actualmente.

Magnitud	Valor actual	Uso en la integración
Mando	δ_{ab} [rad]	Entrada angular del controlador; 0 corresponde a aerofreno cerrado y valores positivos a apertura.
Saturación mecánica	0–$\pi/2\,rad$ (0–90°)	Rango permitido por el modelo actual de aerofreno.
Numero de paneles	4	Paneles desplegables distribuidos alrededor del fuselaje.
Area por panel	$0{,}005\,m^2$	Area geometrica usada para calcular el area proyectada.
Area proyectada maxima	$4 \cdot 0{,}005 \cdot \sin(90°) = 0{,}020\,m^2$	Contribucion maxima de area cuando el aerofreno esta completamente abierto.
Coeficiente aerodinamico	$C_{d,ab} = 1{,}2$	Incremento de arrastre asociado al aerofreno desplegado.
Posicion aerodinamica	$0{,}30\,m$ desde la punta	Punto usado para representar el desplazamiento efectivo del centro de presiones durante el descenso.
PWM de mando	500–$2500\,\mu s$	Rango electrico disponible; el cierre se asocia al extremo inferior y la apertura maxima al extremo superior.
Velocidad de referencia del servo	$0{,}10\,s/60°$ a $8{,}4\,V$	Equivale a $600°/s$ ($10{,}47\,rad/s$) sin carga; valor usado como limite inicial de velocidad (MIUZ26).
Retardo de mando	$0{,}020\,s$	Retardo de simulacion adoptado para la actuacion por servo antes de ensayos de banco.

Tabla 3. Valores actuales de integración del aerofreno

2.1.5. Tobera y sistema de propulsión

La tobera se plantea como un conjunto orientable para TVC, accionado mediante servomotores dedicados que generan las deflexiones de pitch y yaw. Su función es transformar las consignas del controlador en una desviación física del vector de empuje. El esquema de la Figura 4 ilustra esta idea: el empuje total se descompone en componentes laterales al inclinar la tobera, mientras que las fuerzas aerodinamicas se aplican en el centro de presiones y dependen de la actitud y de la velocidad relativa, no del PWM directamente.

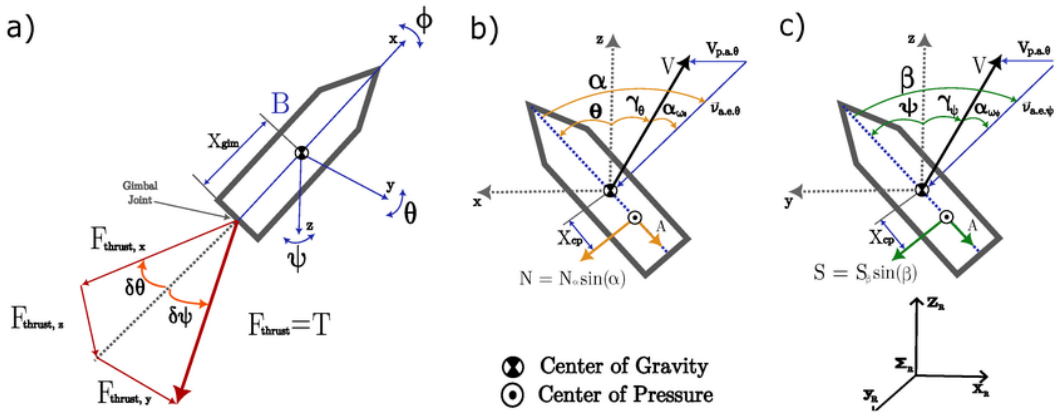


Figura 4. Esquema de tobera orientable con TVC y fuerzas aerodinamicas asociadas. Fuente: dos Santos y Oliveira (SANT24).

Desde el punto de vista mecanico, el servo no inclina la tobera por si solo: mueve un brazo, este transmite desplazamiento a una varilla o biela, y el conjunto de gimbal transforma ese desplazamiento en deflexion angular. La Figura 5 muestra un mecanismo TVC basado en piezas impresas y dos servos; se utilizara como mecanismo de referencia para este TFG, con una arquitectura muy similar de servo–brazo–biela–gimbal adaptada despues a la geometria, cargas y dimensiones propias del cohete.

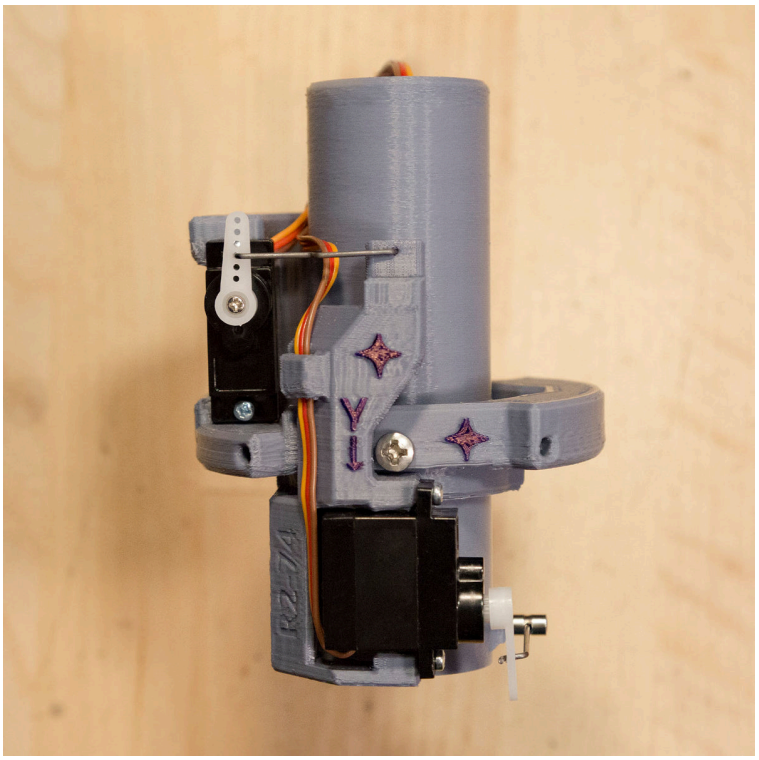


Figura 5. Mecanismo TVC tomado como referencia de diseno con servos y soporte impreso. Fuente: BPS.Space (BPSS26).

En el modelo, la tobera se trata como un actuador mecanico con rango angular limitado, saturacion propia, velocidad maxima y mando PWM desde la PCB. El recorrido electrico del servo sirve como escala de mando, pero la deflexion real queda limitada por la geometria del soporte, por los topes mecanicos del conjunto y por la holgura que aparezca en las uniones. Por

ello, la calibracion de banco mide la relacion real entre microsegundos de PWM y grados de deflexion de la tobera antes de cerrar el limite de vuelo.

2.1.6. Almacenamiento y telemetria de banco en la Raspberry Pi

Durante un ensayo de vuelo o banco integrado, la Raspberry Pi debe conservar localmente las medidas de sensores, mandos enviados a los actuadores, estado estimado y variables principales de ejecucion para su analisis posterior. Esta memoria local sigue siendo el canal robusto previsto para no depender de un enlace radio o Ethernet durante una prueba real.

Adicionalmente, para validacion en laboratorio se ha implementado una telemetria de banco por Ethernet entre la Raspberry y el ordenador. Este enlace no se considera telemetria de vuelo final: sirve para visualizar sensores, comprobar unidades, guardar CSV en el PC y revisar que el mapeo PWM permanece neutro o saturado segun corresponda. El protocolo de banco usa mensajes JSON Lines sobre TCP y mantiene la ignicion y la propulsion fuera del programa.

2.1.7. Sistema de alimentación

La alimentación de la electrónica se realizará mediante una placa de expansión **X728 V2.5** de Geekworm, alimentada por celdas recargables de litio 18650 (GEEK26). Esta placa proporciona la etapa de alimentación principal para la PCB y la electrónica embarcada. La Figura 6 muestra la placa comercial usada como referencia de integración; la imagen del fabricante aparece montada sobre una Raspberry Pi 4, mientras que en este TFG la integración objetivo se mantiene con Raspberry Pi 3B+.

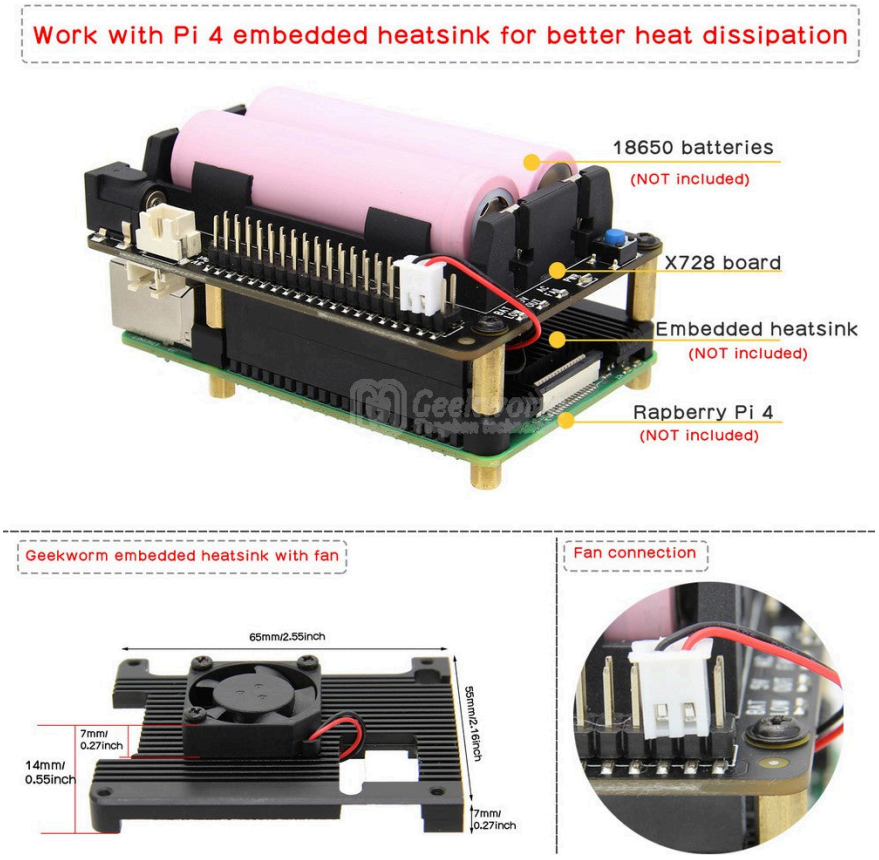


Figura 6. Placa de expansión Geekworm X728 para alimentación con celdas 18650. Fuente: Geekworm (GEEK26).

La celda seleccionada para el prototipo es una JESSPOW 18650 *button top* recargable de ion-litio, $3{,}7\,V$ y $3300\,mAh$, mostrada en la Figura 7. La ficha comercial del fabricante (JESS26) se usa solo para identificar la celda y sus datos nominales.



Figura 7. Celdas JESSPOW 18650 *button top* de $3{,}7\,V$ y $3300\,mAh$ empleadas como referencia de batería. Fuente: JESSPOW (JESS26).

Los servomotores se alimentarán con baterías 18650 independientes de la alimentación de control. Esta separación reduce el acoplamiento de transitorios y caídas de tensión hacia la electrónica de sensores y control. Se adopta provisionalmente una capacidad de corriente de $2{,}5\,A$ como requisito de diseño del banco, no como dato confirmado del servo; deberá sustituirse por el consumo medido, incluido el pico de bloqueo, antes de cerrar el dimensionado eléctrico.

2.1.8. Resumen de componentes

La selección de componentes del sistema queda cerrada para la arquitectura desarrollada en esta memoria. El conjunto final se resume en:

- **Control embarcado:** Raspberry Pi 3B+, empleada como computador de vuelo capaz de ejecutar MATLAB/Simulink y actuar funcionalmente como unidad de control.
- **Sensado:** IMU LSM6DSV32X para aceleraciones y velocidades angulares, barómetro MS5607-02BA03 para presión y altitud con toma de presión estática, y magnetómetro MMC5983MA como referencia de rumbo.
- **Interconexión de sensores:** PCB propia con configuración nominal I^2C y resistencias de $0\,\Omega$ para permitir alternativas de montaje por SPI o bus mixto IMU–magnetómetro.
- **Actuación:** servomotores Miuzei RC Servo 45KG 270deg para las deflexiones de la tobera TVC y un servomotor dedicado para el aerofreno, gobernados mediante PWM.

- **Propulsión y control aerodinámico:** tobera orientable para TVC y aerofreno como actuador de arrastre, ambos tratados con limitaciones mecánicas de recorrido y velocidad.
- **Alimentación:** placa de expansión Geekworm X728 V2.5 alimentada con celdas JESSPOW 18650 *button top* de $3{,}7\,V$ y $3300\,mAh$ para la electrónica, y baterías 18650 independientes para los servos.
- **Registro de datos:** almacenamiento local en la Raspberry Pi de sensores, mandos, estado estimado y variables principales para análisis posterior.
- **Banco de ensayo:** subsistema de célula de carga basado en galgas extensométricas en puente de Wheatstone, acondicionamiento, ADC, adquisición y soporte mecánico para caracterizar el empuje.

Con esta definición, la memoria trata la lista anterior como referencia cerrada de componentes del prototipo. Los apartados siguientes desarrollan la célula de carga y la PCB que integran esta arquitectura.

2.2. Subsistema de célula de carga

LA validación experimental del sistema de propulsión requiere medir empuje de forma repetible para caracterizar las curvas de empuje, necesarias para definir la entrada de empuje del sistema de control y validar la construcción del motor. En este capítulo se documentan el diseño de esta, las simulaciones del circuito de adquisición, los ensayos de los diferentes motores y el análisis de las curvas resultantes.

2.2.1. Función dentro del proyecto

La célula de carga se plantea como el elemento principal de medida para caracterizar los ensayos estáticos del motor. Su misión es obtener una curva fuerza–tiempo suficientemente fiable para determinar la entrada del sistema. Al no ser el empuje un parámetro controlable directamente, esta curva se emplea en las simulaciones previas a los ensayos con prototipos.

2.2.2. Requisitos de diseño

La célula de carga se ha diseñado para las necesidades del IRT, de forma que pueda medir cargas de hasta $1500\,N$ de pico en ensayos estáticos. En este proyecto, sin embargo, el valor de referencia de trabajo será de $250\,N$ de pico. Esta diferencia permite que la medida se realice lejos de la saturación del sensor y deja margen frente a transitorios de ignición, vibraciones, errores de alineación y posibles ensayos posteriores de mayor empuje.

Magnitud	**Valor base**	**Criterio de diseño**
Capacidad de diseño	$1500\,N$ pico	Límite de carga objetivo para el banco del IRT.
Empuje usado en este TFG	$250\,N$ pico	Valor de referencia para los ensayos de este proyecto.
Rango útil de medida	0–$1500\,N$	Cubre el pico de diseño y mantiene margen suficiente frente al pico de trabajo.
Ventana temporal de registro	$\geq 3{,}5\,s$	Cubre ignición, quema y apagado del ensayo estático.
Frecuencia de muestreo	$\geq 100\,Hz$	Compatible con el ADC y suficiente para capturar transitorios sin aliasing.

Tabla 4. Requisitos actualizados del subsistema de célula de carga

2.2.3. Sistema de medida

Se utilizará una célula de carga basada en galgas extensométricas, integrada en puente de Wheatstone de cuatro galgas (full-bridge) y acompañada por una etapa de acondicionamiento y adquisición. La cadena mínima de medida incluye:

- puente de Wheatstone, encargado de transformar la fuerza axial en una deformación repetible;
- circuito de acondicionamiento, con rechazo de ruido, ganancia definida, referencia de medida y filtrado paso bajo;

- convertidor analógico–digital (ADC) y registro temporal sincronizado con el disparo del ensayo;

- soporte mecánico que garantice una aplicación esencialmente axial de la carga (hecho por el departamento de propulsión del IRT).

La etapa de adquisición del puente de galgas se muestra en la Figura 8. El circuito final simulado usa un amplificador de instrumentación AD620 como primera etapa y dos amplificadores operacionales TL082, representados en LTspice como `ADTL082`, para la ganancia adicional y el filtrado de salida (ANAL26a, TEXA26). Esta solución se elige por el tamaño y disponibilidad de los encapsulados, que resultan más cómodos de montar y manipular en la placa de adquisición del banco. Antes de fijar esta versión se estudió una alternativa con AD8221 y ADA4077; esa opción se abandonó por el tamaño de encapsulado y por la dificultad de integración en el prototipo, y las hojas de características de AD8221 y ADA4077 se citan como fuentes de componente (ANAL26b, ANAL26c).

En el esquema, el puente de galgas se representa con resistencias de $200\,\Omega$ y una rama variable R_{var} para barrer el desequilibrio durante la simulación. La salida diferencial del puente entra en el AD620. Con $R_G = 270\,\Omega$, esta primera etapa tiene una ganancia aproximada de 184. Después se utiliza una etapa no inversora con ADTL082 de ganancia $2{,}2$ y una etapa final de ganancia unitaria que implementa un filtro paso bajo activo de segundo orden tipo Sallen–Key, por lo que la ganancia total del circuito es aproximadamente $404{,}8$.

La ganancia de la primera etapa queda fijada por la resistencia externa R_G del AD620 (ANAL26a). La relación de diseño puede escribirse en sus dos formas equivalentes:

$$G_1 = 1 + \frac{49{,}4\,\mathrm{k\Omega}}{R_G}, \qquad R_G = \frac{49{,}4\,\mathrm{k\Omega}}{G_1 - 1} \tag{32}$$

Con el valor montado en el esquema,

$$G_1 = 1 + \frac{49{,}4 \cdot 10^3}{270} = 183{,}963 \simeq 184 \tag{33}$$

La segunda etapa con ADTL082 se configura como amplificador no inversor (TEXA26). Por tanto,

$$G_2 = 1 + \frac{R_{12}}{R_{11}} = 1 + \frac{1{,}2\,\mathrm{k\Omega}}{1\,\mathrm{k\Omega}} = 2{,}2 \tag{34}$$

La ganancia total de la cadena de acondicionamiento se obtiene como el producto de las ganancias de cada bloque:

$$G_{\mathrm{tot}} = G_1 G_2 G_3 \simeq 183{,}963 \cdot 2{,}2 \cdot 1 = 404{,}719 \simeq 404{,}8 \tag{35}$$

El acondicionamiento combina dos tipos de filtrado. En la entrada del AD620 se usa un filtrado pasivo simétrico con resistencias de $51\,\mathrm{k\Omega}$, un condensador diferencial de $1\,\mathrm{nF}$ y condensadores de modo común de $100\,\mathrm{pF}$. Para una red RC pasiva de primer orden, la frecuencia de corte se estima mediante:

$$f_c = \frac{1}{2\pi RC}, \qquad R = \frac{1}{2\pi f_c C} \tag{36}$$

Esta expresión se aplica a los polos RC simples, pero no describe por sí sola la tercera etapa de salida. La etapa final con ADTL082 se implementa como un filtro paso bajo activo de segundo orden tipo Sallen–Key, siguiendo la formulación de diseño de filtros activos de Texas Instruments

(TEXA23). Para esa celda, formada por R_{13}, R_{14}, C_4 y C_5, la frecuencia natural de referencia se estima como:

$$f_0 \simeq \frac{1}{2\pi\sqrt{R_{13}R_{14}C_4C_5}} \tag{37}$$

Con los valores del esquema, $R_{13} = R_{14} = 15\,\mathrm{k\Omega}$, $C_4 = 100\,\mathrm{nF}$ y $C_5 = 50\,\mathrm{nF}$, el filtro queda en el orden de 150–160 Hz, coherente con la frecuencia de corte de referencia indicada en la simulación, $f_c \simeq 159\,\mathrm{Hz}$, y con una frecuencia de muestreo de $1\,\mathrm{kHz}$.

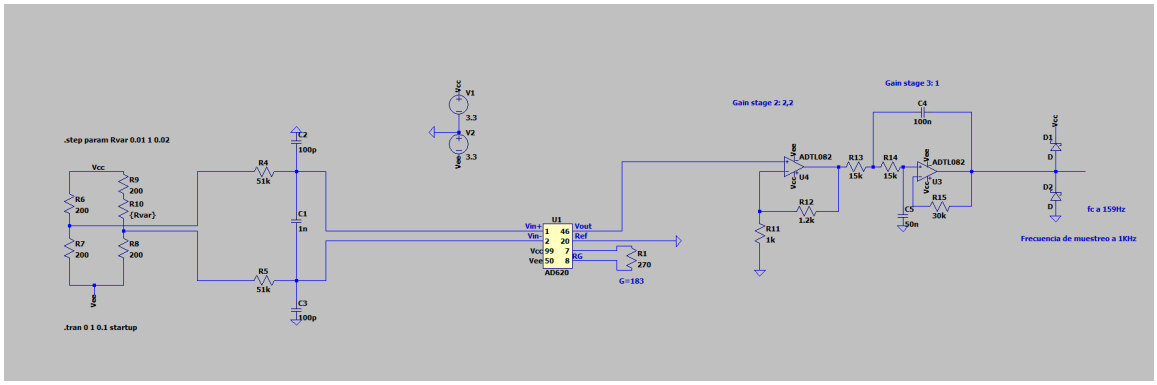


Figura 8. Esquema LTspice del circuito final de acondicionamiento y adquisición del puente de galgas, con AD620 y etapas ADTL082.

La simulación de la Figura 9 barre el desequilibrio del puente de galgas y verifica que la salida permanece dentro del rango admisible de lectura. A la salida se han incluido diodos de protección conectados hacia la alimentación del microcontrolador y masa para limitar sobretensiones o excursiones negativas antes de la entrada analógica. Además, el filtro activo Sallen–Key de salida se fija con una frecuencia de corte aproximada $f_c = 159\,Hz$. Con una frecuencia de muestreo de $1\,kHz$, la frecuencia de Nyquist es $500\,Hz$, de modo que el filtro reduce componentes de alta frecuencia antes de acercarse a ese límite.

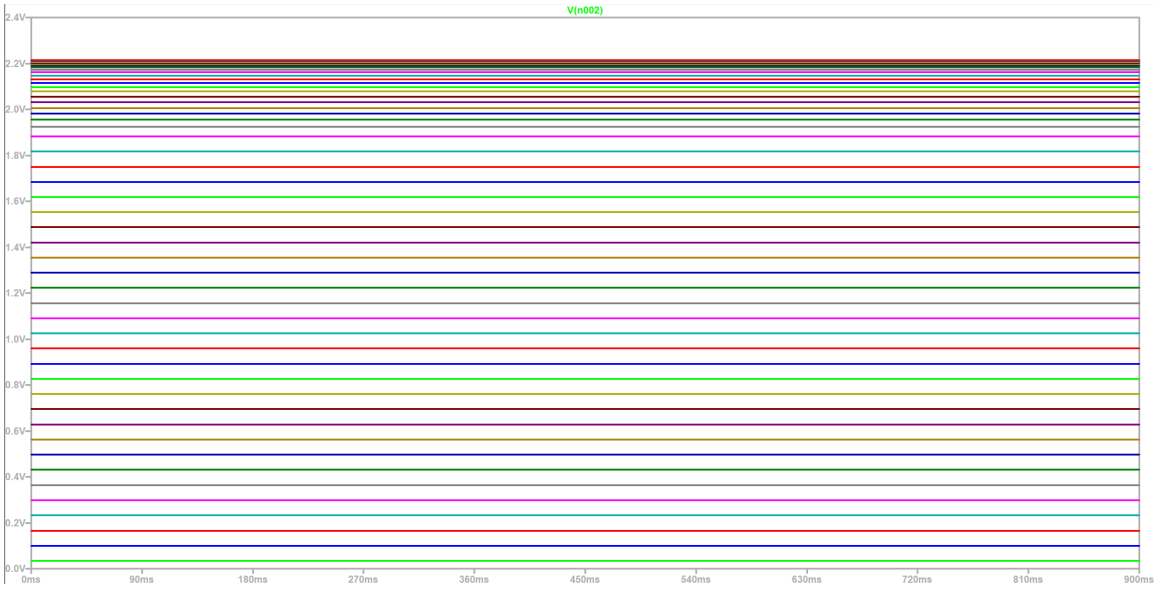


Figura 9. Barrido transitorio de la salida del circuito final de adquisición. Cada línea corresponde a un valor distinto de desequilibrio del puente.

La Figura 10 resume el mismo barrido en régimen estacionario, relacionando la resistencia variable del puente con la tensión de salida. Para cada valor de R_{var} se toma el valor final del transitorio simulado. En el barrido, al aumentar R_{var} desde $0{,}01\,\Omega$ hasta $1\,\Omega$, V_{out} pasa de aproximadamente $0{,}034\,V$ a $2{,}213\,V$. La curva es monótona y muestra un comportamiento prácticamente lineal en el tramo inicial, donde el puente trabaja cerca de su punto de equilibrio y la variación diferencial de tensión es proporcional al desequilibrio. Al acercarse a la saturación, la pendiente disminuye y la respuesta pasa a tener un comportamiento de tipo logarítmico o asintótico. Este aplanamiento puede deberse, por un lado, a que R_{var} modifica una rama de un divisor resistivo y deja de cumplirse la aproximación lineal de pequeña señal; y, por otro, a que el AD620, las etapas ADTL082 y los diodos de protección empiezan a comprimir la salida al aproximarse al margen alto disponible antes de los raíles de alimentación.

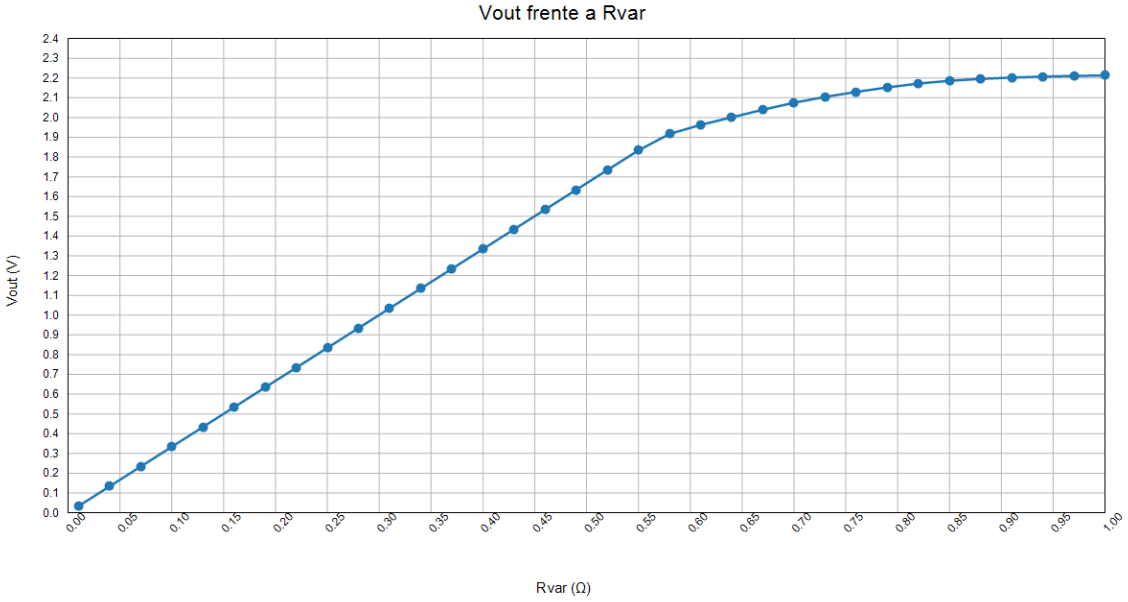


Figura 10. Relación simulada entre R_{var} y la tensión de salida V_{out} del circuito de adquisición con AD620.

La salida del sistema puede modelarse aproximádamente mediante una relación lineal entre la lectura digital y la fuerza aplicada:

$$F = K\,(V_{\mathrm{adc}} - V_0) \tag{38}$$

donde F es la fuerza estimada, K es la ganancia equivalente del conjunto célula–circuito de acondicionamiento y V_0 es el offset medido en vacío, alrededor de la referencia de $2{,}5\,V$.

2.2.4. Integración mecánica

El principal riesgo de una célula de carga en esta aplicacion combina saturación y esfuerzos parásitos. Si la fuerza de empuje no atraviesa el eje de medida, la célula puede recibir componentes de flexión o cortadura que distorsionen la lectura. Por ello, el diseño del banco incluye guías, apoyos y alineación geométrica suficientes para que la medida represente la componente axial del empuje y no el comportamiento del banco de ensayos.

También se prevé aislamiento frente a vibraciones externas, topes mecánicos de seguridad y una fijación que permita desmontaje y recalibración sin modificar la geometría del ensayo.

2.2.5. Calibración y tratamiento de señal

La calibración se ha realizado con masas conocidas, distribuidas en varios puntos del rango útil de trabajo. El procedimiento previsto incluye, como mínimo, tara inicial, carga ascendente, carga descendente y estimación del error. Con ello se cuantifican no linealidad, histéresis y deriva (drift). Sin embargo, por falta de tiempo, se verá hasta donde se puede llegar.

Una vez calibrada la célula, el tratamiento digital de la señal filtra ruido de alta frecuencia sin ocultar los transitorios de ignición y apagado.

2.3. Desarrollo de la PCB

El desarrollo de la PCB constituye el paso necesario para transformar la selección de componentes del Capítulo 2 en una electrónica integrable dentro del cohete. En esta fase se ha diseñado y fabricado una PCB para la aviónica del cohete; por ello, este capítulo documenta el proceso de ingeniería seguido, las decisiones de diseño y el resultado físico disponible. La validación funcional completa de la placa se trata como una fase posterior de integración y ensayo.

2.3.1. Objetivo y alcance

La PCB concentra en un volumen reducido la adquisición de sensores, las señales de control y los elementos auxiliares de diagnóstico. La alimentación principal no forma parte de esta placa: queda situada en el módulo externo descrito en el Subsección 2.1.7. Junto a la función eléctrica, el diseño se ha planteado con restricciones mecánicas claras, en particular vibración, aceleraciones elevadas y montaje dentro de la caja de electrónica.

El resultado documental y físico disponible en esta fase incluye el esquemático de la caja de electrónica incluido en el Documento II de planos, concretamente en Sección E.1. También se incorporan el plano eléctrico detallado de la PCB en Sección E.2, el layout de placa en Sección E.3 y la placa fabricada. Estos planos no sustituyen a los ensayos funcionales, pero fijan la arquitectura eléctrica de partida y sirven para revisar alimentaciones, señales, conectividad y enrutado.

2.3.2. Requisitos de diseño

La PCB se ha definido a partir de un conjunto de requisitos derivados del proyecto global:

- minimizar masa y volumen para que la placa quepa en el fuselaje;
- mantener compatibilidad con buses digitales de sensores y con las señales del sistema de control;
- recibir la alimentación desde el módulo externo del Subsección 2.1.7, manteniendo retornos, desacoplos y protecciones locales coherentes.

Estos requisitos obligan a tratar la PCB como un subsistema de integración, no solo como un soporte pasivo para soldar componentes.

2.3.3. Arquitectura eléctrica

La arquitectura de la placa se ha organizado en bloques funcionales bien separados. La alimentación principal procede del módulo externo descrito en el Subsección 2.1.7; por tanto, la PCB se limita a comunicar los sensores y actuadores con la Raspberry Pi, actuando como interfaz eléctrica entre la electrónica embarcada y el computador de vuelo. A continuación se sitúa la electrónica de sensado, donde conviven la IMU, el barómetro y el magnetómetro ya seleccionados. La placa tambien reserva salida hacia actuadores e interfaces para futuras etapas de comunicaciones.

2.3.4. Configuracion de buses de sensores

Los sensores seleccionados soportan comunicacion digital mediante SPI o I^2C, de acuerdo con las interfaces documentadas por sus fabricantes (STMI26, TECO26, MEMS26). I^2C, abreviatura de *Inter-Integrated Circuit*, es un bus serie compartido de dos lineas principales, datos y reloj, en el que varios dispositivos pueden convivir mediante direcciones distintas. SPI, *Serial Peripheral Interface*, es tambien una comunicacion serie, pero usa lineas separadas de reloj, entrada y salida de datos y seleccion de dispositivo; suele ofrecer mayor velocidad a costa de mas pistas y mas señales de seleccion.

Para mantener flexibilidad durante las primeras pruebas de placa, el esquematico incorpora resistencias de $0\,\Omega$ como enlaces de configuracion: soldando o retirando estos componentes puede seleccionarse la ruta electrica de bus o aislar una señal concreta durante depuracion. Esta solucion reduce el riesgo de rediseñar la placa si una primera estrategia de comunicacion presenta problemas de integridad de señal, direccionamiento o temporizacion. Las tres configuraciones previstas son:

- **Todos los sensores por I^2C**: IMU, barometro y magnetometro comparten el bus I^2C hacia la Raspberry Pi. Es la topologia mas simple en numero de pistas y la configuracion nominal de esta placa.

- **Todos los sensores por SPI**: los sensores se conectan por SPI hacia la Raspberry Pi, usando lineas comunes de reloj y datos y señales de seleccion independientes. Esta opcion prioriza velocidad y separacion de dispositivos, pero aumenta el numero de conexiones.

- **Topologia mixta**: el magnetometro se comunica por I^2C con la IMU; la IMU recibe y procesa esa informacion localmente y el resto de datos se transmite hacia la Raspberry Pi por SPI. Esta configuracion reduce el cableado directo del magnetometro hacia la Raspberry y concentra la salida principal en la IMU.

La configuracion nominal de vuelo se fija, no obstante, en I^2C para todos los sensores. Esta decision simplifica el conexionado con la Raspberry Pi 3B+, reduce el numero de lineas necesarias para agrupar IMU, barometro y magnetometro, y deja SPI o la topologia mixta como alternativas de montaje o diagnostico. La consecuencia directa es que el firmware o modelo ejecutado en la Raspberry queda encargado de gestionar direcciones, resistencias de pull-up, frecuencia de bus y tiempos de lectura de forma compatible con la frecuencia de control.

La placa montada para la configuracion nominal se muestra en la Figura 11. En ella se adopta la topologia I^2C para todos los sensores y no se montan las resistencias asociadas a la ruta SPI. La Figura 12 corresponde a una configuracion equipada con todos los enlaces de seleccion, por lo que conserva la posibilidad de probar distintas topologias de bus durante la puesta en marcha.

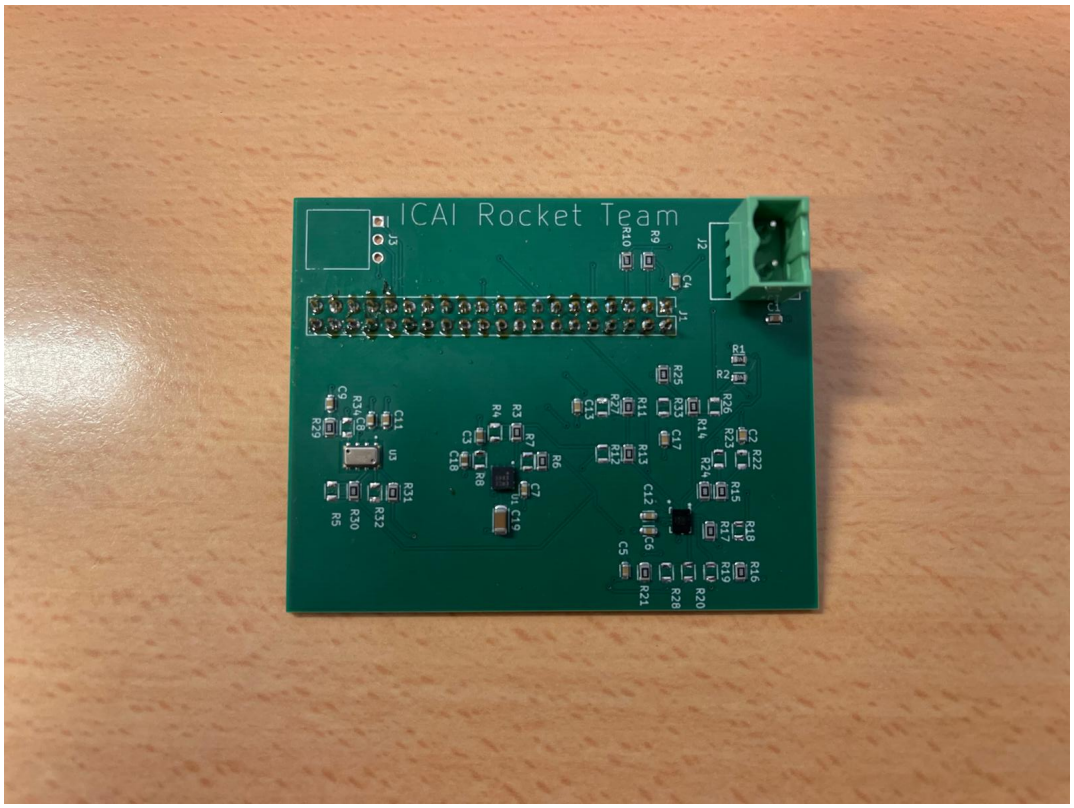


Figura 11. PCB fabricada con la topologia nominal I^2C seleccionada: todos los sensores quedan agrupados en I^2C y las rutas SPI no se montan.

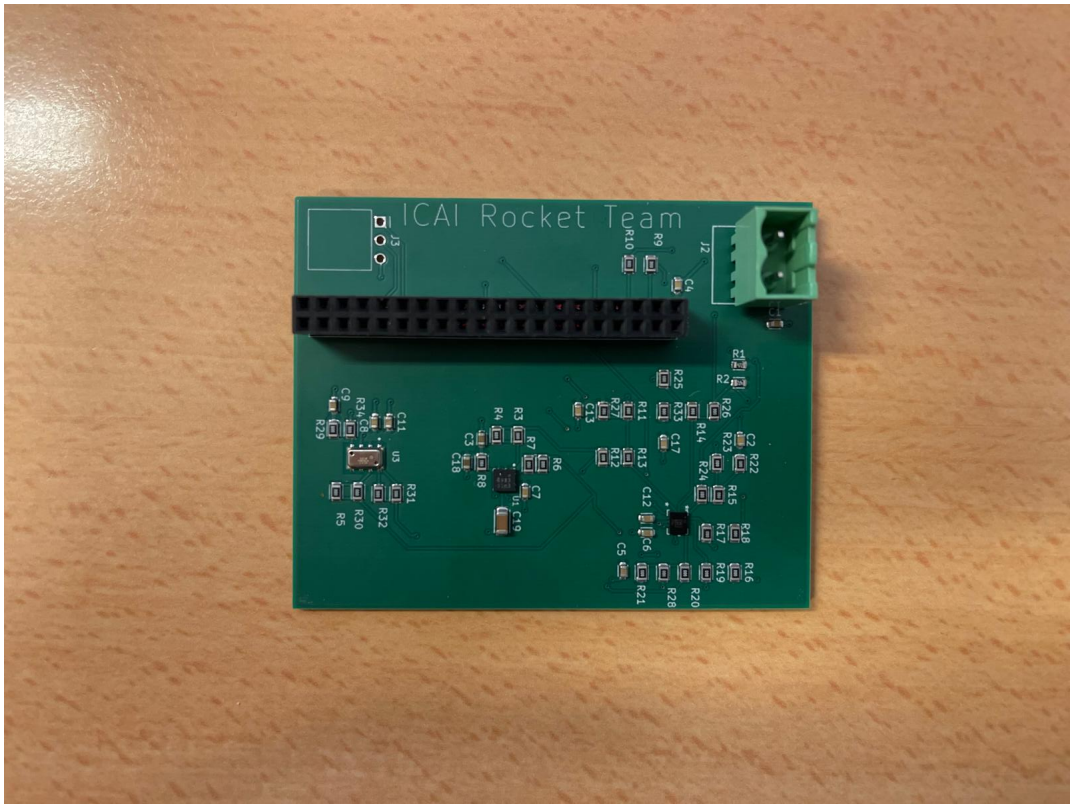


Figura 12. PCB equipada con los enlaces de configuracion, manteniendo la posibilidad de seleccionar fisicamente distintas topologias de bus durante las pruebas.

2.3.5. Interfaz con actuadores TVC

La Raspberry Pi 3B+ genera tres señales PWM y la PCB las encamina hasta el conector J3, de acuerdo con la documentacion GPIO de Raspberry Pi (RASP26b). Estas tres señales corresponden a los dos servos del TVC, pitch y yaw, y al actuador del aerofreno. El contrato electrico mantiene pulsos PWM de 500 a $2500\,\mu s$ con centro en $1500\,\mu s$ para los servos, de acuerdo con la ficha comercial usada como punto de partida (MIUZ26).

La alimentacion de estos servomotores se mantiene separada de la alimentacion de la PCB. De este modo, los transitorios de corriente de los actuadores no cargan la rama de la electronica de sensado ni la Raspberry Pi, reduciendo el riesgo de jittering en las señales PWM, reinicios o ruido acoplado en los sensores.

2.3.6. Criterios de layout

El desarrollo del esquemático es solo la primera mitad del trabajo. En el paso a layout la placa se ha resuelto como una PCB bicapa. Se ha usado una capa como red de $3{,}3\,V$ y la otra como plano de masa, lo que facilita el acceso a ambas referencias desde los integrados y reduce recorridos de retorno.

- se han colocado condensadores de desacoplo lo más cerca posible de cada circuito integrado. Estos condensadores son pequeños depósitos locales de carga conectados entre alimentación y masa; absorben variaciones rápidas de corriente y estabilizan la tensión que ve cada integrado;
- se ha buscado reducir la longitud de las pistas para limitar ruido acoplado y caídas de tensión en señales de comunicación y alimentación;
- se han evitado bucles de masa grandes que favorezcan acoplos electromagnéticos;
- en la zona del magnetómetro se han evitado planos o pistas cercanas de masa y alimentación, ya que corrientes próximas pueden alterar la medida del campo magnético.

En un cohete de pequeño tamaño, una mala decisión de layout puede traducirse en ruido en sensores, caídas de tensión locales o comunicaciones inestables aunque el esquema sea correcto. Por eso el diseño se ha tratado como parte de la electrónica de medida, no solo como un ejercicio de enrutado.

2.3.7. Verificación previa a fabricación

Antes de fabricar la placa se verificaron las alimentaciones, las polaridades, la coherencia de conectores, el consumo previsto y el montaje dentro de la caja de electrónica.

Durante el diseño se omitieron los taladros de fijación y no se alineó el conector de la Raspberry Pi con la posición real que ocupa en la placa. Esta limitación no compromete la integración final, ya que la conexión entre la Raspberry Pi y la PCB se realizará mediante cables.

2.3.8. Validación funcional inicial

Tras recibir la placa se realizó una validación funcional inicial. La comprobación se centró en verificar que no existieran cortocircuitos entre las líneas de alimentación y masa antes de conectar la Raspberry Pi, los sensores o los actuadores.

Capítulo 3

Diseño mecánico del cohete

El diseño mecánico condiciona el comportamiento del cohete. Para este trabajo se ha tomado como punto de partida el modelo desarrollado por el ICAI Rocket Team; los planos asociados se recogen en el anexo del reporte técnico (IRT26). A partir de esa base, en este capítulo se tratan con mayor detalle los cambios ya realizados y las modificaciones que se quieren introducir en el modelo para adaptarlo al aterrizaje retropropulsivo con TVC. La geometría del fuselaje, la posición del centro de gravedad, la posición del centro de presiones, la rigidez de la estructura y el módulo de TVC de la tobera determinan tanto el comportamiento dinámico como el vuelo.

3.1. Arquitectura mecánica

La arquitectura mecánica se divide en fuselaje, módulo de aviónica, módulo de propulsión, sistema de tobera/TVC y elementos desplegables como patas o aerofrenos. Para cada módulo se fijan volumen disponible, masa estimada, puntos de fijación y accesibilidad de montaje.

Elemento	**Función**	**Parámetro a cerrar**	**Unidad**
Fuselaje	Soporte estructural y envolvente aerodinámica.	Longitud, diámetro, material, espesor y momentos de inercia.	m, kg, kg*m^2
Módulo de aviónica	Alojamiento de sensores, PCB y alimentación.	Conexionado a actuadores y fijaciones.	m, kg
Módulo de propulsión	Integración del motor y transmisión de cargas.	Empuje máximo, curva de empuje y temperatura máxima.	N, K
Sistema TVC	Orientación del vector de empuje.	Momento, ángulo máximo y holguras.	m, rad
Aerofrenos	Incremento controlado del arrastre.	Número de paneles, área, posición, ángulo máximo y dinámica.	m, m^2, rad

Tabla 5. Parámetros mecánicos

3.2. Criterios de diseño

Los criterios mecánicos prioritarios son minimizar masa sin comprometer rigidez, mantener alineación entre eje del cohete y línea nominal de empuje, separar zonas calientes de la electrónica y permitir inspección tras cada ensayo. También se verifica que la distribución de masa resultante mantiene coherencia con el modelo de propiedades másicas usado en simulación.

3.3. Estructuras principales del cohete

El diseño estructural toma como punto de partida el cohete de competición del IRT 2026 preparado por el ICAI Rocket Team para la competición SURTAM 2026 y documentado en el reporte técnico final (IRT26). En este TFG se adopta la envolvente estructural y propulsiva como referencia, pero no se trasladan las soluciones de *recovery*, aviónica ni CanSat del informe, porque esos subsistemas se sustituyen por la arquitectura propia del proyecto.

Magnitud	**Valor reportado**	**Uso en el TFG**
Longitud total	$1{,}150\,m$	Envolvente exterior de referencia.
Diámetro máximo	$0{,}084\,m$	Diámetro aerodinámico y escala de estabilidad.
Masa sin motor	$2{,}339\,kg$	Referencia estructural previa a integrar el motor.
Masa completa	$2{,}814\,kg$	Masa completa reportada para la configuración de lanzamiento.
Centro de gravedad	$0{,}682\,m$	Medido desde la punta de la ojiva.
Centro de presiones	$0{,}818\,m$	Medido desde la punta de la ojiva.
Margen de estabilidad	$1{,}61$ calibres	Comprobación de estabilidad pasiva en ascenso.

Tabla 6. Valores mecánicos reportados del cohete de competición del IRT 2026

Segun el reporte tecnico final (IRT26), la arquitectura mecanica mantiene una construccion modular. La ojiva se fabrica en ABS con perfil tangente, longitud de $0{,}20\,m$ y base de $0{,}084\,m$, lo que da una relación de esbeltez aproximada de $2{,}38$, adecuada para vuelo subsónico. Los tramos resistentes usan tubo Quantum, que es PVC ligero de densidad $1{,}07\,\mathrm{g/cm^3}$ frente a los $1{,}39\,\mathrm{g/cm^3}$ del PVC convencional, y una funda exterior de cartón de $0{,}696\,m$. Los conectores se plantean en PETG y las aletas, anillos de centrado y piezas sometidas a mayor carga se fabrican en contrachapado de abedul de $6\,mm$. El PLA queda restringido a piezas internas auxiliares y no al cuerpo resistente del cohete. La vista de referencia de OpenRocket se recoge en el capítulo de modelado; en este capítulo, las figuras 13 y 14 muestran el montaje mecánico previsto y el detalle del compartimento en el que se fija la electrónica del proyecto.

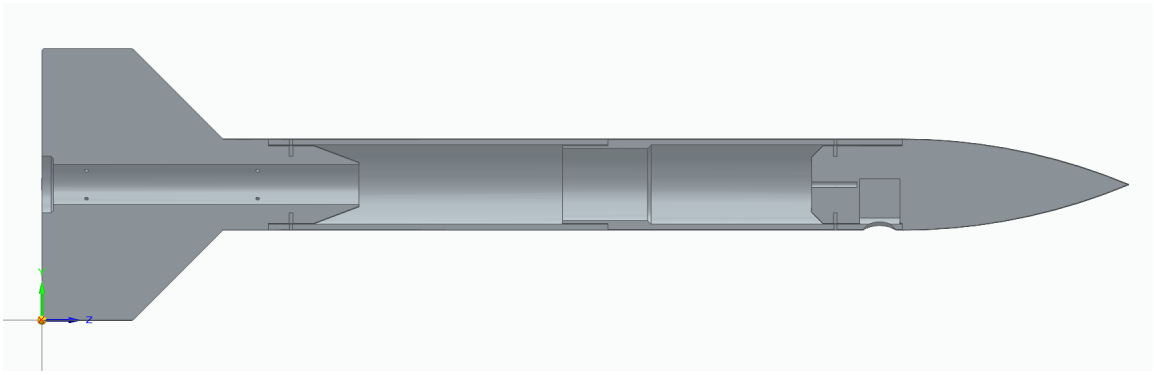


Figura 13. Vista de conjunto del montaje mecanico previsto para el cohete. Fuente: ICAI Rocket Team (IRT26).

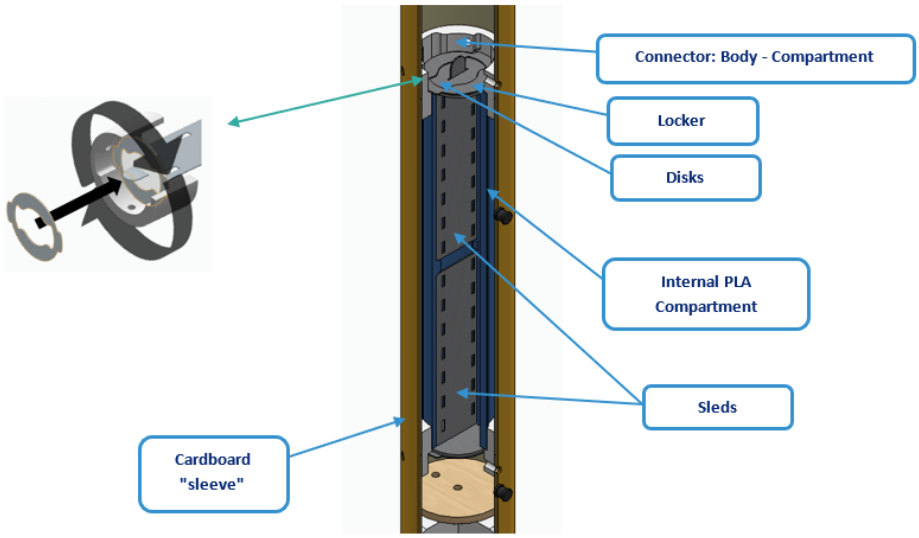


Figura 14. Detalle del compartimento interno y de los puntos de sujecion mecanica de la electronica. Fuente: ICAI Rocket Team (IRT26).

En el tramo inferior, el motor se aloja en un tubo interno de cartón centrado mediante tres anillos. Las aletas atraviesan ranuras del fuselaje y se unen al tubo del motor para transmitir cargas al eje resistente. La sección inferior se cierra mediante piezas de ABS y uniones atornilladas con insertos, de forma que el cohete pueda montarse y desmontarse para inspección y modificaciones.

La estructura transmite el empuje axial del motor sin desalinear la línea de acción respecto al eje del cohete. También mantiene la rigidez suficiente para que el sistema TVC no introduzca holguras que reduzcan la autoridad de control. Por este motivo, los anclajes del motor, la unión de la tobera, los acopladores internos y las fijaciones de la aviónica se consideran parte del modelo dinámico, no solo piezas de soporte.

3.4. Aerofrenos y elementos desplegables

Los aerofrenos se consideran parte de la arquitectura objetivo del cohete. El mecanismo previsto está formado por cuatro aletas desplegables alrededor del fuselaje, accionadas de forma simétrica para aumentar el arrastre sin introducir momentos laterales apreciables. Su mando se

representa mediante el ángulo de apertura δ_{ab}, expresado en radianes: $\delta_{ab} = 0$ corresponde a aerofrenos cerrados y valores positivos corresponden a paneles desplegados.

Desde el punto de vista mecánico, el aerofreno modifica el área proyectada frente al flujo y, con ello, la fuerza de drag disponible para control de energía y altitud. En el modelo de simulación actual se usa una primera aproximación con coeficiente $C_{d,ab}$ configurable y área efectiva proporcional al seno del ángulo de apertura limitado mecánicamente:

$$A_{ab,ef} = N_{ab}\, A_{panel}\, \text{sen}\left(\text{mín}\left(\text{máx}\left(\delta_{ab}, 0\right), \delta_{ab,max}\right)\right) \tag{39}$$

$$D_{ab} = q\, C_{d,ab}\, A_{ab,ef} \tag{40}$$

En estas expresiones N_{ab} es el número de paneles, A_{panel} es el área de cada aleta, $\delta_{ab,max}$ es el límite mecánico de apertura y q es la presión dinámica. La combinación de $\text{máx}(\delta_{ab}, 0)$ y $\text{mín}(\cdot, \delta_{ab,max})$ se ha introducido para representar los límites físicos del aerofreno: impide aperturas negativas y satura el despliegue en el ángulo mecánico máximo. Como configuración inicial del simulador se emplean cuatro paneles, $A_{panel} = 0{,}005\, m^2$, $C_{d,ab} = 1{,}2$, $\delta_{ab,max} = 90°$ y una posición axial aproximada de $0{,}30\, m$ medida desde la punta de la nariz. Estos valores no se interpretan como diseño final: fijan una interfaz verificable entre mecánica, aerodinámica y control.

Magnitud	**Símbolo**	**Criterio de diseño**
Número de paneles	N_{ab}	Cuatro aletas distribuidas de forma simétrica.
Ángulo de mando	δ_{ab}	Radianes; 0 cerrado y positivo desplegado.
Límite mecánico	$\delta_{ab,max}$	Saturación física del mecanismo antes de aplicar el modelo aerodinámico.
Área efectiva	$A_{ab,ef}$	Proporcional al área proyectada por el ángulo de apertura.

Tabla 7. Contrato mecánico y aerodinámico inicial de aerofrenos

La ubicación de los aerofrenos se elige de forma que no interfiera con la electrónica, el sistema de recuperación, las cargas de propulsión ni la unión de los tramos del fuselaje. También se comprueba su efecto sobre el centro de gravedad y sobre el centro de presiones en posición cerrada y desplegada. Si el despliegue no es simétrico, el simulador incorpora el momento adicional correspondiente y el control lo trata como una perturbación o como un fallo de actuación.

También se consideran elementos estructurales auxiliares las patas de aterrizaje, rail buttons, acopladores, soportes internos y caja de aviónica. Cada uno introduce masa, desplazamiento del centro de gravedad y posibles interferencias aerodinámicas. Por ello, cualquier modificación de estas piezas se refleja en el presupuesto de masas y en el cálculo de estabilidad.

3.5. Estabilidad pasiva y modificaciones realizadas

La estabilidad pasiva se evalúa comparando la posición del centro de gravedad con la del centro de presiones. Para vuelo hacia delante, el cohete es estáticamente estable si el centro

de presiones queda por detrás del centro de gravedad. El margen se expresa normalmente en calibres:

$$SM = \frac{x_{CP} - x_{CG}}{D} \tag{41}$$

En el reporte final del cohete de competición del IRT 2026 (IRT26) se toman $x_{CP} = 0{,}818\,m$, $x_{CG} = 0{,}682\,m$ y $D = 0{,}084\,m$, lo que proporciona un margen aproximado de $1{,}61$ calibres para la configuración completa. Esto confirma que el modelo base tiene estabilidad pasiva suficiente durante ascenso, pero el margen no se considera fijo: el modelo lo recalcula al variar la masa con el tiempo y al cambiar la configuración aerodinámica.

La apertura de los aerofrenos modifica la estabilidad pasiva porque introduce una superficie resistente adicional a una posición concreta del fuselaje. Esa contribución altera el centro de presiones efectivo y, por tanto, el margen SM. El signo del cambio depende de la posición relativa entre los aerofrenos, el centro de gravedad y el sentido de la corriente: si la resultante aerodinámica equivalente queda por detrás del centro de gravedad, aumenta la tendencia restauradora; si queda demasiado adelantada, puede reducir el margen. Por esta razón no se supone que el aerofreno sea siempre estabilizador, sino que se comprueba su efecto en configuración cerrada, parcialmente desplegada y desplegada.

En el caso nominal, los cuatro paneles se despliegan de forma simétrica alrededor del fuselaje. Así, el aerofreno se usa para aumentar el arrastre y modificar la estabilidad pasiva sin generar un momento lateral de mando. En el modelo, el mando δ_{ab} se satura entre 0 y $\delta_{ab,max}$, se calcula el área efectiva $A_{ab,ef}$ y se obtiene el incremento de arrastre mediante $D_{ab} = q\,C_{d,ab}\,A_{ab,ef}$. Esa fuerza se aplica en una posición axial equivalente del aerofreno y se combina con el resto de contribuciones aerodinámicas para actualizar el centro de presiones efectivo empleado en el cálculo del margen.

Esta aproximación permite representar el efecto principal de los aerofrenos en la estabilidad sin introducir todavía una caracterización aerodinámica detallada. Si el despliegue no es simétrico, el efecto deja de ser puramente pasivo y aparece un momento lateral adicional; en ese caso el simulador lo trata como perturbación o fallo de actuación, no como condición nominal de estabilidad. Los valores de $C_{d,ab}$, área de panel y posición axial se mantienen por tanto como parámetros revisables hasta validar el diseño con geometría final, CFD o ensayo.

La modificación principal observada en el diseño es la incorporación y ajuste de aletas posteriores. En el modelo OpenRocket del reporte (IRT26, NISK13) aparecen tres aletas de forma libre, con espesor de $0{,}006\,m$ y contorno definido por puntos. Su efecto es desplazar el centro de presiones hacia la cola y aumentar el margen de estabilidad. Sin embargo, aumentar demasiado el margen también incrementa el momento restaurador y exige más autoridad al TVC durante maniobras de actitud, de modo que la estabilidad pasiva se equilibra con la autoridad de control activo.

La segunda modificación relevante es el reparto de masas internas: motor, aviónica, recuperación, acopladores y masas auxiliares. Mover masa hacia la ojiva adelanta el centro de gravedad y aumenta la estabilidad pasiva; mover masa hacia la cola la reduce. Por ello, cada cambio mecánico actualiza el centro de gravedad y comprueba que el margen permanece positivo en las fases de vuelo previstas.

3.6. Tobera y sistema TVC

La tobera es el elemento que convierte la energía del motor en empuje útil y, al mismo tiempo, el punto físico donde se introduce la actuación TVC. Por ello se trata como un subsistema propulsión–mecánica–control, no como una pieza aislada.

3.6.1. Función de la tobera

En el modelo de control, la tobera proporciona un empuje T cuya dirección puede desviarse respecto al eje longitudinal del cohete. Para pequeños ángulos, el momento de control depende principalmente del empuje, de la deflexión y del brazo de palanca entre la línea de acción del empuje y el centro de gravedad.

El mecanismo de referencia que se va a usar para materializar esa orientación se muestra en la Figura 15. La misma imagen aparece en el capítulo de hardware porque fija la arquitectura servo–brazo–biela–gimbal: el servo no aplica directamente una deflexión de tobera, sino que desplaza un brazo y una transmisión mecánica que inclinan el conjunto orientable.

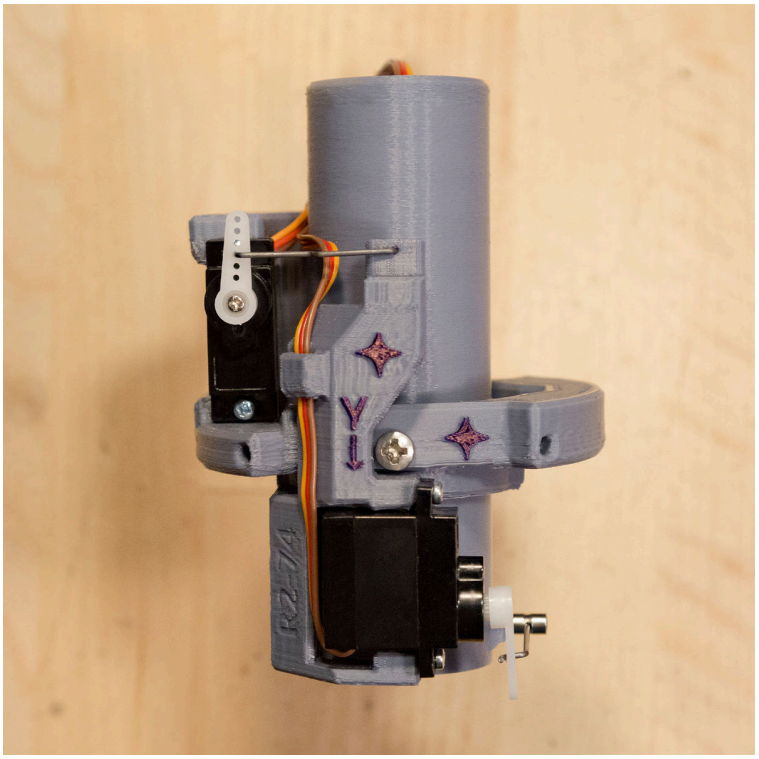


Figura 15. Mecanismo TVC tomado como referencia mecánica para la tobera orientable que se empleará en el prototipo. Fuente: BPS.Space (BPSS26).

En la arquitectura de aterrizaje retropropulsivo, el conjunto motor–tobera contempla dos cargas sólidas: una carga de ascenso, encendida en el lanzamiento, y una carga de descenso que permanece inactiva hasta recibir la orden de ignición durante la bajada. Ambas cargas se tratan como curvas de empuje impuestas a partir de sus respectivos instantes de activación, no como motores regulables en empuje. Por ello, el sistema TVC se plantea para orientar el empuje disponible tanto durante el ascenso como durante el frenado final, y el controlador decide el encendido de la carga de descenso para que el contacto con el suelo se produzca con una velocidad vertical muy cercana a $0\,m/s$.

La siguiente expresión define el par ejercido por el TVC:

$$M_{TVC} \simeq T\, l_{TVC}\, \delta \tag{42}$$

donde T es el empuje, l_{TVC} es el brazo efectivo y δ es la deflexión de la tobera en radianes. El esquema de la Figura 16 muestra esa relación física: la deflexión angular del vector de empuje introduce componentes laterales de fuerza y el brazo entre el gimbal y el centro de gravedad transforma esas fuerzas en momento de control.

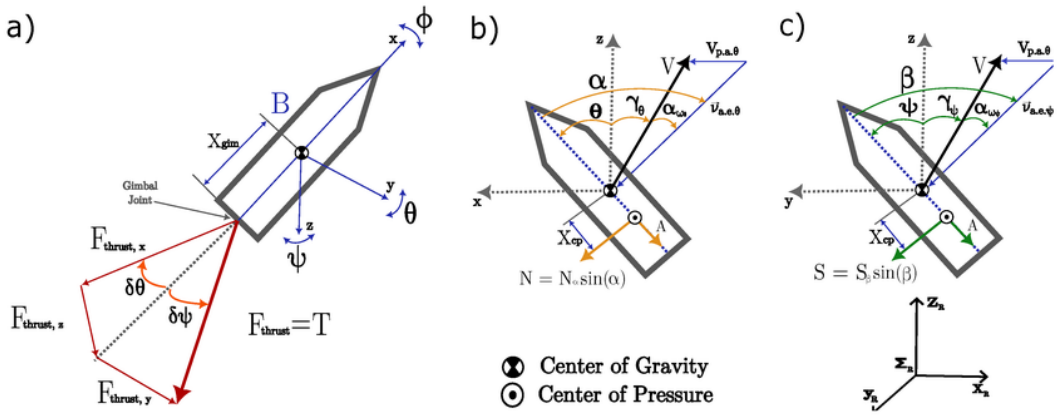


Figura 16. Esquema de TVC donde se identifican la deflexión del vector de empuje y la distancia del gimbal al centro de gravedad, usada como referencia del brazo efectivo l_{TVC}. Fuente: dos Santos y Oliveira (SANT24).

3.6.2. Criterios de diseño de la tobera

En una primera fase se pensó en diseñar un modelo propio de tobera orientable desde cero, incluyendo la pieza, el soporte del gimbal y la transmisión entre el servo y la tobera. Finalmente se seleccionó un diseño TVC existente como base mecánica, mostrado en la Figura 15, porque ya integra la arquitectura servo–brazo–biela–gimbal y permite concentrar el desarrollo del prototipo en la adaptación al cohete (BPSS26).

El funcionamiento es parecido al de un brazo robótico: el servo mueve un eslabón, ese desplazamiento se transmite mediante una biela y el conjunto convierte el giro del actuador en una inclinación controlada de la tobera. Con esta arquitectura, la fase actual se centra en adaptar el mecanismo al motor y al soporte del cohete, manteniendo como variables de ajuste la alineación, la holgura, el recorrido disponible y la saturación angular real.

3.6.3. Límites de actuación

El modelo de tobera usa deflexión máxima, velocidad máxima de deflexión, retardo, ancho de banda de servo y saturación del mando. Para el diseño se adopta un rango de deflexión de $\pm 5°$ a $\pm 15°$, equivalente a $\pm 0{,}0873$–$\pm 0{,}2618\, rad$. El extremo inferior evita que la autoridad de control resulte demasiado baja, ya que el par TVC crece de forma aproximadamente lineal con $T\, l_{TVC}\, \delta$. El extremo superior limita la pérdida de componente axial de empuje, mantiene la aproximación de pequeño ángulo usada en el modelo y reduce las cargas, holguras y recorridos exigidos al conjunto servo–brazo–biela–gimbal.

Con el servo Miuzei de $270°$ se adopta un contrato inicial de actuador para simulación y diseño de control a partir de su ficha comercial (MIUZ26). El recorrido electrico total del servo no se aplica directamente a la tobera: el mecanismo TVC mantiene una saturación fisica de tobera mucho menor, mientras que el PWM del servo se usa como señal electrica intermedia. Esta separacion evita confundir el recorrido del servo con la deflexion angular real del chorro.

Parámetro	Símbolo	Valor inicial	Origen
Deflexión máxima	δ_{max}	$\pm 5°$–$\pm 15°$ ($\pm 0{,}0873$–$\pm 0{,}2618\,rad$)	Modelo
Velocidad máxima	$\dot{\delta}_{max}$	$600°/s$ nominal sin carga	Ficha comercial del servo (MIUZ26)
Ancho de banda	f_{servo}	$8\,Hz$ inicial	Modelo de simulación
Retardo	τ_d	$20\,ms$ inicial	Modelo de simulación

Tabla 8. Parámetros del modelo TVC

3.6.4. Interfaz con simulación y control

La salida del subsistema de tobera hacia el simulador se expresa como fuerzas y momentos en ejes cuerpo, con unidades SI y signos documentados. La entrada desde el controlador queda limitada por el modelo de actuador antes de aplicarse a la dinámica del cohete. Esta separación evita que el controlador ordene maniobras que el mecanismo real no pueda ejecutar.

Capítulo 4

Programa de MATLAB

EXISTEN cuatro bloques diferenciados dentro del software del proyecto. El primero es el simulador no lineal 6DOF, cuya planta física se evalúa en `state_derivatives.m`. El segundo es el modelo Simulink `COHETE_CONTROL_SYSTEM.slx`, que encapsula esa misma planta y la conecta con el control implementado en `rocket_feedback_controller.m` y `rocket_landing_controller.m`. El tercero es la interfaz gráfica, que actúa como punto de entrada para configurar parámetros, ejecutar simulaciones e inspeccionar estados, mandos y estimación. El cuarto es la capa de despliegue de banco para Raspberry Pi 3B+, preparada para llevar el modelo a hardware con sensores por I2C y servos por PWM. Este capítulo documenta qué está implementado, qué ecuaciones usa la planta y qué papel tiene cada bloque dentro del flujo MATLAB/Simulink.

4.1. Objetivo y alcance del programa

El objetivo del programa no es solo dibujar una trayectoria, sino disponer de un entorno trazable para probar decisiones de diseño antes de llevarlas al hardware. Por ello el simulador se ha organizado alrededor de una configuración común y una planta dinámica no lineal separada de la generación de mandos. Sobre esa misma planta se prueban cuatro modos principales: vuelo vertical sin TVC, maniobras con TVC en lazo abierto, controlador discreto de actitud con TVC y controlador LQR.

La implementación principal trabaja con las siguientes hipótesis de primer orden:

- unidades SI en todos los parámetros físicos;
- marco inercial local NED para posición;
- velocidades lineales y angulares expresadas en ejes cuerpo;
- actitud representada mediante cuaterniones $[q_0, q_1, q_2, q_3]$;
- altura de apogeo suficientemente baja para despreciar la curvatura terrestre y la variación de gravedad con la altura;
- empuje no regulable durante el tiempo de quemado, salvo escalado de mando en perfiles de prueba;
- TVC limitado por saturación angular, y en el autopiloto también por retardo, ancho de banda y velocidad máxima.

Estas hipótesis no cierran el diseño experimental final, pero sí fijan una base coherente para comparar configuraciones, estudiar sensibilidad y preparar la integración con control y hardware.

4.2. Estructura del software

La función `state_derivatives.m` es el núcleo donde se evalúa $\dot{x} = f(t, x, u)$ para la planta 6DOF. Alrededor de ella se organizan las capas de configuración, ejecución, interfaz gráfica, control, Simulink, sensores, estimación, propulsión, despliegue en Raspberry y validación. La tabla siguiente resume todas esas capas.

Capa	**Papel dentro del flujo**
Configuración y ejecución MATLAB	Define parámetros físicos y numéricos, construye el estado inicial, ejecuta la integración temporal y genera casos de prueba básicos.
Planta física 6DOF	Calcula masa e inercia, atmósfera, viento, gravedad, fuerzas aerodinámicas, TVC, resultantes y derivadas de estado; `state_derivatives.m` es solo la función que agrupa esa evaluación dinámica.
Control del simulador base	Implementa vuelo vertical sin TVC, perfiles TVC en lazo abierto y controlador discreto de actitud con filtrado, retardo y servo.
Rama LQR	Contiene el flujo de linealización, diseño LQR y simulación alternativa de control en espacio de estados.
Bloques Simulink de planta	Proporciona wrappers, empaquetado del estado, bloques funcionales `rocket_block_*` y modelos Simulink que reutilizan la misma planta.
Arquitectura Simulink-first	Agrupa configuración, planta, controlador, sensores, EKF, propulsión de dos cargas y los modelos `CONTROL_COHETE.slx` y `COHETE_CONTROL_SYSTEM.slx`.
Interfaces gráficas	Permiten editar parámetros, seleccionar modos de control, ejecutar simulaciones y revisar trayectoria, estados, mandos, sensores y estimación.
Banco Raspberry	Prepara el contrato de banco con Raspberry Pi 3B+, sensores I^2C, PWM de servos y despliegue seco o real sin habilitar ignición.
Validación	Contiene pruebas unitarias, de integración y smoke tests para física local, Simulink, GUI, sensores, EKF, control y Raspberry.

Tabla 9. Estructura completa del software MATLAB/Simulink

La traza de ejecución común es: configuración, planta 6DOF, sensores simulados, estimación, controlador, actuadores y registros de salida. En MATLAB puro esa traza se ejecuta desde `simulate_rocket.m`; en Simulink se reparte entre bloques de planta, estimación y control; en la GUI se configura e inspecciona sin duplicar ecuaciones físicas. Por tanto, `state_derivatives.m` queda situado como una pieza interna de la planta, no como la estructura completa del software.

4.3. Vector de estado y convenciones

El vector de estado tiene trece componentes:

$$x = \begin{bmatrix} p_N \\ p_E \\ p_D \\ u \\ v \\ w \\ q_0 \\ q_1 \\ q_2 \\ q_3 \\ p \\ q \\ r \end{bmatrix}^T \tag{43}$$

donde (p_N, p_E, p_D) es la posición NED en metros, (u, v, w) es la velocidad lineal en ejes cuerpo en m/s, (q_0, q_1, q_2, q_3) es el cuaternion de actitud y (p, q, r) es la velocidad angular en rad/s. En NED la coordenada p_D es positiva hacia abajo, por lo que la altitud empleada en gráficas y eventos se calcula como:

$$h = -p_D \tag{44}$$

La matriz de rotación usada para pasar de NED a cuerpo procede de la expresión estándar de cuaterniones. Para $q = [q_0, q_1, q_2, q_3]$, el programa implementa:

$$R_{n\to b} = \begin{bmatrix} 1-2(q_2^2+q_3^2) & 2(q_1q_2-q_0q_3) & 2(q_1q_3+q_0q_2) \\ 2(q_1q_2+q_0q_3) & 1-2(q_1^2+q_3^2) & 2(q_2q_3-q_0q_1) \\ 2(q_1q_3-q_0q_2) & 2(q_2q_3+q_0q_1) & 1-2(q_1^2+q_2^2) \end{bmatrix} \tag{45}$$

La transformación inversa se obtiene por traspuesta, $R_{b\to n} = R_{n\to b}^T$, al ser una matriz ortonormal. Se emplean cuaterniones en lugar de ángulos de Euler porque el lanzamiento vertical sitúa al cohete cerca de configuraciones con cabeceo elevado, donde una parametrización Euler puede perder independencia entre ejes y producir la singularidad conocida como bloqueo de cardán. Con cuaterniones la actitud se representa mediante cuatro componentes normalizados, sin depender de una secuencia de giros sucesivos; por eso la integración de la actitud sigue siendo continua aunque el cohete pase por orientaciones próximas a la vertical.

4.4. Modelo dinámico 6DOF

La dinámica de `state_derivatives.m` parte de las ecuaciones clásicas de cuerpo rígido en seis grados de libertad. La formulación de referencia se toma del artículo *Integrated architecture for navigation and attitude control of low-cost suborbital launch vehicles*, de dos Santos y Oliveira (SANT24). El modelo recoge la forma general:

$$\dot{p}^{\,n} = R_{b\to n} v^b \tag{46}$$

donde p^n y v^b son la posición y la velocidad ya introducidas en el vector de estado, $R_{b\to n}$ es la transformación de ejes cuerpo a NED definida como traspuesta de $R_{n\to b}$, y el punto indica derivada temporal.

$$m\dot{v}^b = F^b - \omega^b \times m v^b \tag{47}$$

donde v^b y ω^b son las velocidades lineal y angular ya definidas en el estado, m es la masa instantánea del cohete en kg y F^b es la fuerza externa total en ejes cuerpo en N. El término $\omega \times mv$ aparece porque la derivada de la velocidad se calcula en un sistema de referencia rotante solidario al cohete.

$$J\dot{\omega}^b = M^b - \omega^b \times J\omega^b \tag{48}$$

donde ω^b es la velocidad angular ya definida, J es la matriz de inercia instantánea respecto al centro de masas, expresada en ejes cuerpo en kg m^2, y M^b es el momento externo total en ejes cuerpo en N m. El término $\omega \times J\omega$ es el término giroscópico de la dinámica rotacional.

La cinemática de actitud se integra mediante:

$$\dot{q} = \frac{1}{2}\Omega(\omega)q \tag{49}$$

donde q es el cuaternión de actitud ya definido en el vector de estado y $\Omega(\omega)$ es la matriz cinemática que convierte velocidad angular en derivada de cuaternión.

$$\Omega(\omega) = \begin{bmatrix} 0 & -p & -q & -r \\ p & 0 & r & -q \\ q & -r & 0 & p \\ r & q & -p & 0 \end{bmatrix} \tag{50}$$

donde p, q y r son las componentes de $\omega^b = [p, q, r]^T$ ya introducidas. En la notación aeroespacial usada aquí, p es la velocidad angular de alabeo alrededor del eje x_b, q es la velocidad angular de cabeceo alrededor del eje y_b y r es la velocidad angular de guiñada alrededor del eje z_b, todas en rad/s. Por tanto, el símbolo q que aparece dentro de $\Omega(\omega)$ es una componente escalar de velocidad angular, distinta del cuaternión de actitud $q = [q_0, q_1, q_2, q_3]^T$ definido antes.

Tras cada paso relevante se normaliza el cuaternión para limitar la deriva numérica. Esta normalización no cambia la actitud física, pero evita que errores acumulados por integración destruyan la propiedad $\|q\| = 1$.

4.5. Fuerzas y momentos implementados

En cada evaluación de la planta se suman tres fuerzas en ejes cuerpo: empuje, aerodinámica y gravedad. El momento total suma el momento de TVC y los momentos aerodinámicos.

4.5.1. Empuje con TVC

La ley de control del simulador base devuelve:

$$u = \begin{bmatrix} T & \mu_p & \mu_y \end{bmatrix}^T \tag{51}$$

donde T es el empuje en N, μ_p la deflexión de pitch en rad y μ_y la deflexión de yaw en rad. La fuerza de propulsión se calcula proyectando el empuje T sobre los ejes cuerpo cuando la tobera se deflecta en pitch y yaw, siguiendo la formulación TVC de dos Santos y Oliveira (SANT24):

$$F_T^b = \begin{bmatrix} T\cos\mu_p\cos\mu_y \\ -T\cos\mu_p\sin\mu_y \\ -T\sin\mu_p \end{bmatrix} \tag{52}$$

El momento asociado al gimbal se calcula con el brazo ℓ entre el punto de giro y el centro de masas:

$$M_T^b = \begin{bmatrix} 0 \\ -T \sin \mu_p \, \ell \\ T \cos \mu_p \sin \mu_y \, \ell \end{bmatrix} \tag{53}$$

Estas ecuaciones se han usado porque relacionan directamente la deflexión de la tobera con fuerza y momento en ejes cuerpo, que es la interfaz natural de la dinámica 6DOF. Para pequeños ángulos se obtiene la aproximación habitual $M_y \simeq -T\ell\mu_p$ y $M_z \simeq T\ell\mu_y$.

4.5.2. Gravedad

La gravedad se modela como constante porque la altura de apogeo del caso estudiado es pequeña frente al radio terrestre. Con esa escala de vuelo, la curvatura de la Tierra y la variación de g con la altura se consideran despreciables para la simulación:

$$g(h) = g_0 \tag{54}$$

En NED, la gravedad apunta en el eje down:

$$F_g^n = \begin{bmatrix} 0 \\ 0 \\ mg_0 \end{bmatrix}, \qquad F_g^b = R_{n \to b} F_g^n \tag{55}$$

El propio código deja preparada la extensión $g = g_0(R_T/(R_T + h))^2$, pero actualmente no se usa porque el rango de altitud simulado es pequeño frente al radio terrestre.

4.5.3. Viento y velocidad relativa

El modelo ambiental actual separa dos conceptos: la atmósfera, que se evalúa con el modelo ISA, y el viento, que entra como perturbacion de la velocidad relativa respecto al aire. La configuración guarda como referencia el campo de Aerocinca, en Alcolea de Cinca, con cota de referencia $213\,m$ MSL, techo autorizado de $1000\,m$ AGL, viento medio de $3{,}472\,m/s$, sector dominante WNW/NW y carril de lanzamiento de $3{,}0\,m$. Se ha utilizado una previsión de Open Meteo para el 11 de junio de 2026 como referencia de campo; este archivo se usará para simular las perturbaciones de viento y ráfagas.

La función `wind_model(altitude, config, t)` devuelve siempre un vector de viento en el marco local NED, $v_w^n = [V_N, V_E, V_D]^T$, en m/s. Si el perfil está desactivado, devuelve el vector nulo. Si está activado, usa `config.environment.wind` como viento base y selecciona uno de cuatro modos:

- `constant`: mantiene el viento base constante, que es el comportamiento histórico del simulador.
- `logarithmic`: escala el viento con una ley logarítmica de cizalladura usando la rugosidad superficial z_0 y una altura de referencia z_{ref}.
- `power_law`: aplica una ley potencial con exponente α; por debajo de z_{ref} conserva el viento de referencia para evitar que el viento colapse artificialmente cerca del aterrizaje.
- `sinusoidal_gust`: suma una ráfaga determinista, de amplitud vectorial NED, frecuencia y fase configurables.

Las dos leyes verticales implementadas pueden escribirse como:

$$v_w^n(h) = \frac{\ln(z/z_0)}{\ln(z_{ref}/z_0)}\, v_{w,ref}^n, \qquad z = \max(h, z_0) \tag{56}$$

$$v_w^n(h) = \left(\frac{\max(h, z_{ref})}{z_{ref}}\right)^{\alpha} v_{w,ref}^n \tag{57}$$

En el modo de ráfaga sinusoidal se usa:

$$v_w^n(t) = v_{w,ref}^n + A_g \sin(2\pi f_g t + \phi_g) \tag{58}$$

La altura que recibe el modelo de viento es la altura local sobre el punto de lanzamiento, usada como AGL por la dinámica. Antes de calcular aerodinámica, el viento se transforma a ejes cuerpo y se resta de la velocidad del cohete:

$$v_{air}^b = v^b - R_{n\to b} v_w^n \tag{59}$$

Esta diferencia es importante: las fuerzas aerodinámicas dependen de la velocidad relativa respecto al aire, no de la velocidad respecto al suelo. Por eso el viento no se suma como una fuerza externa directa; modifica la presión dinámica, el número de Mach, el ángulo de ataque y el resbalamiento a través de v_{air}^b.

4.6. Atmósfera ISA

El bloque `atmosphere_model.m` implementa una atmósfera ISA simplificada a partir de la atmósfera estándar estadounidense de 1976 (NOAA76). En la troposfera, hasta $11000\,m$, la temperatura se calcula con gradiente lineal:

$$T(h) = T_0 + Lh \tag{60}$$

donde $T_0 = 288{,}15\,K$ y $L = -0{,}0065\,K/m$. La presión se obtiene mediante la fórmula barométrica:

$$P(h) = P_0 \left(\frac{T(h)}{T_0}\right)^{-\frac{g_0}{RL}} \tag{61}$$

y la densidad mediante gas ideal:

$$\rho(h) = \frac{P(h)}{R\,T(h)} \tag{62}$$

La velocidad del sonido se calcula como:

$$a(h) = \sqrt{\gamma R T(h)} \tag{63}$$

Por encima de la troposfera el programa mantiene temperatura constante y aplica una caída exponencial de presión. En la práctica, las simulaciones actuales no dependen críticamente de esa extensión porque el vuelo validado permanece por debajo de ese rango.

4.7. Modelo aerodinámico

El modelo aerodinámico calcula fuerzas y momentos a partir de la velocidad relativa en ejes cuerpo. La presión dinámica se define como:

$$\bar{q} = \frac{1}{2}\rho V_{air}^2 \tag{64}$$

El ángulo de ataque total se calcula con:

$$\alpha = \tan^{-1}\left(\frac{\sqrt{v^2 + w^2}}{u}\right) \tag{65}$$

El uso de `atan2` distingue vuelo hacia delante y hacia atrás, de modo que el modelo representa ángulos de ataque hasta $180°$. Esta decisión fue introducida para evitar que el cohete mantuviera un comportamiento artificialmente estable cuando cae de cola tras el apogeo.

4.7.1. Arrastre

La ecuación base se toma del informe técnico Estes TR-11, *Aerodynamic Drag of Model Rockets*, que presenta la formulación clásica de arrastre para cohetes modelo (GREG70):

$$D = C_D \frac{1}{2}\rho V^2 A_{ref} \tag{66}$$

El coeficiente a ángulo de ataque cero se obtiene por suma de componentes:

$$C_{D0} = C_{D,nose} + C_{D,body} + C_{D,base} + C_{D,fins} + C_{D,int} \tag{67}$$

Sobre ese valor se aplica un incremento subsónico con Mach:

$$C_{D,M} = C_{D0}\left(1 + 0{,}25M^2\right), \qquad M < 0{,}9 \tag{68}$$

Para ángulo de ataque se incorpora arrastre inducido:

$$C_{D,i} = \frac{C_L^2}{\pi AR e}, \qquad C_L = C_{L\alpha}\alpha, \qquad AR = \frac{2b}{C_R + C_T} \tag{69}$$

donde b es la envergadura de aleta, C_R la cuerda raíz, C_T la cuerda de punta y e el factor de eficiencia. La envergadura b representa la distancia que sobresale la aleta desde el fuselaje hasta su borde exterior. La cuerda raíz C_R es la longitud de la aleta en la zona unida al cuerpo del cohete, medida aproximadamente en la dirección longitudinal, mientras que la cuerda de punta C_T es la longitud equivalente en el extremo libre de la aleta. Con estas dos cuerdas se aproxima el área en planta de una aleta trapezoidal como $S_f = b(C_R + C_T)/2$, de modo que la relación de aspecto usada por el modelo puede escribirse como $AR = b^2/S_f = 2b/(C_R + C_T)$. El parámetro e es adimensional y corrige la idealización de aleta perfecta: valores cercanos a uno representan una distribución de sustentación más eficiente, y valores menores aumentan el arrastre inducido al recoger pérdidas por geometría no elíptica, interferencia con el fuselaje y efectos tridimensionales. La forma de esta corrección de arrastre inducido se toma de la documentación técnica de OpenRocket (NISK13). La diferencia respecto a OpenRocket no está en la expresión básica, sino en su uso dentro del simulador del TFG: se reevalúa en cada paso con la geometría local de la configuración del cohete de competición del IRT 2026, con $C_L = C_{L\alpha}\alpha$ y con un valor constante $e = 0{,}7$ para cerrar el modelo. La fuerza de arrastre se aplica opuesta a la velocidad relativa:

$$F_D^b = -\bar{q} A_{ref} C_D \frac{v_{air}^b}{\|v_{air}^b\|} \tag{70}$$

Para altos ángulos de ataque, el código mezcla el valor anterior con una estimación de cuerpo en tumbado basada en el tratamiento de cuerpos en caída de OpenRocket (NISK13). De OpenRocket se toman los coeficientes de perfil empleados para el cuerpo y las aletas en esa situación ($C_{D,body} = 0{,}56$ y $C_{D,fin} = 1{,}42$), junto con el criterio de eficiencia equivalente por número de aletas. La diferencia del modelo MATLAB es que no reproduce el solucionador completo de OpenRocket: escala esas contribuciones al área de referencia del simulador, conserva el mayor valor entre el arrastre nominal y el arrastre de cuerpo tumbado, y activa la transición con una función suave propia. La mezcla se activa entre $35°$ y $75°$ mediante un polinomio suave:

$$s_{AoA}(\tau) = \tau^2(3 - 2\tau), \qquad \tau = \frac{\alpha - \alpha_0}{\alpha_1 - \alpha_0} \tag{71}$$

con $\alpha_0 = 35°$ y $\alpha_1 = 75°$. En la implementación, τ se interpreta como una coordenada normalizada de transición: vale 0 cuando $\alpha \leq \alpha_0$, vale 1 cuando $\alpha \geq \alpha_1$ y toma valores intermedios entre ambos límites. El polinomio $s_{AoA}(\tau) = 3\tau^2 - 2\tau^3$ se usa porque cumple $s_{AoA}(0) = 0$, $s_{AoA}(1) = 1$ y, además, $s'_{AoA}(0) = s'_{AoA}(1) = 0$. Por tanto, no introduce saltos bruscos ni en el valor ni en la pendiente de las fuerzas aerodinámicas. El coeficiente final se obtiene como una interpolación del tipo:

$$C_D = (1 - s_{AoA}) C_{D,nom} + s_{AoA} \ \text{máx}\,(C_{D,nom}, C_{D,tumbado}) \tag{72}$$

Con $s_{AoA} = 0$ se conserva el modelo nominal de pequeño ángulo de ataque, y con $s_{AoA} = 1$ domina la estimación de cuerpo en tumbado. Esto evita aplicar un modelo de pequeño ángulo cuando el cohete presenta el cuerpo de lado al flujo.

4.7.2. Fuerza normal y momento aerodinamico

La fuerza normal se calcula con un coeficiente proporcional a $\sin\alpha$:

$$C_N = (1 - s_{AoA})^2 C_{N\alpha} \sin\alpha \tag{73}$$

El término $(1 - s_{AoA})^2$ reduce la fuerza normal restauradora cuando el vehículo entra en régimen de alto ángulo de ataque, de forma coherente con el cambio hacia un comportamiento de cuerpo en tumbado. Aquí s_{AoA} es el factor de mezcla definido en la ecuación 71; es adimensional y no representa la variable compleja s de Laplace. La dirección de la fuerza normal se toma opuesta a la componente transversal de la velocidad:

$$n_t = \frac{1}{\sqrt{v^2 + w^2}} \begin{bmatrix} 0 \\ v \\ w \end{bmatrix}, \qquad F_N^b = -\bar{q} A_{ref} C_N n_t \tag{74}$$

El momento restaurador se obtiene con el producto vectorial entre el brazo desde el centro de gravedad al centro de presión y la fuerza normal:

$$M_{rest}^b = r_{CG \to CP}^b \times F_N^b \tag{75}$$

El centro de presión y el centro de gravedad están definidos como distancias desde la punta, mientras que el eje x cuerpo apunta en la dirección longitudinal del cohete. Por ello el código transforma el signo del brazo para mantener coherencia entre la geometría y los ejes cuerpo.

4.7.3. Amortiguamiento aerodinámico

El amortiguamiento de pitch y yaw usa las expresiones cuadráticas de la documentación técnica de OpenRocket (NISK13). Las constantes $0{,}275$ para el cuerpo y $0{,}3$ para aletas se toman de esa formulación. La diferencia respecto a OpenRocket está en la implementación: aquí se evalúan como términos locales en ejes cuerpo con el centro de gravedad instantáneo, se aplica el signo manualmente para oponerse a la velocidad angular, se limita el número de aletas expuestas al caso del simulador y se multiplica pitch/yaw por una ganancia adicional cuando aumenta la mezcla de alto ángulo de ataque. Para el cuerpo se usa:

$$M_{d,body} = 0{,}275\,\rho\, r_t \left(l_{fore}^4 + l_{aft}^4\right) \dot{\theta}^{\,2} \tag{76}$$

y para aletas:

$$M_{d,fin} = 0{,}3\,\rho\, N_f A_f \xi_f^3 \dot{\theta}^{\,2} \tag{77}$$

El signo se aplica siempre en oposición a la velocidad angular. En roll no se ha trasladado el modelo completo de OpenRocket; se mantiene de momento un término lineal simplificado:

$$M_{d,roll} = \bar{q} A_{ref} \frac{L_{ref}^2}{V_{air}} C_{d,p} p \tag{78}$$

En las ecuaciones 76 y 77, $M_{d,body}$ y $M_{d,fin}$ son las magnitudes de momento amortiguador debidas, respectivamente, al cuerpo cilíndrico y al conjunto de aletas. La densidad del aire es ρ en kg/m^3; $r_t = d_{ref}/2$ es el radio exterior del fuselaje usado por el modelo; l_{fore} y l_{aft} son las longitudes de fuselaje por delante y por detrás del centro de gravedad instantáneo; y $\dot{\theta}$ representa la velocidad angular transversal que se quiere amortiguar. En la implementación no es un nuevo estado: para cabeceo se sustituye por q y para guiñada por r, ambas en rad/s. El término N_f es el número de aletas expuestas usado por el simulador, A_f es el àrea en planta de una aleta y ξ_f es la distancia efectiva entre el centro de gravedad y el centro aerodinámico de las aletas, aproximado mediante el centro de presiones. Las potencias cuarta y tercera hacen que las zonas alejadas del centro de gravedad pesen más en el amortiguamiento, porque generan mayor brazo aerodinámico.

En la ecuación 78, $M_{d,roll}$ es el momento de amortiguamiento alrededor del eje longitudinal del cohete. La presión dinámica es $\bar{q} = \frac{1}{2}\rho V_{air}^2$, A_{ref} es el àrea de referencia, L_{ref} la longitud de referencia, V_{air} la velocidad relativa respecto al aire, $C_{d,p}$ un coeficiente adimensional de amortiguamiento de roll y p la velocidad angular de roll en ejes cuerpo. En el código actual se usa $C_{d,p} = -0{,}1$, por lo que el signo de esta contribución ya queda incluido en el propio coeficiente. En cambio, las ecuaciones de pitch y yaw se calculan como magnitudes positivas y después se multiplican por $-\,\mathrm{sign}(q)$ o $-\,\mathrm{sign}(r)$ para que el momento resultante se oponga siempre al giro. Estos términos fueron introducidos para reducir oscilaciones no realistas tras el apogeo, especialmente cuando el vehículo entra en altos ángulos de ataque y la velocidad aerodinámica es baja.

4.8. Propiedades másicas variables

El consumo de propelente se modela lineal durante el tiempo de quemado:

$$m_p(t) = \begin{cases} m_{p0}, & t < 0 \\ m_{p0}\left(1 - \frac{t}{t_b}\right), & 0 \le t \le t_b \\ 0, & t > t_b \end{cases} \tag{79}$$

La masa total es:

$$m(t) = m_{dry} + m_p(t) \tag{80}$$

El centro de gravedad se calcula como media ponderada:

$$x_{CG}(t) = \frac{m_{dry} x_{CG,dry} + m_p(t) x_{CG,p}}{m(t)} \tag{81}$$

Las inercias se aproximan escalando linealmente la contribución del propelente restante:

$$J_x(t) = J_{x,dry} + J_{x,p}\frac{m_p(t)}{m_{p0}}, \qquad J_y(t) = J_z(t) = J_{t,dry} + J_{t,p}\frac{m_p(t)}{m_{p0}} \tag{82}$$

Esta simplificación es razonable para una primera versión porque mantiene la variación de masa, CG e inercia sin introducir todavía un modelo de grano, tanque o geometría interna de propelente.

4.9. Integración temporal y eventos

El simulador principal de MATLAB, ejecutado por `simulate_rocket.m`, usa `ode45`, un método Runge–Kutta adaptativo de MATLAB, con tolerancias relativas y absolutas definidas en `rocket_config.m`. La función de eventos detecta dos condiciones:

- contacto con el suelo cuando h cae por debajo del umbral configurado;
- apogeo cuando la velocidad vertical ascendente cambia de signo.

La velocidad vertical se obtiene transformando la velocidad body a NED:

$$v^n = R_{b\to n} v^b, \qquad v_{up} = -v_D^n \tag{83}$$

El evento de aterrizaje solo se considera válido si el cohete desciende con $v_{up} < -0{,}5\,m/s$. Esta condición evita que el integrador detenga la simulación en el instante inicial, cuando el cohete está en el suelo pero todavía no ha despegado.

En el simulador cerrado con autopiloto TVC se usa un paso fijo y una integración RK4 explicita:

$$x_{k+1} = x_k + \frac{\Delta t}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right) \tag{84}$$

con:

$$\begin{aligned} k_1 &= f(t_k, x_k, u_k) \\ k_2 &= f(t_k + \Delta t/2, x_k + \Delta t k_1/2, u_k) \\ k_3 &= f(t_k + \Delta t/2, x_k + \Delta t k_2/2, u_k) \\ k_4 &= f(t_k + \Delta t, x_k + \Delta t k_3, u_k) \end{aligned} \tag{85}$$

La razón de usar paso fijo en este caso es representar de forma explicita la temporización discreta del computador de vuelo, filtros y servo.

La elección del integrador depende del flujo que se esté ejecutando:

- **Referencia MATLAB con `ode45`:** se usa en `simulate_rocket.m`, especialmente en vuelo vertical, perfiles de lazo abierto y ejecuciones desde la GUI MATLAB clásica. El paso adaptativo permite resolver la trayectoria 6DOF con tolerancias numéricas y eventos de apogeo y contacto con el suelo.

- **RK4 explícito de paso fijo en `simulate_rocket_tvc_closed_loop.m`:** se usa cuando se ensaya el autopiloto TVC en lazo cerrado dentro de MATLAB. El paso fijo representa la temporización discreta del computador de vuelo, los filtros, retardos y dinámica de servo.

- **Wrapper clásico de Simulink con `ode45`:** se usa en `rocket_6dof_model.slx`. Este modelo encapsula la planta continua de MATLAB en Simulink y permite compararla con el simulador base.

- **Simulink-first discreto:** se usa en `COHETE_CONTROL_SYSTEM.slx` con solver `FixedStepDiscrete` y el integrador `STATE_INTEGRATOR`. Planta, controlador, sensores, EKF y actuadores trabajan con un tiempo de muestreo común; la planta calcula $\dot{x}$ y el integrador actualiza el estado.

- **Lazo discreto de banco Raspberry:** no se emplea como solver físico principal de vuelo, sino como temporización de ejecución para sensores y PWM. En esta capa interesa validar frecuencias de control, latencia, saturaciones y límites de actuador.

4.10. Perfiles de control en lazo abierto

El simulador base acepta cualquier función con la firma conceptual:

```
u = control_law(t, x, config)
```

Los perfiles implementados son:

- vuelo vertical: empuje durante el quemado y TVC nulo;

- TVC constante: empuje durante el quemado y deflexiones fijas saturadas;

- TVC hold+ramp: deflexión constante durante una fracción del quemado y retorno lineal a cero antes de apagar motor.

El pseudocódigo del perfil *hold+ramp* es:

```
si t > t_burn:
    T = 0
    mu_pitch = 0
    mu_yaw = 0
si t <= hold_fraction*t_burn:
    mu_pitch = saturar(mu_pitch_cmd)
    mu_yaw   = saturar(mu_yaw_cmd)
si hold_fraction*t_burn < t <= ramp_fraction*t_burn:
    fase = (t - t_hold)/(t_ramp - t_hold)
    mu_pitch = (1 - fase)*mu_pitch_cmd
    mu_yaw   = (1 - fase)*mu_yaw_cmd
si t > ramp_fraction*t_burn:
    mu_pitch = 0
    mu_yaw = 0
```

Código 1. Pseudocódigo del perfil TVC hold+ramp

Estos perfiles no son controladores autónomos; son herramientas de validación para comprobar que la planta responde correctamente a empuje, deflexión y momento TVC.

4.11. Autopiloto TVC en lazo cerrado

El autopiloto TVC de ascenso implementa una capa de actitud alineada con una arquitectura de Raspberry Pi 3B+ y dos servos TVC. El aterrizaje retropropulsivo se aborda en la rama Simulink-first mediante el modo `Landing`, que decide la ignición de descenso a partir del estado, calcula una demanda vertical de frenado y mantiene la actitud; la planta, por su parte, mantiene enclavada la ignición de la segunda carga. Aun así, sigue siendo un prototipo nominal, validado principalmente en viento cero, y no el guiado completo final con perturbaciones, retardos reales y estimación/hardware validados en banco.

La arquitectura hardware asociada queda definida por una Raspberry Pi 3B+ ejecutando MATLAB/Simulink, sensores de la PCB conectados por I^2C y servos TVC mandados por PWM. CubePilot queda descartado como computador de vuelo objetivo en esta iteración. Las resistencias de $0\,\Omega$ de la PCB configuran físicamente alternativas de bus durante pruebas, pero el contrato nominal de vuelo se fija en I^2C.

4.11.1. Medida geometrica de inclinacion

En lugar de controlar directamente ángulos de Euler, el autopiloto calcula el vector vertical expresado en ejes cuerpo:

$$e_{up}^{b} = R_{n\to b} \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} \tag{86}$$

El error de inclinacion se obtiene como:

$$e_{tilt} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \times e_{up}^{b} \tag{87}$$

Las componentes de este vector se usan como errores equivalentes de pitch y yaw. Esta decisión está documentada en `tvc_controller_handoff_2026-04-13.md`: evita la singularidad asociada a describir una orientación vertical mediante Euler alrededor de $-90°$.

4.11.2. Filtrado de sensores

Las medidas de inclinacion y velocidad angular se filtran con un filtro paso bajo discreto de primer orden:

$$y_k = y_{k-1} + \alpha\,(x_k - y_{k-1}) \tag{88}$$

donde:

$$\alpha = 1 - e^{-2\pi f_c \Delta t} \tag{89}$$

El parámetro f_c es la frecuencia de corte. En la configuración por defecto se usan $20\,Hz$ para inclinación y $25\,Hz$ para giroscopos. El módulo también incluye ruido opcional de sensor, desactivado por defecto.

4.11.3. Ley PD con autoridad TVC

La ley de actitud traduce una frecuencia natural deseada y un amortiguamiento en ganancias proporcionales y derivativas. La autoridad angular en pitch se calcula como:

$$a_{TVC,p} = \frac{T\ell}{J_y} \tag{90}$$

y en yaw:

$$a_{TVC,y} = \frac{T\ell}{J_z} \tag{91}$$

Con $\omega_n = 2\pi f_{att}$ y amortiguamiento ζ, las ganancias quedan:

$$k_p = \frac{\omega_n^2}{a_{TVC}}, \qquad k_d = \frac{2\zeta\omega_n}{a_{TVC}} \tag{92}$$

La forma del mando es:

$$\mu_{p,cmd} = \mu_{p,trim} + k_{p,p}e_p + k_{d,p}q_m + k_i \int e_p\,dt \tag{93}$$

$$\mu_{y,cmd} = \mu_{y,trim} - \left(k_{p,y}e_y + k_{d,y}r_m + k_i \int e_y\,dt\right) \tag{94}$$

El signo de yaw se invierte para mantener coherencia con la definicion de momento de la tobera en la planta. La acción integral existe en la estructura, pero su ganancia por defecto es cero; el comportamiento nominal actual es PD filtrado con saturacion.

4.11.4. Dinámica de servo, retardo y PWM

El mando de servo incluye tres efectos:

- retardo puro mediante una cola discreta;
- dinámica de primer orden con ancho de banda equivalente;
- limite de velocidad angular del servo.

La dinámica de primer orden usa la misma forma de filtro:

$$\mu_{req,k} = \mu_{act,k-1} + \alpha_s(\mu_{cmd,k-d} - \mu_{act,k-1}) \tag{95}$$

donde d es el número de muestras de retardo. Después se aplica:

$$\Delta\mu_k = \text{sat}\left(\mu_{req,k} - \mu_{act,k-1}, -\dot{\mu}_{max}\Delta t, \dot{\mu}_{max}\Delta t\right) \tag{96}$$

y finalmente:

$$\mu_{act,k} = \text{sat}\left(\mu_{act,k-1} + \Delta\mu_k, -\mu_{max}, \mu_{max}\right) \tag{97}$$

Para registrar la señal compatible con PWM se usa:

$$PWM = PWM_0 + \Delta PWM\,\text{sat}\left(\frac{\mu_{act}[deg]}{\mu_{FS}[deg]}, -1, 1\right) \tag{98}$$

Tras seleccionar el servo Miuzei RC Servo 45KG 270deg (MIUZ26), el contrato eléctrico inicial se fija en PWM de 500 a $2500\,\mu s$, con neutro en $1500\,\mu s$ y recorrido eléctrico de $270°$. Para el registro PWM del simulador se toma como escala media $\mu_{FS} = 135°$, mientras que la deflexión física aplicada a la tobera sigue saturada por el límite de TVC de la planta.

La ficha comercial del servo (MIUZ26) declara una velocidad sin carga de $0{,}10\,s/60°$ a $8{,}4\,V$, equivalente a un limite nominal de $600°/s$. En la simulación Simulink-first se usa este valor como limite inicial de velocidad, junto con $8\,Hz$ de ancho de banda y $20\,ms$ de retardo. Estos dos últimos valores no proceden de un ensayo del servo, sino que son parámetros conservadores para representar una dinámica no ideal hasta disponer de medidas en banco.

4.12. GUI de simulación

La GUI de simulación modifica parámetros reales de `rocket_config` sin editar código. Agrupa parámetros de cohete, masa, aerodinámica, propulsión, entorno, simulación, condiciones iniciales y control. Al ejecutar, decide entre dos rutas:

- si el modo es lazo abierto, construye una `control_law` y llama a `simulate_rocket`;
- si el modo es el autopiloto TVC, llama al simulador de lazo cerrado del módulo de control local.

La GUI calcula y muestra trayectoria 3D, huella en planta, altitud, velocidad vertical, actitud, velocidades angulares, aceleración vertical, presión dinámica, Mach, ángulo de ataque, masa, C_D y mandos TVC. También exporta el último resultado al workspace como estructura con tiempos, estados, eventos, configuración, métricas y señales de control.

4.13. Modelo Simulink

La rama Simulink-first se ejecuta mediante `COHETE_CONTROL_SYSTEM.slx`, generado por `build_cohete_simulink_model.m`. El modelo no duplica la física en bloques elementales: conecta bloques `MATLAB Function` que llaman a las mismas funciones externas de configuración, planta, control, sensores y estimación. La cadena principal queda:

`OPEN_LOOP_U_CMD` → `FEEDBACK_CONTROLLER` → `PLANT_6DOF` → `STATE_INTEGRATOR`.

El estado integrado conserva el contrato $x = [pos_{NED}; v_b; q_{wxyz}; \omega_b]$ y la planta conserva por compatibilidad la interfaz legacy $u_{plant,legacy} = [T(t), \delta_{tvc,pitch}, \delta_{tvc,yaw}, \delta_{ab}]^T$. En esa interfaz, $T(t)$ es el empuje que recibe la planta despues de aplicar curvas de empuje y logica de ignicion, no una variable de control continuo. La planta devuelve la pendiente $\dot{x}$, el mando saturado u_{sat} y la bandera de aterrizaje. La integración se hace fuera del bloque de planta con un integrador discreto de Simulink, de forma que la física, la integración temporal y los logs quedan separados.

Por tanto, no se debe trasladar sin matiz la frase anterior de `ode45` a este modelo. En Simulink hay dos casos distintos: el wrapper clásico `rocket_6dof_model.slx` se genera con solver `ode45` para comparar una planta continua equivalente al simulador MATLAB; en cambio, `COHETE_CONTROL_SYSTEM.slx` usa solver `FixedStepDiscrete`, paso `MODEL_INI.PARAM.SAMPLING_TIME` y el bloque `STATE_INTEGRATOR` con integración discreta tipo Forward Euler porque integra control, sensores, EKF, actuadores y lógica de dos cargas sólidas en tiempo discreto. En banco Raspberry se conserva esa idea de muestreo fijo para validar temporización, sensores y PWM, no para sustituir al solver de referencia de la planta.

El controlador generado trabaja con tres modos seleccionables. En modo `Open-loop` reproduce la referencia de mando. En modo `EKF attitude hold` usa el estado estimado x_{est} para amortiguar el error de actitud por cuaternion y las velocidades angulares en ejes cuerpo. En modo `Landing`, implementado en `rocket_landing_controller.m`, mantiene la actitud y calcula una demanda vertical de frenado; la planta no interpreta esa demanda como throttle continuo, sino que la limita con el empuje disponible de dos cargas sólidas.

La segunda carga de descenso se activa por decision de estado dentro de la planta, en `cohete_mfcn_plant_step.m`. El criterio de `descent_ignition_decision.m` usa altitud AGL, velocidad vertical, masa actual, aceleracion de frenado disponible y una ventana finita de quemado. Cuando la condicion se cumple, el instante de ignicion queda fijado de forma irreversible y `solid_dual_thrust_profile.m` suma el perfil de la carga de descenso al de ascenso. Por tanto, el contrato final deseado no es un *throttle* continuo: combina la orden irreversible $ignite_{desc}$ con las deflexiones de TVC y aerofreno. El modelo generado mantiene todavia un campo $T(t)$ en `u_cmd` como interfaz legacy de planta para compatibilidad con los bloques actuales.

Las señales registradas quedan separadas en tres grupos:

- estado y eventos: `coheteStateLog` y `coheteLandingFlagLog`;
- actuación: `coheteUCmdLog` y `coheteUSatLog`;
- sensores: `coheteSensorLog`;
- estimación: `coheteStateEstLog` y `coheteEkfCovDiagLog`.

La GUI Simulink-first `cohete_simulink_gui.m` expone estos modos sin modificar `rocket_config.m`. Además de trayectoria, estados, control y estimación, incluye diagnósticos de perfil vertical, viento y medidas numéricas: altitud AGL, velocidad vertical, aceleración vertical, viento NED, módulo de viento, ráfaga incremental, actuadores y errores de estimación cuando existen. El selector de tema permite revisar las gráficas con fondo claro u oscuro, y la acción de ampliar vista abre la pestaña activa en una ventana independiente para inspección. Estas vistas son diagnósticas: no cambian la dinámica ni el contrato de actuación.

El Documento de Código Fuente recoge el pseudocódigo del flujo Simulink y de la GUI de diagnóstico.

4.13.1. Modelo de sensores y EKF con ruido simulado

Para preparar el paso desde planta ideal a control realimentado, el rediseño Simulink-first incorpora una capa de sensores y una primera versión de filtro de Kalman extendido. El objetivo no es sustituir todavía la validación en banco, sino evitar que el controlador futuro lea directamente el estado perfecto de la planta. La cadena implementada es:

$$\texttt{PLANT\_6DOF} \rightarrow \texttt{SENSOR\_MODEL} \rightarrow \texttt{EKF\_ESTIMATOR}.$$

El bloque `SENSOR_MODEL` recibe el estado ideal x y el mando saturado u_{sat}, y genera el vector de medida:

$$y_{sensor} = \begin{bmatrix} a_b \\ \omega_b \\ h_{baro} \\ m_b \end{bmatrix} \tag{99}$$

donde a_b es la fuerza específica medida por la IMU en ejes cuerpo en m/s^2, ω_b es la velocidad angular en rad/s, h_{baro} es la altitud barométrica en m y m_b es el campo magnético expresado en ejes cuerpo en T. El barómetro se obtiene como $h_{baro} = -p_D$, coherente con la convención NED del resto de la planta.

El ruido de sensores se simula como una perturbación gaussiana de media nula:

$$y_m = h_s(x, u_{sat}) + \nu, \qquad \nu_i = \sigma_i \eta_i, \qquad \eta_i \sim \mathcal{N}(0, 1) \tag{100}$$

En la implementación, la secuencia de ruido se genera de forma determinista a partir del tiempo de muestreo, el canal y una semilla fija. Esta decisión hace repetibles los ensayos automáticos, pero no caracteriza experimentalmente los sensores reales. Las desviaciones tipo usadas son parámetros de simulación iniciales: por alcance temporal del proyecto no da tiempo a implementar esa caracterización en el modelo real. La calibración de acelerómetro, giroscopio, barómetro y magnetómetro se probará en banco, pero no quedará integrada en el modelo real.

El filtro de Kalman extendido no reconstruye una trayectoria completa al final de la simulación, sino que actualiza el estado muestra a muestra. En cada instante conserva dos objetos: el estado estimado $\hat{x}$ y la matriz de covarianza P, que representa la incertidumbre asociada a cada componente del estado y sus correlaciones. La formulación sigue el esquema recursivo de predicción y corrección del filtro de Kalman (KALM60); en este proyecto se usa la variante extendida porque la dinámica 6DOF, la actitud mediante cuaterniones y la proyección del campo magnético son no lineales.

El filtro estima el mismo estado de trece componentes que la planta:

$$\hat{x} = \begin{bmatrix} \hat{p}_{NED} \\ \hat{v}_b \\ \hat{q}_{wxyz} \\ \hat{\omega}_b \end{bmatrix} \tag{101}$$

La primera mitad del ciclo es la predicción. El bloque toma el estado estimado anterior, mantiene constante el mando saturado u_{sat} durante el intervalo de muestreo y evalúa `state_derivatives.m` con ese estado. La integración usada dentro del EKF es Euler explícito:

$$\hat{x}_{k|k-1} = \hat{x}_{k-1|k-1} + T_s f\left(\hat{x}_{k-1|k-1}, u_{sat,k-1}\right) \tag{102}$$

Como $f(\cdot)$ no es lineal, la matriz de transición F_k no se escribe a mano. La implementación la aproxima con diferencias finitas centrales alrededor de $\hat{x}_{k-1|k-1}$, usando los pasos definidos en `CONFIG_EKF.m`: metros para posición, m/s para velocidad, incrementos pequeños en el cuaternión y rad/s para velocidad angular. Con ese jacobiano local se propaga la covarianza:

$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q \tag{103}$$

La matriz Q recoge incertidumbre del modelo de proceso: discretización, errores de dinámica, simplificaciones aerodinámicas y posibles discrepancias entre el mando saturado usado por el filtro y el comportamiento real. Después de cada predicción se normaliza el cuaternión estimado para conservar $\|\hat{q}\| = 1$; si no se hiciera, pequeños errores numéricos acabarían convirtiendo la actitud en una rotación no válida.

La segunda mitad del ciclo es la corrección con sensores. Aunque `y_sensor` contiene acelerómetro, giroscopio, barómetro y magnetómetro, la actualización inicial del EKF usa solo:

$$z_k = \begin{bmatrix} \omega_b \\ h_{baro} \\ m_b \end{bmatrix} \tag{104}$$

La medida esperada se calcula como $z = h_e(\hat{x})$: el giroscopio se compara con $\hat{\omega}_b$, el barómetro con $-\hat{p}_D$ y el magnetómetro con el campo magnético NED configurable proyectado a ejes cuerpo mediante el cuaternión estimado. Igual que en la predicción, el jacobiano de medida H_k se obtiene por diferencias finitas, ahora alrededor de $\hat{x}_{k|k-1}$.

La innovación mide la discrepancia entre lo que han medido los sensores y lo que el filtro esperaba medir desde su estado predicho:

$$r_k = z_k - h_e(\hat{x}_{k|k-1}) \tag{105}$$

Con esa innovación se calcula la covarianza de innovación S_k y la ganancia de Kalman K_k:

$$S_k = H_k P_{k|k-1} H_k^T + R, \qquad K_k = P_{k|k-1} H_k^T S_k^{-1} \tag{106}$$

La matriz R procede de las desviaciones tipo configuradas para giroscopio, barómetro y magnetómetro. Si R es grande, el filtro interpreta que la medida es poco fiable y la corrección pesa menos; si $P_{k|k-1}$ es grande, el filtro reconoce que su predicción es incierta y da más peso a la medida. Esta ponderación es el motivo por el que el EKF no sustituye sin más el estado por el sensor, sino que combina modelo y medida.

La corrección final actualiza el estado y vuelve a normalizar el cuaternión:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k r_k \tag{107}$$

Para la covarianza se usa la forma de Joseph:

$$P_{k|k} = (I - K_k H_k) P_{k|k-1} (I - K_k H_k)^T + K_k R K_k^T \tag{108}$$

Esta forma es más estable numéricamente que restar directamente $K_k H_k P$, porque conserva mejor la simetría y la positividad de la covarianza en presencia de redondeo (ZANE13). El código, además, simetriza P al final del paso y aplica un suelo pequeño a la diagonal para evitar varianzas negativas por error numérico.

La aceleración de la IMU queda disponible en `y_sensor` y en los registros de simulación, pero no se usa todavía como medida fuerte en la corrección. La razón es que la aceleración específica depende de empuje, aerodinámica y masa; emplearla sin cerrar cuidadosamente el modelo de entrada podría contar dos veces la misma información dinámica. Su integración queda

pendiente para una iteración posterior del estimador, junto con sesgos, retardos I2C y calibración real de sensores.

La implementación se reparte entre la configuración de sensores, el generador de ruido, el modelo de medida, la configuración del EKF y el paso discreto del estimador. En el modelo generado, el estimador se ejecuta mediante bloques `MATLAB Function` y funciones externas, no mediante una S-Function de nivel 2. En Simulink se registran medidas, estado estimado, diagonal de covarianza, innovación y estado interno del filtro para revisión posterior.

4.14. Programa de adquisicion de datos y banco Raspberry Pi

Como paso intermedio entre la simulacion y la integracion embarcada se ha preparado un programa de adquisicion de datos y banco para Raspberry Pi 3B+. Su objetivo es comprobar el funcionamiento de los sensores, la PCB y el registro de datos en el ordenador. Para ello, la capa ejecutable valida la temporizacion, el contrato de sensores, la telemetria hacia el ordenador y el mando PWM de actuadores en modo seguro antes de conectar el sistema integrado.

La cadena software se divide en dos lados. En la Raspberry se ejecuta el servidor TCP de banco `cohete_rpi_ethernet.py`. El arranque seguro se realiza con `cohete_rpi_bringup.py` o con `cohete_rpi_start_bench.py`.

En el ordenador se usa `cohete_pc_monitor.py` para consola/CSV o la GUI `cohete_pc_realtime_gui.py` para graficas en vivo. El protocolo entre ambos es `JSON Lines`: cada mensaje ocupa una linea TCP independiente codificada en UTF-8.

La interfaz de sensores mantiene el mismo contrato que Simulink-first:

$$y_{sensor} = \left[a_b^T,\ \omega_b^T,\ h_{baro},\ m_b^T\right]^T \tag{109}$$

El vector contiene aceleracion especifica en ejes cuerpo $[m/s^2]$, velocidad angular en ejes cuerpo $[rad/s]$, altitud barometrica estandar $[m]$ y campo magnetico en ejes cuerpo $[T]$. El mando aplicado por el banco conserva la interfaz legacy de planta, pero fuerza el empuje a cero:

$$u_{bench,legacy} = [0,\ \delta_{tvc,pitch},\ \delta_{tvc,yaw},\ \delta_{ab}]^T \tag{110}$$

Por tanto, el programa no gobierna ignicion ni propulsion real. Las deflexiones TVC y el angulo de aerofreno se conservan en radianes y se transforman en pulsos PWM para servos de banco.

El servidor TCP de la Raspberry escucha por defecto en el puerto `8765`. En un ensayo tipico se inicia primero el programa de banco en la Raspberry mediante una sesion SSH ya configurada por el operador. El operador entra en la carpeta donde esta instalado el paquete de banco y lanza el script de arranque seguro, que prepara el servidor sin activar PWM real ni movimiento de servos.

Desde el ordenador se abre despues el cliente de monitorizacion indicando el nombre o direccion de red de la Raspberry, el puerto de telemetria y, si se desea, un archivo CSV de salida. La GUI de PC usa el mismo puerto, puede reconectar automaticamente, guardar la ventana de muestras en CSV y mantiene `servo_enabled=false` por defecto. El servidor Raspberry envia al conectarse un mensaje `hello` con el modo de banco, el periodo nominal de muestreo y el protocolo. Despues publica mensajes `telemetry` con sensor, inclinacion, mando, PWM y estado de hardware. El PC puede enviar ordenes `set_control`, `set_reference`, `set_manual` y `stop`. Si los servos no estan habilitados, el mando aplicado se fuerza a neutro aunque se calcule una correccion por inclinacion.

El mapeo PWM usa el rango configurado para el servo seleccionado, con minimo $500\,\mu s$, centro $1500\,\mu s$ y maximo $2500\,\mu s$. Antes de escribir en los pines se aplican saturacion angular y limite de velocidad al TVC. La configuracion editable queda en `cohete_rpi_config.json`, donde se fijan el tiempo de muestreo, los limites de TVC, el *rate limit*, los pines GPIO, los signos de control y el rango angular del aerofreno. Los pines de banco por defecto son `GPIO12` para pitch TVC, `GPIO13` para yaw TVC y `GPIO18` para aerofreno.

Por seguridad, el modo con hardware real exige activar explicitamente `-hardware` y `-allow-hardware`. El movimiento inicial de servos exige ademas `-servos-enabled`. Sin esos argumentos el programa registra telemetria y calcula PWM, pero no mueve actuadores. En ningun modo existe salida de ignicion.

El modo de sensores `auto` intenta usar el bus I^{2}C de la Raspberry y cae a un modo seguro si falta algun dispositivo. Tambien existe `i2c-imu-only`: conserva acelerometro y giroscopio reales cuando solo esta disponible la IMU, rellena barometro y magnetometro con valores neutros y mantiene la longitud de `y_sensor`. Esta decision permite validar roll/pitch y flujo de telemetria sin presentar medidas de barometro o magnetometro como si fueran datos reales cuando no estan presentes.

4.15. Rama de control LQR

La rama de control más formal basada en espacio de estados se organiza alrededor de los archivos `CONFIG_ROCKET.m`, `CONFIG_MODEL.m`, `LINEARIZE_MODEL.m`, `CONFIG_CONTROL.m`, `DESIGN_SFC.m` y `RUN_SIMULATION.m`.

El modelo lineal de pitch se obtiene alrededor de vuelo vertical en hovering. Para el estado $x_p = [\theta, q]^T$ y entrada δ_{tvc}:

$$\dot{\theta} = q \tag{111}$$

$$\dot{q} = \frac{mgd_{cg}}{J_y}\theta - \frac{T_0\ell}{J_y}\delta_{tvc} \tag{112}$$

La primera parte representa una inestabilidad pendular debida a la separación axial entre CG y punto de giro del gimbal. La segunda parte es la autoridad TVC linealizada para pequeños ángulos. La linealización queda:

$$A_p = \begin{bmatrix} 0 & 1 \\ \frac{mgd_{cg}}{J_y} & 0 \end{bmatrix}, \qquad B_p = \begin{bmatrix} 0 \\ -\frac{T_0\ell}{J_y} \end{bmatrix} \tag{113}$$

El modelo de altitud usado en esa rama es:

$$\dot{z} = v_z, \qquad \dot{v}_z = -\frac{c_{ab}}{m}\delta_{ab} \tag{114}$$

donde:

$$c_{ab} = \frac{1}{2}\rho S_{ab} C_{D,ab} V_{op} \tag{115}$$

Este modelo de aerofrenos es una aproximación de diseño para control de altitud; no forma parte todavía del contrato final de aterrizaje retropropulsivo.

El diseño LQR se hace en discreto. La planta continua se discretiza como:

$$A_d = e^{AT_s} \tag{116}$$

$$B_d = \int_0^{T_s} e^{A\tau} B \, d\tau \tag{117}$$

El código aproxima la integral de B_d mediante una serie finita. Para control con acción integral se aumenta el estado con la integral del error, en línea con la formulación LQI del mismo trabajo de dos Santos y Oliveira (SANT24). En forma conceptual:

$$x_{a,k} = \begin{bmatrix} x_k \\ \xi_k \end{bmatrix}, \qquad \xi_{k+1} = \xi_k + T_s(r_k - y_k) \tag{118}$$

La ganancia se obtiene con `dlqr`, que minimiza:

$$J = \sum_{k=0}^{\infty} \left(x_{a,k}^T Q x_{a,k} + u_k^T R u_k \right) \tag{119}$$

y aplica:

$$u_k = -K x_k - K_i \xi_k \tag{120}$$

En la simulación `RUN_SIMULATION.m`, el controlador de pitch usa θ y q extraídos del cuaternion y satura la deflexión TVC. La altitud usa aerofrenos solo durante descenso. Esta rama es útil para estudiar LQR/LQI, pero queda pendiente reconciliarla con la interfaz del simulador base, donde el mando base es $[T, \mu_p, \mu_y]^T$, frente a la interfaz usada por `RUN_SIMULATION.m`, en la que aparece $[T, \delta_{tvc}, \delta_{tvc,yaw}, \delta_{ab}]^T$.

4.16. Validación disponible

El proyecto ya dispone de ejemplos y pruebas automatizables. Los ejemplos principales son:

- `vertical_flight.m`: vuelo vertical sin TVC, con cálculo de apogeo y comparación energética simplificada;
- `tvc_flight.m`: maniobra de pitch con TVC en lazo abierto;
- `tvc_autopilot_closed_loop.m`: caso con viento lateral e inclinación inicial, usando autopiloto discreto.

La estimación energética de apogeo usada como referencia sencilla en el ejemplo vertical procede de conservar energía en ascenso balistico sin drag:

$$h_{ideal} = \frac{V_{burnout}^2}{2g} \tag{121}$$

La diferencia entre este valor y el apogeo simulado se interpreta como pérdida por arrastre y no idealidades del perfil.

En pruebas, existen unidades para atmósfera y aerodinámica, integración para simulación completa y una prueba de autopiloto TVC cerrado. Esta última comprueba que la simulación completa, que los límites de actuador no se superan, que la inclinación al burnout se reduce respecto a la perturbación inicial y que el aterrizaje se detecta y clampa correctamente. Para la rama Simulink-first se han añadido pruebas específicas de sensores y estimación. `test_sensor_measurement_model.m` verifica el contrato de medida y_{sensor}, `test_sensor_noise_model.m` comprueba que el ruido simulado es reproducible,

`test_ekf_step.m` valida un paso aislado del EKF y `test_simulink_ekf_io.m` comprueba que el modelo generado registra el estado estimado y la diagonal de covarianza. También, `test_simulink_visual_io.m` verifica que los bloques `SENSOR_MODEL` y `EKF_ESTIMATOR` están presentes y conectados. Para el programa de banco Raspberry, `test_raspberry_pi_bench.py` comprueba el mapeo PWM, las saturaciones, el rate limit y la ejecución en modo seco sin hardware.

4.17. Pseudocódigo del flujo principal

El flujo principal del simulador puede resumirse como:

```
cargar configuracion del cohete
construir estado inicial x0
definir ley de control u(t,x,config)

para cada evaluacion del integrador:
    normalizar cuaternion
    calcular masa, CG e inercias
    calcular atmosfera y viento
    evaluar mando [T, mu_pitch, mu_yaw]
    calcular fuerza de empuje con TVC
    calcular velocidad relativa al aire
    calcular fuerzas y momentos aerodinamicos
    proyectar gravedad a ejes cuerpo
    sumar fuerzas y momentos
    aplicar ecuaciones Newton-Euler
    devolver dx/dt

detectar apogeo y aterrizaje
calcular metricas y graficas
```

Código 2. Pseudocódigo general del simulador MATLAB

Este pseudocódigo refleja la separación central del programa: la planta física no sabe si el mando procede de una tabla, de una GUI, de un autopiloto discreto o de un LQR. Esa separacion deja ampliar el proyecto sin reescribir la dinámica base.

Capítulo 5

Resultados

Este capítulo recoge los resultados experimentales en bancos de pruebas de distintos subsistemas del banco de pruebas.

5.1. Simulación de vuelo

La Figura 17 muestra el resultado de vuelo obtenido con el programa de control. La simulación se ejecutó con el modelo `COHETE_CONTROL_SYSTEM.slx`, paso de integración de $0{,}01\ s$, modo de control *Landing* y la configuración local del cohete de competición del IRT 2026 del repositorio. El evento de contacto se produjo en $t = 31{,}91\ s$.

En los tres paneles, el eje horizontal representa el tiempo de simulación en segundos. El panel superior usa el eje vertical para la altitud AGL $[m]$ y la curva azul representa la posición vertical del cohete. El panel central muestra la velocidad vertical V_z positiva hacia arriba $[m/s]$ en naranja. El panel inferior muestra la aceleración vertical A_z $[m/s^2]$ en verde. Las líneas verticales punteadas señalan los eventos de burnout, apogeo, ignición de la segunda carga y contacto.

Las métricas principales de esta ejecución fueron: altitud máxima $559{,}05\ m$, velocidad máxima $114{,}40\ m/s$, velocidad vertical final $-0{,}85\ m/s$, aceleración vertical máxima $82{,}06\ m/s^2$ y desplazamiento horizontal final $110{,}21\ m$. La segunda carga aparece marcada cerca de $30\ s$ y reduce la velocidad vertical antes del contacto. Estos valores son resultados del modelo actual y no deben confundirse con los resultados oficiales del vuelo convencional con paracaídas incluidos en el informe técnico del ICAI Rocket Team.

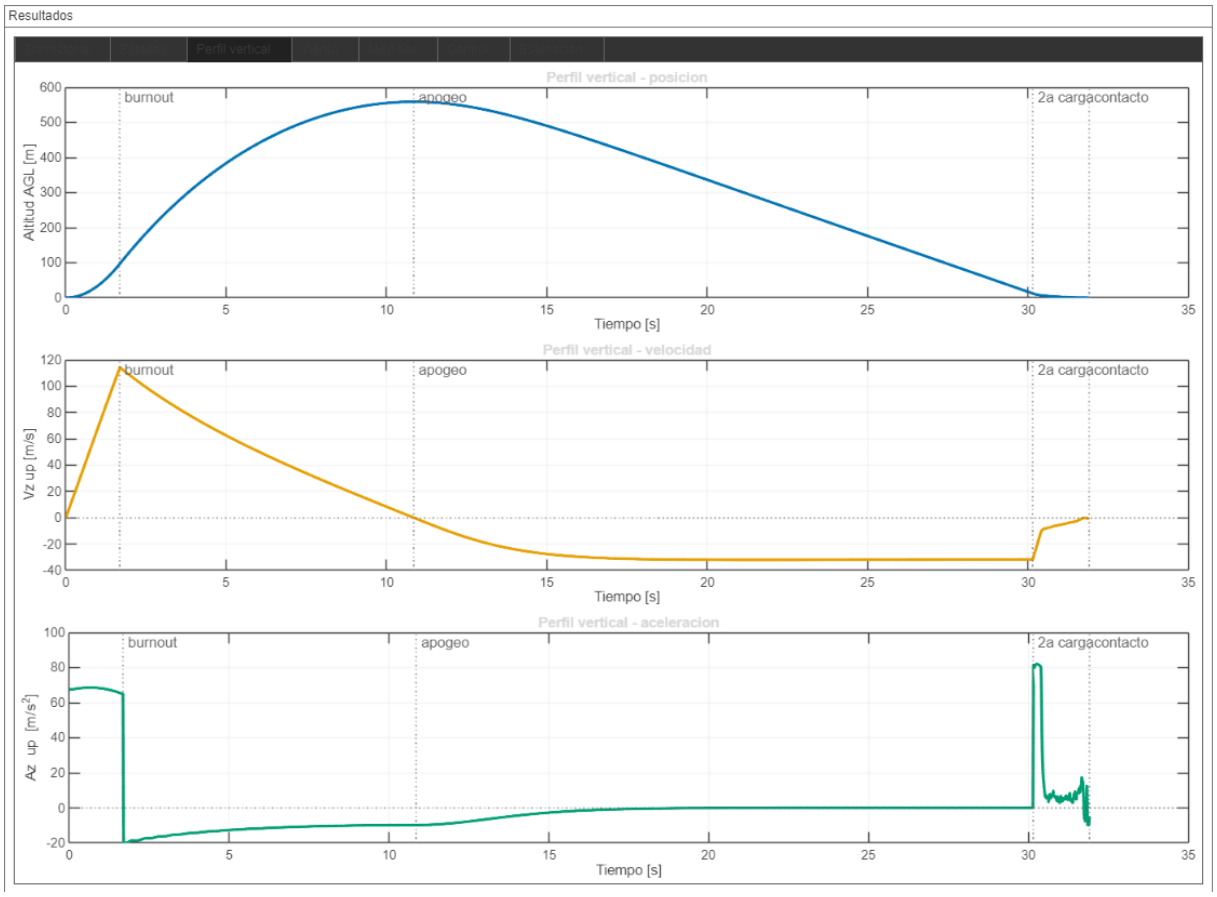


Figura 17. Resultado de simulación de vuelo obtenido desde el programa de control. El eje X es el tiempo $[s]$; azul: altitud AGL $[m]$; naranja: velocidad vertical $[m/s]$; verde: aceleración vertical $[m/s^2]$; líneas punteadas: eventos de vuelo.

5.2. Control

La Figura 18 resume las salidas de control registradas por la misma GUI. En todos los subgráficos, el eje horizontal es el tiempo $[s]$. El eje vertical cambia según el canal: empuje T $[N]$, deflexión TVC de pitch $[deg]$, deflexión TVC de yaw $[deg]$ y ángulo de aerofreno o bandera de aterrizaje. La línea discontinua negra representa el mando calculado `u_cmd` cuando aparece, mientras que la línea continua de color representa el mando aplicado tras saturaciones `u_sat`. Los colores de `u_sat` son azul para empuje y aerofreno, verde para pitch TVC y naranja para yaw TVC; la línea punteada magenta representa la bandera `landing`.

En el canal de empuje se observa la carga de ascenso, el tramo balístico y la contribución de la carga de descenso antes del contacto. Los canales TVC se mantienen dentro del límite configurado y el aerofreno alcanza su saturación de $90°$ durante la fase descendente.

La comparación `x`/`x_est` de esta ejecución dio un error RMS de altitud de $0{,}047\,m$, error máximo absoluto de altitud de $0{,}162\,m$, error máximo de posición de $5{,}86\,m$ y error máximo de velocidad de $3{,}84\,m/s$. Estos resultados validan que el flujo de simulación, sensores simulados, EKF inicial y controlador se ejecutan de forma coherente dentro de la arquitectura actual.

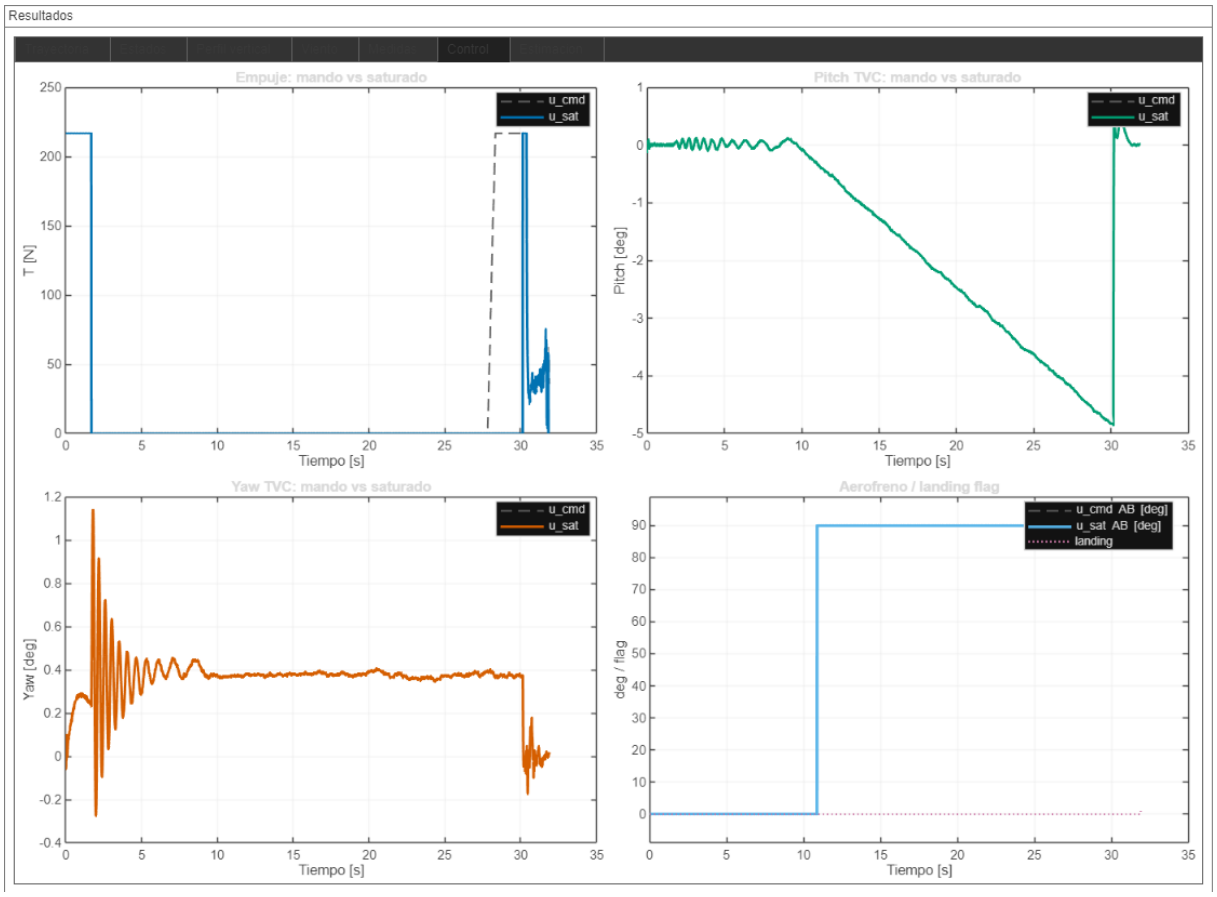


Figura 18. Resultado de control obtenido desde la GUI Simulink-first. Eje X: tiempo $[s]$. Línea discontinua negra: mando calculado `u_cmd`. Líneas continuas: mando aplicado `u_sat`; azul en empuje/aerofreno, verde en pitch TVC y naranja en yaw TVC.

5.3. Célula de carga

El subsistema de célula de carga queda documentado en el capítulo de hardware mediante el puente de galgas, la etapa de amplificación y el filtro de salida. En esta fase se dispone ya de una evidencia de laboratorio del montaje de adquisición, mostrada en la Figura 19. La fotografía documenta el prototipo de acondicionamiento sobre placa de pruebas y la observación de la señal con osciloscopio.

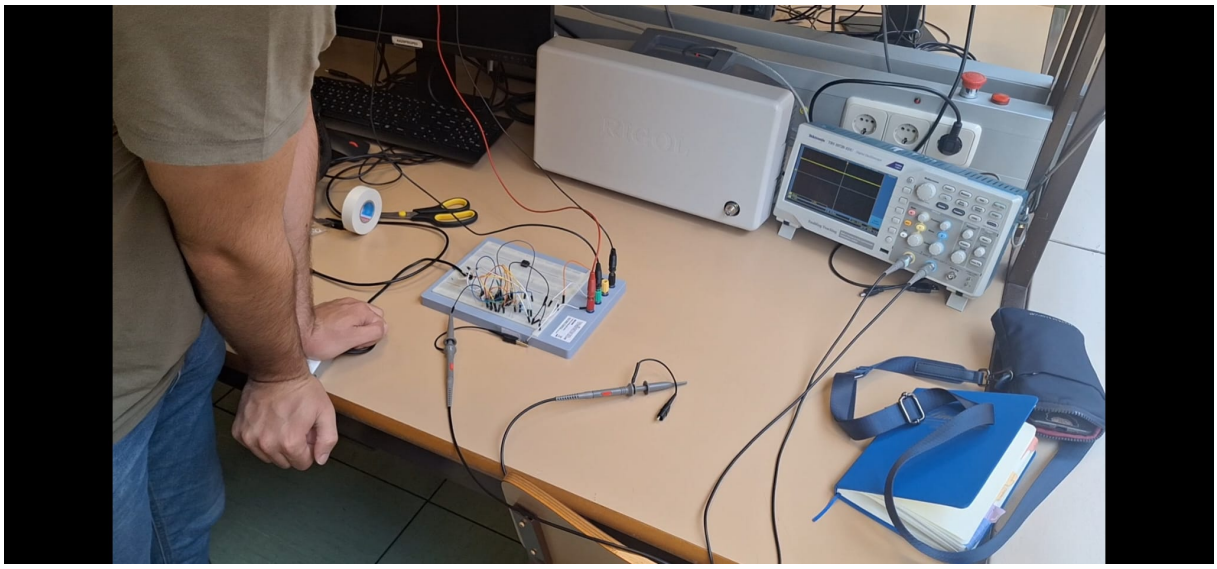


Figura 19. Montaje de laboratorio del sistema de adquisición de la célula de carga, con prototipo en placa de pruebas y osciloscopio para observar la señal acondicionada.

5.4. Ordenador de vuelo y sensores

Para esta prueba la Raspberry Pi 3B+ se conectó al ordenador mediante red Ethernet y el servidor de banco se relanzó desde el PC mediante una sesión SSH ya configurada. Esta forma de trabajo se eligió por comodidad operativa: permite trabajar solo desde el ordenador, sin tener teclado y ratón propios para la Raspberry y sin pasar archivos manualmente de un equipo a otro en cada iteración. El flujo no depende de una dirección concreta en la memoria: una vez que ambos equipos están en la misma red, el PC abre el cliente contra el servidor TCP de la Raspberry y guarda la telemetría recibida en CSV para su análisis. El objetivo del ensayo fue comprobar la lectura de sensores, la integración con la PCB y el registro de datos en el ordenador.

La integración física de banco se muestra en la Figura 20. En ella se ve la PCB de aviónica conectada al conjunto Raspberry y la GUI de telemetría ejecutándose en el PC. Esta fotografía se usa como evidencia del montaje de banco; la validación eléctrica completa de la placa y de los actuadores queda limitada a las comprobaciones descritas en esta memoria.

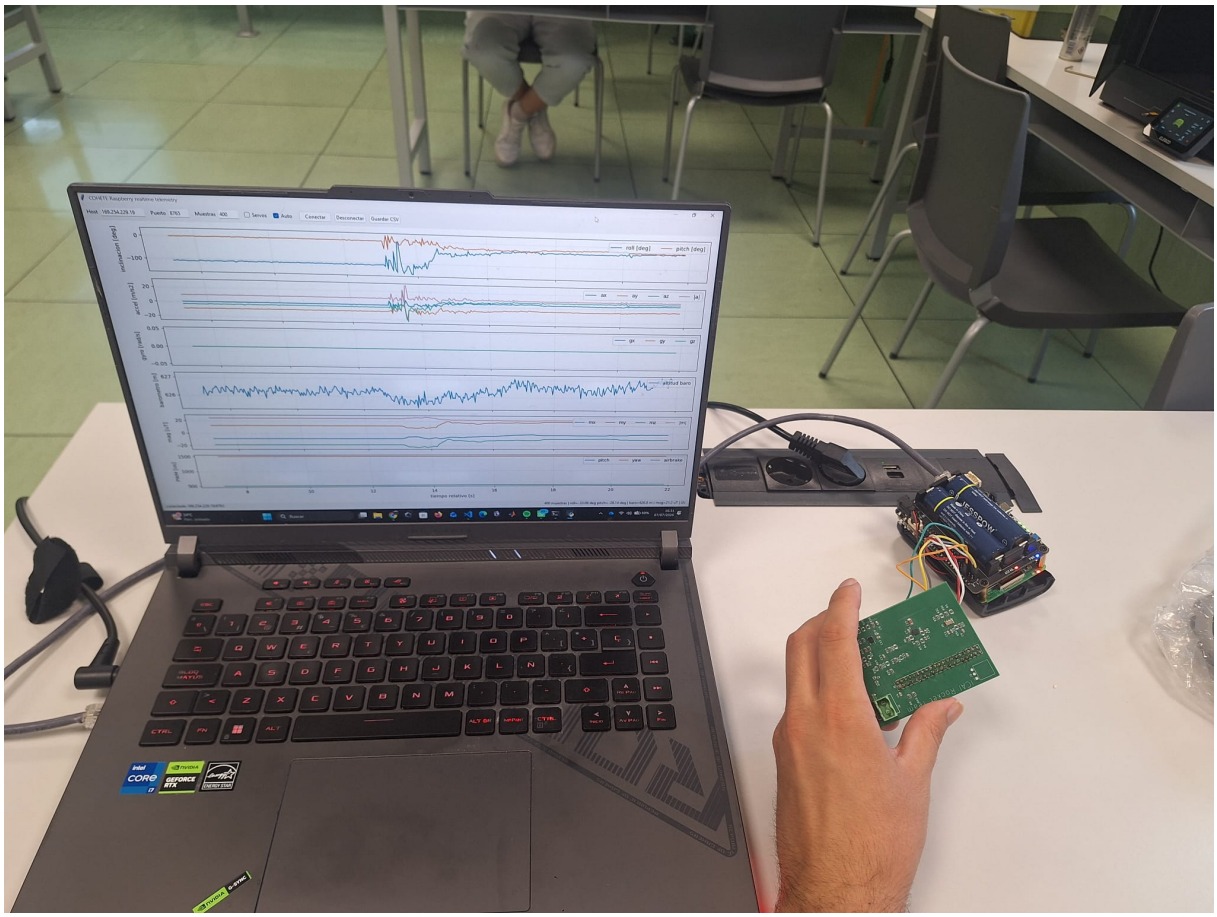


Figura 20. Banco de integración de la PCB de aviónica con Raspberry Pi 3B+ y GUI de telemetría en el ordenador.

La prueba de adquisición usó el servidor TCP `cohete_rpi_ethernet.py` en el puerto `8765` y el cliente PC `cohete_pc_monitor.py`. La comunicación emplea una línea JSON por mensaje. La Raspberry envía `telemetry` con `y_sensor`, inclinación estimada, mando seguro y PWM; el PC puede enviar `set_control`, `set_reference`, `set_manual` y `stop`. Durante esta captura los servos permanecieron deshabilitados, no se activó PWM real y la propulsión no forma parte del banco.

La Figura 21 resume 821 muestras recibidas durante $30{,}00\,s$ mientras se movía manualmente la placa. En todos los paneles, el eje horizontal es el tiempo desde el inicio de la captura $[s]$. En el primer panel, el eje vertical es la inclinación $[deg]$: azul para roll y naranja para pitch. En el segundo panel, el eje vertical es la aceleración $[m/s^2]$: azul para a_x, verde para a_y, naranja para a_z y gris para la norma $|a|$. En el tercer panel, el eje izquierdo muestra la altitud barométrica estándar $[m]$ en azul claro y el eje derecho la norma del campo magnético $|B|$ $[\mu T]$ en rosa. En el cuarto panel, el eje vertical es el pulso PWM $[\mu s]$: azul para pitch TVC, verde para yaw TVC y naranja para aerofreno.

La inclinación estimada recorrió aproximadamente de $-79{,}1°$ a $135{,}3°$ en roll y de $-73{,}7°$ a $79{,}7°$ en pitch. La norma media de aceleración fue $9{,}92\,m/s^2$, la altitud barométrica estándar se mantuvo entre $777{,}30\,m$ y $778{,}58\,m$, y la norma media del magnetómetro fue $45{,}73\,\mu T$. El PWM de TVC permaneció en $1500\,\mu s$ y el aerofreno en $500\,\mu s$, coherente con el banco seguro sin actuación. El resultado relevante aquí es la llegada de telemetría real con movimiento apreciable y la coherencia de unidades.

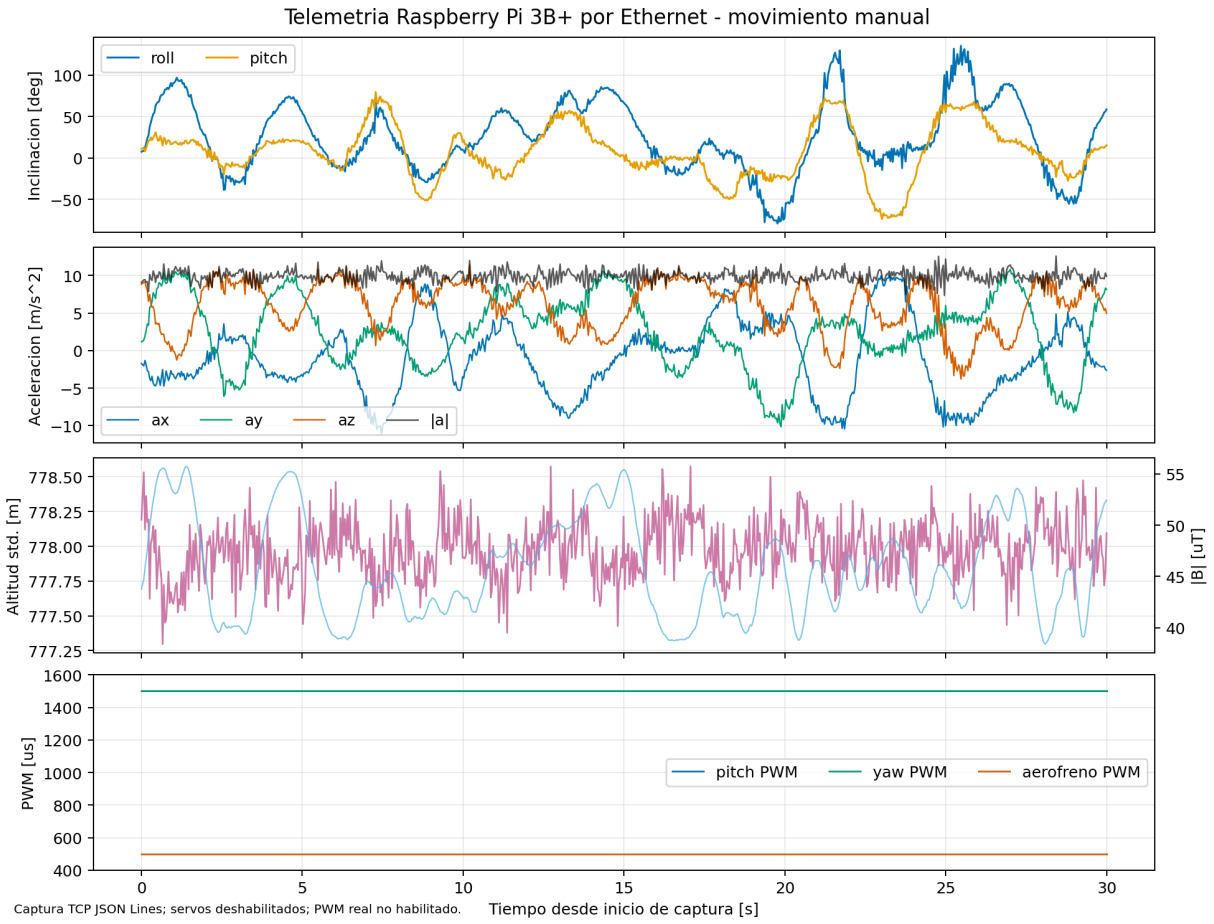


Figura 21. Telemetría real recibida desde la Raspberry Pi 3B+ por Ethernet. Eje X: tiempo $[s]$. Panel 1: roll azul y pitch naranja $[deg]$. Panel 2: a_x azul, a_y verde, a_z naranja y $|a|$ gris $[m/s^2]$. Panel 3: altitud azul claro $[m]$ y $|B|$ rosa $[\mu T]$. Panel 4: PWM pitch azul, yaw verde y aerofreno naranja $[\mu s]$.

Capítulo 6

Conclusiones

El trabajo ha permitido construir una arquitectura completa para estudiar el aterrizaje retropropulsivo de un cohete mediante TVC, integrando modelo físico, simulación, control, selección de hardware, PCB, adquisición de sensores y documentación técnica. El resultado no es todavía un sistema de vuelo cerrado, pero sí una base verificable sobre la que continuar el desarrollo con criterios de trazabilidad y seguridad.

6.1. Conclusiones técnicas

En simulación se ha consolidado una planta 6DOF con unidades SI, posición NED, velocidades en ejes cuerpo y actitud por cuaternión $[q_0, q_1, q_2, q_3]$. La GUI Simulink-first permite modificar parámetros, ejecutar el modelo y revisar resultados de trayectoria, perfil vertical, viento, control, sensores y estimación. La ejecución documentada en resultados demuestra que la cadena Simulink puede representar ascenso, tramo balístico, ignición de descenso y contacto con el suelo, con métricas extraíbles de forma reproducible.

En control se ha separado el contrato actual de planta del contrato objetivo de vuelo con cargas sólidas. La interfaz heredada conserva $u_{cmd} = [T; \delta_{tvc,pitch}; \delta_{tvc,yaw}; \delta_{ab}]^T$; en cambio, el empuje de descenso debe proceder de una decisión irreversible de ignición. Esta distinción es importante: el controlador no debe tratar una carga sólida como un actuador de empuje continuo. La rama actual permite validar TVC, aerofreno, EKF inicial y saturaciones, pero queda pendiente completar el adaptador final hacia `ignite_desc` y medir retardos reales.

En hardware se ha cerrado una arquitectura basada en Raspberry Pi 3B+, sensores I^2C, PCB propia y servos PWM para TVC/aerofreno. La prueba de banco realizada confirma que el ordenador de vuelo puede enviar telemetría al PC por Ethernet con formato JSON Lines y que el PC puede guardar una captura CSV con unidades coherentes. Por seguridad, el banco mantiene deshabilitados la ignición, la propulsión y el movimiento de servos salvo activación explícita.

6.2. Limitaciones

Las principales limitaciones siguen estando en la validación física. Falta calibrar los sensores frente a referencias conocidas, medir latencia I^2C, jitter PWM, consumo de servos, holgura mecánica, zona muerta y velocidad bajo carga. También falta validar la célula de carga con datos experimentales de banco, así como cerrar el montaje mecánico final del TVC y del aerofreno.

La simulación incluye una representación útil del aterrizaje retropropulsivo, pero no sustituye un ensayo integrado. Los resultados del capítulo anterior son coherentes con el modelo actual; no garantizan comportamiento real hasta que el hardware, los actuadores, las masas reales, los retardos y la propulsión se midan y se incorporen a la planta.

6.3. Trabajo futuro

No se ha conseguido cerrar del todo la construcción del cohete. Faltaría por diseñar la tobera TVC concreta para el modelo que se está usando, definiendo su geometría, su mecanismo de orientación, sus límites mecánicos, el montaje con los servos y la integración con la estructura real.

También faltaría diseñar y construir los aerofrenos del cohete, incluyendo su mecanismo de despliegue, el actuador, los topes mecánicos, la integración con la aviónica y la caracterización de su efecto aerodinámico. Una vez cerrados estos elementos, el paso decisivo será realizar lanzamientos experimentales y comparar los datos de vuelo con los resultados de simulación. Solo con esos ensayos podrá comprobarse si el cohete se comporta en vuelo de forma coherente con el modelo desarrollado.

Bibliografía

[NISK13] **Niskanen, S.**, *OpenRocket technical documentation*. Technical documentation, OpenRocket, version 13.05, 2013.
OpenRocket technical documentation v13.05

[GREG70] **Gregorek, G. M.**, *Aerodynamic Drag of Model Rockets*. TR-11 Model Rocket Technical Report, Estes Industries, Inc., Penrose, Colorado, 1970.

[NOAA76] **NOAA, NASA, U.S. Air Force**, *U.S. Standard Atmosphere, 1976*. NOAA-S/T 76-1562, NASA-TM-X-74335, U.S. Government Printing Office, Washington, 1976. Ultima consulta: 17/06/2026.
https://ntrs.nasa.gov/citations/19770009539

[STUL88] **Stull, R. B.**, *An Introduction to Boundary Layer Meteorology*. Kluwer Academic Publishers, Dordrecht, 1988.

[PETE78] **Peterson, E. W. y Hennessey, J. P.**, *On the Use of Power Laws for Estimates of Wind Power Potential*. Journal of Applied Meteorology, vol. 17, pp. 390–394, 1978.

[KALM60] **Kalman, R. E.**, *A New Approach to Linear Filtering and Prediction Problems*. Journal of Basic Engineering, vol. 82, n. 1, pp. 35–45, 1960. DOI: 10.1115/1.3662552.

[IRT26] **ICAI Rocket Team**, *Technical Report: L1 Rocketry Competition - SURTAM 2026. Mision II, Flight Alcolea de Cinca*. Universidad Pontificia Comillas, reporte técnico final interno, 15 de junio de 2026.

[THRU06] **ThrustCurve.org**, *Cesaroni 364I212-14A motor data and RASP thrust curve*. Base de datos de motores de cohetería; fichero RASP de fabricante aportado por Casey Hatch, 30 de mayo de 2006. Última consulta: 09/06/2026.
https://www.thrustcurve.org/motors/Cesaroni/364I212-14A/
https://www.thrustcurve.org/simfiles/5f4294d20002e9000000013b/

[OPEN26] **Open-Meteo**, *Weather Forecast API. Snapshot horario para Alcolea de Cinca, 11 de junio de 2026*. Consulta realizada el 11/06/2026; coordenadas $41{,}7342°$, $0{,}0622°$ y elevación $213\,m$.
https://open-meteo.com/en/docs

[STMI26] **STMicroelectronics**, *LSM6DSV32X iNEMO inertial module datasheet*. Hoja de características oficial del fabricante. Última consulta: 02/07/2026.
https://www.st.com/resource/en/datasheet/lsm6dsv32x.pdf

[TECO26] **TE Connectivity**, *MS5607-02BA03 barometric pressure sensor datasheet*. Hoja de características oficial del fabricante. Última consulta: 02/07/2026.
TE Connectivity MS5607-02BA03 datasheet

[MEMS26] **MEMSIC**, *MMC5983MA 3-axis magnetic sensor datasheet*. Hoja de características oficial del fabricante. Última consulta: 02/07/2026.
MEMSIC MMC5983MA datasheet

[ANAL26a] **Analog Devices**, *AD620 low cost, low power instrumentation amplifier datasheet*. Hoja de características oficial del fabricante. Última consulta: 02/07/2026.
https://www.analog.com/media/en/technical-documentation/data-sheets/AD620.pdf

[TEXA26] **Texas Instruments**, *TL08xx JFET-input operational amplifiers datasheet*. Hoja de características oficial del fabricante. Última consulta: 02/07/2026.
https://www.ti.com/lit/ds/symlink/tl082.pdf

[TEXA23] **Texas Instruments**, *Active Low-Pass Filter Design*. Application Report SLOA049D, revisión D, febrero de 2023; incluye la formulación de diseño Sallen–Key. Última consulta: 14/07/2026.
Texas Instruments, Active Low-Pass Filter Design

[ANAL26b] **Analog Devices**, *AD8221 precision instrumentation amplifier datasheet*. Hoja de características oficial del fabricante. Última consulta: 02/07/2026.
https://www.analog.com/media/en/technical-documentation/data-sheets/AD8221.pdf

[ANAL26c] **Analog Devices**, *ADA4077-1/ADA4077-2/ADA4077-4 operational amplifier datasheet*. Hoja de características oficial del fabricante. Última consulta: 02/07/2026.
https://www.analog.com/media/en/technical-documentation/data-sheets/ADA4077-1_4077-2_4077-4.pdf

[MIUZ26] **Miuzei**, *RC Servo 45KG 270deg digital servo, ASIN B0CZDXW2FH*. Pagina comercial Amazon.es indicada para el prototipo; no se ha localizado una hoja de caracteristicas independiente. Datos pendientes de validación en banco. Última consulta documental: 22/05/2026.
https://www.amazon.es/dp/B0CZDXW2FH

[RASP26a] **Raspberry Pi Ltd.**, *Raspberry Pi 3 Model B+ product page and specification*. Documentación oficial del producto; incluye procesador, memoria, conectividad, cabecera GPIO, alimentación y soporte PoE. Última consulta: 16/06/2026.
https://www.raspberrypi.com/products/raspberry-pi-3-model-b-plus/

[RASP26b] **Raspberry Pi Ltd.**, *Raspberry Pi computer hardware documentation: GPIO and the 40-pin header*. Documentación oficial de hardware; incluye disponibilidad de PWM por software y PWM por hardware en GPIO12, GPIO13, GPIO18 y GPIO19. Última consulta: 16/06/2026.
https://www.raspberrypi.com/documentation/computers/raspberry-pi.html

[RASP26c] **Raspberry Pi Ltd.**, *Raspberry Pi OS documentation*. Documentación oficial del sistema operativo; describe Raspberry Pi OS como sistema operativo oficial para Raspberry Pi basado en Debian. Última consulta: 16/06/2026.
https://www.raspberrypi.com/documentation/computers/os.html

[MATH26] **MathWorks**, *Build and Deploy Your First Simulink Model to Raspberry Pi*. Documentación oficial de Raspberry Pi Blockset sobre generación, despliegue y ejecución de modelos Simulink en hardware Raspberry Pi. Última consulta: 14/07/2026.
MathWorks: despliegue de Simulink en Raspberry Pi

[GEEK26] **Geekworm**, *X728 Raspberry Pi UPS Power Management Board*. Pagina de producto de la placa de gestion de alimentacion X728; documenta montaje con celdas 18650 y conexion a Raspberry Pi. Última consulta: 29/06/2026.
https://geekworm.com/products/x728

[JESS26] **JESSPOW**, *18650 Button Top Rechargeable Battery - 4 Pack*. Ficha de producto del fabricante; identifica celda recargable Li-ion 18650 *button top*, $3{,}7\,V$, $3300\,mAh$ y tamaño $18 \times 67\,mm$. Última consulta: 29/06/2026.
https://jesspow.com/products/18650-button-top-rechargeable-battery-4-pack

[SANT24] **dos Santos, P. y Oliveira, P.**, *Integrated architecture for navigation and attitude control of low-cost suborbital launch vehicles*. Acta Astronautica, vol. 222, pp. 52–68, 2024. DOI: 10.1016/j.actaastro.2024.05.039. Última consulta: 02/07/2026.
https://doi.org/10.1016/j.actaastro.2024.05.039

[BPSS26] **BPS.Space**, *Thrust Vector Control 3D Files*. Pagina de producto y documentacion visual del mecanismo TVC con servos y piezas impresas. Última consulta: 29/06/2026.
https://bps.space/products/thrust-vector-control

[SENE87] **Séneca, L. A.**, *Hercules furens*, v. 437. Texto latino y comentario de la edición de Dickinson College Commentaries, basada en el texto crítico de J. G. Fitch. Última consulta: 14/07/2026.
Hercules furens, vv. 414–438

[KENN62] **Kennedy, J. F.**, *Address at Rice University on the Nation's Space Effort*. Discurso pronunciado en Rice University, Houston, 12 de septiembre de 1962. John F. Kennedy Presidential Library and Museum. Última consulta: 14/07/2026.
Transcripción oficial del discurso

[ZANE13] **Zanetti, R. y DeMars, K. J.**, *Joseph Formulation of Unscented and Quadrature Filters with Application to Consider States*. Journal of Guidance, Control, and Dynamics, vol. 36, n. 6, pp. 1860–1864, 2013. DOI: 10.2514/1.59935.
https://doi.org/10.2514/1.59935

[UNIT15] **United Nations**, *Transforming our world: the 2030 Agenda for Sustainable Development*. Resolución A/RES/70/1 de la Asamblea General de las Naciones Unidas, 2015.
https://sdgs.un.org/2030agenda

Apéndice A

Alineación con los Objetivos de Desarrollo Sostenible

A.1. Planteamiento original recogido en el Anexo B

La idea original del proyecto partió de la reutilización de lanzaderas como vía para reducir residuos, aprovechar mejor los recursos y abaratar el acceso al espacio. En el Anexo B presentado al inicio del Trabajo Fin de Grado, la alineación con los Objetivos de Desarrollo Sostenible se formuló en los siguientes términos.

A.1.1. Objetivo 12: Producción y consumo responsables

Este TFG se alinea con el punto 12.5, ya que la reutilización de lanzaderas espaciales permitiría reducir mucho la generación de desechos (restos de la primera etapa), que normalmente acaban en el mar. Por otra parte, también se alinearía con el 12.1, ya que sería una utilización eficiente de los recursos naturales y, por último, con el 12.6, ya que fomentaría que las empresas utilizaran este método en vez de alternativas no reutilizables, debido a sus costes.

A.1.2. Objetivo 9: Industria, innovación e infraestructura

Este TFG se alinea con el punto 9.1, ya que el abaratamiento del coste de los lanzamientos ha permitido la construcción de constelaciones de satélites que han dado acceso a personas de todo el mundo a internet.

Asimismo, se alinea con el 9.4, ya que permite a las empresas del sector ser más sostenibles al reutilizar toda una etapa.

Y, por último, con el 9.5, ya que el acceso a internet en países en desarrollo permite a sus investigadores estar más parejos con el resto del mundo.

Este planteamiento constituye la motivación original. La evaluación siguiente contrasta esa idea con el alcance que finalmente se ha desarrollado y validado en el TFG. De acuerdo con la Agenda 2030, se distingue entre la contribución técnica demostrada, la capacidad que el proyecto prepara para el futuro y los impactos que todavía no se han medido (UNIT15).

A.2. Cumplimiento específico en el trabajo realizado

A.2.1. Contribución al ODS 12

La idea central del Anexo B era reducir el carácter desechable de las primeras etapas mediante su recuperación. El trabajo realizado no ha demostrado todavía la reutilización física del cohete, pero sí ha construido la base de ingeniería necesaria para estudiarla de forma reproducible.

En relación con la meta 12.5, el simulador no lineal 6DOF y el modelo Simulink permiten analizar trayectorias, perturbaciones, leyes de control y estrategias de encendido antes de consumir motores o fabricar nuevas iteraciones del prototipo. El banco con Raspberry Pi mantiene la ignición deshabilitada y permite comprobar adquisición, estimación y mandos PWM sin efectuar un lanzamiento. Estas herramientas no prueban una reducción de residuos ya conseguida, pero sí reducen el riesgo de descartar hardware por errores detectables previamente en simulación o banco.

El uso eficiente de recursos naturales mencionado como punto 12.1 en el texto original se concreta en la Agenda 2030 mediante la meta 12.2. La contribución específica del TFG es la reutilización de modelos, configuraciones y procedimientos de ensayo. La arquitectura modular de sensores, PCB, actuadores y software permite sustituir o recalibrar subsistemas sin rehacer necesariamente el conjunto completo.

Respecto a la meta 12.6, la trazabilidad entre requisitos, modelos, pruebas y documentación introduce una práctica de desarrollo reproducible. Esta base permite que futuras iteraciones del ICAI Rocket Team comparen alternativas por prestaciones, número de ensayos, consumo de componentes y posibilidad de reutilización, aunque el TFG no ha evaluado todavía esos indicadores mediante un análisis de ciclo de vida.

A.2.2. Contribución al ODS 9

La contribución más directa se produce en las metas 9.4 y 9.5. Para la meta 9.4, se ha desarrollado una cadena digital MATLAB/Simulink que integra dinámica, aerodinámica, masa variable, control, estimación y actuadores no ideales. Esta infraestructura de modelado permite reutilizar configuraciones y ensayos virtuales, y prepara el estudio de un vehículo recuperable. La mejora de sostenibilidad industrial sigue siendo una capacidad potencial, porque no se ha construido ni reutilizado una etapa completa.

Para la meta 9.5, el resultado concreto es una plataforma de investigación y aprendizaje formada por el simulador 6DOF, el control de actitud mediante TVC, la lógica de ignición de la carga de descenso, el estimador EKF, la GUI de simulación y el banco Raspberry Pi con sensores I2C, servos PWM y PCB de aviónica. El conjunto permite continuar con identificación de actuadores, pruebas HIL, validación del estimador y ensayos de guiado sin comenzar de nuevo la arquitectura.

La relación con la meta 9.1 conserva el sentido de contexto planteado en el Anexo B: lanzamientos más asequibles pueden facilitar infraestructuras satelitales y ampliar el acceso a comunicaciones. Sin embargo, este TFG no ha estudiado constelaciones, cobertura de internet ni costes de lanzamiento. Por ello, esta relación se presenta como motivación indirecta y no como un resultado atribuible al trabajo realizado.

A.3. Alcance de la contribución

El cumplimiento específico demostrado consiste en haber desarrollado y conectado herramientas de simulación, control, estimación y adquisición de banco orientadas a un futuro cohete reutilizable con aterrizaje propulsivo. También se ha establecido una arquitectura modular que permite repetir ensayos y evolucionar subsistemas con trazabilidad.

Quedan fuera de la validación actual la reutilización real del cohete, la caracterización completa de la carga de descenso y del mecanismo TVC, el balance de ciclo de vida, la cuantificación de residuos evitados y la reducción del coste por lanzamiento. Estos aspectos deberán evaluarse cuando existan prototipos completos, ensayos físicos y datos de fabricación y operación.

Apéndice B

Programa MATLAB y Simulink

B.1. Criterio de documentacion del codigo

Esta parte no reproduce los archivos completos del repositorio. El codigo real evoluciona durante el proyecto y seria poco util copiar cientos de lineas sin contexto. En su lugar se documenta el funcionamiento mediante pseudocodigo, contratos de entrada y salida, y fragmentos cortos de MATLAB cuando una linea concreta fija una convencion fisica.

El criterio seguido es el mismo que en el resto de la memoria: unidades SI, posicion en NED, velocidades lineales en ejes cuerpo, actitud con cuaternion $[q_0, q_1, q_2, q_3]$ y deflexiones TVC en radianes. Cuando el codigo usa grados es solo para entrada de usuario, salida grafica o conversion a PWM.

B.2. Mapa funcional del software

Bloque	**Entradas y salidas**	**Funcionamiento**
`rocket_config.m`	Parametros fisicos y numericos	Centraliza geometria, masa, atmosfera, propulsion, aerodinamica, entorno y condiciones iniciales.
`simulate_rocket.m`	Configuracion y ley de control; devuelve tiempo, estado y eventos	Prepara el estado inicial, ejecuta `ode45`, detecta apogeo y aterrizaje, y calcula metricas de vuelo.
`state_derivatives`	t, x, configuracion y mando; devuelve $\dot{x}$	Evalua fuerzas, momentos, propiedades masicas y ecuaciones Newton–Euler 6DOF.
`wind_model.m` y aerodinamica	Altitud, configuracion, tiempo y velocidad relativa	Calculan atmosfera, viento NED, velocidad respecto al aire, drag, fuerza normal y momentos aerodinamicos.
Componentes software	Buses, configuracion, control, sensores y EKF	Prepara la arquitectura Simulink-first y los contratos entre planta, controlador, sensores y estimador.
GUI Simulink-first	Parametros editados por usuario; devuelve logs y graficas	Configura el modelo generado, ejecuta Simulink, extrae senales y muestra diagnosticos.

Tabla 10. Mapa funcional del codigo documentado

B.3. Flujo principal de simulacion MATLAB

El flujo MATLAB empieza cargando una configuracion unica. Esa estructura evita que la geometria, la masa o el entorno queden repartidos entre scripts. La simulacion no sabe de antemano si el mando procede de una ley abierta, de una GUI o de un controlador; solo exige una funcion con la interfaz $u(t, x, config)$.

```
config = rocket_config();
x0 = estado_inicial(config);
control_law = @(t, x, cfg) perfil_de_mando(t, x, cfg);
[t, x] = ode45(@(t, x) state_derivatives(t, x, config, control_law), ...);
```

Código 3. Esqueleto funcional de una simulacion MATLAB

El pseudocodigo completo de una evaluacion de planta es:

```
normalizar cuaternion q = [q_0 q_1 q_2 q_3]
calcular altitud h = -p_D
si hay contacto con suelo y velocidad vertical descendente:
    devolver dxdt = 0

calcular masa, centro de gravedad e inercias
calcular atmosfera, gravedad y viento NED
transformar viento a ejes cuerpo
calcular velocidad relativa al aire
leer empuje disponible T(t) y deflexiones [delta_tvc_pitch, delta_tvc_yaw, delta_ab]
calcular fuerza y momento de empuje TVC
calcular fuerzas y momentos aerodinamicos
proyectar gravedad a ejes cuerpo
sumar fuerzas y momentos
aplicar dinamica traslacional, rotacional y cinematica de cuaternion
```

Código 4. Pseudocodigo de `state_derivatives`

La linea de contacto con suelo no representa una pata de aterrizaje. Es un cierre numerico para detener la integracion cuando el vehiculo alcanza la cota de suelo y sigue descendiendo. Por eso el lanzamiento desde el suelo no queda bloqueado: el codigo usa una puerta temporal y comprueba el signo de la velocidad vertical.

B.4. Planta 6DOF y convenciones de estado

El estado integrado tiene trece componentes:

$$x = [p_N,\ p_E,\ p_D,\ u,\ v,\ w,\ q_0,\ q_1,\ q_2,\ q_3,\ p,\ q,\ r]^T \tag{122}$$

La posicion se expresa en NED y la altitud usada para graficas se calcula como $h = -p_D$. Las velocidades u, v, w y p, q, r estan en ejes cuerpo. Esta eleccion simplifica la suma de fuerzas y momentos porque empuje TVC, drag, fuerza normal y amortiguamiento se calculan en el sistema unido al cohete.

Las ecuaciones implementadas se reducen en el codigo a tres operaciones principales:

```
d_pos_ned = R_body_to_ned * vel_body;
d_vel_body = (F_total_body - cross(omega_body, mass * vel_body)) / mass;
d_omega_body = J \ (M_total_body - cross(omega_body, J * omega_body));
```

Código 5. Operacion central de la planta 6DOF

En el archivo real las inercias se manejan como componentes principales J_x, J_y y J_z, por lo que la division se escribe componente a componente. La forma matricial anterior resume la misma operacion y deja visible el termino giroscopico $\omega \times J\omega$.

B.5. Mando, actuadores y limites

Conviene distinguir dos niveles. La planta Simulink-first conserva una interfaz legacy de entrada, usada para evaluar la fisica, pero no es el contrato final del controlador:

$$u_{plant,legacy} = [T(t),\ \delta_{tvc,pitch},\ \delta_{tvc,yaw},\ \delta_{ab}]^T \tag{123}$$

En esa interfaz, $T(t)$ es el empuje ya disponible para la planta despues de evaluar el perfil de propulsion y el estado irreversible de ignicion; no representa un throttle continuo calculado por el controlador. Con cargas solidas, el contrato objetivo de control es:

$$u_{control} = [ignite_{desc},\ \delta_{tvc,pitch},\ \delta_{tvc,yaw},\ \delta_{ab}]^T \tag{124}$$

Las deflexiones se expresan en radianes. El subindice `ab` representa el angulo de apertura del aerofreno, no un mando binario. Esta distincion evita mezclar el actuador aerodinamico con el TVC.

En la planta, el TVC se convierte directamente en fuerza y momento:

```
F_thrust_body = [T*cos(dp)*cos(dy);
                -T*cos(dp)*sin(dy);
                -T*sin(dp)];
M_thrust_body = [0;
                -T*sin(dp)*ell;
                 T*cos(dp)*sin(dy)*ell];
```

Código 6. Fuerza y momento de empuje con TVC

Donde `dp` y `dy` son las deflexiones de pitch y yaw. La variable `ell` representa ℓ, el brazo de palanca entre el gimbal y el centro de masas. En pequenos angulos, $M_y \simeq -T\ell\delta_{tvc,pitch}$ y $M_z \simeq T\ell\delta_{tvc,yaw}$. Por esta razon, una deflexion acumulada durante un tramo sin empuje puede producir un transitorio angular al encender una carga solida.

Por tanto, cuando aparece `T` en el codigo documentado, debe leerse como campo legacy de planta o resultado del modelo de propulsion, no como accion libre de control.

B.6. Modelo Simulink-first

El subapartado de Simulink documenta `COHETE_CONTROL_SYSTEM.slx`. Este modelo resuelve la integracion temporal de los estados de la planta, toma los datos de

`CONFIG_ROCKET.m` y organiza los bloques principales como funciones MATLAB externas. Esta estructura permite modificarlos desde codigo y controlar de forma mas directa su comportamiento. La cadena principal es:

$$\texttt{OPEN\_LOOP\_U\_CMD} \rightarrow \texttt{FEEDBACK\_CONTROLLER} \rightarrow \texttt{PLANT\_6DOF} \rightarrow \texttt{STATE\_INTEGRATOR}$$

La rama de sensores y estimacion queda conectada como:

$$\texttt{PLANT\_6DOF} \rightarrow \texttt{SENSOR\_MODEL} \rightarrow \texttt{EKF\_ESTIMATOR} \rightarrow \texttt{FEEDBACK\_CONTROLLER}$$

El flujo discreto puede resumirse asi:

```
para cada muestra k:
    u_cmd = feedback_controller(x_est, u_ref, u_prev, CONTROL)
    si CONTROL_MODE == Landing:
        calcular demanda vertical y actitud hold con x_est
    [x_dot, u_sat, landing_flag] = plant_6dof(t, x, u_cmd)
    si la planta detecta condicion de segunda carga:
        fijar t_ign_desc de forma irreversible y sumar T_desc(t - t_ign_desc)
    x(k+1) = x(k) + Ts * x_dot
    y_sensor = sensor_model(x, u_sat, SENSORS)
    [x_est, P] = ekf_estimator(y_sensor, u_sat, x_est, P, EKF)
    registrar x, u_cmd, u_sat, y_sensor, x_est y landing_flag
```

Código 7. Pseudocodigo del modelo Simulink-first generado

La integracion de estado del Simulink-first no reutiliza `ode45`: se hace fuera del bloque de planta, mediante un integrador discreto de Simulink configurado con el paso de `MODEL_INI.PARAM.SAMPLING_TIME`. Esto permite inspeccionar la pendiente $\dot{x}$ y mantener separados el calculo fisico, la integracion temporal y los registros de senales. Las salidas principales agrupan estado, mandos, actuadores saturados, sensores, estimacion, covarianza del EKF y bandera de aterrizaje. El modo `Landing` conserva el contrato legacy u_{cmd}, pero la disponibilidad real de empuje la decide la planta mediante el perfil de dos cargas solidas y el estado irreversible de la ignicion de descenso.

El uso de cada método de integración queda separado por caso:

- Referencia continua: `ode45` se usa en `simulate_rocket.m` y en el wrapper clásico `rocket_6dof_model.slx` para simulación continua y detección de eventos.
- RK4 de paso fijo: autopiloto TVC cerrado en MATLAB. Se usa para representar el ciclo discreto de control, filtros y servos.
- Simulink-first discreto: el modelo generado usa solver de paso fijo e integrador `STATE_INTEGRATOR` cuando conecta planta, controlador, sensores, EKF, actuadores y lógica de propulsión de dos cargas.
- Lazo fijo Raspberry: programa de banco. Se usa para temporización real de sensores y PWM; no sustituye la simulación física de referencia.

B.6.1. Bloques MATLAB Function del modelo

Los cuatro bloques funcionales del modelo se apoyan en funciones MATLAB externas. Esta decision permite validarlo con `checkcode` y pruebas unitarias. Su responsabilidad se reparte asi:

- `FEEDBACK_CONTROLLER`: selecciona modo de control, aplica actitud hold o aterrizaje, y limita TVC por saturacion y velocidad maxima.
- `PLANT_6DOF`: calcula $\dot{x}$, empuje disponible, saturaciones finales y bandera de aterrizaje.
- `SENSOR_MODEL`: construye $y_{sensor} = [a_b^T, \omega_b^T, h_{baro}, m_b^T]^T$ con ruido configurable.
- `EKF_ESTIMATOR`: predice el estado con la misma dinamica 6DOF y corrige con giroscopio, barometro y magnetometro.

La aceleracion de la IMU queda registrada, pero todavia no se usa como correccion fuerte del EKF. El motivo es que depende de empuje, masa y aerodinamica; usarla sin cerrar ese contrato podria introducir informacion duplicada en el filtro.

B.6.2. GUI Simulink-first

La GUI GUI Simulink-first es una capa de configuracion y diagnostico. No modifica `rocket_config.m`; copia los cambios a estructuras de workspace como `MODEL_INI`, `PROPULSION_INI` y `CONTROL_INI`, ejecuta Simulink y extrae logs.

```
leer tablas de la GUI
validar parametros y unidades
actualizar MODEL_INI, PROPULSION_INI y CONTROL_INI
ejecutar sim('COHETE_CONTROL_SYSTEM')
extraer logs de estado, control, sensores y EKF
calcular metricas de altitud, velocidad, aceleracion y viento
dibujar trayectorias, perfil vertical, viento, control y estimacion
```

Código 8. Funcionamiento de la GUI Simulink-first

Las graficas diagnosticas no cambian la simulacion. El perfil vertical muestra altitud AGL, velocidad vertical y aceleracion vertical, con marcas de burnout, apogeo, segunda carga y contacto cuando aparecen. La vista de viento recalcula `wind_model(altitude, config, t)` para representar componentes NED, magnitud y rafaga incremental. La vista `Medidas` resume tiempos, velocidades, aceleraciones, viento, actuadores y errores de estimacion con unidades. La vista `Estimacion` compara el estado de planta con `x_est`; ya no se expone la comparacion antigua contra `simulate_rocket.m` en esta GUI. El selector de tema y la accion de ampliar vista solo afectan a la inspeccion visual.

B.7. Programa de adquisicion Raspberry

El programa de banco Raspberry se divide en un servidor embarcado y un cliente de ordenador. La Raspberry lee sensores reales o simulados, empaqueta `y_sensor`, estima inclinacion desde acelerometro, calcula un mando seguro de TVC y envia telemetria por TCP. El ordenador abre la

conexion, solicita el estado seguro de control y guarda las muestras en CSV o las muestra en la GUI de tiempo real.

```
Raspberry:
    preparar eth0 y servicio SSH si hace falta
    abrir servidor TCP en 0.0.0.0:8765
    enviar hello con protocolo json-lines-v1
    mientras el banco este activo:
        leer sensores I2C, IMU-only o dry-run
        construir y_sensor = [accel; gyro; baro; mag]
        estimar roll y pitch desde acelerometro
        recibir ordenes set_control, set_reference o set_manual
        si servo_enabled es false:
            forzar TVC neutro y aerofreno seguro
        convertir mando angular a PWM [us]
        emitir mensaje telemetry como una linea JSON

Ordenador:
    conectar a la IP de la Raspberry y puerto 8765
    enviar set_control con servo_enabled=false por defecto
    recibir telemetry, actualizar graficas y guardar CSV
    cerrar socket o enviar stop segun el cliente usado
```

Código 9. Pseudocodigo del programa de adquisicion Raspberry–PC

El contrato mantiene unidades SI salvo el PWM, que se expresa en microsegundos. En la version actual el empuje se fuerza a cero en la capa hardware. Por tanto, el banco no gobierna ignicion ni propulsion real; solo comprueba temporizacion, formato de datos, lectura de sensores, limites de actuador y mapeo PWM. PWM real requiere `-hardware -allow-hardware`, y el movimiento de servos exige tambien `-servos-enabled`.

B.8. Validacion asociada

La validacion del codigo se reparte entre pruebas MATLAB, pruebas Simulink y pruebas de banco. Los tests principales comprueban configuracion, componentes fisicos locales, burnout, ignicion de segunda carga, sensores, ruido reproducible, EKF, logs Simulink, GUI y runtime Raspberry. Esta documentacion no sustituye esos tests; solo explica que hace cada bloque para que el lector pueda seguir el flujo antes de revisar los archivos fuente.

Apéndice C

Reporte técnico SURTAM 2026 del ICAI Rocket Team

El documento reproducido a continuación se incorpora como apéndice documental de este Trabajo Fin de Grado porque se utilizará el mismo modelo de cohete para este TFG, modificando la parte de aviónica, *recovery* y *payload* para adaptarlo a este proyecto. La autoría de este reporte no corresponde de forma individual al autor de este TFG. El informe fue elaborado por el conjunto del ICAI Rocket Team para la competición de SURTAM 2026 (IRT26). Las cuarenta páginas siguientes son una reproducción íntegra de ese documento externo, no contenido original de esta memoria.

Technical Report: L1 Rocketry Competition - SURTAM 2026



MISION II

FLIGHT ALCOLEA DE CINCA

18 – 19 JULY 2026

15 JUNE

ICAI Rocket Team

Universidad Pontificia Comillas

TABLE OF CONTENTS

I. Introduction and presentation of the team and candidate ---- 3
Introduction and purpose ---- 3
The ICAI Rocket Team ---- 3
Candidate presentation ---- 3
Flight mission objectives ---- 3
Team resources and project management ---- 4
II. Concept of operations (CONOPS) ---- 6
Operational framework and environment ---- 6
System states and mission phases ---- 8
Personnel roles and interfaces ---- 10
Actionable launch-day master Checklist (SURTSR-L1-GEN-002) ---- 11
III. System Architecture ---- 15
Structural System ---- 15
Propulsion System ---- 17
Avionics System ---- 17
Hardware architecture ---- 18
Software architecture ---- 18
Recovery System ---- 19
Payload System (CanSat Integration) ---- 19
IV. Compliance with Requirements ---- 20
V. Simulation and Flight Profiles ---- 24
VI. Risk Assessment ---- 26
VII. Engineering Drawings ---- 28
Annex I ---- 40

I. Introduction and presentation of the team and candidate

This section provides an overview of the ICAI Rocket Team, outlines the specific objectives of our L1 certification mission, and presents the credentials of the candidate leading the flight operations. It serves as a formal introduction to the project's purpose within the context of the SURTAM 2026 Competition.

Introduction and purpose

The document constitutes the technical report for the L1 certification flight of the ICAI Rocket Team, prepared for the SURTAM 2026 competition in Alcolea de Cinca (Huesca). The purpose of this report is to present the design, engineering analysis, simulation data, and operational procedures of the rocket system. This submission serves as evidence of our commitment to safety, technical rigor and the successful completion of the Tripoli rocketry association L1 certification requirements.

The ICAI Rocket Team

The ICAI Rocket Team is a student-led organization based at the Universidad Pontificia Comillas (ICAI). Our mission is to promote aerospace engineering excellence through the design, fabrication, and testing of high-power rocketry systems. Composed of multidisciplinary engineering students, the team focuses on applying theoretical knowledge from the classroom to real-world aerospace challenges. This organization allows students to work in a realistic engineering environment, where design decisions must be justified through analysis, testing, safety considerations, and operational feasibility.

Candidate presentation

The candidate for the L1 Certification flight is Héctor Sánchez Valencia, a member of the ICAI Rocket Team and an industrial engineering student at Universidad Pontificia Comillas ICAI. He had always been fascinated by space, to the point that he spent a considerable amount of time deciding whether to pursue Industrial Engineering or Aerospace Engineering. He ultimately chose Industrial Engineering, but he never gave up on his other dream. He tried to join the ICAI Rocket Team, which had originally been founded in 2019, but discovered that it had disappeared in 2020 due to the pandemic and the lack of new members. As a result, he decided to re-establish the team together with Mario Mora in 2024.

Flight mission objectives

As our first official launch, our primary goal is to achieve and execute a safe and successful flight that demonstrates the reliability of our design. Specifically, as a team, we aim to complete a stable flight, that is aligned with the simulations and successfully deploy the recovery system to recover the rocket without structural damage, ensuring it remains in good condition so it can be launched again.

Moreover, our most significant achievement will be working cohesively to apply our engineering knowledge to this challenge.

Team resources and project management

The development of the Dédalo vehicle requires a strategic management structure capable of coordinating a technically complex project within a limited time frame. The challenge is not only to design and manufacture a rocket, but also to organize the people, resources, knowledge, and external relationships needed to make the project viable. For this reason, project management in the ICAI Rocket Team is treated as an essential part of the engineering process.

The project is developed in an environment where uncertainty is constant. Technical problems, supplier limitations, manufacturing constraints, testing needs, and competition deadlines require the team to adapt continuously. Therefore, the management approach is based on clear responsibilities, fast communication, practical decision-making, and coordination between departments. The objective is to keep the project moving while maintaining technical rigor, safety, and team cohesion.

The first pillar of the project management strategy is team building and internal organization. From the beginning, the team understands that the success of the rocket depends directly on the people building it. The ICAI Rocket Team is not only a group of students working on separate tasks, but a multidisciplinary engineering team where each member contributes to a shared objective. This is especially important in a project that requires commitment outside the classroom, continuous learning, and a high level of trust between departments.

To organize the work, the team is divided into five departments: Aerodynamics, Avionics, Propulsion, Recovery, and Strategy.

- **Aerodynamics** is responsible for the structural and aerodynamic design of the rocket, including the geometry, materials, stability, and mechanical integration of the airframe.
- **Avionics** is responsible for the electronic systems, sensors, flight computer, data acquisition, and flight monitoring architecture.
- **Propulsion** is responsible for the motor selection, propulsion hardware, launch preparation, and safety aspects related to the energetic components.
- **Recovery** is responsible for the parachute system, deployment mechanism, descent control, and post-flight recovery of the vehicle.
- **Strategy** is responsible for external representation, sponsor relations, administrative coordination, communication, and the general direction that supports the continuity of the team.

This departmental structure allows the team to divide the workload while keeping the rocket as one integrated system. Each subsystem depends on the rest, so coordination between departments is a constant requirement. Changes in mass, layout, materials, electronics, or recovery configuration affect the global performance of the vehicle. For this reason, the team works through continuous communication between technical areas, shared reviews, and direct collaboration between members.

The management model also focuses on continuity. The project is not limited to one launch or one generation of students. The team seeks to create a working structure that can be transferred, improved, and reused in future projects. Documentation, internal mentoring, and collaboration between experienced and new members are essential to preserve knowledge and avoid starting from zero each year. This reflects one of the central values of the ICAI Rocket Team: building a team that grows through people, not only through technical results.

The second pillar of the project management strategy is brand building. From the beginning, the team understands that sponsors and collaborators need to see more than an idea. They need to see a serious project,

a committed team, and visible progress. This is especially relevant because the team is being rebuilt after a previous interruption, so credibility must come from execution, consistency, and real engineering work.

For this reason, the team does not present the rocket as an isolated technical object, but as the first step of a broader initiative. The objective is to build a recognizable and reliable university aerospace project with long-term potential. The website (**https://icai-rocket-team.base44.app/**), LinkedIn page (**https://es.linkedin.com/company/icairocketteam**), and Instagram account (**@icairocketteam**) support this objective by communicating the team's progress, showing the work behind the project, and creating a connection with companies, students, alumni, and followers.

Brand building is therefore part of the project infrastructure. It helps attract future members, strengthen external relationships, give visibility to the technical work, and create confidence among potential sponsors. The team aims to show that the ICAI Rocket Team is not only preparing for one competition, but building the foundations for future launches, future collaborations, and future generations of students.

Overall, the project management approach combines technical coordination, internal organization, team culture, and external positioning. The main objective is to make the Dédalo project viable while consolidating a team that can learn, adapting, and growing together.

This same approach is reflected in the funding structure of the project. Since the team is still in a consolidation phase, financing is not treated only as an administrative task, but as part of the collective responsibility required to move the project forward. One of the first financial decisions is the implementation of a team membership contribution, which allows the team to cover initial needs while sponsor relations and institutional support continue to develop. The response from the members is immediate, reinforcing the idea that the rocket is built not only through technical work, but also through shared commitment.

Table 1 presents the income and expenses considered for this project phase. Income is grouped into four categories: team membership fees, donations, sponsors, and ICAI support. Expenses are grouped according to the five departments of the team: Aerodynamics, Avionics, Propulsion, Recovery, and Strategy. This structure provides a clear view of how resources are obtained and how they are assigned to the different areas of the project.

Income category	Income (€)	Expense (€)	Expense department
Team membership fees	475,00 €	336,52 €	Aerodynamics
Donations	100,00 €	662,00 €	Avionics[1]
Sponsors	550,00 €	161,58 €	Propulsion
ICAI contribution	113,29 €	58,19 €	Recovery
		20,00 €	Strategy
TOTAL INCOME	**1.238,29 €**	**1.188,29 €**	TOTAL EXPENSES
BALANCE	**0 €**		

Table 1: SURTAM-J26 Poroject Income and Expenses

[1] Avionics budget consists of 112 € PCB (paid) and ~500€ from sponsors (not fully used). So, numbers are indicative not all actual expenses

A relevant point from the income structure is that team membership fees amount to 475 €, which represents approximately 44% of the direct funding obtained before the ICAI balancing contribution. This is important not only from an economic perspective, but also from a management perspective, because it shows a high level of internal commitment and responsibility towards the project. In the same way, the 550 € assigned to sponsors reflects the first step in the external funding strategy that the team is developing through its brand-building and partnership efforts.

On the expense side, the table shows that resources are mainly concentrated in the technical departments directly involved in the rocket system. Avionics represents one of the largest expense areas, with 662 €, due to the electronic components and sponsored material required for the flight system. Aerodynamics, Propulsion, and Recovery also concentrate essential costs related to the physical construction, launch capability, and safe recovery of the vehicle. Strategy expenses remain limited and are mainly related to support activities required to manage the project externally and administratively.

It is also important to note that some resources provided by ICAI are not fully quantified in the table, such as access to fabrication laboratories, 3D printing capabilities, tools, equipment, and technical facilities. These resources are essential for the development of the rocket and represent a significant form of institutional support, even when they do not appear as direct monetary expenses.

II. Concept of operations (CONOPS)

This section outlines the operational framework and environmental boundaries for the ICAI Rocket Team's L1 certification mission at Alcolea de Cinca. Following the top-down methodology of Systems Engineering, this Concept of Operations (CONOPS) defines the behavioural states of the vehicle and its critical interfaces with the ground launch personnel. By structuring operational roles and strict safety criteria, this framework provides a systematic blueprint to mitigate ground hazards and ensure a safe, disciplined, and fully traceable execution of the flight profile.

Operational framework and environment

Geographical boundaries and airspace constraints

The physical envelope of the launch operation is defined by the coordinates of the Aerocinca aerodrome at 41°43’33.9”N, 0°06’39.9”E, featuring a base elevation of 213 meters above mean sea level. This specific location establishes a strict ceiling constraint of 1000 meters for the authorized airspace waiver. Dédalo launch vehicle is engineered to achieve a nominal apogee of 550 meters approximately, which provides a safety margin of 45% below the authorized maximum altitude, ensuring no risk of airspace violation. The flat, unpopulated agricultural topography of the surrounding terrain serves as the primary landing zone, accommodating the recovery phase within a controlled radius.

Local wind profile and meteorological data analysis

Site meteorological data for the launch area, illustrated in figures 1,2 and 3, indicates a predominance of wind vectors originating from the West-Northwest (WNW) and Northwest (NW) sectors at both boundaries. According to the hourly mean velocity profiles, the local wind speed reaches a daily minimum at approximately 10:00 UTC, recording values of 3.0 m/s at a height of 50 meters and 3.5 m/s at 200 meters. Conversely, peak velocities occur late in the afternoon between 18:00 and 20:00 UTC, where speeds escalate to 5.5 m/s at 50 meters and up to 7.5

m/s at 200 meters. This wind distribution dictates that the optimal operational window is constrained between 06:00 and 11:00 UTC to remain safely below the Tripoli Unified Safety Code maximum sustained surface wind limit of 20 MPH (8.9 m/s).

Operational weather thresholds and launch pad alignment

Beyond wind vectors, the system's operational go/no-go criteria require a cloud coverage of less than 50% and the absolute absence of lightning or convective activity within a 16km radius of the pad. To counteract the micro-meteorological drift caused by the dominant WNW/NW wind patterns during the guided ascent phase, the launch rail will be mechanically tilted. Depending on real-time anemometer readouts on launch day, a precise compensation will be applied, ensuring the trajectory remains stable and the landing velocity of 6.4 m/s takes place within the designated recovery zone.

The environmental conditions at Alcolea de Cinca were analysed using wind profile data at 50m and 200m above ground level as well as the average daily profile of wind velocity, as shown in the figures below.

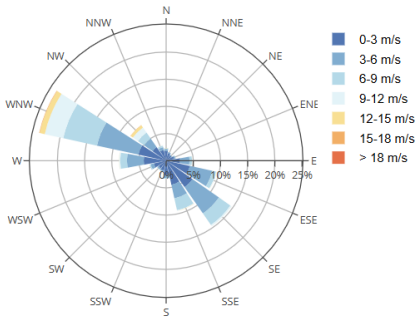


Figure 1: Wind rose analysis for Aerocinca at 50m altitude.

The predominant vector orientation from the West-Northwest (WNW) sector influences the ballistic drift profile during the initial phase of the guided ascent.

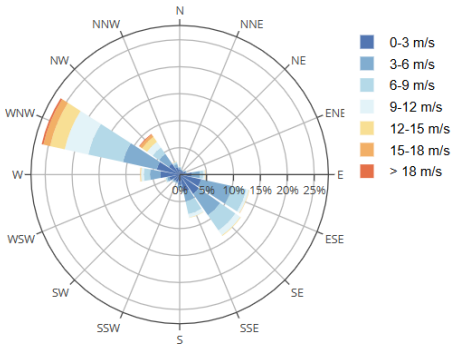


Figure 2: Wind rose analysis for Aerocinca at 200m altitude.

As altitude increases, the wind velocity gradients indicate a persistence of the WNW/NW prevailing direction, which is critical for estimating the trajectory compensation required for the Dédalo vehicle.

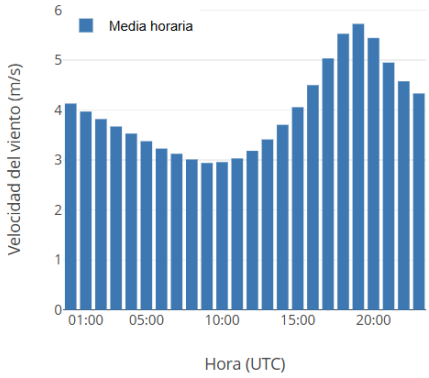


Figure 3: Average daily profile of wind velocity.

System states and mission phases

To establish a highly reliable behavioural architecture for the rocket, the internal software and hardware configurations are discretized into six distinct system states. This finite-state machine framework governs the operational mode of the main flight computer, sensors, integrated CanSat payload and recovery system, ensuring systematic transitions and strict hazard isolation throughout the mission.

STANDBY / SAFE: This is the initial state used for storage and transport. In this state, the vehicle remains unpowered and with the motor outside its compartment. Both the propulsion system and the CO2 mechanisms for parachute deployment are physically and electronically isolated. The flight computer and the CanSat remain inactive, which prevents data logging, accidental actuator activation, and battery drainage.

SELF-TEST: Activated immediately upon receiving the manual power-on command from the avionics operator. In this state, the main flight computer (dsPIC33 MCU) executes a comprehensive Built-In Self-Test (BIST) routine to initialize and verify the hardware integrity of all onboard sensors (barometer, IMU, and temperature sensor) via I2C and SPI communication buses. Concurrently, the Petit FatFs library initializes the SD card subsystem and pre-allocates the flight telemetry CSV file. If any communication or sensor diagnostic check fails, a dedicated fault LED is triggered to alert the launch crew; if all automated self-tests pass successfully, the system automatically transitions to the ARMED state.

ARMED: Activated automatically once the vehicle successfully passes the Self-Test sequence and is stabilized on the launch platform. In this state, the avionics execute real-time sensor calibration (barometric zeroing and IMU bias alignment) while the CanSat payload completes its ground synchronization. The video camera is turned on, and the ignition system is connected, awaiting the firing signal.

POWERED FLIGHT: Automatically activated at time t = 0 s when the Cesaroni Pro38 I212 motor ignites and the onboard accelerometer registers a sustained vertical acceleration threshold exceeding 2g. The flight computer begins high-frequency data logging (minimum 100 Hz), storing inertial, pressure, and temperature measurements. The CanSat subsystem switches to active data acquisition mode. This state remains active during the entire 1.7-second motor burn time, during which the vehicle experiences a maximum acceleration of 82.7 m/s^2 and exits the launch rail at approximately 20 m/s.

COASTING FLIGHT: This state initiates automatically upon motor burnout t = 1.7s as the vehicle decelerates inertially toward its maximum altitude. The system achieves its maximum velocity of 113 m/s at the boundary of this transition. Both the flight computer and the CanSat continue with data acquisition.

RECOVERY / DESCENT: Begins at the instant the avionics detect the zero-crossing point of the vertical velocity, signalling the nominal apogee of 550 meters. The flight computer sends the activation signal to the recovery system (CO2 parachute deployment). The velocity of the vehicle at that moment is around 6.3 m/s. Once the 2,5 m parachute inflates, the system transitions to a stabilized velocity descent configuration with a nominal landing velocity of 6.4 m/s. The avionics continue with data collection.

POST-FLIGHT SAFE: Activated when the inertial measurement unit (IMU) detects zero acceleration across all three axes and the barometer detects a constant pressure for a sustained period of 5 seconds, indicating ground touchdown (around t = 100 s of flight). The flight computer finalizes data logging and safely saves all open data files to the SD storage device. Once this process is finished, the avionics safely disconnect and power off.

INITIAL STATE	MISSION PHASE	OPERATIONAL CONDITIONS	SYSTEM ACTIONS	STATE TRANSITION EVENT	NEXT STATE
1. STANDBY / SAFE	Storage & transportation	Rocket disassembled. Electronic systems disconnected and powered off. Motor out of position.	-	Manual command from avionics operator	2. SELF-TEST
2. SELF-TEST	Pre-launch checkout / Ground operations	Rocket assembled. Avionics powered on via manual switch. Sensors undergoing automated boot sequence. Camera and ignition circuits remain isolated.	MCU initialization. Sensor communication integrity verification (I2C/SPI). SD card validation and CSV pre-allocation via Petit FatFs. LED fault monitoring active.	All automated sensor diagnostics and storage self-tests pass successfully.	3. ARMED
3. ARMED	Pad operations / pre-launch	Rocket assembled and on rail. Avionics and CanSat powered on and acquiring data. Motor in position with ignition system connected. Camera turned on.	Calibration of avionics sensors. CanSat operational. Start of camera recording Ignition system prepared.	Significant change in acceleration or altitude	4. POWERED FLIGHT
4. POWERED FLIGHT	Powered ascent	Motor ignited. Sustained acceleration greater than 2G. Vehicle ascending vertically.	Avionics and CanSat acquiring data. Kalman filter initialization at state entry. Motor active. Camera recording.	Motor Burnout (t = 1.7 s)	5. COASTING FLIGHT
5. COASTING FLIGHT	Inertial ascent	Motor burned out and hot. Vehicle ascending inertially. Progressive deceleration.	Avionics and CanSat acquiring data. Camera recording.	Avionics detect vertical velocity zero-crossing point	6. RECOVERY / DESCENT
6. RECOVERY / DESCENT	Stabilized descent	Controlled parachute descent.	Parachute system deployment. Avionics and CanSat acquiring data. Camera recording.	Ground touchdown detected	7. POST-FLIGHT SAFE
7. POST-FLIGHT SAFE	Ground operations / Recovery	Rocket motionless on the ground. Acceleration across all 3 axes stable at 0g and constant barometric pressure.	End of the Kalman filter cycle at state entry. Avionics and CanSat running an infinite loop. Camera recording.	Data saving and manual shutdown by the recovery team	SYSTEM POWERED OFF (END OF MISSION)

Table 2: SURTAM-J26 Mission Phase and State Transition Matrix.

To close the Systems Engineering loop, the flight data retained on the onboard SD card throughout the mission is mathematically validated post-landing. Since the avionics remain powered to guarantee data integrity, the compiled .csv telemetry log is extracted and processed via a dedicated ground station dashboard. This automated software tool instantly parses the logged sensor metrics (Kalman filter estimates, barometric altitude, and multi-axis acceleration), generating comprehensive analytical flight profiles to evaluate system performance against the nominal simulation baselines.

SIMULATION-BASED FLIGHT PROFILE TIMELINE

To complement the state transition logic, the nominal flight profile parameters extracted from the OpenRocket simulation attached below in the report are mapped chronologically below. These figures establish the technical baseline for real-time telemetry validation during the mission:

- **Launch pad departure (t = 0 to t ≈ 0.2 s):** The vehicle accelerates along the rail, achieving a safe lift-off velocity of 20 m/s at rail exit.

- **Powered ascent (t = 0 to t = 1.7 s):** The Cesaroni Pro38 I212 motor delivers a maximum acceleration of 82,7 m/s^2. At motor burnout (t = 1.7 s), Dédalo reaches its maximum velocity of 113 m/s.
- **Inertial coasting (t = 1.7 to t = 10.8 s):** The vehicle decelerates inertially for 9.1 s until it reaches the zero vertical velocity crossover point.
- **Apogee deployment (t = 10.8 s):** The nominal apogee is achieved at an altitude of 550 m. The CO2 recovery system triggers at a local velocity of 6.3 m/s.
- **Stabilized descent (t = 10.8 to t ≈ 100 s):** The 91.4 cm parachute inflates completely, establishing a constant descent profile with a nominal landing velocity of 6.4 m/s. Touchdown occurs at approximately t = 100 s of total flight time.

Personnel roles and interfaces

To ensure safe, disciplined and highly coordinated ground operations during the SURTAM 2026 competition at Alcolea de Cinca, the ICAI Rocket Team deployment crew is structured into a specialized operational command hierarchy. The launch crew consists of 9 multi-departmental engineering students, mapping internal management positions directly into standardized aerospace field roles. This structure guarantees clear lines of authority, dedicated safety oversight, and explicit personnel interfaces during pad operations.

LAUNCH CREW ROLE DISTRIBUTION AND FIELD HIERARCHY

TEAM MEMBER	IRT INSTITUTIONAL ROLE	FIELD OPERATION ROLE	PRIMARY LAUNCH-DAY RESPONSIBILITIES
Héctor Sánchez	IRT co-director / Head of Avionics	Launch Officer (LO) / Flight Candidate	Overall mission commander. Coordinates countdown, commands state transitions, and acts as the L1 Candidate interacting with the Tripoli LCO.
Mario Mora	IRT co-director / Head of design	Range Safety Officer (RSO)	Co-leads field crew. Holds absolute veto power over the launch window based on weather and structural criteria.
Agnès Ribera	Head of aerodynamics	Structural Integrity Specialist (SIS)	Preflight mechanical inspections, structural fastening, fin alignment tracking, and post-flight damage analysis.
Carolina Pueche	Head of propulsion	Propulsion & Pad Safety Officer (PPSO)	Leads propulsion and pad pyrotechnic operations. Responsible for motor casing inspection, ignition circuit deployment, pad arming, and executing misfire/abort protocols.
Pablo Losada	Head of recovery	Recovery & Payload Officer (RPO)	Responsible for packing the 2.5m parachute, disarming the CO2 actuator, validating the integrated CanSat payload telemetry, and leading visual tracking and field retrieval sweeps.
Marcos Alonso	Propulsion department member	Propulsion Systems Assistant (PSA)	Assists the PPSO with motor hardware audits, rail mechanical loading, and ignition circuit connections. Executes manual igniter extraction and core safing during aborts.
Álvaro Pérez	Avionics department member	Avionics & Telemetry Officer (ATO)	Manages dsPIC33 MCU boot sequences, automated sensor BIST diagnostics , camera standby/recording activation, rail sensor calibration, and post-flight CSV data extraction.
Blanca Serrano	Aerodynamics department member	Structural & Field Recovery Assistant (SFRA)	Co-executes pre-flight structural integrity checks, launch rail mechanical loading, post-flight field retrieval sweeps, and structural impact damage audits alongside the SIS.

Table 3: ICAI Rocket Team Launch-Day Operational Roles.

HUMAN-TO-HUMAN INTERFACES AND COMMUNICATION PROTOCOL

The personnel interfaces are governed by a strict "Call-and-Response" verbal communication protocol controlled by the Launch Officer (LO, Héctor) and supervised by the Range Safety Officer (RSO, Mario). The operational sequence defines three mandatory communication gates:

1. **The Propulsion-to-Pad Interface:** The Propulsion Assistant (Marcos) and Pad Safety Officer (Carolina) must mutually verify motor containment and rail locking. Any mechanical anomaly halts the countdown immediately.

2. **The Avionics-to-Recovery Interface:** The Avionics Officer (Álvaro) and Recovery Specialist (Pablo) must explicitly confirm sensor diagnostic readiness and CO2 deployment wiring continuity prior to arming the vehicle.

3. **The Team-to-Tripoli Interface:** The Launch Officer (Héctor) remains the sole authorized technical channel to communicate with the Tripoli Launch Control Officer (LCO). Once the master checklist is signed off by all departmental heads, Héctor requests final clearance for ignition circuit arming.

Actionable launch-day master Checklist (SURTSR-L1-GEN-002)

Pre-flight assembly and technical inspection (State: 1. STANDBY / SAFE)

This initial phase takes place in the team's preparation area. The vehicle remains completely unpowered, and all energetic subsystems are physically isolated to prevent accidental activation.

Item ID	Operational Step / Description	Responsible Party	Verification
STBY-01	**Structural Integrity Check:** Inspect the "Quantum" tubing and cardboard outer sleeve for cracks, deformation or damage on surface. Verify that all structural joints and fins are perfectly aligned and secure.	SIS (Agnès Ribera) / SFRA (Blanca Serrano)	☐ Verified
STBY-02	**Fastener Torque Audit:** Physically check and tighten all modular assembly connectors and threaded inserts. All fasteners (screws, bolts, adhesive joints) are confirmed tight and complete.	SIS (Agnès Ribera) / SFRA (Blanca Serrano)	☐ Secured
STBY-03	**CM and CP Verification** - Centre of Pressure (CP) is clearly and permanently marked on the airframe - Centre of Gravity (CG) is clearly and permanently marked on the airframe - Stability margin (CP − CG distance) verified and within design specification	SIS (Agnès Ribera) / SFRA (Blanca Serrano	☐ Verified
STBY-04	**Recovery Mechanical Prep:** Inspect the parachute canopy and lines. Carefully fold and pack it into the deployment bag alongside the fire-resistant blast blanket.	RPO (Pablo Losada)	☐ Packed
STBY-05	**CO_2 Actuator Isolation:** Verify that the CO_2 parachute deployment mechanism is mechanically disarmed and that no puncturing pin is under tension.	RPO (Pablo Losada)	☐ Safe
STBY-06	**Propulsion Hardware Audit:** Inspect the Cesaroni 38mm 3-Grain aluminium casing (Model 71022) for scratches or debris. Confirm the delay grain and ejection charge have been completely removed.	PPSO (Carolina Pueche) / PSA (Marcos Alonso)	☐ Inspected
STBY-07	**Ground Logistics Initialization:** Initialize the master countdown timekeeping log, verify team-to-spectator communication lines, and review ground resource inventory.	RSO (Mario Mora)	☐ Initialized

Avionics boot and built-in self-test (State: 2. SELF-TEST)

This phase initializes the electronic core of the vehicle. Sensor health is verified through software protocols prior to moving the rocket onto the launch pad.

Item ID	Operational Step / Description	Responsible Party	Verification
TEST-01	**Avionics Ground Boot:** Insert the micro-SD card into the logging slot and power on the dsPIC33FJ32MC202 Main Control Unit (MCU) flight computer via the hardware switch.	LO (Héctor Sánchez) / ATO (Álvaro Pérez)	☐ Powered
TEST-02	**Sensor Diagnostics & BIST:** Run the automated Built-In Self-Test firmware to verify barometer, IMU, and temperature sensor communication integrity via I2C/SPI buses. Confirm background CSV data logging starts via Petit FatFs. Verify no failure LED is triggered.	LO (Héctor Sánchez) / ATO (Álvaro Pérez)	☐ Passed
TEST-03	**Camera Standby Mode:** Power on the onboard video camera and verify battery life. Set to standby mode (not yet recording).	LO (Héctor Sánchez) / ATO (Álvaro Pérez)	☐ Ready
TEST-04	**Pre-Pad Weather Review:** Review real-time anemometer wind profile data and cloud coverage (50%). Confirm wind parameters remain below the Tripoli safety limits (<8.9 m/s}).	LO (Héctor Sánchez) / RSO (Mario Mora)	☐ Clear

Launch pad operational procedures (State: 3. ARMED)

This phase covers the integration of the vehicle onto the rail, physical loading of the motor, and final handshake authorizations before entering the flight sequence.

Item ID	Operational Step / Description	Responsible Party	Verification
PAD-01	Transport the vehicle from the preparation tent to the SpainRocketry launch rail. Ensure zero mechanical shocks occur during transit. Verify all airframe sections are properly assembled and secured	SIS (Agnès Ribera) / SFRA (Blanca Serrano)	☐ Delivered
PAD-02	**Launch Rail Mechanical Loadin**g: Insert the rocket guide buttons into the rail slot. Verify rail button condition verified: no cracks or deformation. Slide the vehicle down to the launch stopper and verify free axial movement without binding or excessive friction. Verify fins are correctly attached and aligned.	SIS (Agnès Ribera) / SFRA (Blanca Serrano)	☐ Loaded
PAD-03	**Wind Profile Compensation & Rail Alignment:** Check real-time weather vectors. Mechanically tilt the launch rail (up to a maximum of 20º from vertical) to counteract dominant WNW/NW ballistic drift.	LO (Héctor Sánchez) / RSO (Mario Mora)	☐ Aligned
PAD-04	**Motor Integration:** Insert the prepared Cesaroni Pro38-3G Smoky Sam motor casing into the lower fuselage mount. Secure it firmly by tightening the airframe S2 section and locking the set screw.	PPSO (Carolina Pueche) / PSA (Marcos Alonso)	☐ Locked

PAD-05	**Ignition Circuit Deployment:** Insert the E-Match igniter into the motor nozzle throat. Connect the igniter leads to the SpainRocketry launch controller. VERIFY SYSTEM REMAINS PHYSICALLY DISARMED AT THE PAD CONTROLLER.	PPSO (Carolina Pueche) / PSA (Marcos Alonso)	☐ Connected
PAD-06	**Sensor Calibration:** Command the dsPIC33 MCU via software interface to execute real-time sensor calibration, performing barometric zeroing and IMU bias alignment directly on the rail.	LO (Héctor Sánchez) / ATO (Álvaro Pérez)	☐ Calibrated
PAD-07	**Payload Status Check:** Confirm that the integrated CanSat payload within the internal PLA compartment is powered on and successfully transmitting background sensor telemetry to the ground station.	RPO (Pablo Losada)	☐ Telemetry OK
PAD-08	**Visual Recording Activation:** Trigger the onboard video camera recording mechanism. Verify that the status light indicates active recording mode.	ATO (Álvaro Pérez)	☐ Recording
PAD-09	**Range Clearance & Evacuation:** Conduct a strict visual sweep of the immediate pad area. Evacuate all personnel to the designated safety perimeter according to Tripoli safety protocols.	LO (Héctor Sánchez) / RSO (Mario Mora)	☐ Pad Clear
PAD-10	**Timekeeping Sync & Media Logging:** Establish the definitive countdown T-time synchronization and activate ground-station video/photo documentation logs.	GCCO (Juan Conde)	☐ Sync
PAD-11	**Tripoli Handshake & Authorization:** Formally notify the Tripoli Launch Control Officer (LCO) that the Dédalo vehicle is fully armed on the rail. Confirm airspace waiver clearance below the 1000m ceiling.	LO (Héctor Sánchez)	☐ Authorized
PAD-12	**Final Go/No-Go Operational Poll:** Conduct a final verbal call-and-response checklist approval across all departmental heads to clear the vehicle for the automated countdown.	RSO (Mario Mora)	**GO / NO-GO**

Countdown and ignition sequence (Transition to 4. POWERED FLIGHT)

This phase establishes the final safety enforcement perimeter and execution of the firing sequence. All communications follow a strict countdown timeline managed by the ground command.

Item ID	Operational Step / Description	Responsible Party	Verification
IGN-01	**Range Perimeter Enforcement:** Perform a final visual sweep and ensure that *everyone* within the launch site is evacuated to a safe distance of at least 30 feet away from the rocket airframe.	RSO (Mario Mora)	☐ 30 ft Clear
IGN-02	**Ignition Circuit Arming:** Insert the safety key into the SpainRocketry launch control device to energize the pyrotechnic circuit connected to the E-Match igniter.	PPSO (Carolina Pueche) / PSA (Marcos Alonso)	☐ Armed
IGN-03	**Audible Timekeeping Alert:** Shouted out loud the final 5-second countdown to alert all range spectators and track cameras before firing.	RSO (Mario Mora)	☐ Counted
IGN-04	**Ignition Trigger:** Press and hold the ignition button on the Tripoli launching controller until motor ignition occurs.	LO (Héctor Sánchez) / PPSO (Carolina Pueche)	☐ Fired

Misfire and abort contingency protocol

If the automated firing command is triggered but the Cesaroni P38-3G motor fails to ignite properly or at all, this strict emergency sub-checklist overrides standard operations immediately to isolate pad hazards.

Item ID	Operational Step / Description	Responsible Party	Verification
ABRT-01	**Circuit De-energization:** Instantly release the ignition button and remove the safety key from the SpainRocketry launching device. Hault all automated sequences.	PPSO (Carolina Pueche) / PSA (Marcos Alonso)	☐ Disarmed
ABRT-02	**Pad Lockdown Phase:** Order an absolute hold on the pad. Force all personnel to remain at the safe distance perimeter (at least 30 feet away) for a minimum of 2 minutes to prevent hangfire injuries.	RSO (Mario Mora)	☐ 2 Min Hold
ABRT-03	**Single-Person Approach Rule:** Once the 2-minute cooldown expires, authorize *only one designated person* to approach the launch rail to service the vehicle.	PPSO (Carolina Pueche)	☐ Authorized
ABRT-04	**Igniter Safe Extraction:** Disconnect the igniter leads from the launch controller box and carefully pull the E-Match igniter out of the motor nozzle throat BEFORE moving the rocket.	PPSO (Carolina Pueche) / PSA (Marcos Alonso)	☐ Safe
ABRT-05	**Post-Abort Safe State:** Return the main computer to a safe state, download error logs, and disassemble the airframe sections for internal motor casing inspection.	ATO (Álvaro Pérez) / PSA (Marcos Alonso)	☐ Reset

Post-flight recovery and data logging (State: 7. POST-FLIGHT SAFE)

This final phase governs the safe tracking of the landing footprint, hardware retrieval, and software data extraction once touchdown velocity returns to zero.

Item ID	Operational Step / Description	Responsible Party	Verification
RCV-01	**Visual Descent Tracking:** Monitor the parachute descent path from apogee down to touchdown using binoculars. Estimate landing zone coordinate vectors.	RPO (Pablo Losada)	☐ Sighted
RCV-02	**Field Retrieval Sweep:** Navigate the landing field to collect the main airframe sections.	RPO (Pablo Losada) / SIS (Agnès Ribera)	☐ Retrieved
RCV-03	**Structural Damage Audit:** Conduct an immediate physical inspection of the "Quantum" body tube, birch plywood fins, and nose cone joint to verify that no non-standard damage occurred during impact.	SIS (Agnès Ribera) / SFRA (Blanca Serrano)	☐ Inspected

RCV-04	**Manual Avionics Shutdown:** Toggle the external hardware power switch off to safely end the infinite background software loop of the dsPIC33 computer and conclude the mission.	ATO (Álvaro Pérez)	☐ Off
RCV-05	**CSV Telemetry Data Extraction:** Remove the micro-SD card from the flight core. Insert it into the ground station laptop and load the csv log into the automated data dashboard for performance graphing.	ATO (Álvaro Pérez)	☐ Exported

III. System Architecture

Structural System

The rocket has been designed with a modular architecture that separates the propulsion, avionics, recovery system, CanSat, and nose cone. It has an overall length of 1150 mm and a maximum diameter of 84 mm. The rocket's mass is 2339 g (without the motor) and 2814 g in its complete configuration. In terms of stability, it has a margin of 1,61 cal, with the Center of Gravity (CG) located at 68,2 cm and the Center of Pressure (CP) at 81,8 cm.

In general terms, a variety of materials are used to ensure design coherence and support the modular approach. 3D printing (ABS and PLA) is used for complex parts requiring weight customization, 6 mm birch plywood for elements subjected to higher mechanical stress (fins and joints), and "Quantum" tubing for the main structure; additionally, the rocket features an outer cardboard "sleeve". All fixed joints are secured with epoxy (Araldite-type) at the contact areas, while the assembly of the main sections employs fasteners (M2 to M5) and metal inserts, allowing the rocket to be fully assembled and disassembled.

Below is a detailed breakdown of each subsystem, from top to bottom, in terms of its structure and the materials comprising the complete rocket:

1. **Nose Cone Subsystem**
   - **Function:** Provides an optimal aerodynamic profile and serves as the upper attachment point for the recovery system.
   - **Materials:** 3D-printed using ABS.
   - **Design:** A tangent profile (shape factor = 1) is used to achieve a seamless blend with the airframe. This smooth transition prevents the sudden separation of the aerodynamic boundary layer, thereby minimizing turbulence during subsonic flight. Additionally, analysing its design dimensions (20 cm in length and an 8.4 cm base diameter), the resulting Fineness Ratio is approximately:

     $$\frac{L}{D} = \frac{20}{8,4} \approx 2,38$$

     This ratio indicates an efficient aerodynamic profile without being excessively long (target values for subsonic flight typically range between 2 and 3). Finally, a metal eyebolt is threaded into the base to serve as the anchor point for tying the shock cord.

2. **Recovery Subsystem**
   - **Function:** Contains the components that ensure the safe descent and recovery of the rocket.
   - **Materials:** The outer shell is made of "Quantum" tubing.

- **Design:** This module houses the CO_2 ejector, the parachute deployment bag, its fire-resistant blast blanket, and a telemetry system to signal the ejection. The design features a dedicated channel to allow the passage of the shock cord ascending from the rocket's lower section. All these elements will be detailed more comprehensively in the Propulsion System section.

3. **Avionics and CANSAT Subsystem**
   - **Function:** Contains the electronics and the payload (CANSAT).
   - **Materials:** "Quantum" tubing for the exterior; Aluminium, PLA, and ABS for the internal structures.
   - **Design:** This subsystem consists of several interconnected elements. First, an internal compartment, 3D-printed in PLA, is inserted into the "Quantum" tubing. The electronic equipment is mounted on two "sleds" that include routing holes for securing the components with zip ties. These sleds are inserted into the compartment and locked in place using disks alongside sliding and rotary fasteners. The entire assembly is joined to the rest of the rocket body using connectors equipped with threaded inserts.

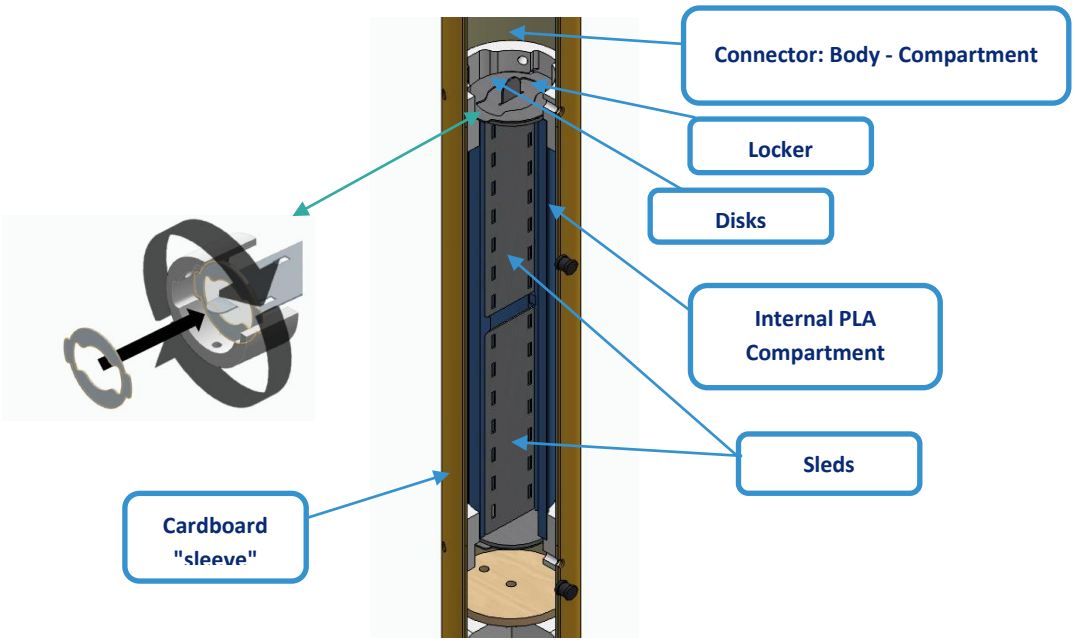


Figure 4: Detailed Avionics and CANSAT Subsystem

4. **Coupler Section Subsystem**
   - **Function:** Joins the upper airframe to the lower section and bears the tension of the shock cord.
   - **Materials:** 6 mm birch plywood disk and "Quantum" tubing.
   - **Design:** This section has a birch plywood attachment disk (bulkhead) glued to the inside of the "Quantum" tubing. This bulkhead has two key features: an off-center hole where the eyebolt is installed to anchor the shock cord, and a central hole through which a threaded rod passes. This threaded rod is responsible for securely holding the upper section and the lower section (motor system) together.

5. **Lower Section (Propulsion) Subsystem**
   - **Function:** Contains the motor, provides stability via the fins, and transfers thrust to the rest of the rocket.
   - **Materials:** 3D-printed ABS parts, internal cardboard tubing, and birch plywood for the fins and 3 centering rings.

- **Design:** The motor mount consists of a cardboard tube to which three centering rings are glued. This assembly is bonded to the inside of the lower fuselage (airframe S1). The fins are inserted through slots in airframe S1 and glued directly to the internal cardboard tube. Finally, a second tube (airframe S2, which features embedded threaded inserts) is screwed onto airframe S1 and locked in place with a set screw, closing the entire assembly.

Propulsion System

The propulsion system for the launch consists in a reloadable rocket motor, which consists in a reloadable grain, casing and inner motor cardboard tube which is glued to the lower section of the rocket with three centering rings.

The grain used is a Cesaroni P38 – 3G Smoky Sam (I212) with the following characteristics:

- **Model:** 71332
- **Size:** 38 mm Reload
- **Delay:** 14 sec
- **Burn time:** 1.7 sec
- **Total Impulse:** 364. 0 Newtons/Second
- **Motor Length:** 245 mm
- **Max Thrust:** 255.2 Newtons
- **Total Mass:** 475.0 g
- **Propellant Mass:** 248.0 g
- **Igniter:** E-Match
- **Manufactured by:** Cesaroni

The cased used is a Cesaroni 38mm 3-Grain Case with the following characteristics:

- **Model:** 71022
- **Material:** 6061 – T6 aluminium tubing
- **Weight:** 121.9 g (4.27 oz)
- **Length/Thickness:** 23.11 cm
- **Manufactured by:** Cesaroni

As the recovery system will oversee the parachute ejection, the delay grain and ejection charge will be taken out to guarantee the correct operation of the recovery system and to avoid damage to the rocket.

Avionics System

The rocket's avionics system is comprised of both hardware and software subsystems responsible for flight monitoring, sensor acquisition, state estimation, data logging, parachute deployment and landing (recovery).

The flight computer used is a Microchip dsPIC33FJ32MC202, referred to throughout this report as the Main Control Unit (MCU). The MCU was selected due to its real-time processing capabilities, deterministic behaviour, low power consumption, integrated peripherals, low cost-efficiency ratio and robustness for embedded aerospace applications.

The avionics architecture was designed with the objectives of reliability, modularity, and ease of debugging during both ground testing and flight operations. The system continuously acquires data from onboard sensors, processes flight-state information and stores telemetry data locally.

Hardware architecture

The avionics hardware consists of the following main components:

- Main Control Unit (MCU): dsPIC33FJ32MC202
- Altimeter / barometric pressure sensor
- Accelerometer sensor
- Temperature sensor
- Battery
- SD card data logging subsystem
- Parachute deployment with a CO_2 charge

The MCU interfaces with the sensors primarily through I2C and SPI communication buses. Sensor measurements are sampled periodically and synchronized through the main flight loop.

Software architecture

The software consists of embedded C firmware for the rocket flight computer. It manages sensor initialization, calibration, self-tests, flight-state detection, telemetry logging, and fault handling. The system uses data from the accelerometer, barometer, and temperature sensor, and combines acceleration and altitude measurements with a Kalman filter to estimate vertical position and velocity during flight.
The flight logic is organized as a state machine shown in the CONOPS. However, there are a few more than those solely for logging more information about the flight. These are all the states:

x

- Power on
- Sensor initialization
- Calibration
- Self-test
- Idle
- Armed
- Countdown
- Ignition
- Liftoff
- Powered ascent
- Coast
- Apogee
- Parachute deployment
- Descent
- Landed
- Recovery
- Abort and error states

Supplier-provided interfaces were not used because of memory limitations. Instead, custom SPI and I2C drivers were developed to communicate with the onboard sensors and peripherals. For SD card storage, the firmware uses Petit FatFs, allowing telemetry to be written to a preallocated CSV file on the SD card.

During flight, the CSV telemetry log stores timestamped sensor readings, state changes, errors, altitude, acceleration, temperature, and Kalman filter estimates. After the flight, the SD card can be connected to a laptop, where a dashboard automatically plots the flight information and provides instant feedback for analysis.

It is important to mention that we follow MISRA C to ensure safe, reliable, and consistent code.

Recovery System

The recovery system consists of two main components: the parachute and the CO2 ejection system. At apogee the synchronized action of these components ensures the proper deployment of the parachute, guaranteeing a stable descent at a reduce terminal velocity to prevent any structural damage to the rocket upon impact.

Parachute assembly

The main parachute is made from standard parachute fabric, nylon ripstop, selected for its durability and lightweight properties. It has a diameter of 2.5 meters. The canopy is circular and is made by sewing together 12 triangular panels, each with a 30º angle. Additionally, it features a centre hole (vent) to stabilize the descent and prevent the rocket from swaying during its fall. High strength nylon lines will be used to connect the parachute to the rocket, providing enough strength to withstand the deployment shock.

Ejection System

The ejection system features a custom ABS 3D printed design housing designed to hold a standard 16-gram CO2 cartridge. The system is purely mechanical and triggers at apogee when the avionics system send a signal to a 7Kg high torque servo. Once activated, the servo releases a spring-loaded pin driven by a 4.7 Kg spring. The instant release of gas creates a pressure differential inside the recovery bay forcing the nosecone to pop off pulling the parachute out for a successful deployment. Ground ejection test will be conducted to verify that the spring consistently puncture the cartridge and the system successfully deploys.

Payload System (CanSat Integration)

The rocket's mechanical acceleration measurement system is an experimental subsystem designed to estimate the rocket's longitudinal acceleration during launch. Its purpose is to provide an independent measurement that can later be compared with the data obtained from an external accelerometer from the avionics' system.

The system is based on a known mass of 20g attached to a calibrated spring. During launch, the rocket accelerates upwards and the mass tends to remain behind due to inertia, causing the spring to stretch. This displacement is measured using a VL53L4CD Time-of-Flight distance sensor. The measured distance is then converted into spring displacement and, from that displacement, into acceleration. The processing unit used for this subsystem is an ESP32 DevKit.

Hardware architecture

The acceleration measurement system hardware consists of the following main components:

- ESP32 DevKit microcontroller
- VL53L4CD Time-of-Flight distance sensor
- Spring with calibrated stiffness
- Mass of 20g
- Reflective white surface attached to the mass
- Mechanical, plastic guide for the mass

- SD card data logging module

The spring is mounted vertically and fixed to the upper part of the housing. The 20 g mass is attached to the lower end of the spring and moves along the longitudinal axis of the rocket. The VL53L4CD sensor is placed below the mass, pointing upwards towards a reflective surface.

During upward acceleration, the mass moves closer to the sensor, reducing the measured distance. Then, the microcontroller uses this change in distance to calculate the displacement of the mass.

The VL53L4CD sensor communicates with the ESP32 using the I2C bus. The SD card module communicates through SPI.

Software architecture

The software runs on the ESP32 DevKit and manages sensor initialization, position calibration, distance acquisition, acceleration calculation and data logging.

At the beginning, the system initializes the VL53L4CD sensor and the SD card. It then performs an initial calibration while the rocket is stationary by averaging several distance measurements. This value is used as the reference position.

During operation, the ESP32 is continuously reading the distance from the VL53L4CD sensor. The firmware converts the mass displacement into acceleration using the known mass and spring stiffness. The system stores the measurements on an SD card in CSV format. After the flight, the SD card can be connected to a computer, and the CSV files can be opened. The data can then be plotted to compare the mechanical acceleration estimate with the external accelerometer measurements.

IV. Compliance with Requirements

1. General Requirements

[SURTSR-L1-GEN-001]	**L1 Certification Candidate**
The candidate must be at least 18 years old to opt for the L1 certification and must be part of the Tripoli association and SURT.	
Compliance Method	Description
Solution	Héctor Sánchez Valencia is an active member of TRA and is registered under SURT for the 2026 competition. Passport/ID verification confirms the age requirement.
Additional Notes (If applicable)	-

[SURTSR-L1-GEN-002]	**Launch Day Checklist**
The technical report must include a detailed launch preparation checklist covering motor assembly, rocket integration, required tools, and pre-launch procedures. The system must not present excessive damage or non-standard repairs before launch.	
Compliance Method	Description & Test
Solution	A launch-day master checklist has developed and integrated into Section II.2.4 of this technical report. The protocol systematically covers pre-flight structural audits, avionics Built-In Self-Tests (BIST), Cesaroni Pro38 motor assembly, launch pad integration, countdown milestones, emergency misfire contingency protocols, and post-flight

	recovery data logging. Each action item is explicitly mapped to a designated operational team role to minimize human error and guarantee strict compliance with Tripoli safety regulations.
Additional Notes (If applicable)	The step-by-step verification procedures are fully synchronized with the 7-state avionics software architecture of the dsPIC33 MCU flight computer to ensure operational and validation consistency.

2. Propulsion and Simulation Requirements

[SURTSR-L1-PROP-001]	**Motor Impulse**
The rocket must use a single certified HPR motor within the H–I impulse range (160–640 Ns), and proof of purchase must be provided.	
Compliance Method	Description
Solution	The propulsion system utilizes a single, commercially certified Cesaroni P38 – 3G Smoky Sam (I212). This motor is an I class motor that provides a total impulse of 364 Ns which falls within the requirements (160-640Ns).
Additional Notes (If applicable)	Proof of purchase is shown in Annex I

[SURTSR-L1-PROP-002]	**Motor-Structure Interface**
The motor must be securely fixed and aligned with the rocket structure using centering rings.	
Compliance Method	Description & Test
Solution	**Description:** The motor mount assembly utilizes three plywood centering rings bonded to the motor mount tube using high-strength epoxy. This multi-ring configuration distributes the thrust load evenly to the airframe and ensures rigid axial alignment. **Test:** Alignment will be verified to ensure concentricity in the forward and aft ends of the frame. Structural security will be checked with a push-test to ensure rigidity.
Additional Notes (If applicable)	Engineering CAD drawings of the centering rings are shown in section VII.

[SURTSR-L1-PROP-003]	**Ascent Trajectory**
The rocket must not exceed 1000 m altitude or deviate outside the launch range. Simulations should typically remain below 600 m apogee.	
Compliance Method	Analysis & Test
Solution	**Analysis:** Flight simulations were conducted using OpenRocket Software using a 3D model of the rocket and the motor mentioned. With nominal values of the launching site as well as wind conditions typical to the area, the apogee reached 520 m which remains under the 600 m. Also, the landing footprint is inside the launch range permitted.
Additional Notes (If applicable)	Simulation graphs and description is showed in section V.

[SURTSR-L1-PROP-004]	CP–CG Stability Margin
The distance between the centre of pressure (CP) and centre of gravity (CG) must be greater than 1.5 calibres.	
Compliance Method	Analysis
Solution	Stability analysis was performed with OpenRocket Software. Based on the final structural mass distribution and the selected motor configuration, the Centre of Gravity (CG) is located at 68,2 cm and the Centre of Pressure (CP) at 81,8 cm, it leaves a margin of 1,61 calibres which is over the 1,5 calibres permitted.
Additional Notes (If applicable)	-

3. Structure Requirements

[SURTSR-L1-STR-001]	Structural Design
The rocket airframe must follow a conventional design including fins; non-conventional configurations are not permitted.	
Compliance Method	Description
Solution	As seen explained in the system architecture, the design has met all requirements and was developed taking the stablished requirements into account. The different components can be viewed in much more detail in the engineering drawings section.
Additional Notes (If applicable)	-
[SURTSR-L1-STR-002]	**CP and CG Marking**
The centre of pressure (CP) must be clearly marked on the rocket airframe.	
Compliance Method	Test
Solution	Centre of Mass (CM) and the Centre of Pressure (CP) will be clearly marked on the rocket airframe. For the CM, the balancing method will be used: the rocket (fully assembled in flight configuration) will be placed horizontally on a pivot point, adjusting its position until perfect balance is achieved. Then, it will be marked on the airframe as the CM. For the CP, the cardboard cutout method will be used. The model will be laid flat on a piece of cardboard, its exact outline will be traced, and it will be cut out. Next, the balance point of this 2D silhouette will be found using the same pivot method as with the CM. Once the cardboard is balanced, the distance from the tip of the silhouette's nose to the balance mark will be measured, and that same measurement will be transferred from the nose of the actual model to mark the CP on the outer airframe. It is important to consider that the cutout method is a conservative approximation that assumes a high angle of attack, placing the CP with an extra margin of safety. For this reason, it will be validated and cross-referenced with the aerodynamic data calculated by OpenRocket.

Additional Notes (If applicable)	-

4. Recovery Requirements

[SURTSR-L1-REC-001]	**Parachute Recovery System**
The rocket must use a parachute-based recovery system; alternative methods are not allowed.	
Compliance Method	Description & Test
Solution	A ABS 3D-printed mechanism that uses a 7kg servo and a 4.7kg spring to pierce a 16g CO2 cartridge, ejecting the nosecone and deploying a 2.5m parachute.
Additional Notes (If applicable)	-

[SURTSR-L1-REC-002]	**Recovery Deployment Timing**
The recovery system must deploy at apogee.	
Compliance Method	Description & Analysis
Solution	The onboard avionics system monitors the flight trajectory in real-time and sends an automated electronic signal to the high-torque servo exactly at apogee.
Additional Notes (If applicable)	-

[SURTSR-L1-ELEC-001]	**Avionics integration in the rocket**
The launch system may optionally include either self-built or commercial functional avionics capable of recording, at a minimum, the apogee altitude.	
Compliance Method	Description & Analysis
Solution	The rocket includes a custom-designed avionics PCB. The board integrates a barometric pressure sensor, an IMU, and a temperature sensor, all controlled by a dsPIC microcontroller. The system is powered by an onboard battery with an estimated autonomy of approximately 2 hours, which is sufficient for pre-launch preparation, flight, and recovery operations. The barometric sensor is used to estimate altitude during flight and detect the apogee event. Once apogee is detected, the microcontroller sends a control signal to the servo actuator of the recovery system, initiating deployment. As a redundancy measure, if apogee detection fails or the barometric data becomes unreliable, the system includes a timed backup trigger that activates the recovery system after a predefined delay.
Additional Notes (If applicable)	-

[SURTSR-L1-REC-003]	**Descent Rate**
The descent velocity must not exceed 10 m/s.	
Compliance Method	Analysis & Test
Solution	The OpenRocket simulations detailed in the 'Simulation and Flight Profiles' section, show that the recovery system and the rest of the rocket body facilitate a highly controlled final landing phase, with a steady terminal velocity that settles between 5 m/s and 6 m/s.
Additional Notes (If applicable)	-

5. Launch Rail Requirements

[SURTSR-L1-LR-001]	Rocket Rail Guides
The rocket must include at least two rail guides compatible with the standard launch rail.	
Compliance Method	Test
Solution	The rocket incorporates three rail guides compatible with the standard launch rail: one located above the centre of mass (CM) in the central compartment section, another near the centre of pressure (CP), and the final one in the fin and motor section.
Additional Notes (If applicable)	-

6. Payload (PL) (optional)

SURTSR-L1-PL-001	Payload
The launch vehicle may contain a payload (either inert or functional payload) of no less than 400g. The payload must have similar dimensions corresponding to a CanSat format.	
Compliance Method	Description & test
Solution	The proposed solution is to design a functional CanSat payload consisting of 3D-printed housing that contains the spring, the mass, the sensors and the microcontroller. The total mass will be ensured to exceed the 400g by adjusting the design of the printed housing.
Additional Notes (If applicable)	-

SURTSR-L1-PL-002	Non – deployable payload
The payload (if integrated into the launch vehicle) must remain inside the rocket fuselage during all phases of flight.	
Compliance Method	Description & test
Solution	The payload will be designed as a non-deployable module, fully integrated inside the rocket. The 3D printed housing will be mechanically fixed to the structure, so that it always remains inside.
Additional Notes (If applicable)	-

V. Simulation and Flight Profiles

These flight simulation plots provide a clear overview of the rocket's aerodynamic behaviour, structural performance, and recovery sequence. The vertical motion profile highlights a maximum vertical velocity of approximately 108 m/s achieved right before motor burnout occurs at around the 2-second mark. Following burnout, the vehicle experiences a smooth deceleration phase until it reaches a stable apogee of 521 m (bellow the 1000m requirement) at exactly 10,6 seconds into the flight, which perfectly triggers the activation of the primary recovery mechanism. The final landing phase demonstrates a highly controlled descent, with the parachute deploying at 6,43 m/s and the rocket touching down at 7,92 m/s, while the vertical descent rate remains safely below the 10 m/s limit required for a compliant recovery sequence.

The remaining performance graphs confirm the vehicle's structural stability and wind drift resistance under active conditions. Additionally, the lateral drift analysis displays that the maximum horizontal distance

travelled from the launch pad throughout the entire flight profile is roughly 245 m. Because the final landing footprint is well under the maximum permissible drift radius of 500m, the simulation confirms a highly reliable, easily recoverable flight path within the designated boundaries of the field.

Velocity off rod	Apogee	Velocity at deployment	Optimum delay	Max. Velocity	Max. Acceleration	Time to apogee	Flight time	Ground hit velocity
19,8 m/s	521 m	6,43 m/s	8,92 s	108 m/s	79,3 m/s^2	10,6 s	89,8 s	7,92 m/s

Table 4. Summary of Flight Parameters

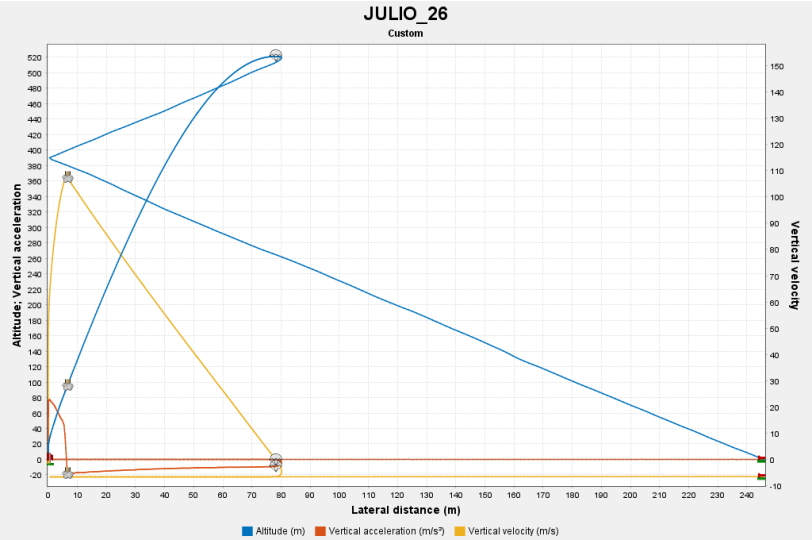


Figure 5. Simulation lateral distance vs vertical velocity

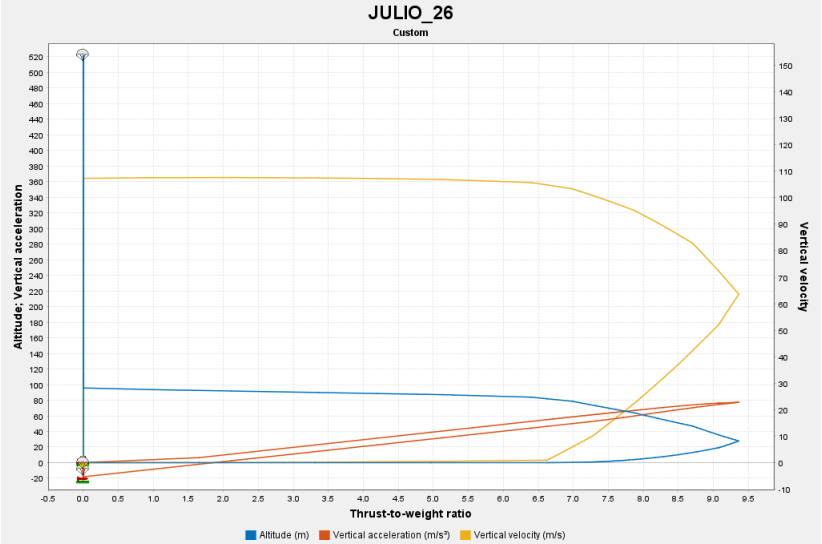


Figure 6. Simulation Thrust-to-weight ratio

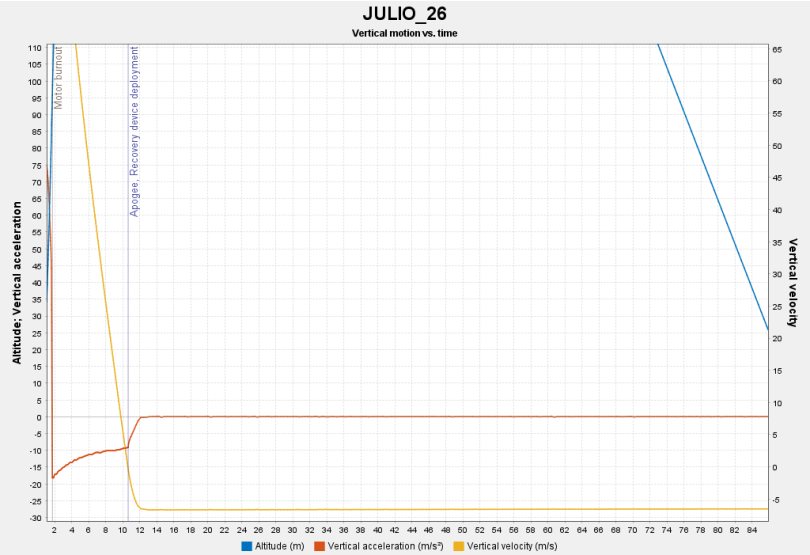


Figure 7. Simulation vertical motion vs time

VI. Risk Assessment

This risk assessment aims to identify, analyse and manage potential risks that could occur on the launch day, to safeguard people, assets, and the overall project success.

Risk/Hazard	Cause	Effect	Mitigation strategy
Ballistic re-entry (Parachute failure)	Failure of the altitude sensor or backup time sensor / Mechanical jam of the parachute inside the body.	The rocket does not deploy the parachutes after the apogee, resulting in high velocity and unsafe impact with the ground.	**Redundancy:** Dual deployment systems (altimeter and timing sensors). **Testing:** Conduct ground testing to ensure the parachute deploys safely before launching date.
Motor forward ejection (Motor flies off its casing and inner tube)	Failure of motor retention mechanism (e.g. loose centering rings, damaged inner motor tube or casing or inadequate epoxy use)	Motor casing shoots directly into the inside of the body of the rocket, damaging the structure as well as the avionics and recovery systems.	**Components:** Use commercially manufactured components. **Inspection:** Verify epoxy integrity in centering rings to the body and to the inner motor tube.

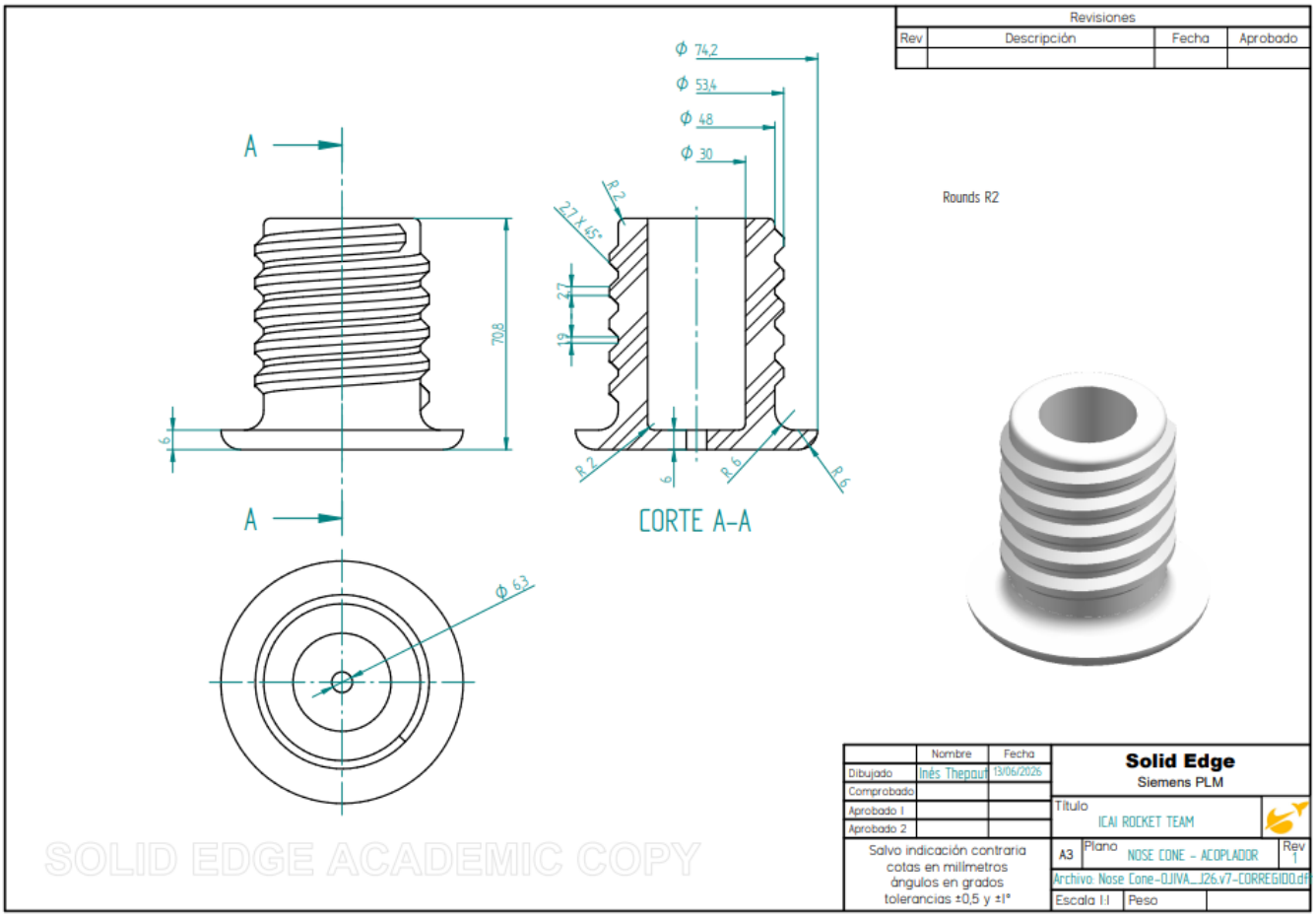
Catastrophic Motor Failure	Manufacturing defect in the motor reloads or structural failure of the casing.	Rapid overpressurization causes the motor to explode.	**Certifications:** Only use official motor reloads. **Safety distance:** Ensure that everyone inside the launching site is standing at a safe distance from the launch pad.
Unstable flight path (Low stability)	Centre of gravity (CG) too far aft due to unexpected mass movements inside the body.	The rocket changes trajectory or flies horizontally after leaving the launch rail. Resulting in a safety hazard.	**Simulation verification:** Constantly check the OpenRocket simulation to ensure the mass distribution is updated. **Pre-flight check:** Before launching, check and measure that the CG is in its right place.
Excessive wind drifts	Launching with high winds or wrongly sized parachute.	The rocket lands outside the permitted landing range.	**Wind limits:** If the wind velocity is over the recommended velocity, abort launch.
Stuck on launch rail	Launch buttons are misaligned or the launch rail is blocked.	Rocket cannot accelerate smoothly off the rails, stability loss at exit or motor burning on the launch pad.	**Inspection:** Before launch, check that the rails are not blocked and the rail buttons are perfectly aligned and run smoothly through the rail. **Maintenance:** Keep the rails and buttons clean and without rust.
Delayed ignition	Poor contact between igniter and grain or weak launch battery.	The motor ignites several seconds after the button is pressed, or fails to ignite fully, creating a severe safety hazard for recovery crews approaching the pad.	**Installation:** Ensure the igniter is pushed all the way to the forward section of the motor. **Safety protocol:** In the event of a misfire, enforce a mandatory wait time before anyone approaches the pad, disconnecting the launch battery first.
Avionics power failure	Loose connections due to vibrations.	Altimeter turns off mid-flight.	**Mechanical security:** Secure avionics to the

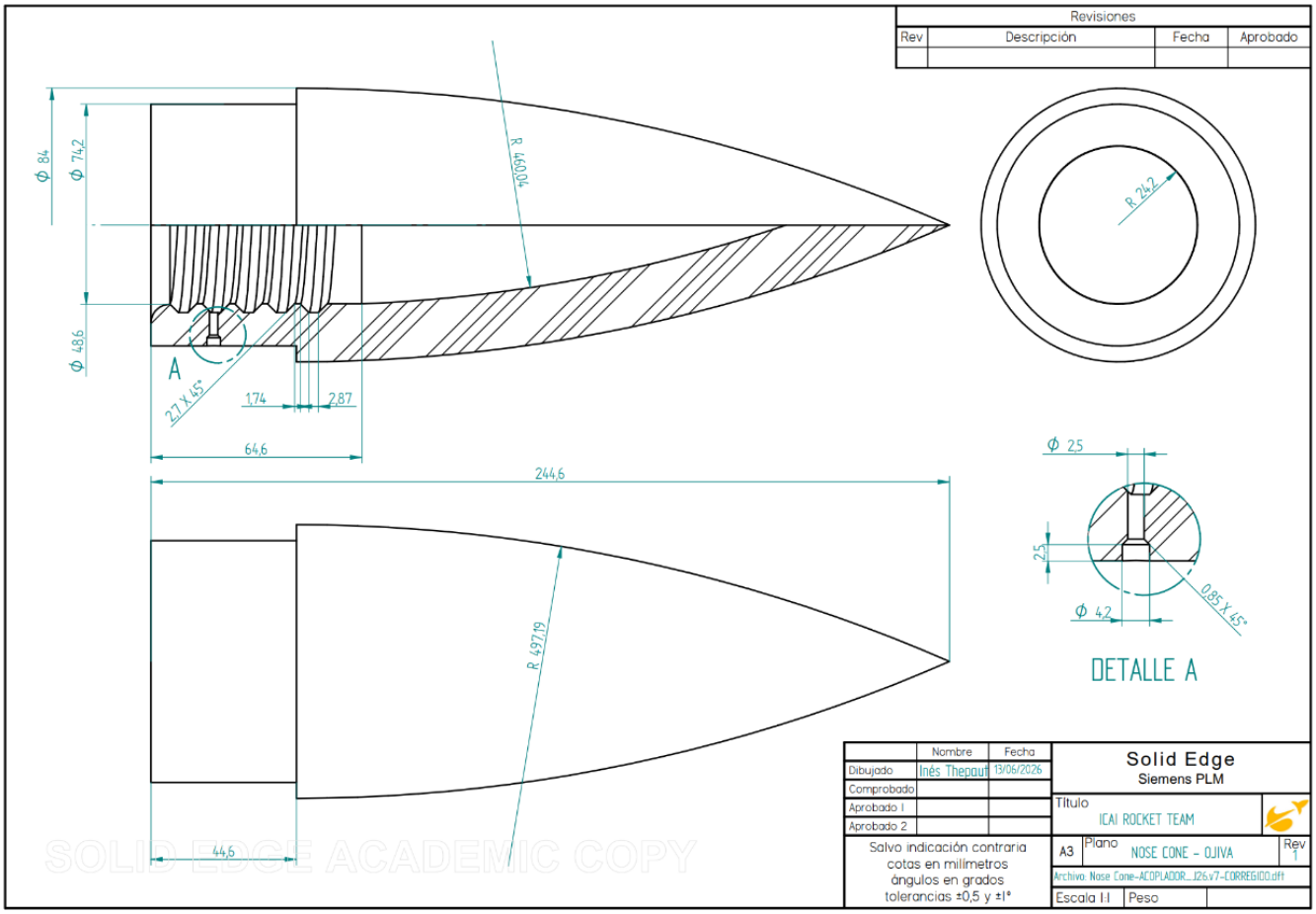
			body with zip-ties or heavy-duty screws. **Inspection:** Before launch, inspect that every connection is secured.
Recovery harness failure	Shock cord snaps due to high deployment forces, using brittle material, or damage from hot ejection charge gases.	The parachute detaches completely from the rocket body. The rocket sections fall with no control causing a dangerous free-fall hazard.	**Materials:** Use strong nylon or Kevlar cord. **Thermal protection:** Wrap the sock cord in the fireproof sleeve to prevent heat damage.

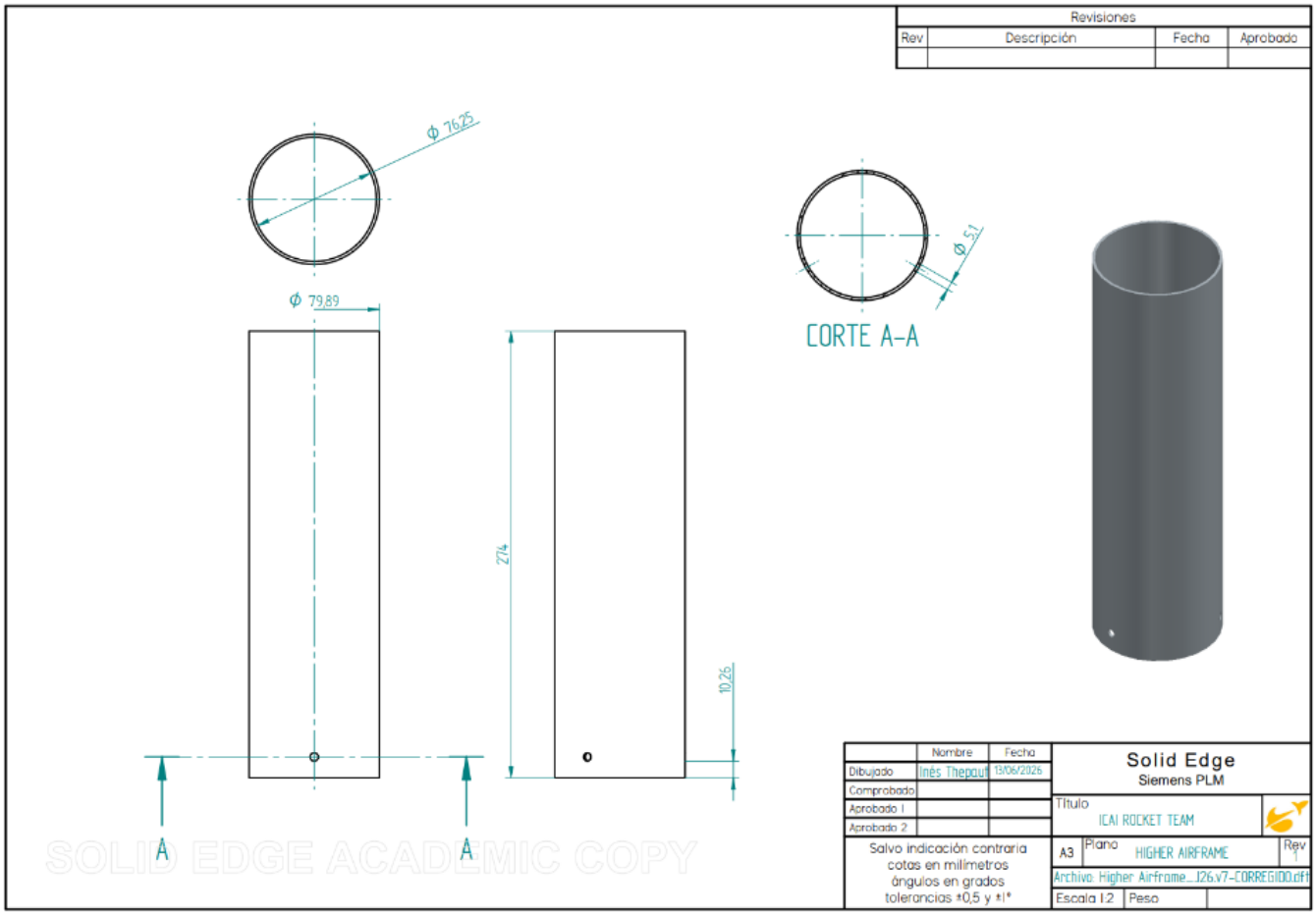
VII. Engineering Drawings

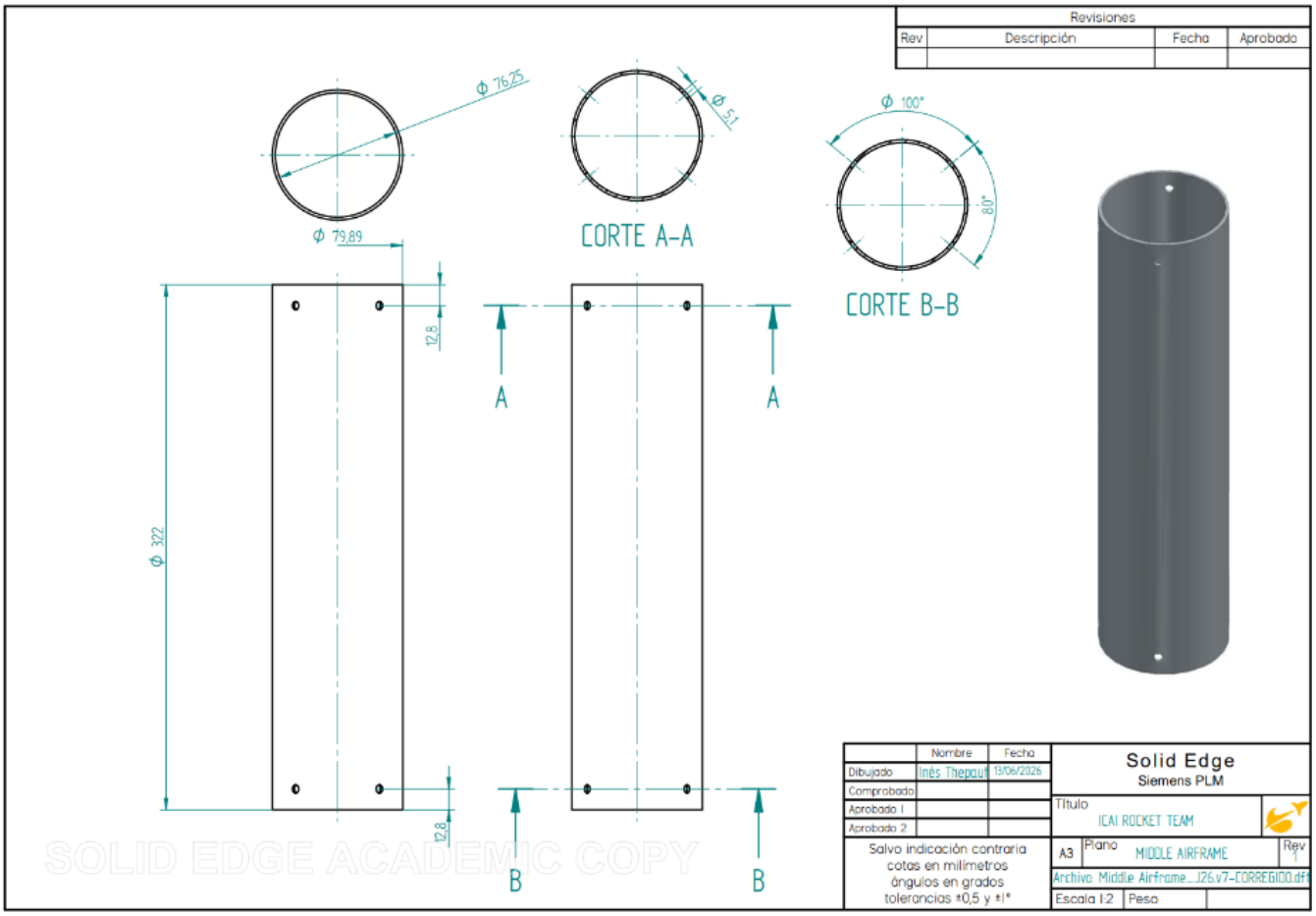
This appendix contains all the technical sheets and blueprints for Dédalo, which were designed and verified using Solid Edge (Siemens PLM). Unless specified otherwise, all component dimensions follow a standardized engineering layout written strictly in millimetres with a general geometric tolerance applied to linear dimensions and angular measurements. The drawings are organized to present clear 2D views for workshop fabrication, internal cross-sections to illustrate how different segments connect, and 3D isometric models to provide a clean visual reference of the components.

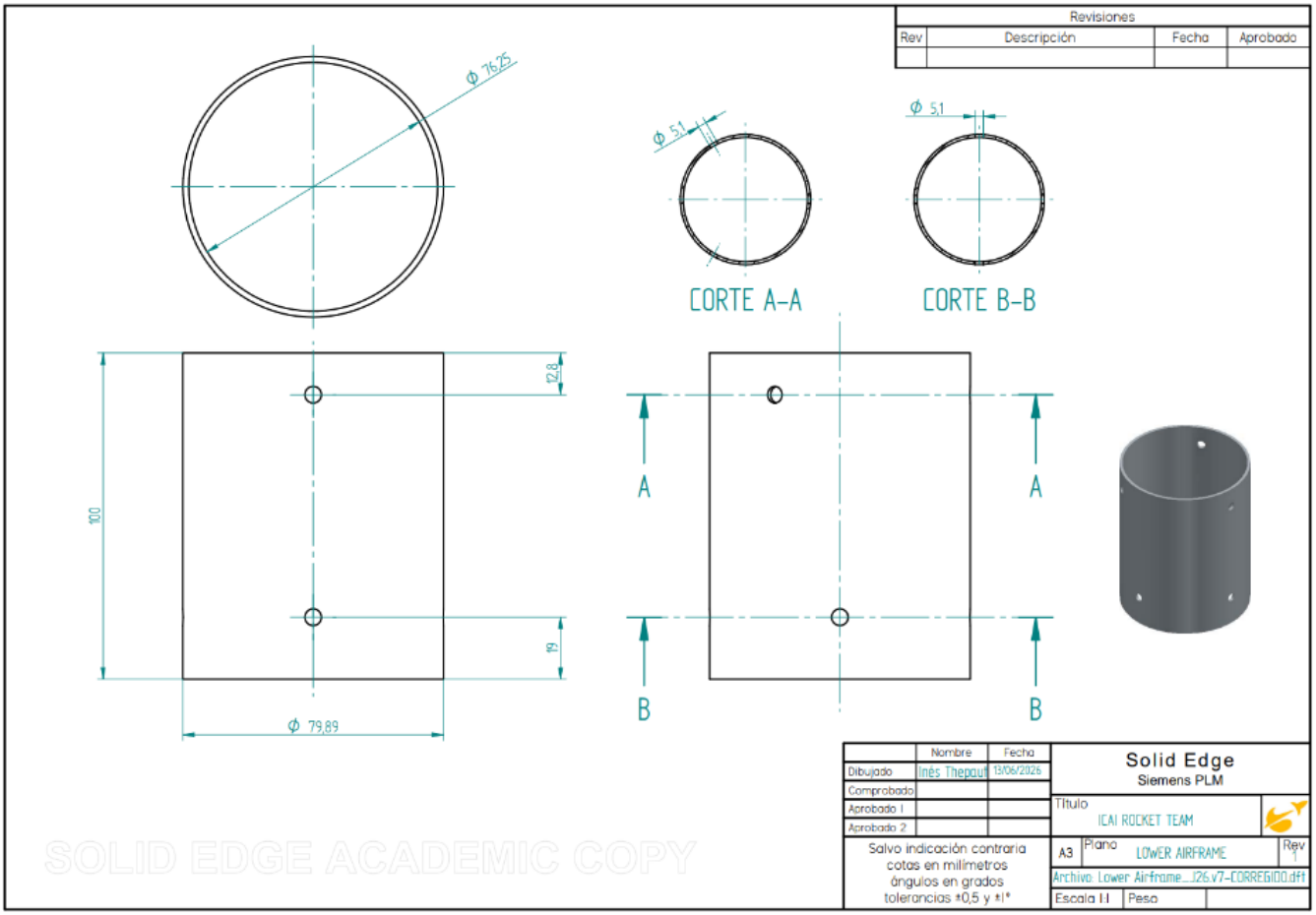
The first group of schematics details the external aerodynamic and structural components, including the ogive nose cone with its base shoulder, the higher airframe tube with radial alignment holes, the central middle airframe coupler, and the lower airframe section. To ensure aerodynamic stability, the prints include the trapezoidal fin geometry featuring independent root tabs designed to slide into the matching side slots of the main motor airframe body, which is securely locked at the top by a threaded cap. The remaining drawings focus on the internal propulsion alignment, recovery mechanisms, and avionics. This includes an inner cardboard protective liner tube and a centering ring to align the engine assembly, an internal compartment structure that tracks and houses the flat modular avionics sled for electronics staging, and a set of internal slot discs, thin interlocking fixing rings, and a thick coupling disc bulkhead acting as the main anchor point for the recovery shock cord.

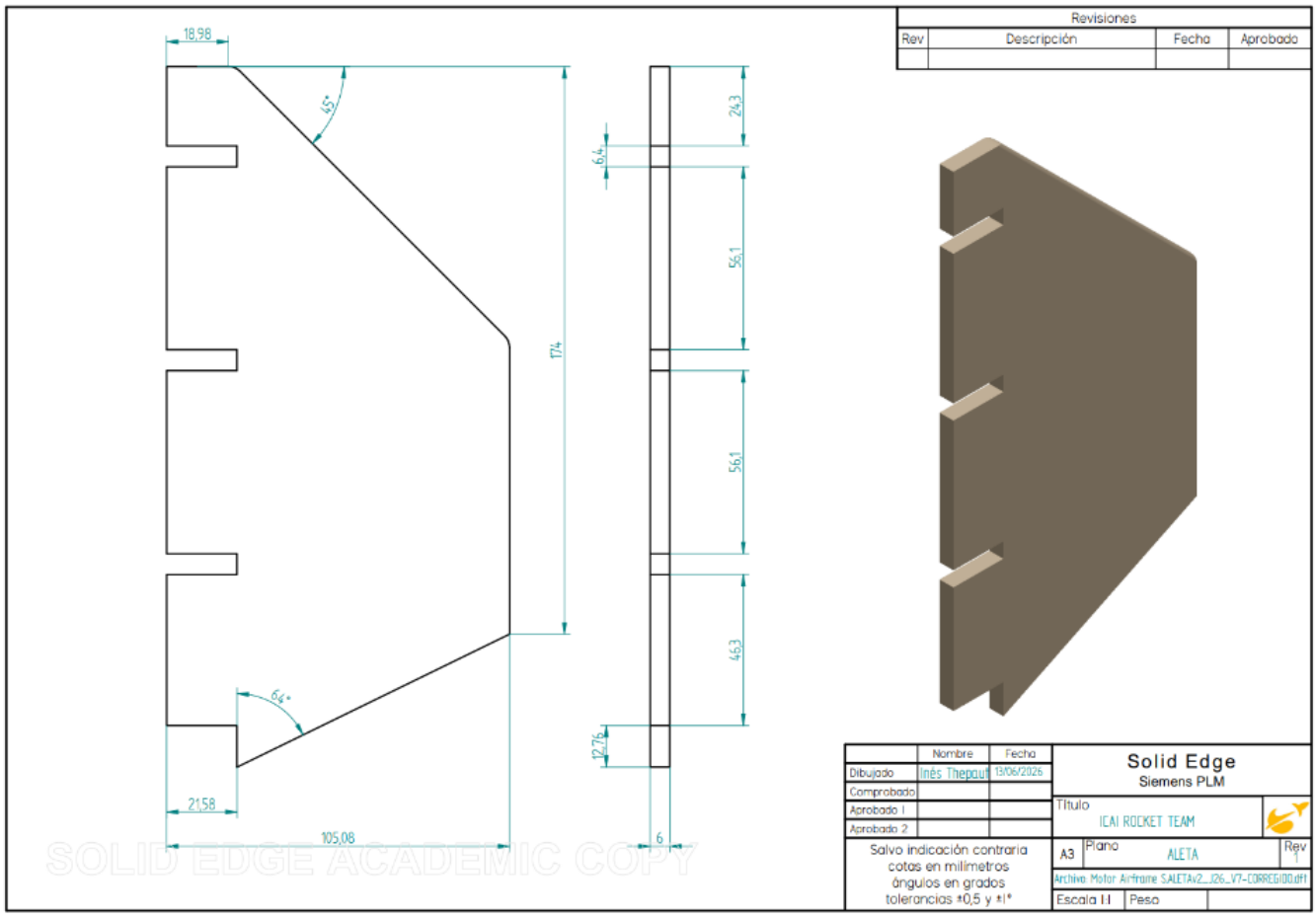

Revisiones
Rev
Descripción
Fecha
Aprobado
Ø 74,2
Ø 53,4
Ø 48
Ø 30
A
A
70,8
6
R 2
2,7 X 45°
2,7
1,9
R 2
6
R 6
R 6
CORTE A-A
Ø 63
Rounds R2
Nombre
Fecha
Dibujado
Inés Thepaut
13/06/2026
Comprobado
Aprobado 1
Aprobado 2
Solid Edge
Siemens PLM
Título
ICAI ROCKET TEAM
Salvo indicación contraria
cotas en milímetros
ángulos en grados
tolerancias ±0,5 y ±1°
A3
Plano
NOSE CONE - ACOPLADOR
Rev
1
Archivo: Nose Cone-OJIVA_J26.v7-CORREGIDO.dft
Escala 1:1
Peso
SOLID EDGE ACADEMIC COPY

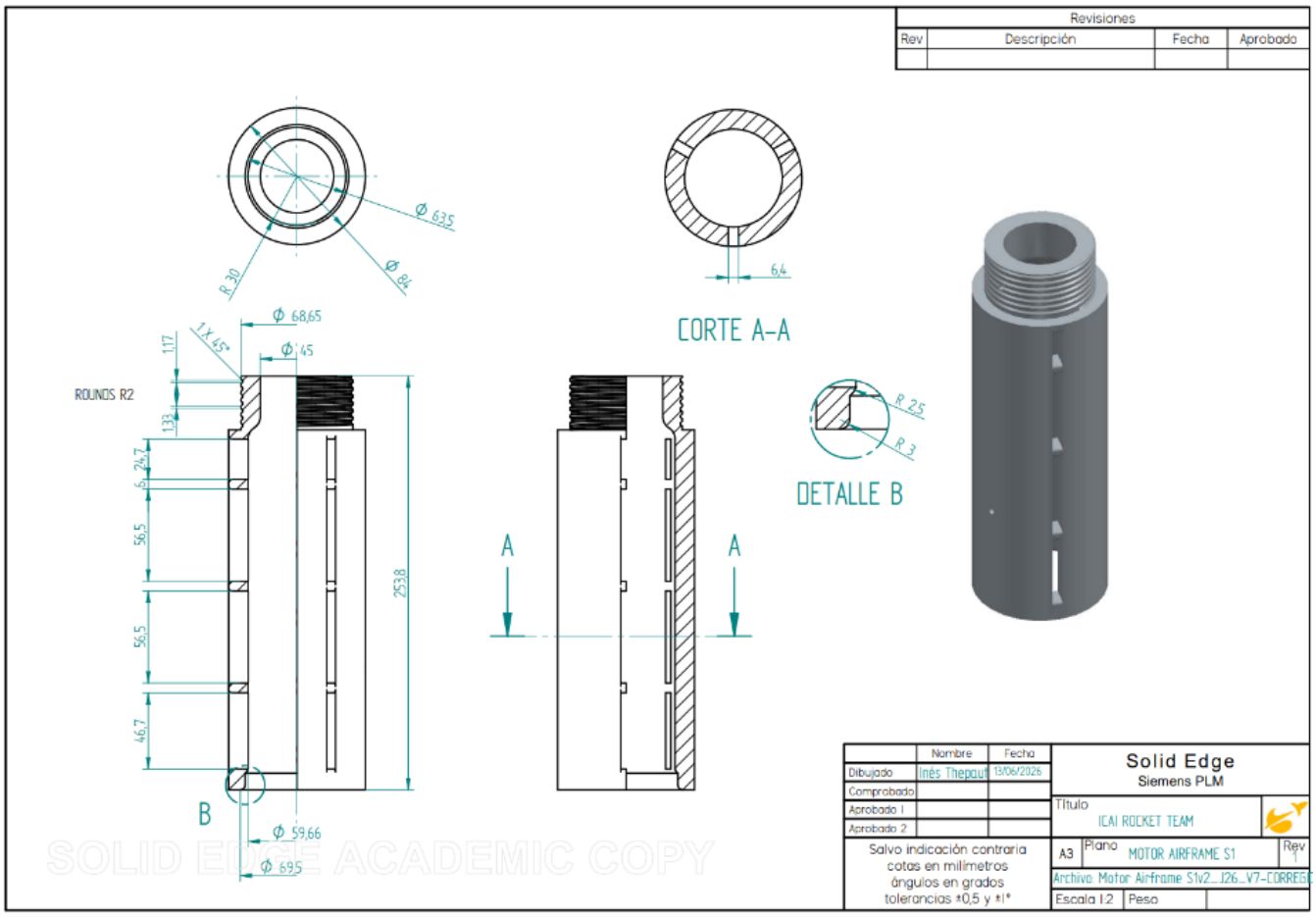

Revisiones
Rev
Descripción
Fecha
Aprobado
Ø 84
Ø 74,2
Ø 48,6
R 460,04
A
2,7 X 45°
1,74
2,87
64,6
244,6
R 497,19
44,6
R 24,2
Ø 2,5
2,5
Ø 4,2
0,85 X 45°
DETALLE A
Nombre
Fecha
Dibujado
Inés Thepaut
13/06/2026
Comprobado
Aprobado 1
Aprobado 2
Solid Edge
Siemens PLM
Título
ICAI ROCKET TEAM
Salvo indicación contraria
cotas en milímetros
ángulos en grados
tolerancias ±0,5 y ±1°
A3
Plano
NOSE CONE - OJIVA
Rev
1
Archivo: Nose Cone-ACOPLADOR_J26.v7-CORREGIDO.dft
Escala 1:1
Peso
SOLID EDGE ACADEMIC COPY

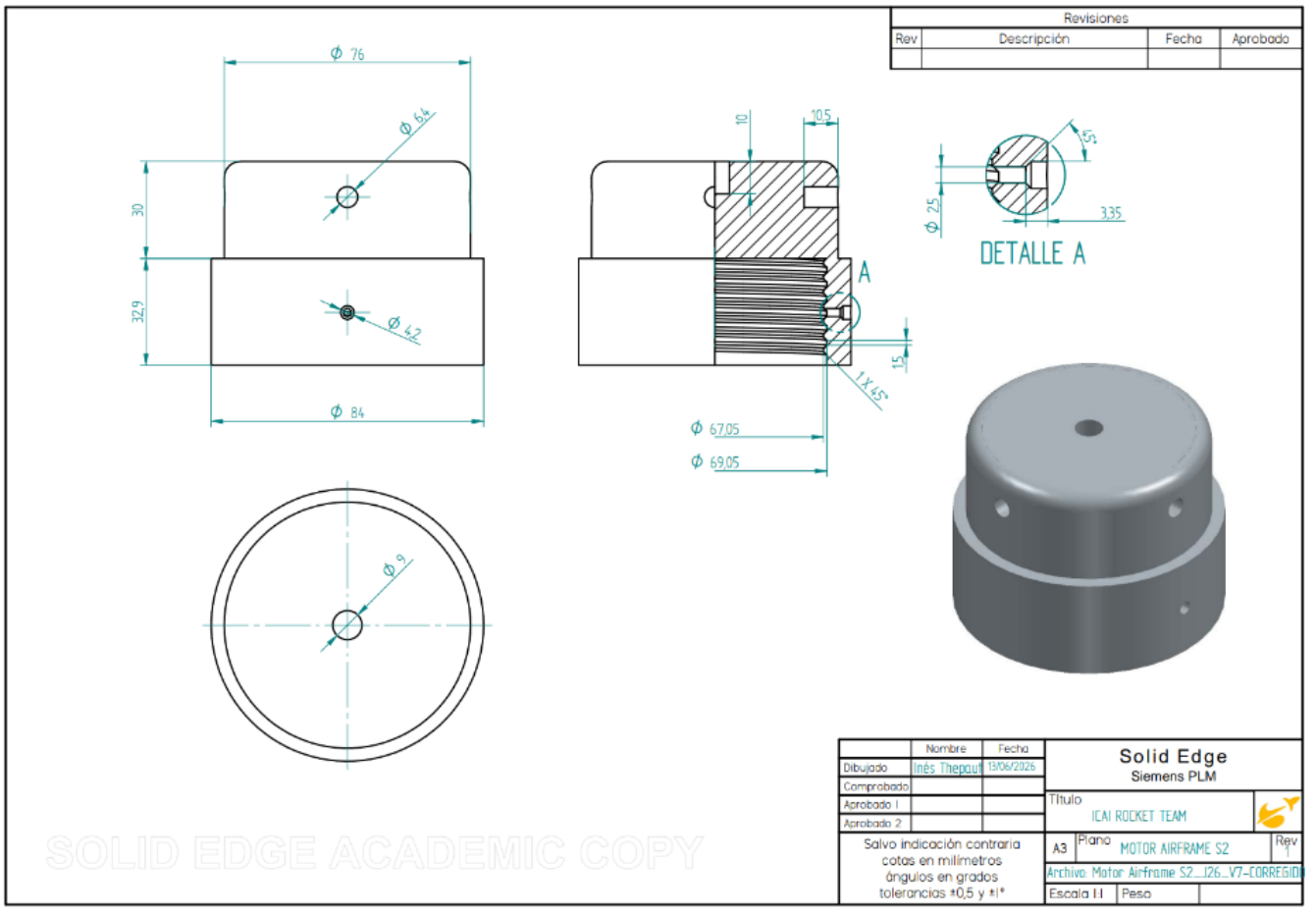

Revisiones
Rev
Descripción
Fecha
Aprobado
ϕ 76,25
ϕ 79,89
ϕ 5,1
CORTE A-A
274
10,26
A
A
Nombre
Fecha
Dibujado
Inés Thepaut
13/06/2026
Comprobado
Aprobado 1
Aprobado 2
Solid Edge
Siemens PLM
TItulo
ICAI ROCKET TEAM
A3
Plano
HIGHER AIRFRAME
Rev
Archivo: Higher Airframe...126.v7-CORREGIDO.dft
Escala 1:2
Peso
Salvo indicación contraria
cotas en milímetros
ángulos en grados
tolerancias ±0,5 y ±1°
SOLID EDGE ACADEMIC COPY

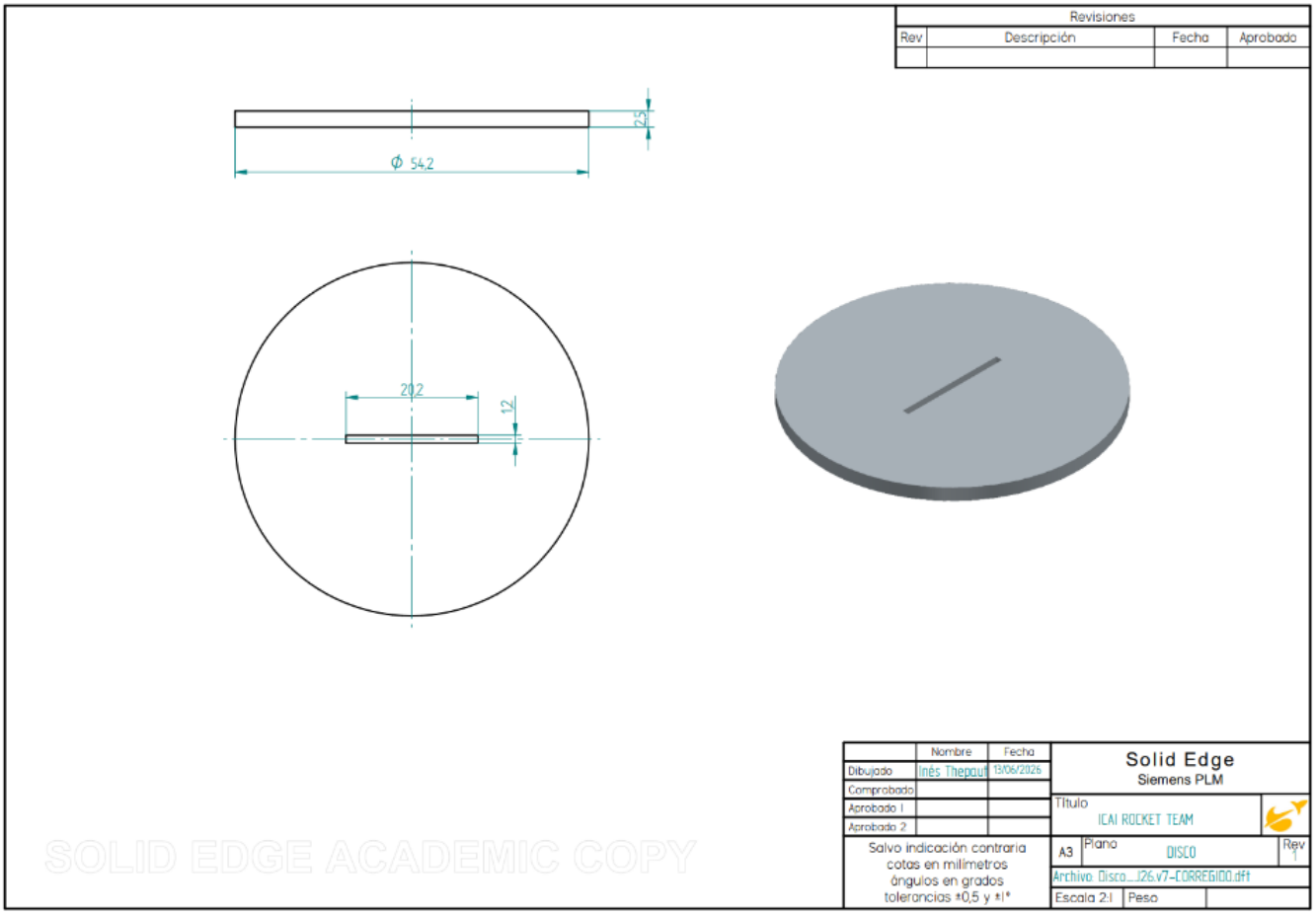

Revisiones
Rev
Descripción
Fecha
Aprobado
ϕ 76,25
ϕ 5,1
ϕ 100°
80°
ϕ 79,89
CORTE A-A
CORTE B-B
12,8
12,8
ϕ 322
A
A
B
B
Nombre
Fecha
Dibujado
Inés Thepaut
13/06/2026
Comprobado
Aprobado 1
Aprobado 2
Solid Edge
Siemens PLM
TItulo
ICAI ROCKET TEAM
A3
Plano
MIDDLE AIRFRAME
Rev
Archivo: Middle Airframe...126.v7-CORREGIDO.dft
Escala 1:2
Peso
Salvo indicación contraria
cotas en milímetros
ángulos en grados
tolerancias ±0,5 y ±1°
SOLID EDGE ACADEMIC COPY

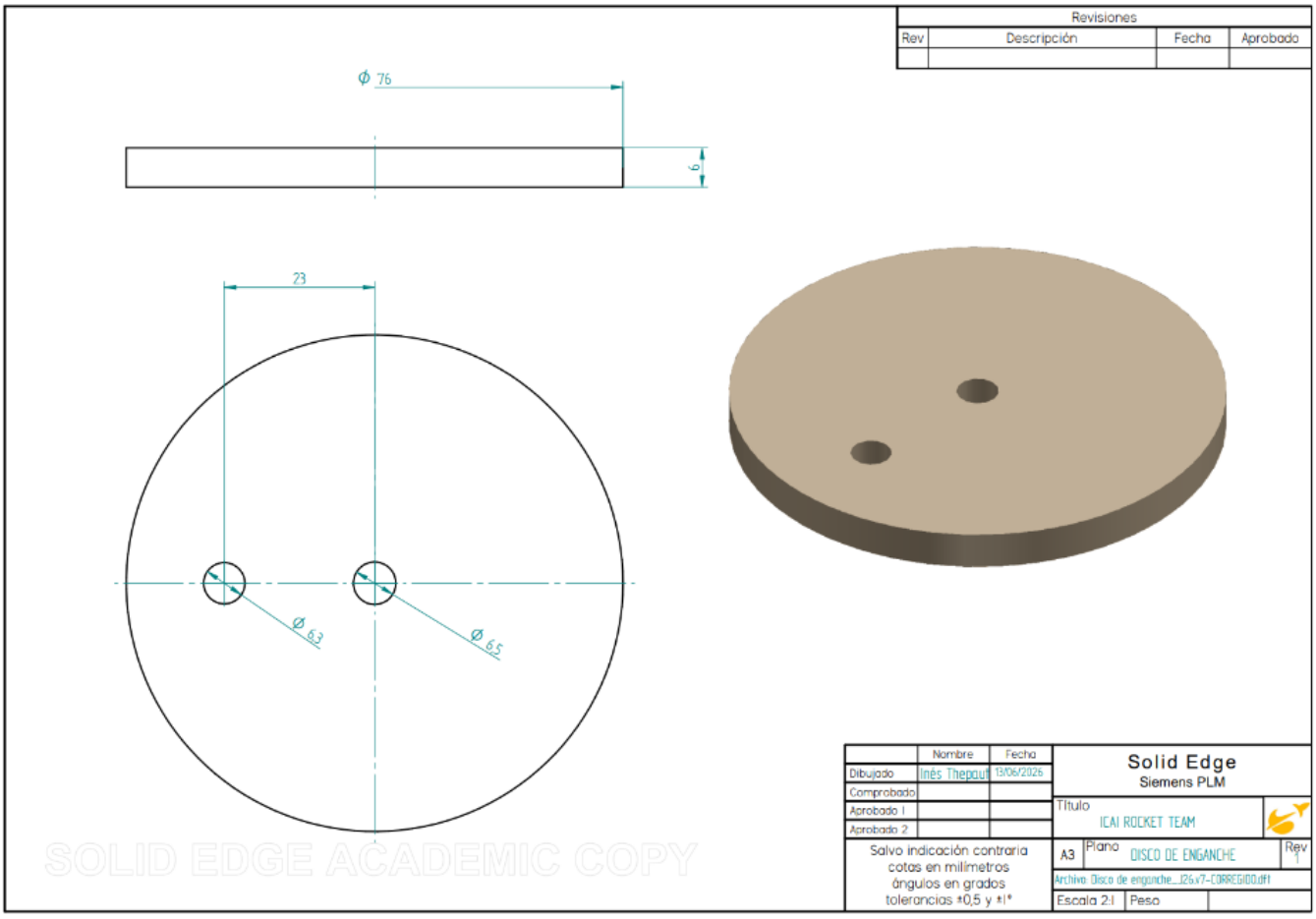

Revisiones
Rev
Descripción
Fecha
Aprobado
Ø 76,25
Ø 5,1
Ø 5,1
CORTE A-A
CORTE B-B
12,8
100
19
Ø 79,89
A
A
B
B
Nombre
Fecha
Dibujado
Inés Thepaut
13/06/2026
Comprobado
Aprobado 1
Aprobado 2
Solid Edge
Siemens PLM
TItulo
ICAI ROCKET TEAM
Salvo indicación contraria
cotas en milímetros
ángulos en grados
tolerancias ±0,5 y ±1°
A3
Plano
LOWER AIRFRAME
Rev
Archivo: Lower Airframe...I26.V7-CORREGIDO.dft
Escala 1:1
Peso
SOLID EDGE ACADEMIC COPY

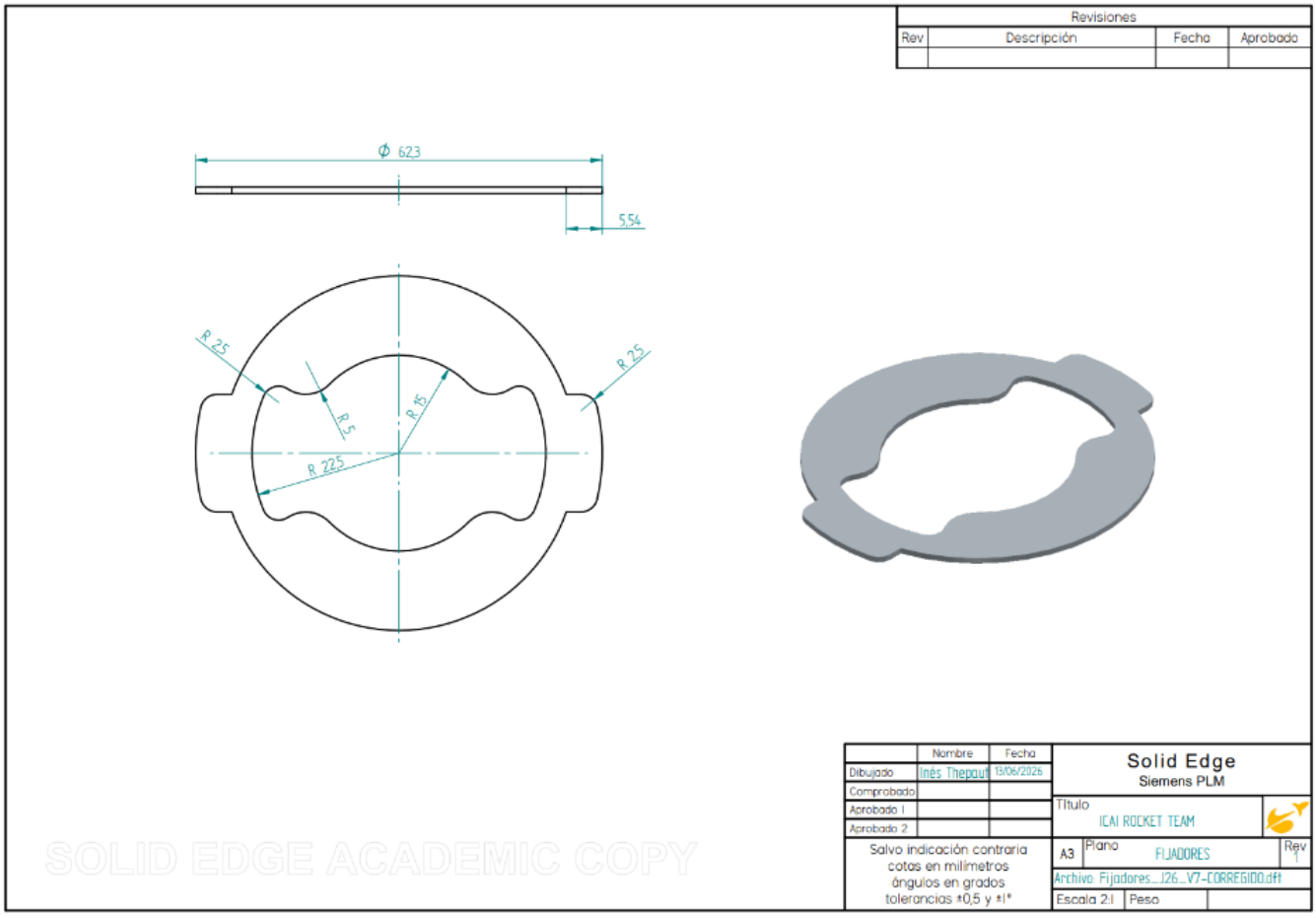

Revisiones
Rev
Descripción
Fecha
Aprobado
18,98
45°
174
64°
21,58
105,08
24,3
6,4
56,1
56,1
46,3
12,76
6
Nombre
Fecha
Dibujado
Inés Thepaut
13/06/2026
Comprobado
Aprobado 1
Aprobado 2
Solid Edge
Siemens PLM
TItulo
ICAI ROCKET TEAM
Salvo indicación contraria
cotas en milímetros
ángulos en grados
tolerancias ±0,5 y ±1°
A3
Plano
ALETA
Rev
Archivo: Motor Airframe SALETAv2...I26...V7-CORREGIDO.dft
Escala 1:1
Peso
SOLID EDGE ACADEMIC COPY

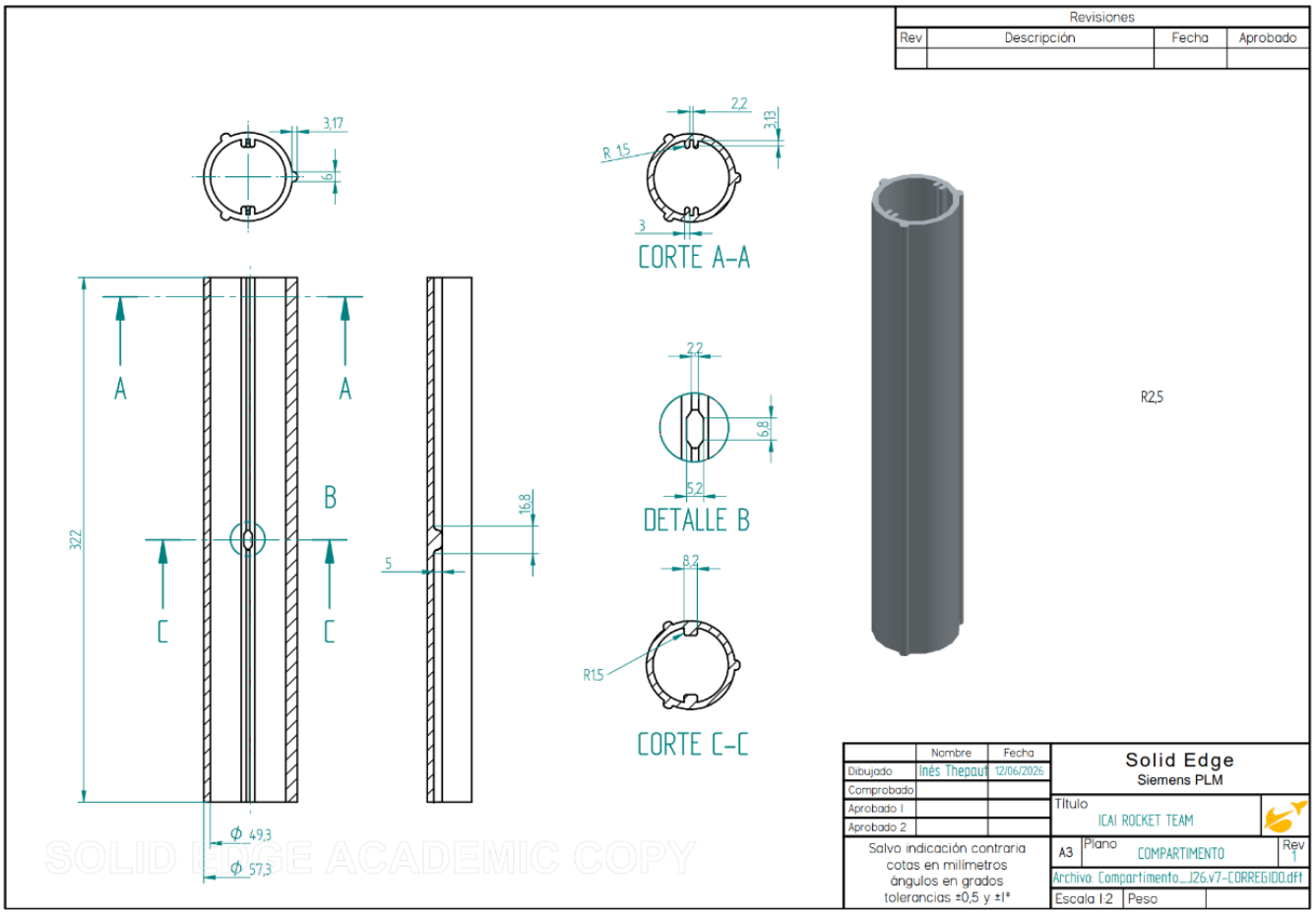

Revisiones
Rev | Descripción | Fecha | Aprobado
Ø 63,5
R 30
Ø 84
Ø 68,65
1,17
1X 45°
Ø 45
ROUNDS R2
13,2
6
24,7
56,5
56,5
46,7
253,8
B
Ø 59,66
Ø 69,5
CORTE A-A
6,4
A
A
R 2,5
R 3
DETALLE B
Nombre | Fecha
Dibujado | Inés Thepaut | 13/06/2026
Comprobado
Aprobado 1
Aprobado 2
Salvo indicación contraria cotas en milímetros ángulos en grados tolerancias ±0,5 y ±1°
Solid Edge
Siemens PLM
Título
ICAI ROCKET TEAM
A3
Plano
MOTOR AIRFRAME S1
Rev
1
Archivo: Motor Airframe S1v2_126_V7-CORREGI
Escala 1:2
Peso
SOLID EDGE ACADEMIC COPY

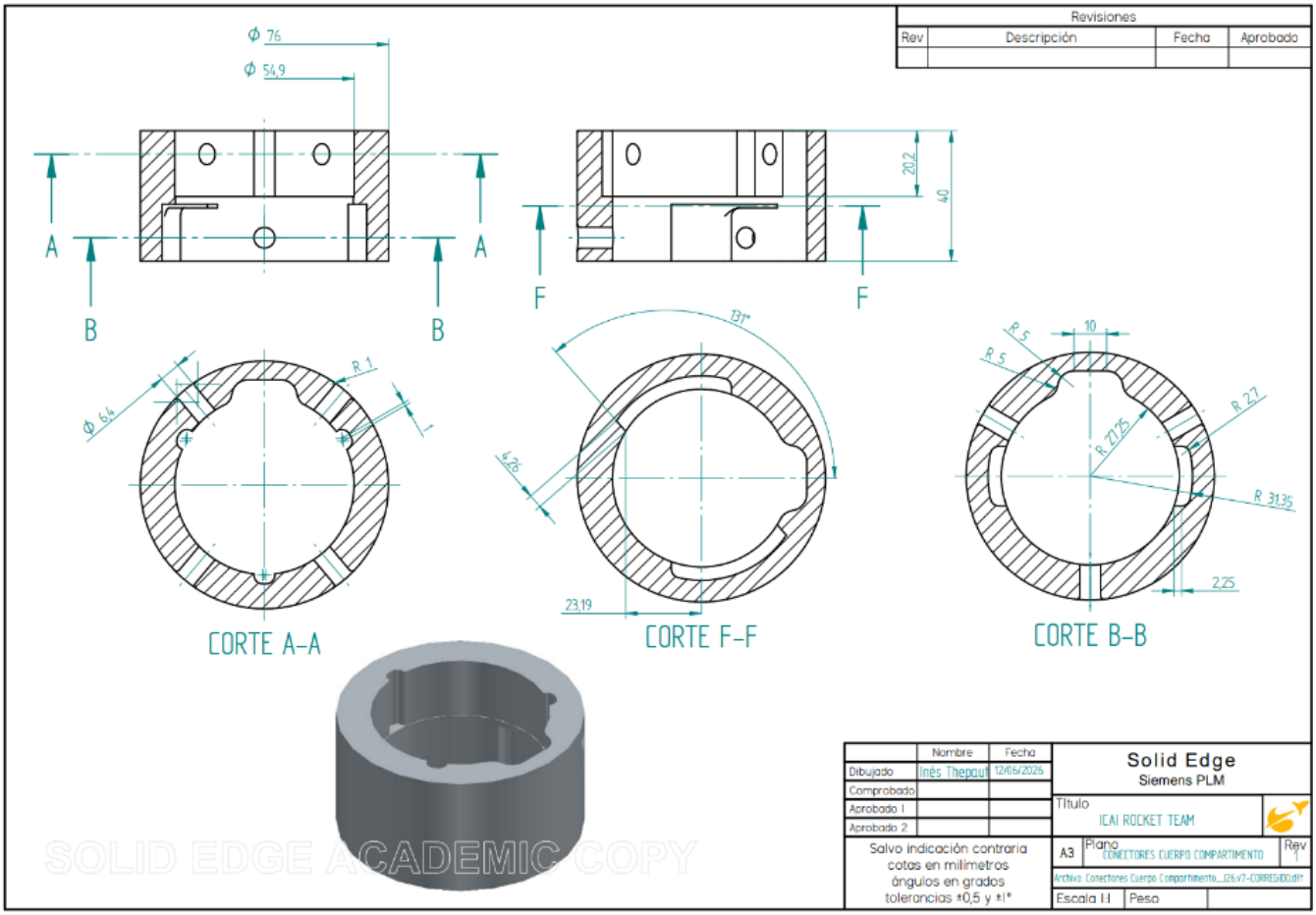

Revisiones
Rev | Descripción | Fecha | Aprobado
Ø 76
Ø 64
30
32,9
Ø 42
Ø 84
Ø 9
10
10,5
A
15
1X 45°
Ø 67,05
Ø 69,05
45°
Ø 25
3,35
DETALLE A
Nombre | Fecha
Dibujado | Inés Thepaut | 13/06/2026
Comprobado
Aprobado 1
Aprobado 2
Salvo indicación contraria cotas en milímetros ángulos en grados tolerancias ±0,5 y ±1°
Solid Edge
Siemens PLM
Título
ICAI ROCKET TEAM
A3
Plano
MOTOR AIRFRAME S2
Rev
1
Archivo: Motor Airframe S2_126_V7-CORREGIDO
Escala 1:1
Peso
SOLID EDGE ACADEMIC COPY

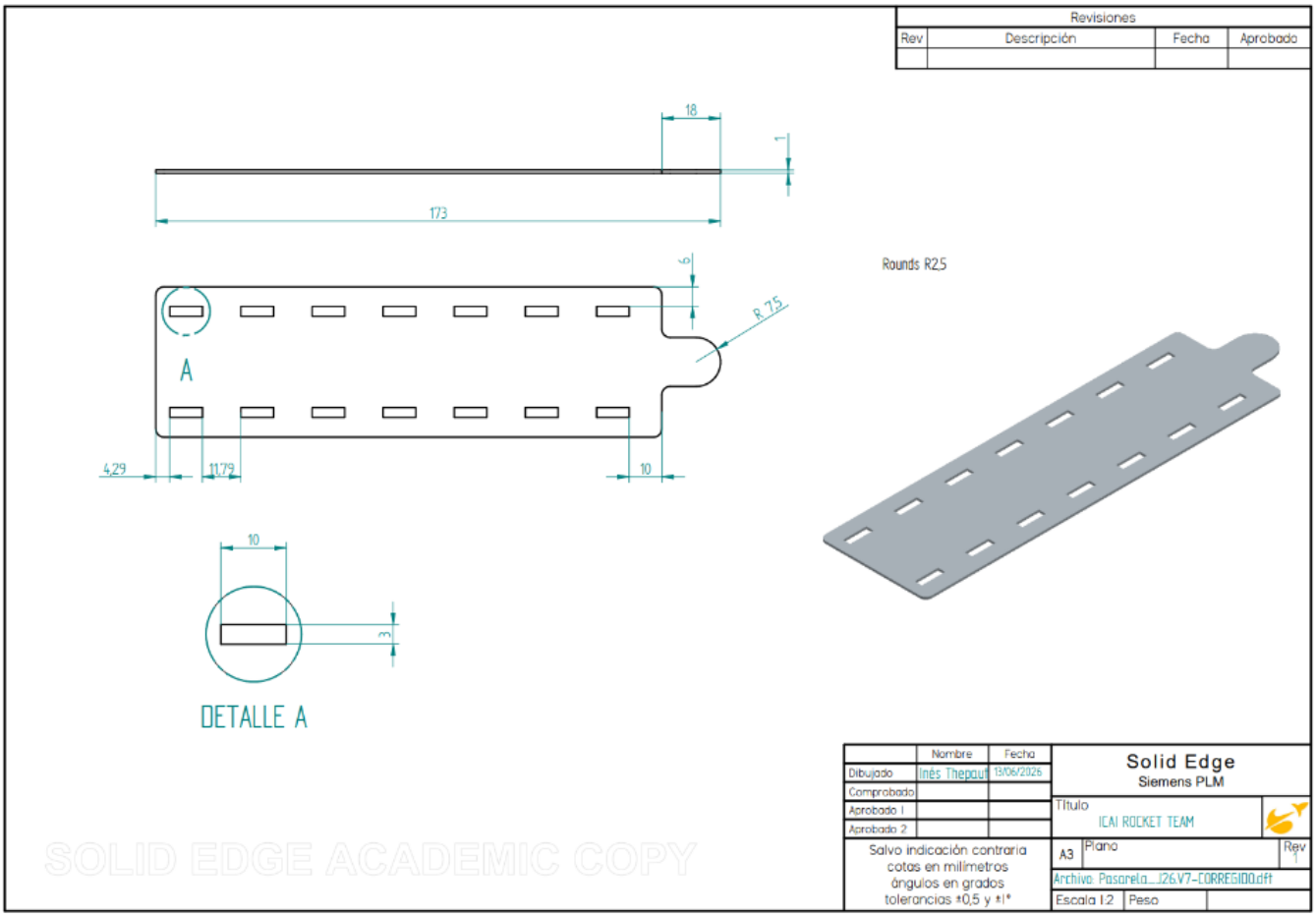

Revisiones
Rev
Descripción
Fecha
Aprobado
25
Ø 54,2
20,2
12
Nombre
Fecha
Dibujado
Inés Thepaut
13/06/2026
Comprobado
Aprobado 1
Aprobado 2
Solid Edge
Siemens PLM
Título
ICAI ROCKET TEAM
Salvo indicación contraria
cotas en milímetros
ángulos en grados
tolerancias ±0,5 y ±1°
A3
Plano
DISCO
Rev
1
Archivo: Disco...126.v7-CORREGIDO.dft
Escala 2:1
Peso
SOLID EDGE ACADEMIC COPY

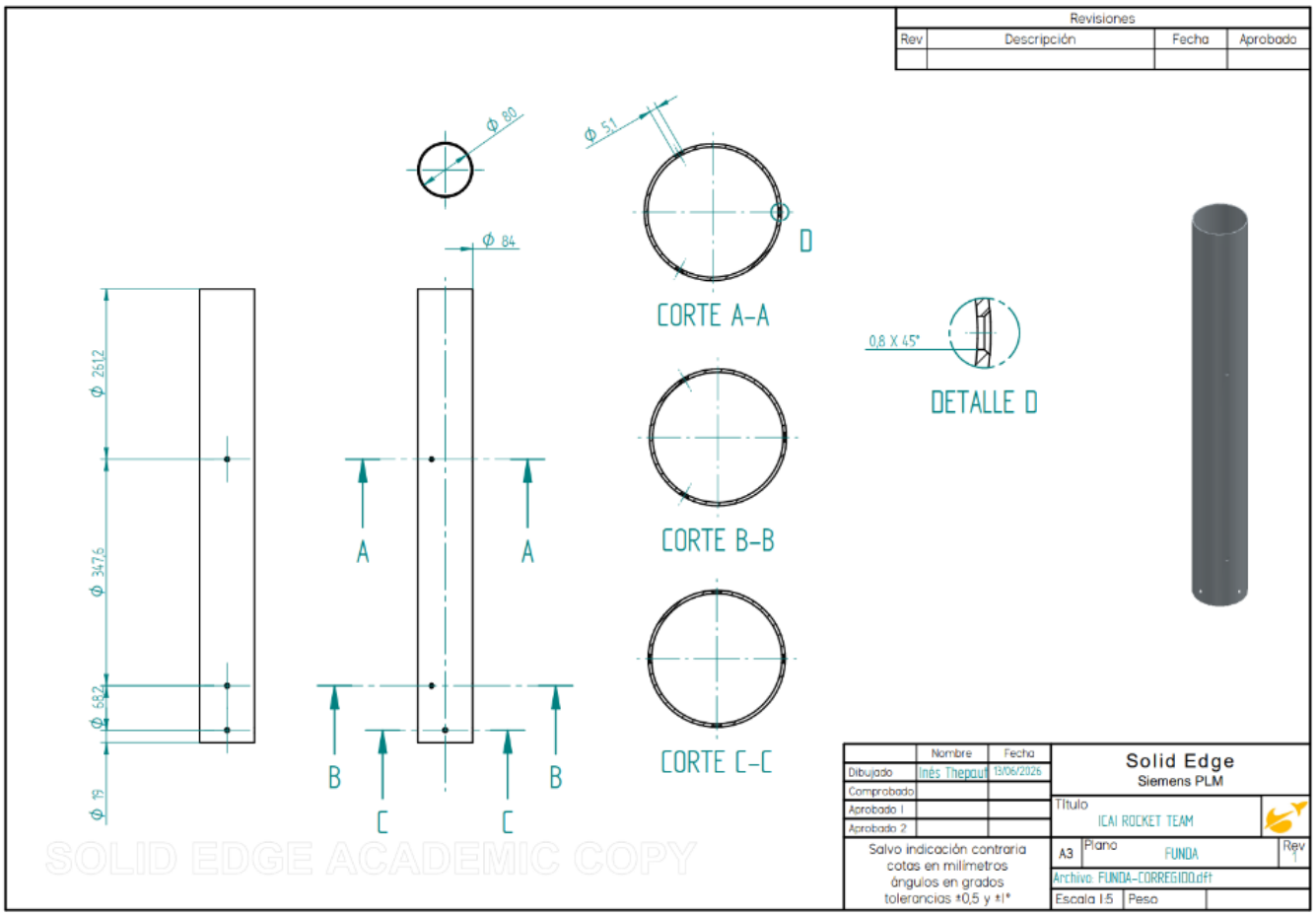

Revisiones
Rev
Descripción
Fecha
Aprobado
Ø 76
6
23
Ø 6,3
Ø 6,5
Nombre
Fecha
Dibujado
Inés Thepaut
13/06/2026
Comprobado
Aprobado 1
Aprobado 2
Solid Edge
Siemens PLM
Título
ICAI ROCKET TEAM
Salvo indicación contraria
cotas en milímetros
ángulos en grados
tolerancias ±0,5 y ±1°
A3
Plano
DISCO DE ENGANCHE
Rev
1
Archivo: Disco de enganche...126.v7-CORREGIDO.dft
Escala 2:1
Peso
SOLID EDGE ACADEMIC COPY


Revisiones
Rev
Descripción
Fecha
Aprobado
Ø 623
5,54
R 25
R 5
R 15
R 225
R 25
Nombre
Fecha
Dibujado
Inés Thepaut
13/06/2026
Comprobado
Aprobado 1
Aprobado 2
Solid Edge
Siemens PLM
TItulo
ICAI ROCKET TEAM
A3
Plano
FIJADORES
Rev
Archivo: Fijadores...126...V7-CORREGIDO.dft
Escala 2:1
Peso
Salvo indicación contraria cotas en milímetros ángulos en grados tolerancias ±0,5 y ±1°
SOLID EDGE ACADEMIC COPY

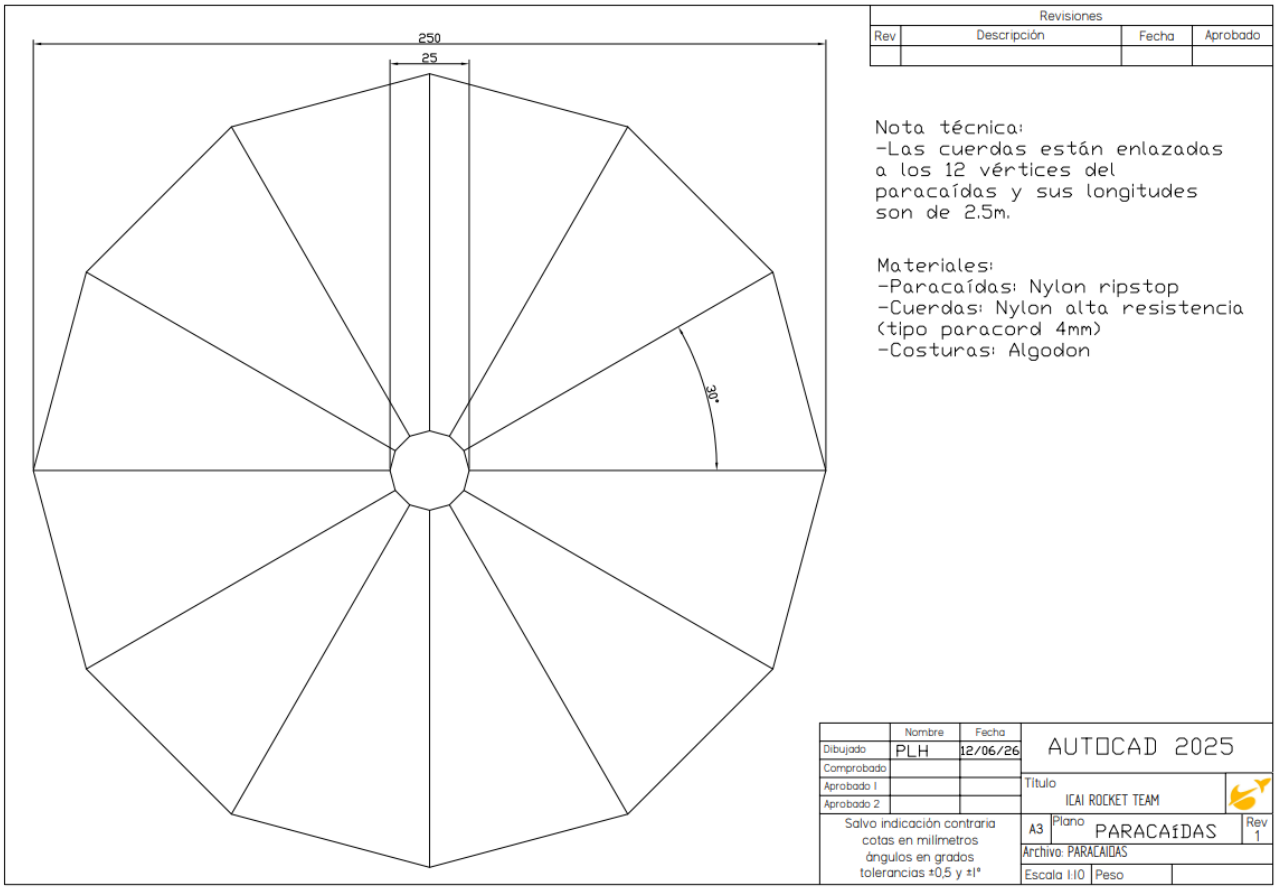

Revisiones
Rev | Descripción | Fecha | Aprobado
CORTE A-A
DETALLE B
CORTE C-C
R2,5
Solid Edge
Siemens PLM
Título
ICAI ROCKET TEAM
Plano
COMPARTIMENTO
Escala 1:2
Peso
SOLID EDGE ACADEMIC COPY

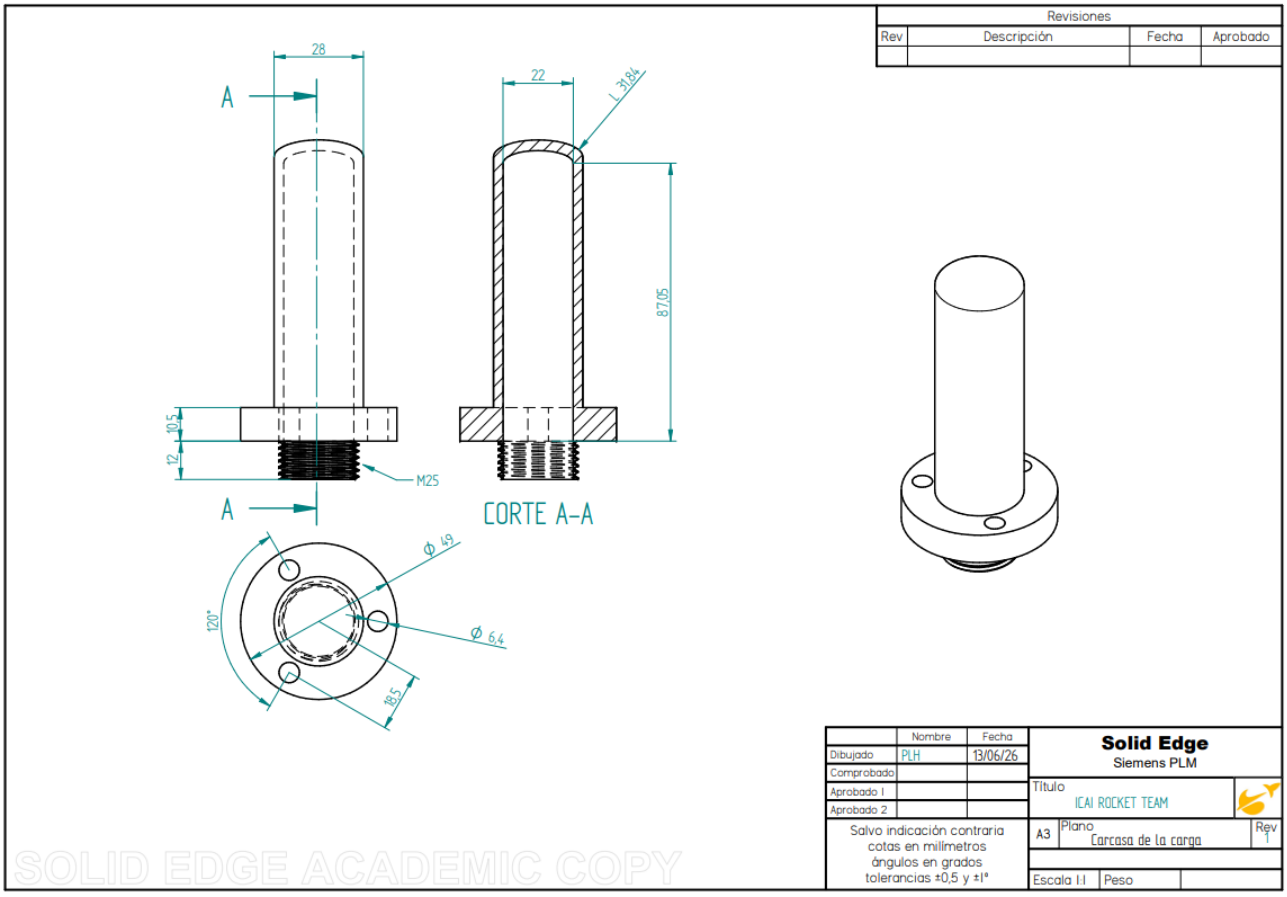

Revisiones
Rev | Descripción | Fecha | Aprobado
CORTE A-A
CORTE F-F
CORTE B-B
Solid Edge
Siemens PLM
Título
ICAI ROCKET TEAM
Plano
CONECTORES CUERPO COMPARTIMENTO
Escala 1:1
Peso
SOLID EDGE ACADEMIC COPY

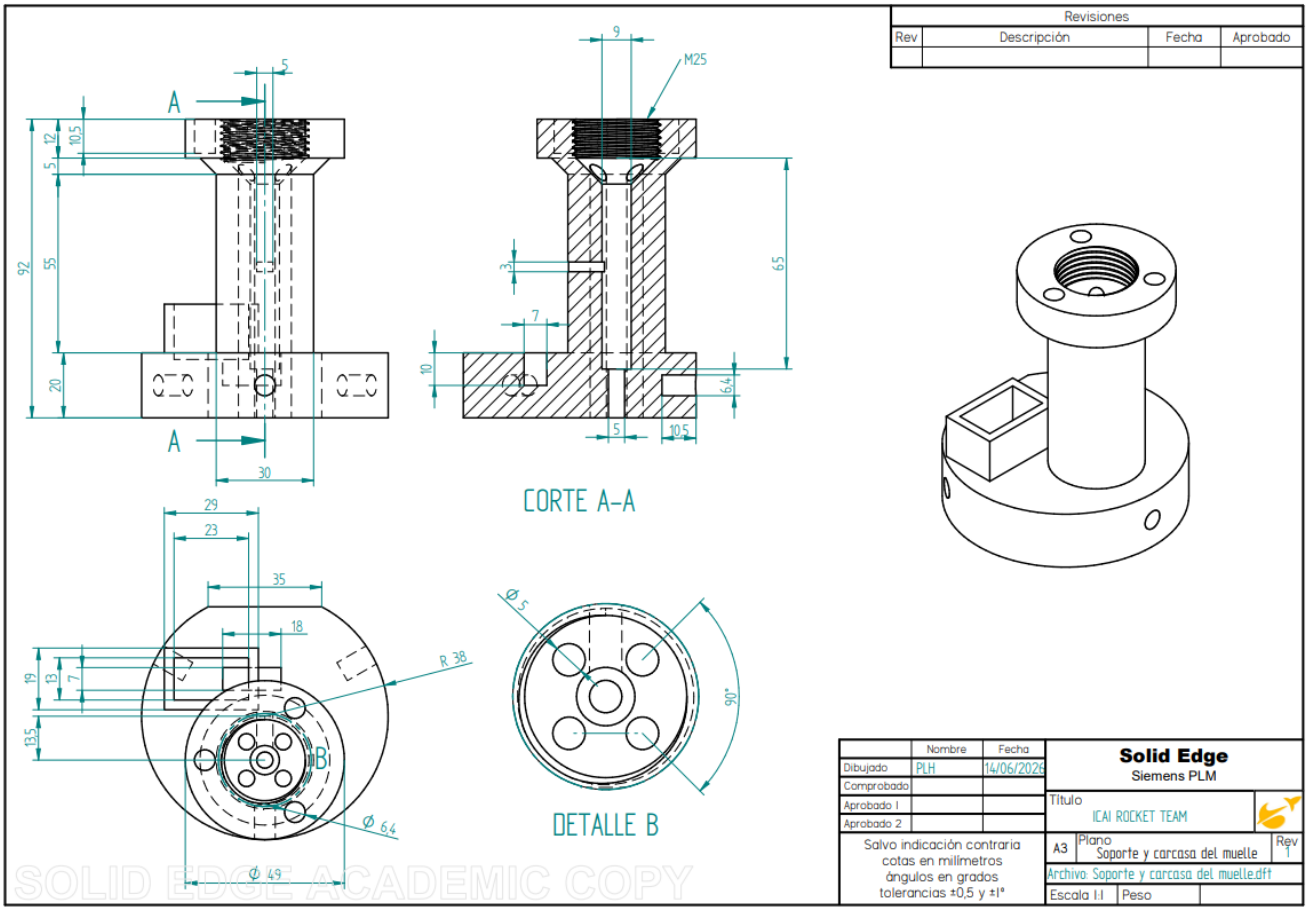

Revisiones
Rev | Descripción | Fecha | Aprobado
18
1
173
6
R 7.5
A
4.29
11.79
10
Rounds R2.5
10
3
DETALLE A
Nombre | Fecha
Dibujado Inés Thepaut 13/06/2026
Comprobado
Aprobado 1
Aprobado 2
Solid Edge
Siemens PLM
Título
ICAI ROCKET TEAM
A3
Plano
Rev
Archivo: Pasarela...126.V7-CORREGIDO.dft
Escala 1:2
Peso
Salvo indicación contraria cotas en milímetros ángulos en grados tolerancias ±0,5 y ±1°
SOLID EDGE ACADEMIC COPY

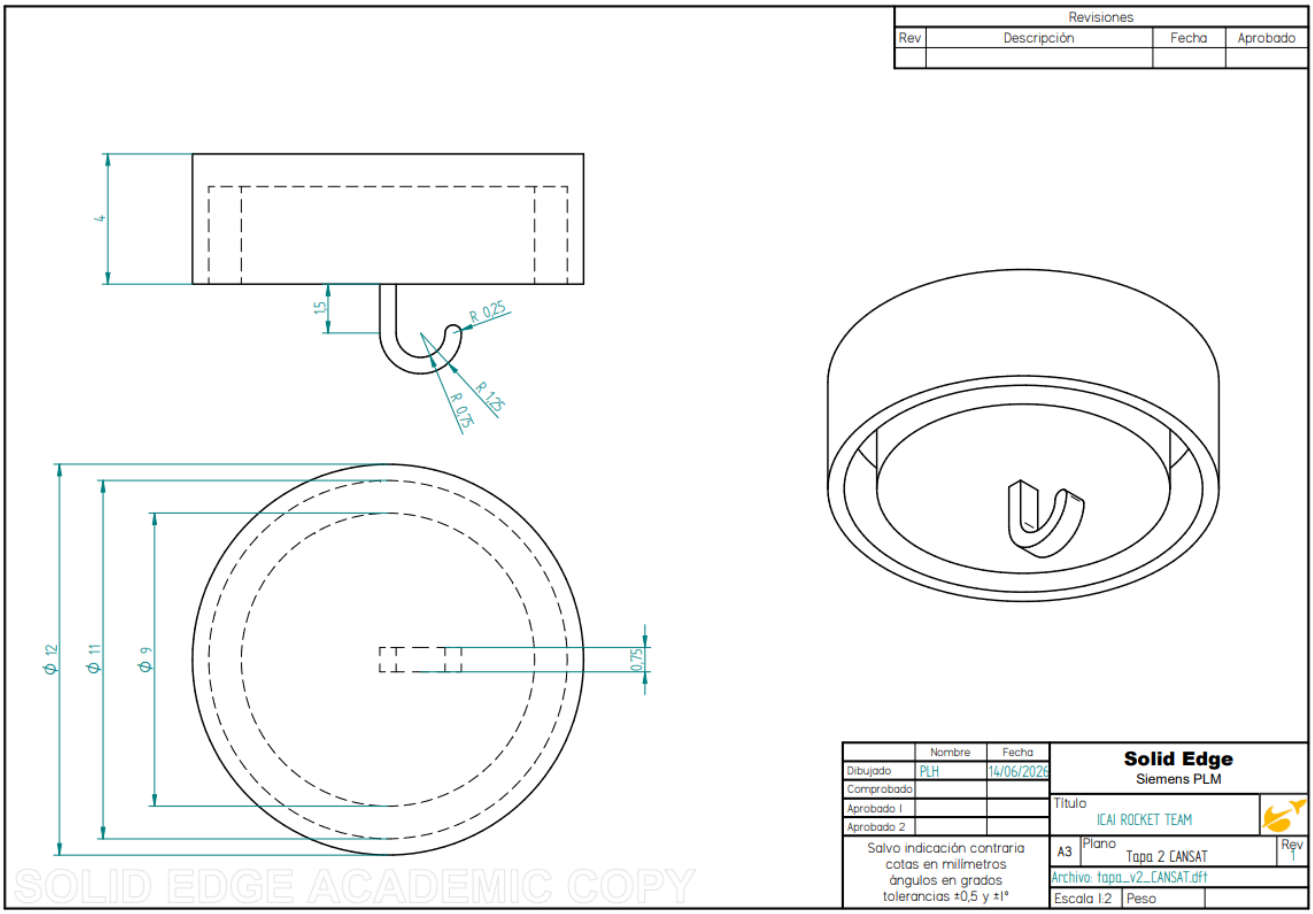

Revisiones
Rev | Descripción | Fecha | Aprobado
Ø 80
Ø 84
Ø 51
D
CORTE A-A
0,8 X 45°
DETALLE D
Ø 261.2
Ø 347.6
Ø 68.2
Ø 19
A
A
CORTE B-B
B
B
C
C
CORTE C-C
Nombre | Fecha
Dibujado Inés Thepaut 13/06/2026
Comprobado
Aprobado 1
Aprobado 2
Solid Edge
Siemens PLM
Título
ICAI ROCKET TEAM
A3
Plano
FUNDA
Rev
Archivo: FUNDA-CORREGIDO.dft
Escala 1:5
Peso
Salvo indicación contraria cotas en milímetros ángulos en grados tolerancias ±0,5 y ±1°
SOLID EDGE ACADEMIC COPY

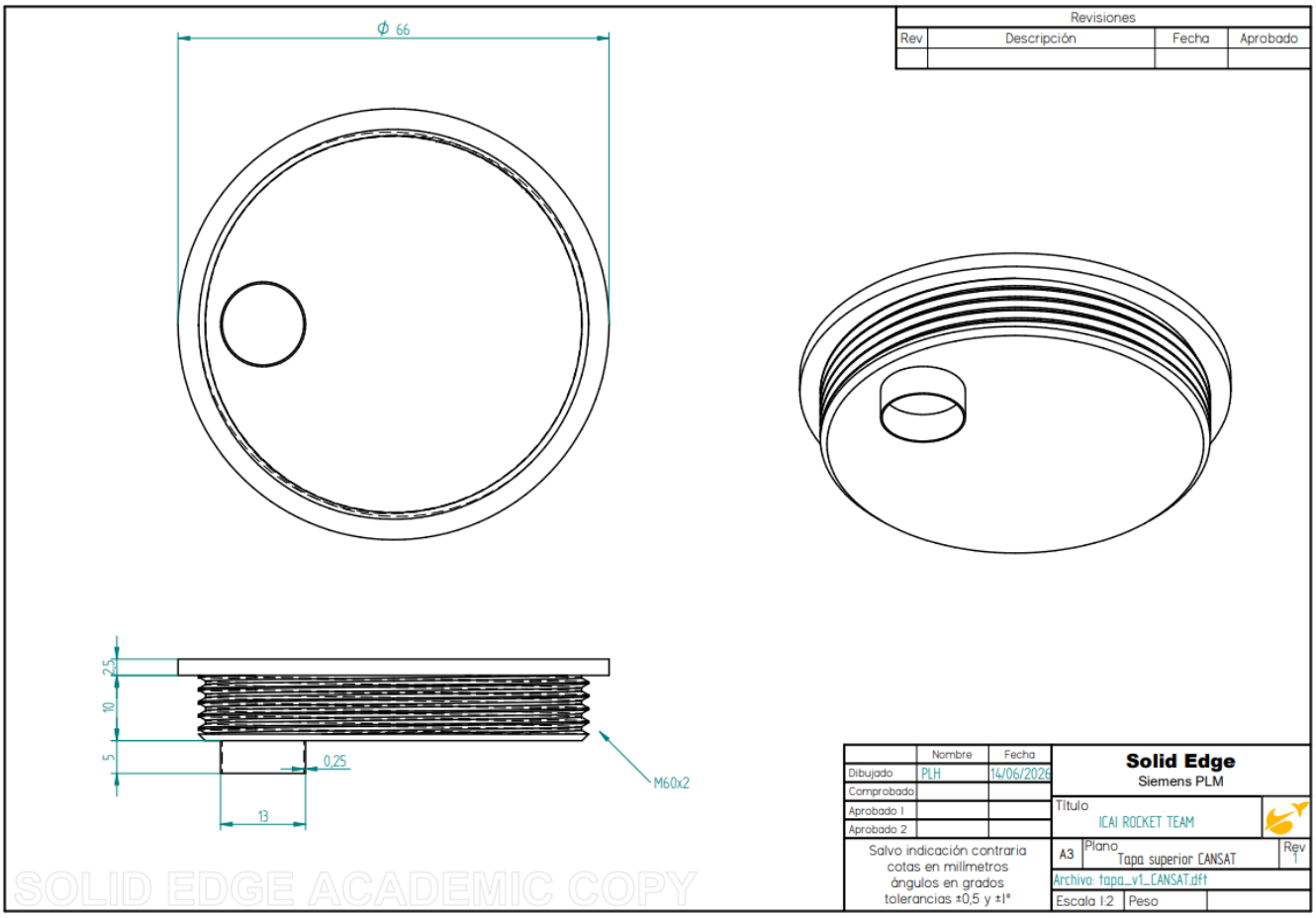
RECOVERY SYSTEMS DRAWINGS

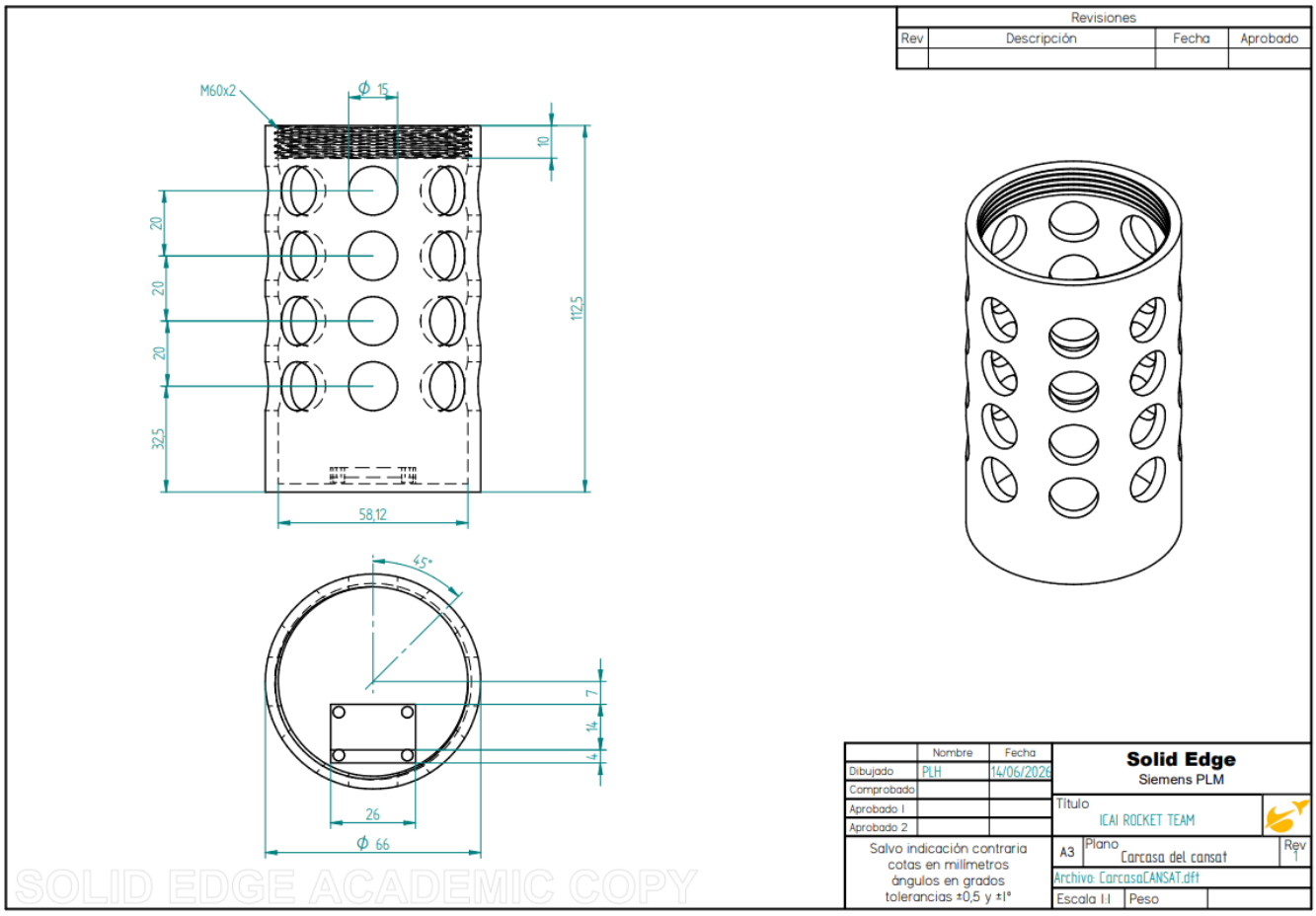


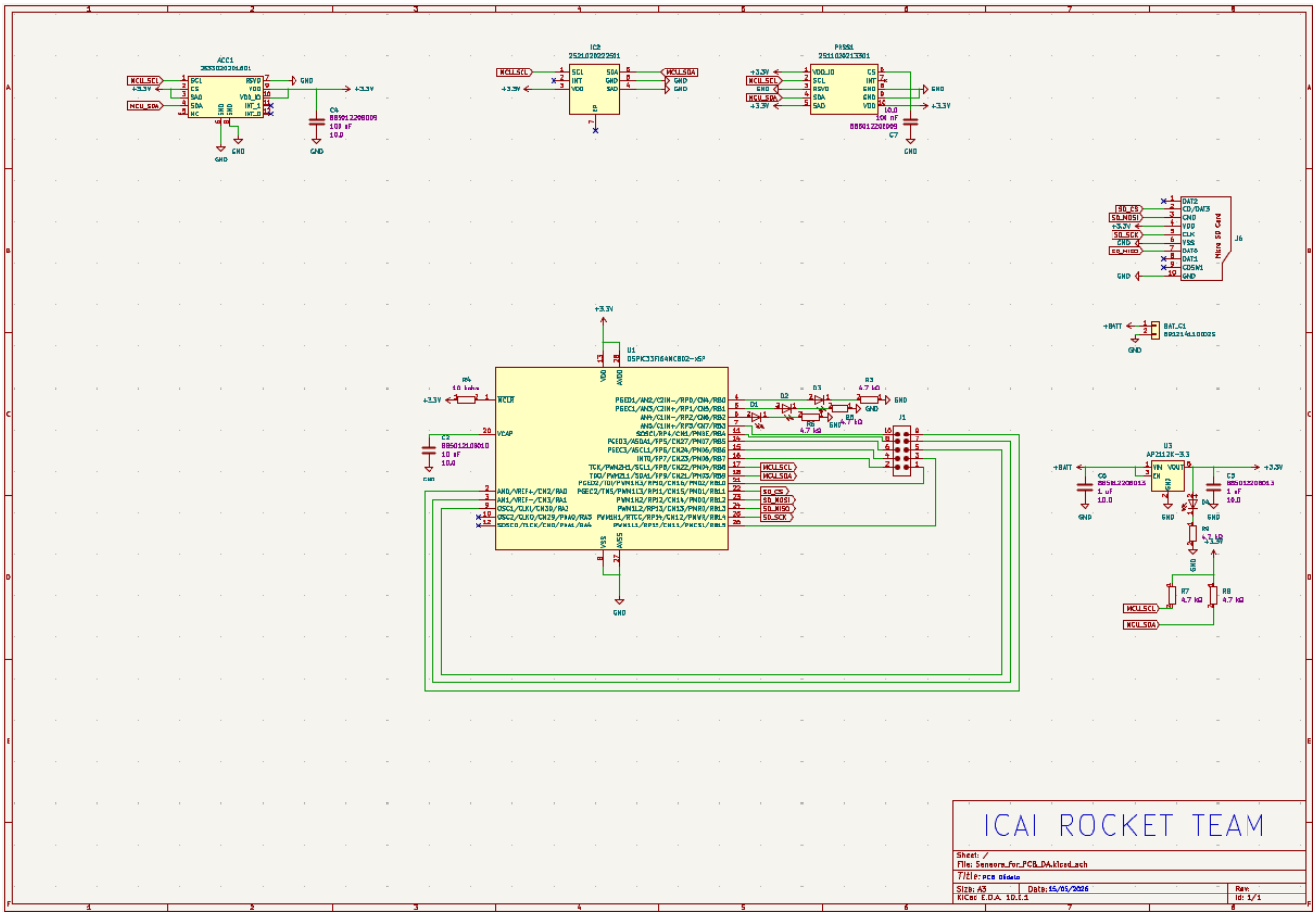



Revisiones
Rev | Descripción | Fecha | Aprobado
A
A
M25
CORTE A-A
DETALLE B
R 38
Ø 64
Ø 49
Ø 5
90°
Nombre | Fecha
Dibujado PLH 14/06/2026
Comprobado
Aprobado 1
Aprobado 2
Salvo indicación contraria cotas en milímetros ángulos en grados tolerancias ±0,5 y ±1°
Solid Edge
Siemens PLM
Título
ICAI ROCKET TEAM
A3
Plano
Soporte y carcasa del muelle
Rev 1
Archivo: Soporte y carcasa del muelle.dft
Escala 1:1
Peso
SOLID EDGE ACADEMIC COPY


Revisiones
Rev | Descripción | Fecha | Aprobado
R 0,25
R 1,25
R 0,75
Ø 12
Ø 11
Ø 9
0,75
Nombre | Fecha
Dibujado PLH 14/06/2026
Comprobado
Aprobado 1
Aprobado 2
Salvo indicación contraria cotas en milímetros ángulos en grados tolerancias ±0,5 y ±1°
Solid Edge
Siemens PLM
Título
ICAI ROCKET TEAM
A3
Plano
Tapa 2 CANSAT
Rev 1
Archivo: tapa_v2_CANSAT.dft
Escala 1:2
Peso
SOLID EDGE ACADEMIC COPY


Revisiones
Rev | Descripción | Fecha | Aprobado
Ø 66
2,5
10
5
0,25
13
M60x2
Nombre | Fecha
Dibujado PLH 14/06/2026
Comprobado
Aprobado 1
Aprobado 2
Salvo indicación contraria
cotas en milímetros
ángulos en grados
tolerancias ±0,5 y ±1°
Solid Edge
Siemens PLM
Título
ICAI ROCKET TEAM
A3
Plano
Tapa superior CANSAT
Rev
1
Archivo: tapa_v1_CANSAT.dft
Escala 1:2
Peso
SOLID EDGE ACADEMIC COPY


Revisiones
Rev | Descripción | Fecha | Aprobado
M60x2
Ø 15
10
20
20
20
32,5
112,5
58,12
45°
7
14
4
26
Ø 66
Nombre | Fecha
Dibujado PLH 14/06/2026
Comprobado
Aprobado 1
Aprobado 2
Salvo indicación contraria
cotas en milímetros
ángulos en grados
tolerancias ±0,5 y ±1°
Solid Edge
Siemens PLM
Título
ICAI ROCKET TEAM
A3
Plano
Carcasa del cansat
Rev
1
Archivo: CarcasaCANSAT.dft
Escala 1:1
Peso
SOLID EDGE ACADEMIC COPY



Annex I

Proof of purchase of the motor is shown below.

SIERRALAB
web commerce ideas

Via Quarto Cagnino 24
20153 Milano – ITALY
VAT No.: IT08505860968
ph. 3334001884
info@sierralab.it

27/05/2026

PRO-FORMA INVOICE

To: ICAI Rocket Team
Alberto Aguilera, 32
Madrid
ESG02857886

QTY	Description	Unit Price	Total Price
1	CTI I212	€ 111,95	€ 111,95
1	Shipping part	€ 4,50	€ 4,50
	TOTAL		**€ 116,45**

Apéndice D

Enlaces a hojas de características

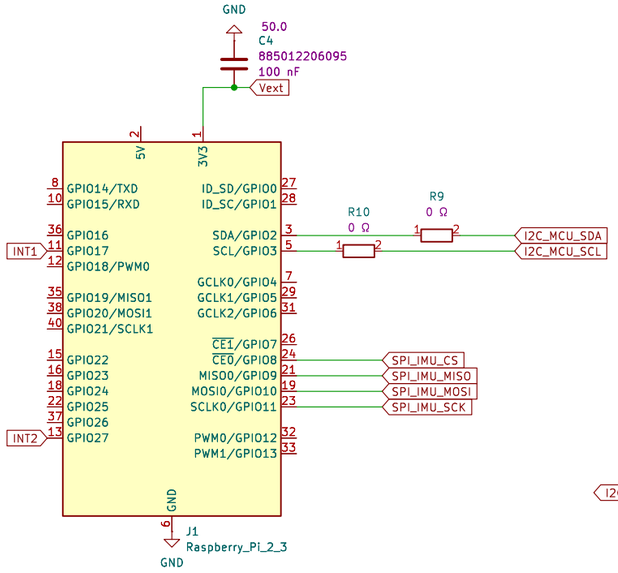
Las hojas de características de los componentes seleccionados se citan a partir de los documentos oficiales del fabricante y de las fichas comerciales identificadas durante el diseño. Para reducir el volumen de la memoria impresa, no se reproducen sus páginas completas en este documento; en su lugar se mantienen enlaces directos a las fuentes originales usadas durante el diseño.

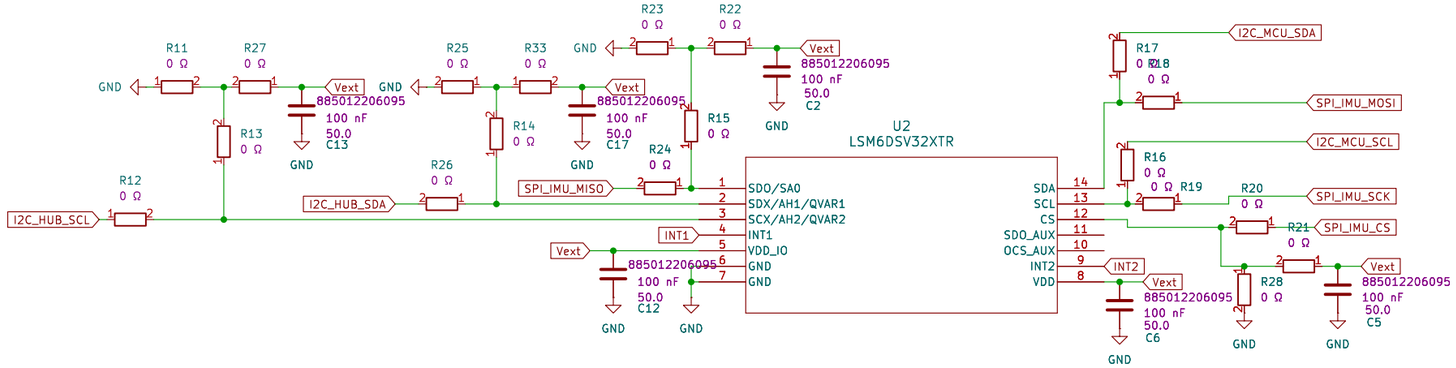
Componente	Enlace a la fuente
STMicroelectronics LSM6DSV32X 6-axis IMU	ST LSM6DSV32X datasheet
TE Connectivity MS5607-02BA03 Barometric Pressure Sensor	TE MS5607-02BA03 datasheet
MEMSIC MMC5983MA 3-axis Magnetic Sensor	MEMSIC MMC5983MA datasheet
Analog Devices AD620 Instrumentation Amplifier	Analog Devices AD620.pdf
TL082 JFET-input operational amplifier, usado en la simulación como ADTL082	Texas Instruments TL082.pdf
Analog Devices AD8221 Precision Instrumentation Amplifier	Analog Devices AD8221.pdf
Analog Devices ADA4077-1/ADA4077-2/ADA4077-4 Operational Amplifier	Analog Devices ADA4077-1_4077-2_4077-4.pdf
Miuzei RC Servo 45KG 270deg digital servo	Ficha comercial ASIN B0CZDXW2FH
JESSPOW 18650 *button top* rechargeable battery	JESSPOW 18650 product sheet
Geekworm X728 V2.5 UPS HAT for Raspberry Pi	Geekworm X728 wiki; X728 hardware notes
Raspberry Pi 3 Model B+ single-board computer	Product brief; reduced schematics; mechanical drawing

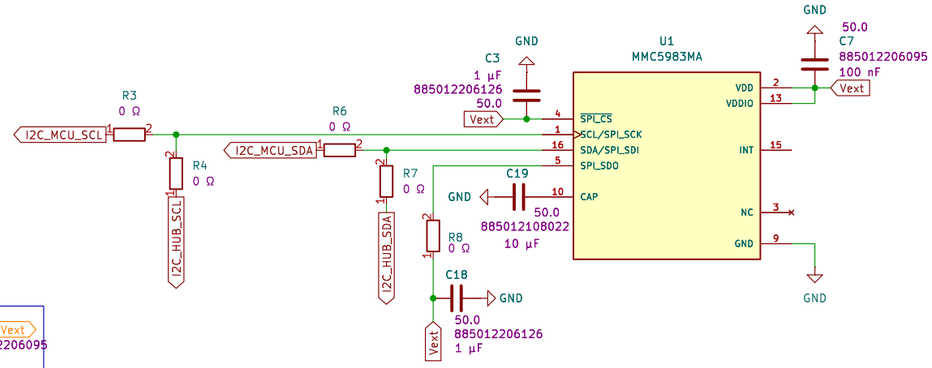
Tabla 11. Enlaces a las fuentes de componentes empleadas en el diseño

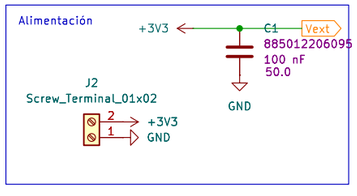
Apéndice E

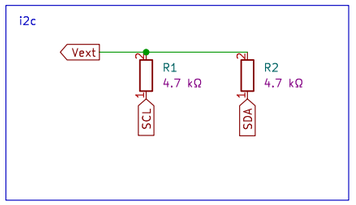
Planos

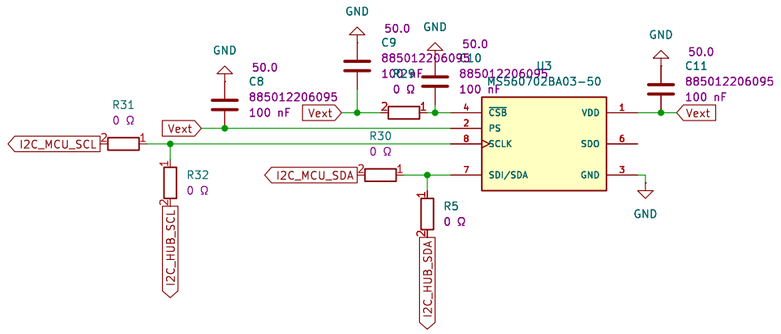












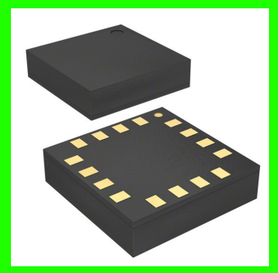
ICAI ROCKET TEAM

Sheet: /Esquematico/ Héctor Sánchez Valencia
File: esquematico.kicad_sch

Title: Caja Electrónica

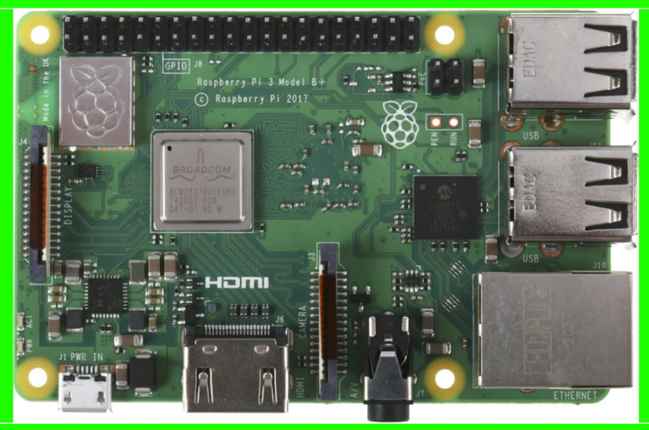
Size: A3	Date: 1/04/2026	**Rev:**
KiCad E.D.A. 9.0.7		Id: 2/2

Magnetómetro



MMC5983MA

Raspberry pi 3b+



6-axis IMU



LSM6DSV32X

Servomotor



Miuzei 45KG RC Servo Motor

Barometro



MS560702BA03-50

Servomotor



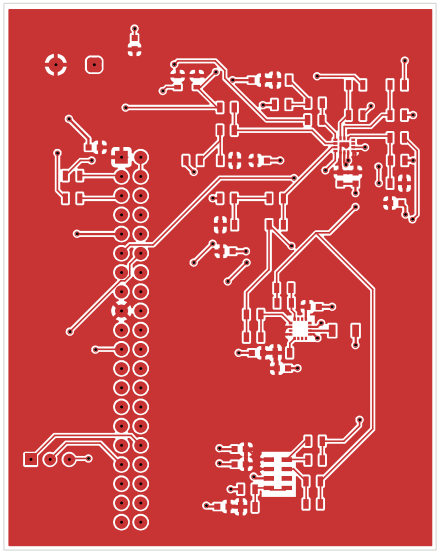
Miuzei 45KG RC Servo Motor

ICAI ROCKET TEAM

Sheet: / Héctor Sánchez Valencia
File: Caja de electrónica.kicad_sch

Title: Caja Electrónica

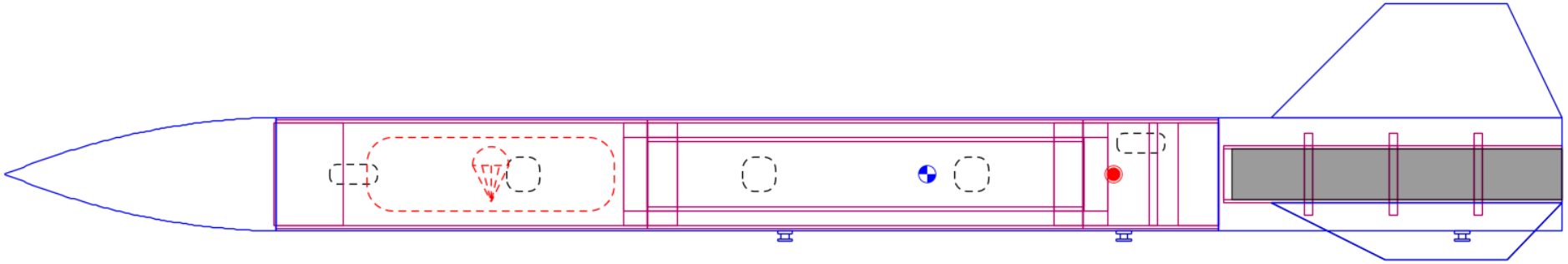
Size: A3	Date:16/03/2026	**Rev:**
KiCad E.D.A. 10.0.1		Id: 1/2



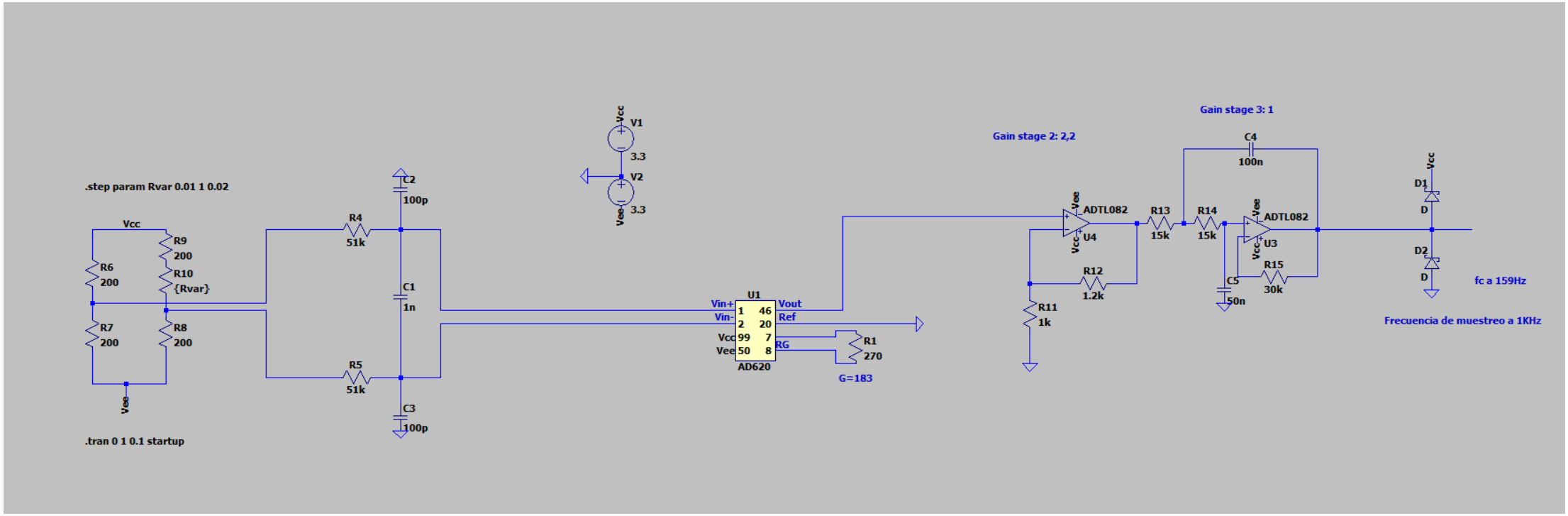
Sheet:		
File: Caja de electrónica.kicad_pcb		
Title:		
Size: A4	Date:	**Rev:**
KiCad E.D.A. 10.0.1		Id: 1/1

SURTAM-J26
Longitud: 115 cm, Diámetro máximo: 8,4 cm
Masa sin motores: 2334 g
Masa con motores: 2809 g

Estabilidad: 1,64 cal / 12 %
CG:68,1 cm
CP:81,9 cm
a M=0,300



Flight configuration: [364l212-14A-5]
Apogeo: 550 m
Velocidad Máx.: 113 m/s , (Número Mach: 0,334)
Aceleración Máx.: 82.8 m/s²


.step param Rvar 0.01 1 0.02
Vcc
R6
200
R9
200
R10
{Rvar}
R7
200
R8
200
Vee
.tran 0 1 0.1 startup
R4
51k
C2
100p
C1
1n
R5
51k
C3
100p
V1
3.3
V2
3.3
U1
Vin+
Vin-
Vcc
Vee
1
2
99
50
46
20
7
8
Vout
Ref
RG
AD620
R1
270
G=183
Gain stage 2: 2,2
ADTL082
U4
R11
1k
R12
1.2k
R13
15k
Gain stage 3: 1
C4
100n
R14
15k
ADTL082
U3
C5
50n
R15
30k
D1
D
D2
D
fc a 159Hz
Frecuencia de muestreo a 1KHz