



Facultad de
INGENIERÍA



Proyecto Final Informe **LAVADORA DE ECUS**

Proyecto Final

Equipo de cátedra

- Ezequiel Blanca
- Juan Zombach
- Cristian Leandro Lukaszewicz

Alumnos

- Federico Bellomi
- Nicolas De Lio
- Elias Jogla



Contenido

1. Introducción.....	7
1.1 Contexto del problema.....	7
1.2 Justificación técnica	8
1.3 Objetivo general	9
1.4 Objetivos específicos	9
2. Marco teórico.....	10
2.1 Limpieza de placas electrónicas	10
2.1.1 Métodos tradicionales	10
2.1.2 Limpieza por aspersión (aplicación al prototipo)	10
2.1.3 Limpieza por ultrasonido (principios y estado del prototipo)	10
2.2 Robots SCARA.....	11
2.2.1 Arquitectura y propiedades esenciales	11
2.2.2 Selección para la aplicación de limpieza de ECUs.....	11
2.3 Cinemática de robots SCARA.....	11
2.3.1 Notación y parámetros geométricos del prototipo	11
2.3.2 Cinemática directa	11
2.3.3 Cinemática inversa.....	11
2.3.4 Jacobiano y control de velocidad.....	12
2.3.5 Workspace y limitaciones prácticas.....	12
2.4 Sistemas de visión artificial	12
2.4.1 Objetivo funcional en el sistema	12
2.4.2 Pipeline de procesamiento propuesto	12
2.4.3 Calibración cámara-robot y conversión píxel→mundo	12
2.5 Microcontroladores y control de motores paso a paso.....	13
2.5.1 Plataforma de control y arquitectura electrónica.....	13
2.5.2 Principios de motores paso a paso y dimensionamiento aplicado	13
2.5.3 Drivers de etapa de potencia y ajustes térmicos.....	13
2.5.4 Dimensionamiento de la fuente de alimentación (cálculo aplicado)	13
3. Descripción general del sistema	14
3.1 Visión general y propósito	14
3.2 Arquitectura funcional (diagrama lógico — descripción)	14
3.3 Flujo operativo (secuencia funcional, paso a paso)	14
3.4 Layout físico y dimensiones principales	15
3.5 Descripción funcional de la lavadora.....	15
3.6 Mecanismo de sujeción y gripper.....	15
3.7 Electrónica, alimentación y comunicaciones.....	15
3.8 Tiempos de ciclo y parámetros de operación	15
BLOQUE I – ROBOT SCARA	16

4. Diseño mecánico del Robot SCARA — (fase de diseño: cálculos previsionales)	16
4.1 Objetivo.....	16
4.2 Supuestos de diseño (valores adoptados a efectos de cálculo)	16
4.3 Cálculo estático de pares (fase de diseño).....	16
4.3.1 Distancias de CG (modelo simple)	16
4.3.2 Momento por gravedad en J1 (paso a paso).....	17
4.3.3 Par disponible (diseño) y margen.....	17
4.4 Estimación dinámica (fase de diseño)	18
4.4.1 Inercia equivalente (modelo puntual) — pasos y números	18
4.4.2 Aceleración de referencia y par dinámico	18
4.4.3 Demanda total (dinámica + gravedad) en J1	18
4.5 Resolución esperada (fase de diseño).....	18
4.6 Resumen numérico (valores previstos por diseño)	18
4.7 Pilar / columna del eje Z — Base estructural y montaje.....	19
4.8 Guía lineal del eje Z	20
4.9 Sistema de contrapeso del eje Z	20
4.10 Fijación mediante perfil D (chaveta plana) del eje Z a la polea	21
4.11 Poleas GT2 paramétricas y plantilla CAD	22
4.12 Mecanismos de transmisión por segmentos (resumen y descripción).....	23
4.13 Fijación de la última polea al siguiente segmento (interfaz de transmisión entre segmentos)	24
4.14 Pilar / soporte de unión entre segmentos.....	24
4.15 Shape Generator aplicado al segundo segmento	24
4.16 Diseño del gripper y ranura de sujeción (interfaz con la pieza adaptadora)	26
4.17 Piezas críticas y consideraciones de tolerancia	27
4.18 Gestión de cableado y trazado mecánico.....	27
4.19 Disipación térmica de la electrónica	27
4.20 Vinculación CAD / Anexos	28
5 Diseño Electrónico del Robot SCARA	30
5.1 Arquitectura electrónica.....	30
5.2 Cálculo de consumo de corriente	31
5.3 Selección y dimensionamiento de fuente de alimentación.....	31
5.4 Análisis de consumo en pines críticos del microcontrolador	32
5.5 Drivers de motores y disipación térmica	32
5.6 Esquemas Electrónicos Scara/Anexos	33
6 Control y Software del Robot SCARA.....	33
6.1 Estrategia de control El control del SCARA está implementado en dos capas:	33
6.2 Implementación de cinemática inversa	34
6.3 Secuencia de homing	35

6.4 Comunicación con la PC	35
6.5 Integración con el sistema de visión.....	36
6.6 Códigos Scara/Anexos.....	36
BLOQUE II – LAVADORA DE ECUs	37
7. DISEÑO MECÁNICO DE LA LAVADORA DE ECUs	37
7.1 Requerimientos funcionales	37
7.2 Diseño del sistema de movimiento XY	37
7.3 Sistema de sujeción de la ECU	38
7.4 Sistema de aspersión	38
7.5 Sistema de limpieza ultrasónica	38
7.6 Diseño estructural y hermetización	39
7.7 Consideraciones de seguridad	39
8. CÁLCULOS MECÁNICOS.....	39
8.1 Cálculo de fuerzas en el sistema XY	39
8.2 Cálculo de presión de aspersión	40
8.3 Selección de bomba	40
8.4 Cálculo de potencia mecánica del sistema XY	40
8.5 Análisis de vibraciones.....	40
9. DISEÑO ELECTRÓNICO DE LA LAVADORA	41
9.1 Arquitectura electrónica.....	41
1 Bloque de control.....	41
2 Bloque de potencia 12 V	41
3 Bloque de potencia 5 V	41
9.2 Cálculo de consumo total del sistema	41
Motores paso a paso.....	41
Bomba automotriz	41
Inyectores de nafta.....	41
Servomotores MG946	41
Consumo electrónica (Arduino + lógica).....	42
Corriente total en línea de 12 V.....	42
Corriente total en línea de 5 V	42
9.3 Dimensionamiento de fuentes de alimentación	42
Fuente 12 V.....	42
Fuente 5 V para servos.....	42
9.4 Sistema de protección contra sobrecorriente.....	43
9.5 Sensores de fin de carrera y lógica de seguridad	43
9.6 Esquema eléctrico general (descripción conceptual).....	43
Conclusión Técnica del Bloque II.....	44
9.7 Lavadora de ECUs / Anexos.....	44

BLOQUE III – SISTEMA DE VISIÓN E INTELIGENCIA ARTIFICIAL.....	44
10. Sistema de Visión Artificial	44
10.1 Selección de cámara	44
10.2 Flujo de procesamiento de imagen.....	44
10.3 Modelo de inteligencia artificial utilizado	45
10.4 Entrenamiento y validación.....	46
10.5 Conversión de coordenadas píxel a coordenadas reales	47
10.6 Comunicación entre PC y microcontrolador	48
10.7 Códigos Vision/Anexos.....	48
BLOQUE IV – INTEGRACIÓN DEL SISTEMA	48
11. Integración del sistema	48
11.1 Arquitectura general.....	48
11.2 Protocolos y señales.....	49
11.3 Topología física del cableado.....	49
11.4 Flujo operativo (secuencia normal de trabajo).....	49
11.5 Lógica de orquestación	49
La PC actúa como maestro: toma decisiones de alto nivel (qué pieza procesar, cuándo iniciar ciclo de lavado).	49
11.6 Tiempos y tolerancias temporales.....	50
11.7 Manejo de errores y políticas operativas.....	50
11.8 Handshake, ACK y trazabilidad	50
11.9 Interfaz de usuario y control	50
11.10 Calibración y parámetros configurables	50
11.11 Logging, trazabilidad y versiones	50
11.12 Diagrama de secuencia (texto).....	51
BLOQUE V – GESTIÓN DEL PROYECTO.....	51
12. Planificación del Proyecto	51
12.1 Diagrama de Gantt.....	51
12.2 Horas estimadas por etapa	51
12.3 Análisis de desvíos	52
BLOQUE VI – RESULTADOS Y CONCLUSIONES	52
15. Ensayos y Pruebas Realizadas	52
16. Resultados y Análisis.....	53
16.1 Resultados obtenidos.....	53
16.2 Comparación con objetivos iniciales.....	53
17. Conclusiones y Mejoras Futuras.....	54
17.1 Conclusiones técnicas.....	54

17.2 Propuestas de Mejora	55
LAVADORA DE ECUS	55
1. Incorporación de sistema de seguridad para solventes inflamables	55
2. Implementación de recirculación y filtrado del fluido	55
3. Mejora estructural y hermetización	55
4. Optimización del sistema ultrasónico	55
5. Sensores adicionales y monitoreo.....	56
6. Mejora del sistema de sujeción.....	56
ROBOT SCARA.....	56
1. Incorporación de encoders absolutos	56
2. Optimización estructural.....	56
3. Mejora del sistema de visión.....	56
4. Implementación de seguridad industrial	57
18. Referencias Bibliográficas	57



1. Introducción

En el ámbito de la electrónica automotriz moderna, las Unidades de Control Electrónico (ECU, Electronic Control Unit) cumplen un rol fundamental en la gestión de los sistemas del vehículo, tales como la inyección de combustible, encendido, control de emisiones y monitoreo de sensores críticos. Estas unidades están compuestas por placas electrónicas de alta densidad, con componentes sensibles a factores ambientales como la humedad, el polvo, residuos químicos y contaminantes atmosféricos.

Uno de los problemas más frecuentes en el sector de reparación electrónica automotriz es el deterioro progresivo de las ECUs debido a la acumulación de suciedad y a la presencia de humedad. Esta situación puede generar desde fallas intermitentes hasta la inutilización completa del sistema, provocando pérdidas económicas significativas y tiempos prolongados de diagnóstico y reparación. Los métodos de limpieza actualmente utilizados (principalmente el cepillado manual con alcohol isopropílico y el lavado por ultrasonido convencional) presentan limitaciones en términos de repetibilidad, eficiencia, estandarización y seguridad del proceso.

En este contexto, surge la necesidad de desarrollar una solución automatizada que permita mejorar la calidad del proceso de limpieza, reducir la intervención humana y aumentar la confiabilidad del procedimiento. El presente proyecto propone el diseño y desarrollo de una célula automatizada compuesta por un robot tipo SCARA y una lavadora de ECUs de doble etapa (aspersión controlada y limpieza ultrasónica), integrada con un sistema de visión artificial basado en inteligencia artificial para la detección y localización automática de las placas electrónicas.

La solución desarrollada contempla no solo la automatización del proceso de limpieza, sino también la integración coordinada entre los subsistemas mecánicos, electrónicos y de control, garantizando precisión en la manipulación, repetibilidad en los movimientos y seguridad en la operación. El robot SCARA se encarga de la manipulación de las placas mediante un sistema pick and place, mientras que la lavadora ejecuta el ciclo de limpieza de manera autónoma. El sistema de visión permite identificar la posición de las ECUs en el área de trabajo, transformando la información visual en coordenadas utilizables por el robot.

El desarrollo del proyecto implicó el diseño mecánico estructural del robot, el dimensionamiento de motores y fuentes de alimentación, el análisis de esfuerzos y torques, el diseño electrónico de potencia y control, la programación del microcontrolador, la implementación de algoritmos de cinemática inversa y la integración con un sistema de procesamiento de imágenes en una computadora externa.

El presente informe describe de manera detallada el proceso de diseño, cálculo, implementación y validación del sistema completo, incluyendo los análisis mecánicos y electrónicos correspondientes, la planificación del proyecto, la evaluación económica y los ensayos realizados para verificar el cumplimiento de los objetivos establecidos.

1.1 Contexto del problema

La evolución de la industria automotriz ha incrementado significativamente la cantidad y complejidad de sistemas electrónicos presentes en los vehículos modernos. Las Unidades de Control Electrónico (ECUs) son responsables de procesar señales provenientes de múltiples sensores y ejecutar estrategias de control en tiempo real, convirtiéndose en elementos críticos para el correcto funcionamiento del vehículo. Debido a su ubicación dentro del compartimiento del motor o en zonas expuestas a condiciones adversas, las ECUs se encuentran sometidas a variaciones térmicas, vibraciones mecánicas, humedad ambiental y posibles filtraciones de líquidos. Con el

paso del tiempo, estas condiciones pueden generar acumulación de suciedad, corrosión en pistas conductoras y degradación de contactos eléctricos, afectando el desempeño del sistema.

En el ámbito de la reparación electrónica automotriz, la limpieza de estas placas constituye una etapa previa fundamental antes de realizar diagnósticos o intervenciones más profundas. Sin embargo, los procedimientos actuales presentan limitaciones importantes. La limpieza manual mediante cepillado y solventes depende en gran medida de la habilidad del operario, carece de repetibilidad y puede resultar insuficiente para remover contaminantes en zonas de difícil acceso. Por otro lado, los sistemas de lavado ultrasónico disponibles comercialmente no siempre están integrados dentro de un proceso automatizado que contemple manipulación, posicionamiento y control del flujo de trabajo.

Esta situación evidencia una oportunidad de mejora mediante la implementación de soluciones automatizadas que integren tecnologías de manipulación robótica, limpieza controlada y visión artificial, permitiendo estandarizar el proceso, aumentar la eficiencia y reducir la intervención humana directa.

1.2 Justificación técnica

El desarrollo de una célula automatizada para la limpieza de ECUs se justifica desde múltiples perspectivas técnicas y operativas. En primer lugar, la automatización del proceso permite garantizar condiciones de operación repetibles y controladas, reduciendo la variabilidad asociada al trabajo manual. Esto resulta especialmente relevante en tareas donde la precisión y el cuidado del componente son fundamentales para evitar daños adicionales.

Desde el punto de vista mecánico, la incorporación de un robot tipo SCARA ofrece ventajas significativas en aplicaciones de manipulación plana, debido a su estructura rígida en el eje vertical y su alta velocidad en movimientos en el plano XY. Estas características lo convierten en una solución adecuada para operaciones de tipo pick and place, como la carga y descarga de placas electrónicas en la lavadora.

En el aspecto electrónico y de control, el diseño integral del sistema permite dimensionar adecuadamente fuentes de alimentación, drivers de motores y protecciones eléctricas, asegurando un funcionamiento seguro y eficiente. Asimismo, la implementación de cinemática inversa y secuencias de homing contribuye a mejorar la precisión y confiabilidad del robot.

Por otra parte, la incorporación de un sistema de visión artificial basado en inteligencia artificial permite automatizar la detección y localización de las placas dentro del área de trabajo, eliminando la necesidad de posicionamiento manual previo y aumentando el grado de autonomía del sistema.

Finalmente, desde una perspectiva académica y de ingeniería, el proyecto integra conocimientos de diseño mecánico, electrónica de potencia, control, programación, procesamiento de imágenes e integración de sistemas, constituyendo una aplicación multidisciplinaria que refleja las competencias adquiridas durante la carrera de Ingeniería Mecatrónica.

1.3 Objetivo general

Diseñar, desarrollar e integrar una célula automatizada compuesta por un robot tipo SCARA, una lavadora de ECUs de doble etapa (aspersión controlada y limpieza ultrasónica) y un sistema de visión artificial basado en inteligencia artificial, con el propósito de optimizar el proceso de limpieza de placas electrónicas automotrices, garantizando precisión, repetibilidad, seguridad y reducción de la intervención humana.

1.4 Objetivos específicos

Para alcanzar el objetivo general planteado, se establecen los siguientes objetivos específicos:

- Diseñar y construir un robot tipo SCARA capaz de realizar movimientos en el plano XY con un eje vertical Z, logrando una repetibilidad mínima de 1 mm y capacidad de manipulación de cargas de al menos 100 g.
- Implementar el modelado matemático del robot, incluyendo el desarrollo de la cinemática directa e inversa, para permitir el posicionamiento preciso del efector final.
- Diseñar un sistema de gripper adecuado para la sujeción segura de placas electrónicas, considerando su geometría y fragilidad.
- Desarrollar una lavadora de ECUs que integre un sistema de aspersión controlada mediante alcohol isopropílico y un sistema de limpieza ultrasónica, garantizando la eliminación de contaminantes superficiales y profundos.
- Dimensionar y diseñar el sistema electrónico completo, incluyendo selección de motores, drivers, fuentes de alimentación, sensores de fin de carrera y protecciones eléctricas, realizando los cálculos correspondientes de corriente, potencia y disipación térmica.
- Implementar un sistema de visión artificial capaz de detectar y localizar automáticamente las placas electrónicas dentro del área de trabajo, utilizando un modelo de inteligencia artificial ejecutado en una computadora externa.
- Desarrollar la comunicación entre la PC y el microcontrolador del robot, permitiendo la transferencia de coordenadas procesadas por el sistema de visión.
- Integrar todos los subsistemas mecánicos, electrónicos y de control en una arquitectura coordinada que permita un funcionamiento autónomo del proceso completo.
- Realizar ensayos de validación para verificar el cumplimiento de los requisitos de repetibilidad, carga, limpieza e integración del sistema.

2. Marco teórico

2.1 Limpieza de placas electrónicas

2.1.1 Métodos tradicionales

La limpieza de placas electrónicas se realiza habitualmente mediante tres enfoques: limpieza manual con solventes, aspersión controlada y ultrasonido. La limpieza manual con alcohol isopropílico u otros disolventes es económica y efectiva para contaminantes superficiales, pero depende altamente de la pericia del operador y carece de reproducibilidad. La aspersión controlada aplica energía mecánica puntual mediante chorros de fluido para desalojar residuos adheridos, siendo una alternativa intermedia entre el tratamiento manual y la inmersión completa. La limpieza por ultrasonido actúa a escala microscópica por cavitación, permitiendo la remoción en zonas inaccesibles para las técnicas anteriores; requiere control de frecuencia, potencia y temperatura del baño.

Cada técnica tiene ventajas y limitaciones en términos de agresividad, compatibilidad con componentes, seguridad (especialmente si se emplean solventes inflamables) y repetibilidad. Para procesos industriales y estandarizados se recomiendan soluciones automatizadas que combinen etapas complementarias (por ejemplo, aspersión seguida de ultrasonido) para maximizar eficacia sin dañar los componentes.

2.1.2 Limpieza por aspersión (aplicación al prototipo)

La etapa de aspersión en el sistema propuesto utiliza inyectores de combustión montados en posición fija y una bomba con presión operativa entre 2.7 y 4.1 bar, rango típico de bombas de inyectores y suficiente para generar chorros con alta energía cinética. Los parámetros operativos relevantes que condicionan la eficacia son: presión de trabajo, caudal, diámetro y geometría de la tobera, distancia nozzle–superficie, ángulo de incidencia y tiempo de exposición.

En la práctica del prototipo, la pieza (ECU) se somete a un ciclo de rotación que expone sucesivamente cara superior e inferior a los chorros. Esta estrategia permite que boquillas fijas cubran ambas caras sin necesidad de mecanismos articulados adicionales para las boquillas, reduciendo complejidad mecánica y puntos de fallo.

2.1.3 Limpieza por ultrasonido (principios y estado del prototipo)

La limpieza por ultrasonido convierte energía eléctrica en ondas mecánicas en el líquido mediante transductores; las oscilaciones inducen cavitación y el colapso de microburbujas genera micro chorros que desprenden contaminantes. Los parámetros críticos son la frecuencia (normalmente 20–400 kHz), la potencia específica (W/L), el tiempo de exposición y la temperatura del baño. Frecuencias bajas (20–40 kHz) favorecen cavitación intensa para suciedad gruesa; frecuencias altas (≥ 100 kHz) producen limpieza más fina y suave.

En el prototipo actual existe una batea y un motor desbalanceado que, en su configuración, sirve como demostrador mecánico, pero no logra la excitación a frecuencias ultrasónicas efectivas. La implementación plena del ultrasonido requiere transductores piezoeléctricos y drivers apropiados. Además, el uso de ultrasonido en presencia de solventes inflamables requiere equipamiento certificado (ATEX u homologaciones equivalentes), ventilación y control térmico.

2.2 Robots SCARA

2.2.1 Arquitectura y propiedades esenciales

Un robot tipo SCARA (Selective Compliance Assembly Robot Arm) es un manipulador caracterizado por articulaciones rotacionales en el plano XY y un grado de libertad prismático o rotacional en el eje Z. Su rigidez en la dirección vertical, la alta velocidad y la precisión en el plano horizontal lo hacen especialmente apto para tareas de pick & place, inserción y manipulación de piezas planas. Para la manipulación de ECUs (componentes relativamente planos y frágiles) el SCARA combina buena velocidad de ciclo con suficiente rigidez y repetibilidad.

2.2.2 Selección para la aplicación de limpieza de ECUs

La elección del SCARA se fundamenta en la necesidad de integrar un sistema de manipulación que realice carga/descarga, posicionamiento frente a boquillas de aspersión y colocación en la batea ultrasónica, todo con mínimo contacto humano. Se valora además la posibilidad de integrar un efector con cámara (eye-in-hand) para detección y sujeción precisa.

2.3 Cinemática de robots SCARA

2.3.1 Notación y parámetros geométricos del prototipo

Denotamos L_1 y L_2 las longitudes de los dos eslabones en el plano XY; en el prototipo $L_1=160$ y $L_2=200$ mm. Las variables articulares son θ_1 (theta 1) y θ_2 (theta 2) que son rotaciones en las uniones base \rightarrow eslabón 1 y eslabón 1 \rightarrow eslabón 2) y z para el eje prismático.

2.3.2 Cinemática directa

Las expresiones de la cinemática directa permiten calcular la posición cartesiana (x,y,z) del efector en función de los ángulos articulares:

$$\begin{aligned}x &= L_1 \cos(\theta_1) + L_2 \cos(\theta_1 + \theta_2) \\y &= L_1 \sin(\theta_1) + L_2 \sin(\theta_1 + \theta_2) \\z &= z\end{aligned}$$

Estas relaciones son la base para establecer la transformación entre espacio articular y espacio cartesiano usada por el planificador del robot.

2.3.3 Cinemática inversa

Dada la posición (x,y) deseada del efector, la obtención de θ_1 y θ_2 se realiza mediante:

$$r^2 = x^2 + y^2 \quad ; \quad \cos(\theta_2) = \frac{r^2 - L_1^2 - L_2^2}{2L_1L_2} ;$$

y

$$\begin{aligned}\theta_2 &= \text{atan2}(\pm\sqrt{1 - \cos^2(\theta_2)}, \cos(\theta_2)) \\ \theta_1 &= \text{atan2}(y, x) - \text{atan2}(L_2 \sin(\theta_2), L_1 + L_2 \cos(\theta_2))\end{aligned}$$

La solución presenta ambigüedad de configuración (dos soluciones—elbow-up/elbow-down) que debe resolverse con criterios basados en límites mecánicos, obstáculos y eficiencia de la trayectoria.

2.3.4 Jacobiano y control de velocidad

El Jacobiano $J(\theta)$ relaciona velocidades articulares $\dot{\theta}$ con velocidades cartesianas $\dot{p}=[\dot{x} \ \dot{y}]^T$ mediante $\dot{p}=J(\theta)\dot{\theta}$. La inversa del Jacobiano (cuando existe) se usa para convertir velocidades cartesianas deseadas en velocidades articulares necesarias. El determinante de J indica singularidades donde la manipulación es problemática; por tanto es esencial planificar trayectorias evitando configuraciones cercanas a dichas singularidades.

2.3.5 Workspace y limitaciones prácticas

El workspace teórico del SCARA en el plano XY viene dado por el conjunto de radios r tales que $|L1-L2| \leq r \leq L1+L2$. En la práctica, las limitaciones mecánicas (topes físicos, interferencias de estructura, rango del efector, espacio de montaje) reducen el workspace útil. Adicionalmente, la interacción con el sistema de visión y la disposición de la lavadora fijan zonas operativas preferentes.

2.4 Sistemas de visión artificial

2.4.1 Objetivo funcional en el sistema

El sistema de visión tiene por objetivo detectar la presencia y estimar la pose (posición y orientación plana) de las ECUs sobre la mesa de trabajo para proporcionar coordenadas que el controlador del robot utilice para el pick & place. La solución en el prototipo combina detección por redes convolucionales y marcadores QR para robustez y precisión.

2.4.2 Pipeline de procesamiento propuesto

El flujo típico se compone de:

1. **Adquisición:** la cámara montada en el efector (eye-in-hand) captura imágenes. En el prototipo se usa una cámara comercial de consumo (Logitech C270 HD 720p), adecuada para demostración; su obturador y tasa de frames deben ser consideradas al programar movimientos.
2. **Preprocesado:** corrección de distorsión, filtrado y normalización del histograma para robustecer la detección ante variaciones de iluminación.
3. **Detección/segmentación:** empleo de un detector de objetos basado en CNN (por ejemplo YOLO u arquitecturas análogas) para obtener cajas delimitadoras y puntos claves.
4. **Estimación de pose:** uso de marcadores QR (uno fijo como referencia y uno variable en cada ECU) para obtener la transformación relativa ECU→referencia; concebir la estimación mediante homografía si el plano es conocido o mediante PnP en caso de tener referencias 3D.
5. **Transformación a coordenadas del robot:** mediante calibración hand–eye convertir la pose en coordenadas del robot y enviar comandos de posicionamiento.

2.4.3 Calibración cámara–robot y conversión píxel→mundo

Se distinguen dos calibraciones necesarias:

- **Calibración intrínseca:** determina parámetros de la cámara (f_x, f_y, c_x, c_y) coeficientes de distorsión radial/tangencial. Se obtiene típicamente con patrones planos (tablero de ajedrez) y librerías como OpenCV.
- **Calibración extrínseca / hand–eye:** define la transformación rígida entre la cámara y la herramienta (efector). En configuraciones eye-in-hand es indispensable para traducir observaciones de la cámara a movimientos del robot

con precisión.

Si la cámara observa esencialmente un plano de trabajo fijo y la altura no varía significativamente, una homografía 2D proporciona una conversión eficaz entre coordenadas de imagen y coordenadas del plano. En casos más generales será necesario estimar la pose 3D usando solvePnP y la transformación hand-eye.

2.5 Microcontroladores y control de motores paso a paso

2.5.1 Plataforma de control y arquitectura electrónica

La electrónica de control del prototipo se basa en un microcontrolador de arquitectura compatible con proyectos de control de múltiples motores (se utiliza la placa **Arduino Mega** sobre la que se monta una shield tipo RAMPS 1.4 para alojar drivers de potencia). Esta configuración facilita la integración de drivers tipo DRV8825, sensores I²C (multiplexor TCA9548A y AS5600) y finales de carrera.

2.5.2 Principios de motores paso a paso y dimensionamiento aplicado

Los motores paso a paso se caracterizan por su paso por revolución (200 pasos = 1,8°), torque de retención y curva de torque a velocidad. En el proyecto se emplean NEMA17 de alto torque con especificación nominal **4.3 kg·cm** ($\approx 0.4217 \text{ N}\cdot\text{m}$) y un NEMA17 pancake para J6 con **1.6 kg·cm** ($\approx 0.1569 \text{ N}\cdot\text{m}$).

Efecto de la reducción: las transmisiones implantadas (relación compuesta 16:1 en J1/J2 y 1:5 en J6) multiplican el torque disponible en la salida de la transmisión, mientras que disminuyen la velocidad y aumentan la resolución angular por paso. Aplicando la reducción 16:1 al motor de 0.4217 N·m, el torque teórico disponible en el eje de salida es $\approx 6.747 \text{ N}\cdot\text{m}$, una magnitud que excede holgadamente la demanda gravitatoria de una ECU de 0.1 kg colocada a $r = 0.36 \text{ m}$ ($\tau \approx 0.353 \text{ N}\cdot\text{m}$). Para dimensionamiento dinámico se debe incluir el término de inercia I y la aceleración angular requerida:

$$\alpha: \tau = I\alpha + \tau_{\text{gravedad}} + \tau_{\text{fricción}}$$

2.5.3 Drivers de etapa de potencia y ajustes térmicos

Los DRV8825 regulan corriente mediante control por chopper y requieren ajuste del límite de corriente acorde al motor. La disipación térmica es significativa; es imprescindible disipadores y ventilación forzada. Ajustar la corriente máxima evita sobrecalentamiento de los drivers y limita la corriente RMS en los motores.

2.5.4 Dimensionamiento de la fuente de alimentación (cálculo aplicado)

Con los consumos nominales aportados se calcula:

- Motores: 3 × NEMA17 alto-torque a 1.7 A/fase ($3 \times 1.7 = 5.1 \text{ A}$) y 1 × pancake a 1.0 A \rightarrow total motores = 6.1 A.
- Aplicando margen de seguridad del 30%: corriente de diseño $\approx 7.93 \text{ A}$.
- Potencia requerida a 12 V: $P = 12 \text{ V} \times 7.93 \text{ A} \approx 95.2$

La elección de una fuente **12 V / 15 A** ofrece margen suficiente para picos de corriente y expansión futura. El circuito de control (Arduino, multiplexor I²C, sensores y drivers de lógica) se alimenta desde USB de la PC; el consumo estimado total de la electrónica de control es $\approx 174 \text{ mA}$ (con margen operativo tomado a 250 mA).

3. Descripción general del sistema

3.1 Visión general y propósito

La célula automatizada integra un manipulador tipo SCARA, una lavadora de ECUs de doble etapa (aspersión y batea) y un sistema de visión montado en el efector del robot. El objetivo es automatizar la carga, limpieza y descarga de placas electrónicas automotrices con repetibilidad y mínima intervención humana. Las ECUs que se procesan disponen previamente de una pieza adaptadora impresa en 3D que incorpora un marcador QR y ranuras autocentrantes para el gripper del robot, estandarizando la sujeción y la referencia visual.

3.2 Arquitectura funcional (diagrama lógico — descripción)

La arquitectura se organiza en cuatro subsistemas principales:

- **Operador / bandeja inicial.** El operario inserta la pata 3D en la ECU y deja las piezas en la mesa de trabajo dentro del alcance del SCARA.
- **Visión y procesamiento (PC).** La cámara del efector captura imágenes; la PC ejecuta el detector y el módulo de IA para localizar QRs y calcular poses. (Se utiliza una cámara comercial Logitech C270 para la etapa de demostración). Logitech
- **Control SCARA.** Control local del manipulador (motor drivers, encoders, firmware) basado en una placa Arduino Mega con shield RAMPS 1.4. Arduino
- **Lavadora (control local).** Control independiente basado en otro Arduino Mega que gestiona bombas, servos, guías lineales y la batea; la comunicación simple entre SCARA y lavadora se realiza vía ESP32 para señales de evento.

El flujo de información es: PC (visión) → Control maestro → Control SCARA → (ESP32) → Control Lavadora, con señales de estado de retorno desde la lavadora hacia el maestro.

3.3 Flujo operativo (secuencia funcional, paso a paso)

1. **Preparación:** el operario instala la pata 3D en cada ECU y las deja sobre la mesa.
2. **Inicio:** el operario pulsa "Play" en la interfaz de la PC para iniciar la rutina.
3. **Escaneo:** el SCARA recorre posiciones de escaneo; la cámara captura imágenes y la PC detecta QRs.
4. **Selección:** al detectar un QR válido, la PC manda la pose al SCARA.
5. **Pick:** el SCARA va a la posición de captura, activa el gripper y toma la pieza (la geometría autocentrante asegura sujeción con tolerancia).
6. **Transferencia:** el SCARA lleva la pieza a la zona de entrega de la lavadora y deposita la ECU en la posición de transferencia.
7. **Cedencia:** el SCARA envía señal (ESP32 → Arduino Lavadora) indicando "pieza colocada"; la lavadora cierra sus grippers internos y confirma la sujeción; el SCARA se retira.
8. **Lavado:** la lavadora ejecuta la secuencia preconfigurada: aspersión cara A, rotación ($\leq 180^\circ$), aspersión cara B, inmersión en batea.
9. **Fin de ciclo:** al completar el ciclo, la lavadora abre sus grippers y envía señal de "listo" al SCARA; el SCARA recoge la ECU y la deposita en la posición de salida.
10. **Repetición/Finalización:** el SCARA reanuda el escaneo. Si no detecta más piezas tras recorrer todas las zonas, finaliza la rutina.

3.4 Layout físico y dimensiones principales

- **Lavadora (caja):** altura del cuerpo = 33 cm; patas = 7 cm → altura total ≈ 40 cm; ancho = 53 cm; profundidad = 33 cm.
- **Separación robot-lavadora:** ≈ 360 mm entre el centro de trabajo del robot y la abertura de la lavadora, distancia calculada para que el gripper realice la transferencia sin interferencias.
- **Carcasa y visibilidad:** las cuatro paredes laterales son acrílico; el frente del prototipo está abierto (sin puerta). La estructura principal es chapa doblada en L para rigidez.
- **Montaje:** actualmente la célula se apoya en patas; se recomienda anclaje al piso para mitigar vibraciones y garantizar repetibilidad.

3.5 Descripción funcional de la lavadora

Estructura y componentes internos: la lavadora es un cubo rectangular con la cara frontal abierta. Internamente incorpora:

- **Pinzas internas y guías verticales:** dos mecanismos verticales con 2 × NEMA17 (tornillo + guía lineal) para ascenso/descenso de los conjuntos de pinzas; finales de carrera implementados para homing y límites.
- **Servomotores: 4 × MG90 (dos por lado):** uno para rotación en Y (voltar la pieza) y otro para cerrar/abrir los grippers.
- **Inyectores fijos en techo:** inyectores de nafta reales conectados a una bomba ON/OFF que entrega 2.7–4.1 bar; las boquillas están fijas, la placa se rota para exponer ambas caras.
- **Batea de base:** recoge líquidos y sirve de zona de inmersión; en prototipo la batea no integra transductores ultrasónicos activos (solo demostrador mecánico).

3.6 Mecanismo de sujeción y gripper

- **Gripper del SCARA:** impreso en PETG, apertura máxima 50 mm; ranura de entrada 38 mm. La pata 3D contiene un perfil autocentrante (vista frontal: rectángulo sobre trapecio que forma un cono guía) que asegura centrado y captura aun con desvíos de unos milímetros.
- **Interfaz de entrega:** el SCARA deposita la ECU en la zona de transferencia y notifica via ESP32 al Arduino de la lavadora; la lavadora cierra sus pinzas internas y confirma antes de iniciar el ciclo. No se usan sensores de fuerza; la geometría mecánica realiza la función de alineado y sujeción.

3.7 Electrónica, alimentación y comunicaciones

- **Subsistemas electrónicos:** SCARA y Lavadora disponen cada uno de una placa **Arduino Mega + RAMPS 1.4** para drivers y sensores; el SCARA alimenta su lógica desde la PC (USB) y desde una fuente 12 V/15 A para potencia. La Lavadora cuenta con fuentes independientes (12 V/20 A y 5 V/20 A).
- **Comunicaciones:** señalización simple entre SCARA y Lavadora por medio de un **ESP32** (GPIOs / pin-level signaling). Actualmente no hay un protocolo de handshake formal; la interacción es basada en eventos discretos y retardos.

3.8 Tiempos de ciclo y parámetros de operación

- **Visión / detección por zona:** ≈ 30 s por zona de escaneo.
- **Pick & place (captura + transferencia):** ≈ 15 s.
- **Ciclo de limpieza (configuración actual):** ≈ 60 s total: 15 s aspersión cara A → rotación → 15 s aspersión cara B → inmersión 20 s en batea.
- **Tiempo total típico por pieza:** entre 1 min 30 s y 2 min (depende de detección, colas y tiempos de transferencia).

BLOQUE I – ROBOT SCARA

4. Diseño mecánico del Robot SCARA — (fase de diseño: cálculos previsionales)

4.1 Objetivo

El objetivo de esta sección es presentar el análisis mecánico previsto en la fase de diseño, los supuestos de masa y geometría que se consideraron al dimensionar el manipulador, los cálculos de pares estáticos y dinámicos basados en esos supuestos, la resolución esperada y las verificaciones estructurales preliminares.

4.2 Supuestos de diseño (valores adoptados a efectos de cálculo)

Estos son los supuestos de diseño que se usarán para todos los cálculos de esta sección. Son intencionalmente conservadores para dar margen en la fase de proyecto:

- Geometría:
 - L1=160 mm (brazo proximal, entre centros).
 - L2=200 mm (brazo distal, entre centros).
 - Alcance máximo: $r_{max} = L_1 + L_2 = 0.360 \text{ m}$
- Masas (supuestas en diseño):
 - Segundo segmento (brazo proximal): $m_2 = 0.70 \text{ kgm}$
 - Tercer segmento (brazo distal): $m_3 = 0.50 \text{ kgm}$
 - Efector (gripper + pieza adaptadora + ECU) supuesto: $m_g = 0.10 \text{ kg}$ (100 g).
- Material y fabricación: piezas impresas en PETG infill $\approx 50\text{--}60 \%$, paredes nominales $\approx 8 \text{ mm}$.
- Actuadores y transmisiones (selección de proyecto):
 - Motores NEMA17 alto-torque con torque nominal $\tau_m = 4.3 \text{ kg}\cdot\text{cm} \approx 0.4217 \text{ N}\cdot\text{m}$
 - Relación de transmisión en J1/J2 tomada: $i = 16:1$
 - Motor J6 pancake y relación 1:5 (no se analiza en detalle aquí).
- Perfil de movimiento de referencia para dinámicas: giro de $90^\circ = \pi$ en $T = 0.5$ con perfil triangular (acelera la mitad del tiempo y desacelera la otra mitad).

4.3 Cálculo estático de pares (fase de diseño)

4.3.1 Distancias de CG (modelo simple)

Asumimos centros de masa en mitad de cada eslabón:

$$r_{m2} = \frac{L_1}{2} = \frac{0.160\text{m}}{2} = 0.08 \text{ m}$$

$$r_{m3} = L_1 + \frac{L_2}{2} = 0.160\text{m} + 0.100\text{m} = 0.260 \text{ m}$$

$$r_{mg} = L_1 + L_2 = 0.360 \text{ m}$$

4.3.2 Momento por gravedad en J1 (paso a paso)

Usamos los supuestos de masa del diseño.

1. Momento debido a $m_2=0.70$ kg en $r_{m2}=0.080$

$$\begin{aligned}M_{m2} &= m_2 \cdot g \cdot r_{m2} \\M_{m2} &= 0.7 \cdot 9.81 \cdot 0.080 \\M_{m2} &= 0.54936 \text{ Nm}\end{aligned}$$

2. Momento debido a $m_3=0.50$ kg en $r_{m3}=0.260$ m:

$$\begin{aligned}M_{m3} &= m_3 \cdot g \cdot r_{m3} \\M_{m3} &= 0.5 \cdot 9.81 \cdot 0.260 \\M_{m3} &= 1.2753 \text{ Nm}\end{aligned}$$

3. Momento debido a $m_g=0.10$ kg en $r_g=0.360$ m:

$$\begin{aligned}M_g &= m_g \cdot g \cdot r_g \\M_g &= 0.1 \cdot 9.81 \cdot 0.360 \\M_g &= 0.35316 \text{ Nm}\end{aligned}$$

4. Momento total en J1 (estático, diseño):

$$\begin{aligned}M_{J1} &= M_g + M_{m2} + M_{m3} \\M_{J1} &= 0.35316 \text{ Nm} + 0.54936 \text{ Nm} + 1.2753 \text{ Nm} \\M_{J1} &= 2.17782 \text{ Nm}\end{aligned}$$

4.3.3 Par disponible (diseño) y margen

- Par nominal motor: $\tau_m=0.4217$ N·m
- Par disponible en salida tras $i=16:1$

$$\tau_{salida} = \tau_m \cdot i = 0.4217 \cdot 16 = 6.7472 \text{ Nm}$$

- Margen previsto:

$$\text{Margen} = \frac{\tau_{salida}}{M_{J1}} = \frac{6.7472 \text{ Nm}}{2.17782 \text{ Nm}} = 3.1$$

Esto nos da un margen aproximado de 3 veces el torque necesario con respecto al torque producido por los motores (asumiendo que no hay perdida por fricción)

4.4 Estimación dinámica (fase de diseño)

4.4.1 Inercia equivalente (modelo puntual) — pasos y números

Aproximamos las masas como puntos en sus CG respecto a J1:

$$\begin{aligned} I_{m2} &= m_2 \cdot r_{m2}^2 = 0.7 \cdot 0.08^2 = 0.00448 \text{ kg} \cdot \text{m}^2 \\ I_{m3} &= m_3 \cdot r_{m3}^2 = 0.5 \cdot 0.260^2 = 0.03380 \text{ kg} \cdot \text{m}^2 \\ I_g &= m_g \cdot r_{mg}^2 = 0.1 \cdot 0.360^2 = 0.01296 \text{ kg} \cdot \text{m}^2 \end{aligned}$$

Suma (inercia total aproximada alrededor de J1):

$$\begin{aligned} I_{J1} &= I_{m2} + I_{m3} + I_g \\ I_{J1} &= 0.00448 \text{ kg} \cdot \text{m}^2 + 0.03380 \text{ kg} \cdot \text{m}^2 + 0.01296 \text{ kg} \cdot \text{m}^2 \\ I_{J1} &= 0.05124 \text{ kg} \cdot \text{m}^2 \end{aligned}$$

4.4.2 Aceleración de referencia y par dinámico

Para 90° en T=0.5 (perfil triangular) la aceleración angular máxima es

$$\alpha = \frac{4\theta}{T^2} = \frac{4(\pi/2)}{0.5^2} = 8\pi = 25.133 \text{ rad/s}^2$$

Par dinámico requerido:

$$\tau_{dinámico} = I_{J1} \cdot \alpha = 0.05124 \text{ kg} \cdot \text{m}^2 \cdot 25.133 \text{ rad/s}^2 = 1.2878 \text{ Nm}$$

4.4.3 Demanda total (dinámica + gravedad) en J1

$$\begin{aligned} \tau_{total\ requerido} &= M_{J1} + \tau_{dinámico} = 2.17782 \text{ Nm} + 1.2878 \text{ Nm} \\ \tau_{total\ requerido} &= 3.46562 \text{ Nm} \end{aligned}$$

En términos de par en el motor (antes de reducción):

$$\tau_{total\ requerido\ en\ motor} = \frac{\tau_{total\ requerido}}{16} = 0.2166 \text{ Nm}$$

Comparación con torque motor nominal $\tau_m=0.4217 \text{ N} \cdot \text{m} \rightarrow$ margen dinámico a nivel motor ≈ 1.94 (sin pérdidas).

4.5 Resolución esperada (fase de diseño)

- Paso motor: 200 pasos/rev $\rightarrow 1.8^\circ$ por paso.
- Con reducción $i=16:1$ la resolución en articulación es de:

$$\Delta\theta_{art} = \frac{1.8^\circ}{16} = 0.1125^\circ = 0.0019635 \text{ rad}$$

- Desplazamiento lineal en extremo ($r = 0.36 \text{ m}$) por paso:

$$\Delta s = 0.36 \text{ m} \cdot 0.0019635 \text{ rad} = 0.000707 \text{ m} = 0.707 \text{ mm}$$

esto sitúa la resolución sin microstepping en $\approx 0.7 \text{ mm}$ en el extremo.

4.6 Resumen numérico (valores previstos por diseño)

- Par estático previsto en J1:

$$M_{J1} = 2.178 \text{ Nm}$$

- Par dinámico previsto (90° en 0.5 s):

$$\tau_{dinamico} = 1.288 \text{ Nm}$$

- Demanda total prevista:

$$\tau_{total\ requerido} = 3.46562 \text{ Nm}$$

- Par disponible en salida (diseño):

$$\tau_{salida} = 6.7472 \text{ Nm}$$

- Margen estático previsto ≈ 3.1 ; margen dinámico a nivel motor ≈ 1.94 .
- Resolución por paso (sin microstepping): $\approx 0.1125^\circ \rightarrow \approx 0.707 \text{ mm}$ en el extremo.

4.7 Pilar / columna del eje Z — Base estructural y montaje

La columna principal del SCARA (figura 4.1) está construida a partir de un perfil UPN80 de acero de 74 cm de altura, soldado a una placa base circular de acero ($\varnothing 400 \text{ mm}$, espesor 6 mm). La base incorpora tres orificios $\varnothing 8 \text{ mm}$ distribuidos a 120° para la fijación mediante tres regatones de goma que actúan como pies niveladores; las roscas en dichos orificios fueron soldadas para facilitar el montaje. Para refuerzo se añadieron dos nervios triangulares (catetos 70 mm) soldados a la cara externa de una de las alas del UPN, aportando rigidez a flexión en la parte inferior de la columna.



Figura 4.1 – Materiales para base del robot

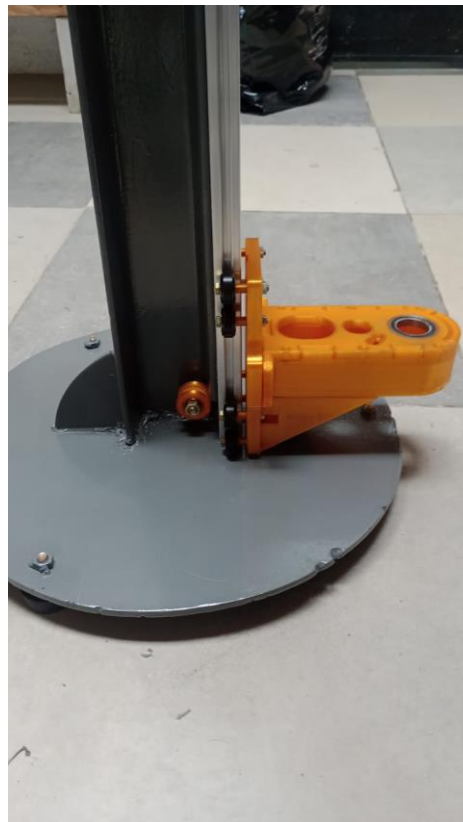


Figura 4.2 – Guía lineal de aluminio + Vslots

Sobre una de las alas del UPN se fija mediante tornillos M6 avellanados un perfil de aluminio 2040. El perfil de aluminio 2040 actúa como guía de desplazamiento del carrito superior; en sus ranuras laterales se montan 8 V-slots fijados a un carro que soporta todo el conjunto SCARA. El apriete de la fijación se diseñó de modo que la cabeza de los bulones quede rasante en la cara exterior del 2040 y no sobresalga ni en la cara externa (para no realizar una interferencia con el carrito) ni en la cara interna del UPN (para no realizar una interferencia con el sistema de contrapeso).

La distancia entre la cara del perfil de aluminio (plano de montaje) y el centro del eje final del primer segmento es de 135 mm; esta cota se emplea en los cálculos de alcance.

4.8 Guía lineal del eje Z

La guía vertical que gobierna el eje Z (Figura 4.2) está implementada mediante el perfil de aluminio 2040 y 8 V-slots montados sobre un carro único. El carro incorpora el hospedaje para soportar el conjunto del SCARA y actúa como interfaz entre la columna y los segmentos rotatorios.

- Recorrido útil (stroke) del carro: ≈ 550 mm (recorrido funcional, dentro del eje total de 740 mm del perfil).
- Final de carrera / homing: el tope inferior está materializado por un micro-switch alojado en la parte baja del carro; al descender, éste hace tope contra la base circular y activa el final de carrera.

Este esquema aporta una solución simple y robusta: la combinación 2040 + V-slots con carro garantiza rigidez lateral suficiente para el sistema en las condiciones de trabajo previstas y facilita el montaje y regulación de la correa abierta que actúa como contrapeso.

4.9 Sistema de contrapeso del eje Z

El contrapeso definitivo se fijó en 2.4 kg (medido). El esquema de operación es el siguiente: en la parte superior del eje Z se aloja la caja reductora (1:5) cuyo eje final entrega potencia a una polea GT2. Desde esta polea parte una correa lineal que corre paralela al ala externa del UPN y va conectada a la parte superior del carro. La correa continua internamente dentro del UPN hasta la parte superior del contrapeso (alojado en el interior del perfil UPN) de modo que las alas del UPN actúan como guía para el contrapeso.

- Longitud de correa abierta (desde la punta superior del contrapeso hasta la punta superior del carro): ≈ 780 mm.
- Implementación: correa GT2 6 mm, correa abierta con punto de anclaje superior en contrapeso y superior en carro; la tensión se ajusta en el montaje.

Este diseño proporciona compensación de peso efectiva, reduciendo significativamente el par estático exigido al motor Z y permitiendo la transmisión 1:5 sin necesidad de un reductor excesivo.



Figura 4.3 – Sistema de contrapeso

4.10 Fijación mediante perfil D (chaveta plana) del eje Z a la polea

En el eje de transmisión que une la salida del reductor y la polea de extracción de correa se emplea una solución de perfil plano (flat / “D”). Sobre un perno de posición $\varnothing 8$ mm se mecanizó una cara plana (flat) con anchura efectiva ≈ 6 mm, de forma que las poleas cuyo alojamiento están diseñadas con orificio en forma de “D” ($\varnothing 5.9$ mm sobre el plano) se ajustan por interferencia y transmiten torsión sin juego apreciable. Las poleas se fijan con tornillos prisioneros que inciden sobre la superficie plana, garantizando la transmisión requerida. Esta solución demuestra ser efectiva en la práctica y evita la necesidad de chavetas separadas.

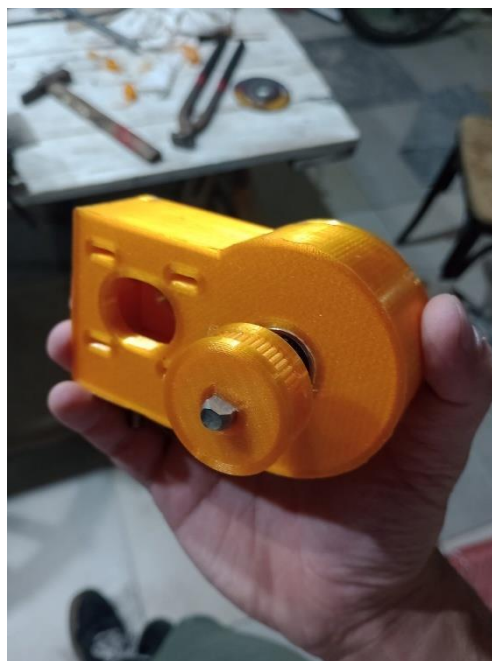


Figura 4.4 – Eje y Poleas perfil D

4.11 Poleas GT2 paramétricas y plantilla CAD

Todas las poleas utilizadas en el robot fueron generadas mediante una plantilla paramétrica en Autodesk Inventor. La plantilla incorpora las fórmulas estándar (pitch, número de dientes, diámetro primitivo, diámetro exterior y diámetros interiores) y permite variar el número de dientes para obtener automáticamente el diámetro correspondiente, manteniendo la compatibilidad con correa GT2 de 6 mm.

En el informe se documenta que las poleas están basadas en dicha plantilla paramétrica (Anexo 4.B contiene los .ipt y .stl exportados). Se refleja además la regla de diseño empleada: seleccionar número de dientes en el módulo GT2, este cambio de dientes dispara todos las demás variables.

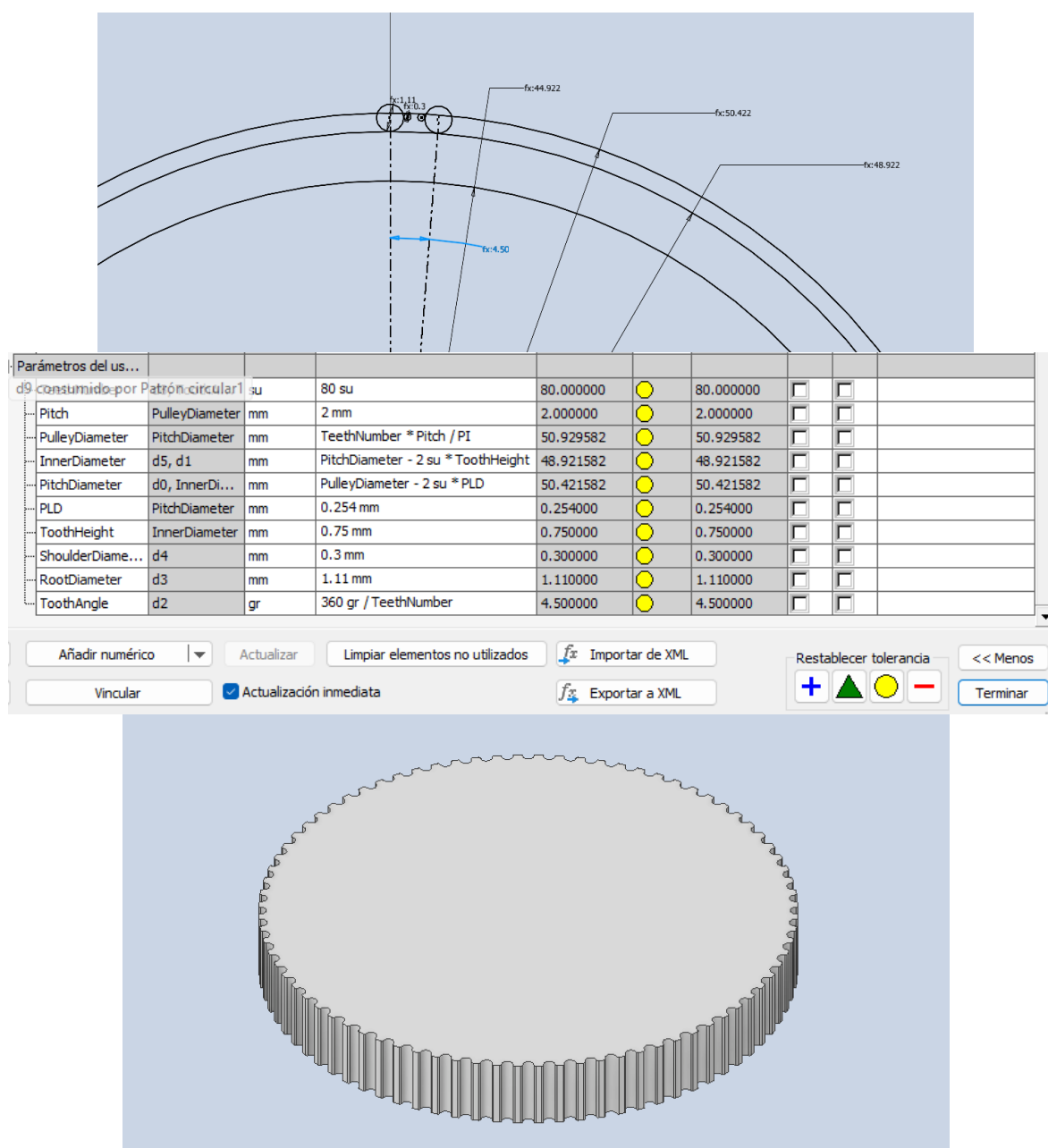


Figura 4.5, 4.6 y 4.7 – Diseño paramétrico perfil GT2

4.12 Mecanismos de transmisión por segmentos (resumen y descripción)

Se documentan a continuación los trenes de transmisión tal y como están implementados:

- **Eje Z:** mecanismo de 1 etapa (dos ejes). Polea del motor NEMA17: **16 dientes**; polea del eje conducido: **80 dientes** → relación **1:5**. Ambos ejes participan en la transmisión (uno entrega potencia, el otro recoge) y la correa hacia el contrapeso se conecta en el eje de salida.
- **Primer segmento (brazo proximal):** mecanismo en 2 etapas (tres ejes).
 - Eje motor: polea **16 dientes**.
 - Eje intermedio (loco) con polea doble: **80 dientes** (inferior) y **25 dientes** (superior).
 - Eje final: polea **80 dientes**.
 - Relaciones: primera etapa **1:5** (80/16), segunda etapa **1:3.2** (80/25). Multiplicadas dan la reducción compuesta de **16:1**.
- **Segundo segmento (brazo distal):** idéntico al primer segmento en topología: primera etapa 1:5 (80/16), segunda etapa 1:3.2 (80/25).
- **Tercer segmento (efector):** mecanismo de 1 etapa (dos ejes): motor polea **16 dientes**, polea final **80 dientes** → relación **1:5**.

En las primeras etapas (donde se monta el NEMA17) los alojamientos para ajustar la posición del motor son ranuras alargadas con forma de chaveta para permitir el tensado de la correa al desplazar el motor y fijarlo con tornillos. En las segundas etapas se emplean idlers (poleas locas) para ajustar la corona de la correa: en el primer segmento el idler empuja la correa hacia adentro; en el segundo segmento la empuja hacia afuera, medidas que permiten un ajuste fino de la tensión y reducen deslizamientos.

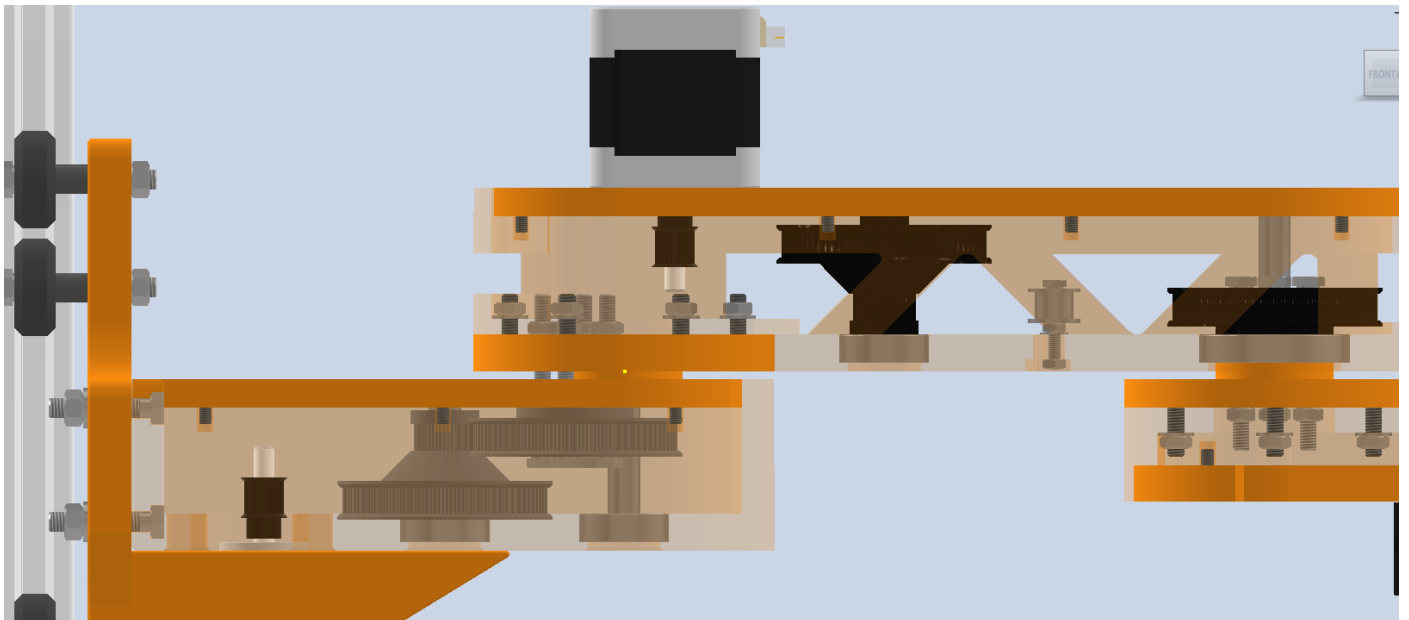


Figura 4.8 – Mecanismos de transmisión primer y segundo segmento

4.13 Fijación de la última polea al siguiente segmento (interfaz de transmisión entre segmentos)

Para garantizar rigidez en la transmisión entre segmentos se emplea el siguiente esquema, probado en el prototipo:

- En el lado que transmite potencia se monta un cojinete 6805 (de mayor diámetro) y, en el extremo opuesto, un 625. La polea que transmite tiene un asiento interno que encaja alrededor del 6805. La polea dispone además de cuatro orificios radiales que permiten pasar tornillos M4 que la fijan mecánicamente al plano del siguiente segmento. Al apretar esos tornillos se comprime el cojinete 6805, lo que elimina cualquier juego entre la cara de la polea y la cara del segmento receptor. Este sistema crea una unión rígida sin necesidad de ejes pasantes adicionales y se usa en los puntos de transmisión al final del primer, segundo y tercer segmento.
- En el lado contrario a la transmisión (no conductor) se mantiene el rodamiento 625, que permite el apoyo radial sin transmitir torsión.

El paso de cables no se realiza a través de esta unión; todos los cables se canalizan por el exterior.

4.14 Pilar / soporte de unión entre segmentos

El soporte que sostiene el rodamiento y fija la polea está realizado con un eje de acero (perno de posición) que limita la desviación lateral. La fuerza de transmisión se realiza mediante la extensión interna de la polea que atraviesa el rodamiento y se fija con tornillos M4 al siguiente segmento. Debido al diseño de sujeción por compresión del cojinete, no se aprecia juego axial significativo y no han sido necesarios topes axiales adicionales en operación normal.

4.15 Shape Generator aplicado al segundo segmento

Se aplicó Shape Generator sobre el segundo segmento con objetivo principal de reducción de masa manteniendo rigidez frente a solicitaciones de flexión. Resumen del proceso y resultados:

- Restricciones y condiciones de contorno: se definió el empotramiento en la interfaz de sujeción al primer segmento (punto de anclaje superior), se preservaron zonas sólidas (rectángulo en la cara inferior para asegurar montaje y unión de rodamientos) y se conservaron los diámetros de los alojamientos de los rulemanes.
- Cargas aplicadas: se distribuyeron cargas que simulan el esfuerzo del motor superior (dividida en cuatro cargas de 0.002 MPa) y la carga transmitida en el eje distal (0.004 MPa), además de la gravedad.
- Objetivo: minimizar masa con restricción de espesor mínimo de pared en caras críticas (superior e inferior) para priorizar resistencia a la flexión.
- Resultados: se probaron topologías con reducción de masa al 20%, 30%, 40% y 50%; la variante seleccionada y fabricada fue la de $\approx 40\%$ de reducción de masa, por ofrecer mejor balance entre rigidez y economía de material. A partir de la topología generada se diseñaron nervios triangulares y se mecanizó la pieza para obtener una geometría apropiada para impresión/maquinado.
- Implicaciones de fabricación: la topología final fue adaptada a las restricciones de impresión en PETG (orientación de capas y soportes) y mecanizado para los alojamientos de rulemanes. Resultados comparativos (masa original vs optimizada) y vistas CAD se adjuntan en Anexo 4.B.

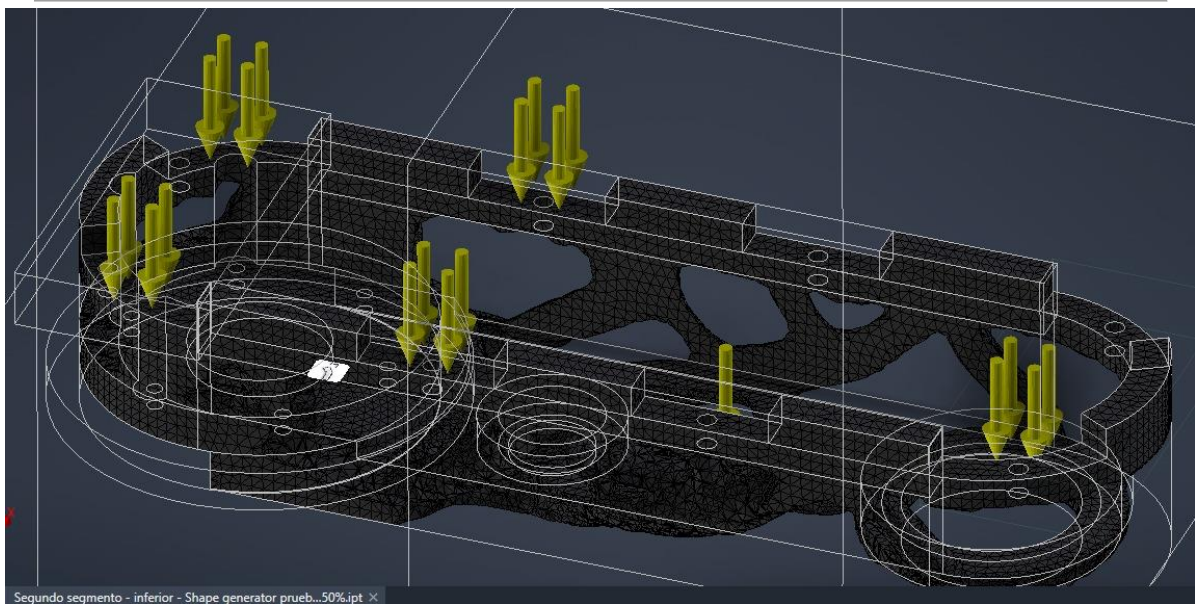


Figura 4.9 – Modelado CAD análisis de Shape generator aplicado a segundo segmento

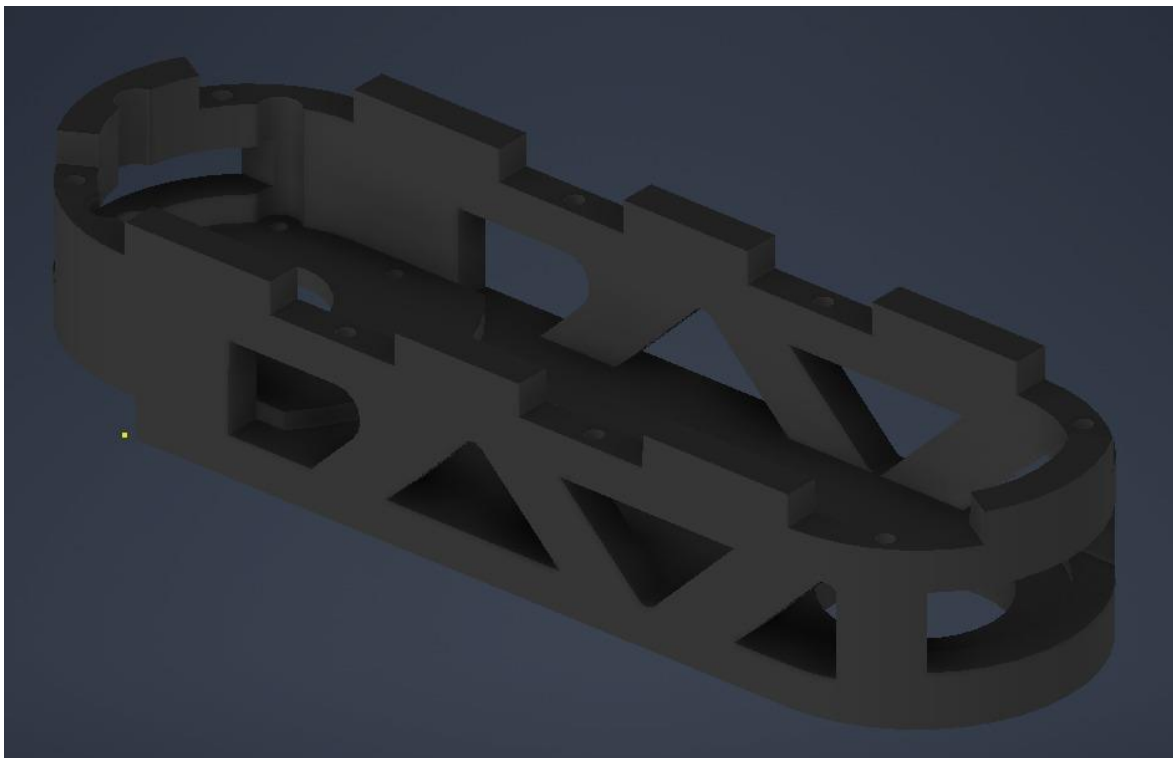


Figura 4.10 – Modelado CAD Segundo segmento forma final

4.16 Diseño del gripper y ranura de sujeción (interfaz con la pieza adaptadora)

El gripper está impreso en PETG y presenta las siguientes características principales:

- Apertura máxima 50 mm; ranura de entrada diseñada en 38 mm.
- Mecanismo de cierre con servo SG90 que actúa sobre un engranaje-cremallera: el servo acciona un piñón que transforma el giro en movimiento lineal de las garras.
- El efector incorpora alojamiento para la cámara y la electrónica del servo.
- La ranura de sujeción presenta un diseño autocentrante: vista en planta, la geometría combina un rectángulo sobre un trapecio cuyos flancos describen un cono guía que gradualmente centra la pieza al entrar; este perfil permite tolerancias de error de posicionamiento del orden de varios mm y asegura un centrado pasivo cuando el gripper cierra.

El diseño prioriza facilidad de montaje y repetibilidad: no se requieren sensores de fuerza para la toma salvo en casos extremos; el centrado geométrico es el mecanismo principal de alineado.

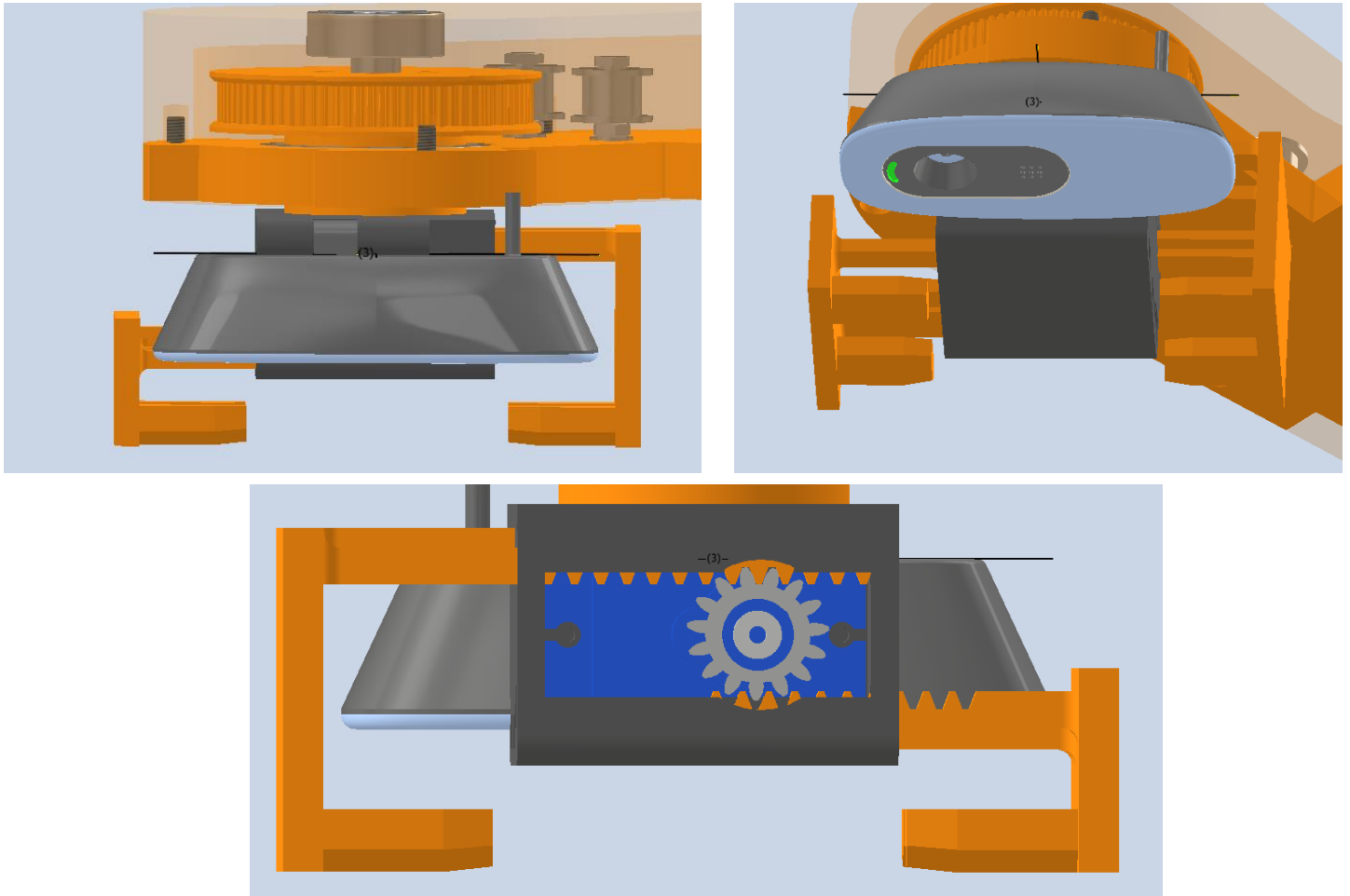


Figura 4.11, 4.12 y 4.13 – Gripper, vista frontal, vista en perspectiva, y vista trasera

4.17 Piezas críticas y consideraciones de tolerancia

Las piezas que requieren especial atención son las estructuras y tapas de cada segmento (primer, segundo y tercer segmento) ya que soportan las mayores sollicitaciones. Si bien no se especifican tolerancias geométricas estrictas en todas las piezas, se debe prestar especial atención a:

- Alojamiento de rulemanes (625 y 6805): inspección de ajuste y concentricidad; asegurar que el asiento sea coaxial y el diámetro permita montaje sin deformaciones.
- Superficies de unión entre segmentos: comprobar planitud y perpendicularidad en montaje para evitar introducción de cargas axiales no deseadas.
- Perno de posición Ø 8 mm que actúa como eje: verificar que no presente run-out apreciable.

En el proceso de montaje se recomienda verificar el ajuste de los rulemanes y corregir con ligeros ajustamientos si se detecta juego.

4.18 Gestión de cableado y trazado mecánico

La gestión de cableado se implementa como un sistema en forma de “árbol” con ramales de distintos diámetros y con uso de cadena portacables tipo piel de serpiente:

- Diámetros empleados: 3 mm, 5 mm, 8 mm y 12 mm en los distintos tramos.
- Arquitectura: los tramos con pocos cables se usan con canal de 3 mm; al agruparse se pasa a 5 mm, 8 mm y finalmente al ramal principal de 12 mm que va hasta la caja de electrónica.
- Conectividad: los motores NEMA17 y encoders disponen de conectores rápidos en sus extremos; en la caja electrónica todos los cables están conectados mediante conectores (no todo es fijo), mientras que en los tramos de brazo se utiliza termocontraíble para proteger uniones.
- Estrategia práctica: cada rama contempla una holgura razonable y un radio mínimo en las curvas para evitar fatiga; el cableado se ancla en puntos fijos en cada segmento con alivio de esfuerzos.

4.19 Disipación térmica de la electrónica

La caja electrónica del SCARA aloja la placa controladora (Arduino Mega + RAMPS), multiplexor y drivers. La solución térmica adoptada consta de un ventilador tipo cooler de PC alimentado a 12 V que extrae el aire caliente de la caja de electrónica de forma continua. La fuente de 12 V está dentro de la misma carcasa del SCARA.

- Observación práctica: la extracción forzada mediante el ventilador mantiene los drivers en régimen térmico aceptable durante ensayos; se recomienda vigilar temperaturas superficiales de los disipadores de los DRV8825 — si superan ~70 °C en operación continua, aumentar flujo de aire o mejorar disipación.
- Disposición: la electrónica se separó físicamente de las piezas impresas para evitar exposición directa al calor; se sugiere mantener camino de flujo entre la entrada (aire fresco) y salida (ventilador) con rejillas y filtros si se opera en ambientes sucios.

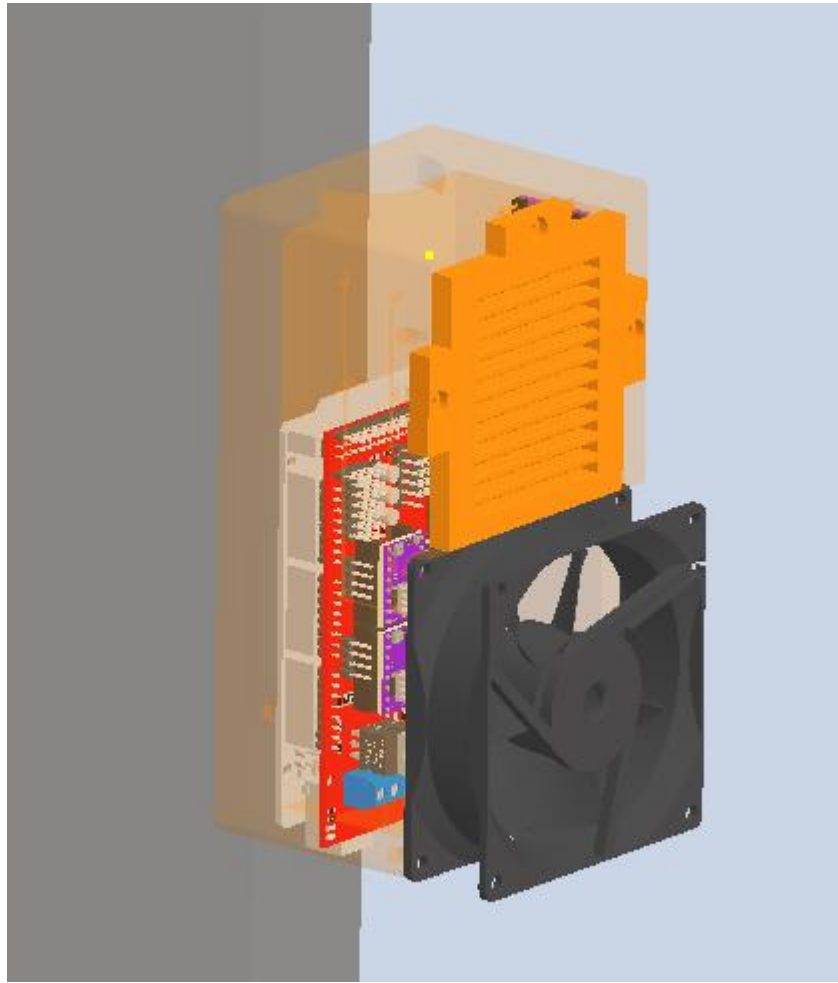


Figura 4.14 – Caja para electrónica con cooler disipador

4.20 Vinculación CAD / Anexos

Los modelos CAD, exports de STL y planos asociados a cada una de las piezas y ensamblajes descritos se han incluido en los anexos:

- Anexo 4.A — [Lista de Ensamblaje y diseño.xlsx](#)
- Anexo 4.B — Piezas [Robot SCARA](#)
- Anexo 4.C — [Ensamblaje SCARA.iam](#) (.ipt)



Figura 4.15 – Ensamblaje robot SCARA

5 Diseño Electrónico del Robot SCARA

5.1 Arquitectura electrónica

Resumen de bloques y flujo de energía / señales

- **PC (GUI Python) ↔ Serial USB → Arduino Mega 2560**
 - Gestiona interfaz de usuario, scripting, recepción de POS ... y envío de comandos (moveJ, home, enable, gripper, s).
- **Arduino Mega 2560 → RAMPS (shield) → drivers de motor (DRV8825) → Motores (NEMA17):**
 - RAMPS 1.4 actúa como placa intermedia para señales STEP/DIR/EN y alimentación 12 V.
 - DRV8825 controla los coils de cada motor vía chopping; potenciómetro Vref para I_{limit}.
- **Motores:**
 - NEMA17 (3 × alto torque; 1 × pancake).
- **Periféricos y sensores:**
 - Encoders magnéticos AS5600 (I²C), TCA9548A (multiplexor I²C), servo gripper SG90 (control PWM), endstops (INPUT_PULLUP).
- **Alimentación:**
 - PSU principal: 12 V, 15 A.
 - Pc 5 V para lógica/servos.

Asignación de pines

- J1 (X): STEP=54, DIR=55, EN=38, endstop X_MIN=3
- J2 (Y): STEP=60, DIR=61, EN=56, endstop Y_MIN=14
- Z: STEP=46, DIR=48, EN=62, endstop Z_MIN=18
- J6 (E0): STEP=26, DIR=28, EN=24, endstop E0_MIN=19
- Gripper servo: pin 11

5.2 Cálculo de consumo de corriente

Datos de diseño

- $3 \times \text{NEMA17 alto torque} \rightarrow 1.7 \text{ A por fase} \rightarrow 3 \times 1.7 \text{ A} = 5.1 \text{ A}$
- $1 \times \text{NEMA17 pancake} \rightarrow 1.0 \text{ A por fase} = 1.0 \text{ A}$
- Total motores:

$$I_{\text{motores}} = 5.1 + 1.0 = 6.1 \text{ A}$$

- Margen de seguridad 30%:

$$I_{\text{margen}} = 6.1 \times 1.3 = 7.93 \text{ A}$$

- PSU nominal considerada: 12 V

Potencia requerida

$$P = V \cdot I = 12 \text{ V} \times 7.93 \text{ A} \approx 95.2 \text{ W}$$

Fuente seleccionada

- Fuente propuesta en proyecto: 12 V, 15 A $\rightarrow P_{\text{fuente}} = 12 \cdot 15 = 180 \text{ W}$.
 - Razonamiento: cubre el consumo calculado ($\approx 95 \text{ W}$) y deja margen amplio para picos, ventilación, bombas u otros periféricos futuros.

Consumo circuito de comando

- Arduino Mega: 100 mA (asumido)
- $4 \times \text{AS5600}: 4 \times 8 \text{ mA} = 32 \text{ mA}$
- TCA9548A: 2 mA
- $4 \times \text{DRV8825 (consumo lógico)}: 4 \times 10 \text{ mA} = 40 \text{ mA}$
Total: $100 + 32 + 2 + 40 = 174 \text{ mA} \rightarrow \text{con margen práctico} \rightarrow 250 \text{ mA}$.

5.3 Selección y dimensionamiento de fuente de alimentación

Requisitos

- Riel 12 V para los drivers/motores: corriente continua estimada con margen $\approx 7.93 \text{ A}$.
- Recomendación: 12 V, 15 A (180 W). Ventajas:
 - Suficiente margen para picos y expansión.
 - Permite alimentar ventilador caja, drivers, y opcionalmente algún periférico DC.

5.4 Análisis de consumo en pines críticos del microcontrolador Límites del microcontrolador (ATmega2560)

- Corriente máxima por pin: 40 mA (límite ABSOLUTO)
- Recomendado (por seguridad y fiabilidad): ≤ 20 mA por pin
- Máx por puerto (grupo de 8 pins): ≈ 100 mA
- Máx consumo total VCC: ≈ 200 mA

Evaluación del diseño actual

- Pines STEP/DIR: señales digitales de bajo consumo (μ A–mA) \rightarrow sin riesgo.
- PC (AS5600 y TCA9548A): consumo de bus μ A–mA \rightarrow sin riesgo.

Conclusión: con la arquitectura actual no hay riesgo de exceder límites de pines del ATmega2560.

5.5 Drivers de motores y disipación térmica

1) Estimación de corriente límite (I_{limit}) a partir de V_{ref}

La relación típica (válida para muchas placas Pololu/compatibles) es aproximadamente:

$$I_{limit} \approx \frac{V_{ref}}{8 \cdot R_{sense}}$$

donde R_{sense} es la resistencia de medida (valor en la placa del drv).

Importante: diferentes placas usan distintas resistencias sense (típicamente 0.05Ω o 0.1Ω).

Si no conocés R_{sense} , conviene comprobar la placa o medir la resistencia entre los pads SENSE.

Calculemos dos casos típicos:

- Si $R_{sense} = 0.1 \Omega$:

$$I_{limit} = \frac{0.85}{8 \times 0.1} = \frac{0.85}{0.8} = 1.0625 \text{ A}$$

\rightarrow Insuficiente para motores con 1.7 A.

- Si $R_{sense} = 0.05 \Omega$:

$$I_{limit} = \frac{0.85}{8 \times 0.05} = \frac{0.85}{0.4} = 2.125 \text{ A}$$

\rightarrow Adecuado para motores configurados a 1.7 A.

Interpretación: dado que trabajás con motores de 1.7 A/fase, la segunda hipótesis ($R_{sense} \approx 0.05 \Omega$) es la coherente si $V_{ref}=0.85$ V. Verifica físicamente R_{sense} impreso en la placa o consulta la documentación del módulo. Si fuese 0.1Ω , V_{ref} debería aumentarse para alcanzar 1.7 A (con las precauciones abajo).

2) Comportamiento térmico y disipación

Contexto: la disipación en el DRV8825 depende de: tensión de alimentación (12 V), corriente por fase (I_{set}), microstepping, $R_{ds(on)}$ de MOSFETs internos y duty cycle. El cálculo exacto requiere $R_{ds(on)}$ y modelo transistor; aquí se da un método práctico + estimaciones conservadoras.

Regla práctica:

- Para $I_{limit} > 1$ A por fase conviene disipador + ventilador.
- Para $I_{limit} \approx 1.7$ A por fase, esperar que los módulos alcancen temperaturas elevadas sin ventilación — la ventilación en la caja es adecuada.

5.6 Esquemas Electrónicos Scara/Anexos

- Anexo 5.A — [Esquema Scara.png](#)

6 Control y Software del Robot SCARA

6.1 Estrategia de control

El control del SCARA está implementado en dos capas:

- Capa baja (firmware Arduino / drivers): control open-loop de motores paso a paso mediante AccelStepper (drivers en modo DRIVER). El firmware (Control SCARA.ino) se encarga del homing, ejecución de trayectorias simples y gestión de enable/disable de drivers, límites por software, stop de emergencia y control del gripper (servo).
 - Pines, límites y home definidos por macros (ej.: $X_STEP_PIN=54$, $J1_HOME_POS$, $J1_MIN/J1_MAX$, etc.).
 - Movimientos no bloqueantes: moveL (objetivos absolutos) y moveJ (deltas por eje) — moveJ construye movimientos relativos por eje y el firmware los ejecuta con AccelStepper::move() y run(). Al finalizar un movimiento se publica por serial la línea "POS J1 ... J2 ... Z ... J6 ...".
 - Homing interrumpible con lectura de endstops y setCurrentPosition() al concluir el homing.
- Capa alta (PC / GUI Python): Actúa como controlador maestro y planificador: genera comandos de alto nivel (GOTO en coordenadas cartesianas), usa la librería de cinemática inversa para calcular ángulos, pasa esos ángulos a unidades/ pasos (con MotionModel / robot_config.json) y envía comandos moveJ/moveL por serial. También gestiona scripts, jog, joystick PS4 y la lógica de integración con visión (p. ej. obtener_ecu_y_guardar() que solicita coordenadas del sistema de visión). Ejemplos de uso: envío de moveJ J1 1000 800 400 (delta, vmax, amax). Ver implementación y parsing en la GUI.

Ventajas de la estrategia actual

- Arquitectura sencilla y robusta para prototipado (firmware simple + procesador potente en PC).
- Separación clara entre la cinemática/planificación (PC) y la ejecución de bajos niveles (firmware).
- Permite operación manual, scripting y teleoperación por joystick.

Limitaciones / riesgos

- Control *open-loop* (steppers sin encoders): sin retroalimentación, ocurren pérdidas de pasos por sobrecarga o colisiones; el firmware no corrige posición en tiempo real.
- Protocolos de comunicación simples (sin ACKs): posible pérdida o desincronización si hay ruido serial.
- Parámetros "duros" en firmware (límites y constantes) que requieren mantenimiento manual si se cambian mecánicos o configuración de microstepping.

6.2 Implementación de cinemática inversa

Librería usada

La cinemática está implementada en `scara_kinematics.py` (módulo propio): ofrece `forward_kinematics(...)` e `inverse_kinematics(...)`. En ese archivo se definen longitudes mecánicas y offsets (ej.: $L1 = 160$, $L2 = 200$, $OFFSET_X = 135$) y utilidades para convertir grados↔radianes.

`scara_kinematics`

Flujo de cálculo

1. Usuario/ rutina indica objetivo cartesiano (X, Y, Z, ϕ_{abs}).
2. La GUI llama `inverse_kinematics(x, y, z, phi_abs)` y obtiene una (o varias) soluciones para $(\theta_1, \theta_2, d, \phi_m)$.

`scara_kinematics`

3. Se convierte cada ángulo/posición en pasos mediante `MotionModel.units_to_steps(eje, valor)` (usa `robot_config.json` con pasos/mm o pasos/deg).
4. Se envía `moveJ` al firmware; el firmware ejecuta el movimiento y, al terminar, publica la posición actual (POS ...) que la GUI usa para actualizar la interfaz.

Consideraciones numéricas

- `inverse_kinematics` devuelve 0 o 2 soluciones (codo arriba/abajo). La GUI elige la primera solución válida o puede elegir según criterio (mínimo movimiento, evitar singularidades).

- Es crítico que los parámetros geométricos (L1, L2, OFFSET_X) sean correctos y estén sincronizados entre la documentación, la librería y cualquier archivo de configuración (para evitar errores de IK).

6.3 Secuencia de homing

Descripción de la secuencia implementada (firmware)

- El comando serial home inicia homing eje a eje: se llama homeAxis(...) para J1, J2, Z. J6 no se mueve (simplemente setCurrentPosition(J6_HOME_POS)).
- homeAxis funciona así:
 - Puede aplicar una inversión temporal de los pines de dirección para asegurar el sentido correcto (tempInvert para J2 está activado en el código actual).
 - Ajusta velocidad con setMaxSpeed / setSpeed y avanza hasta que el endstop cambia de estado (en el código: while (digitalRead(endstopPin) == HIGH) — por lo tanto los endstops están configurados como INPUT_PULLUP y normalmente HIGH).
 - Al detectarse el endstop hace delay(200) y motor.setCurrentPosition(homingPosition) para fijar la posición home definida por macro (J1_HOME_POS, etc.).
- El homing es interrumpible por el comando s (stop).

6.4 Comunicación con la PC

Protocolo actual y elementos clave

- Baudrate: el firmware inicializa Serial.begin(115200); y la GUI abre puerto serial con BAUDRATE definido, timeout y listener. La GUI no bloquea esperando confirmación de los comandos; en cambio, hay un *listener* que muestra las líneas recibidas del Arduino en el log (la GUI interpreta líneas como "POS J1 ..." para actualizar estado).

GUI

- Comandos soportados (por firmware):
 - home — homing completo.
 - moveJ J1 <steps> <vmax> <amax> ... — movimiento relativo por ejes (pares por eje).
 - moveL J1 <target> J2 <target> ... — objetivos absolutos (no bloqueante).
 - enable on/off — energizar / desenergizar drivers.
 - gripper open / gripper close — control servo.
 - s o línea vacía — STOP global.
 - Al terminar movimientos, firmware envía Movimiento finalizado y POS J1 ... con posiciones en pasos.

6.5 Integración con el sistema de visión

Arquitectura de integración

- El sistema de visión (VisionArtificial.py) detecta marcadores ArUco, filtra la pose y ejecuta YOLO on-demand / periódico para clasificar tipo de ECU y asociar bounding box → marcador; además expone un servidor TCP que responde a GET devolviendo {"type":..., "X":..., "Y":..., "Z":..., "Yaw":...} con la posición en marco *world* (ya aplicada la corrección por offsets del tipo de ECU).

VisionArtificial

- La GUI ejecuta obtener_ecu_y_guardar() que invoca el módulo de visión y luego realiza:
 1. Conversión de la pose cartesiana de visión → IK (scara_kinematics.inverse_kinematics).
 2. Conversión de ángulos a pasos (MotionModel).
 3. Envío de comando moveJ al firmware para ejecutar pick/place.

GUI

Puntos críticos y comprobaciones

- Hand-eye / calibración: la precisión de la conversión pixel→world→robot depende completamente de:
 - calibración intrínseca de la cámara (archivo calibracion_camara.npz que el vision code carga),
 - definición correcta de known_markers_world (matrices T_{w_m} en mm),
 - y la coherencia de unidades entre visión y cinemática (mm/°).
Si estos no están ajustados, la IK recibirá coordenadas erróneas y la manipulación fallará.
- Asociación bbox ↔ marker: el código usa distancia centro bbox ↔ centroid marker con umbral MATCH_MAX_DIST_PIX. Debes verificar este umbral en distintas condiciones de iluminación/zoom.
- Tiempo y sincronización: la GUI llama al servidor de visión y espera resultado; visión puede ejecutar YOLO on-demand y toma varios frames — la latencia debe medirse y tolerancias definidas.

6.6 Códigos Scara/Anexos

- Anexo 6.A — Código/SCARA/Complementos [motion_model.py](#)
- Anexo 6.B — Código/SCARA/Complementos [scara_kinematics.py](#)
- Anexo 6.C — Código/SCARA [Control SCARA.ino](#)
- Anexo 6.D — Código/SCARA [GUI_Scara.py](#)

BLOQUE II – LAVADORA DE ECUs

7. DISEÑO MECÁNICO DE LA LAVADORA DE ECUs

7.1 Requerimientos funcionales

La lavadora de ECUs fue concebida como un módulo automatizado destinado a la limpieza controlada de placas electrónicas automotrices. El sistema debe aceptar placas con dimensiones máximas de:

- Largo: 125 mm
 - Ancho: 105 mm
 - Espesor de PCB: 1,6 mm
 - Peso máximo considerado: 150 g
- El sistema opera con una sola ECU por ciclo, con un tiempo objetivo total de aproximadamente 30 minutos, incluyendo etapa de aspersión y limpieza ultrasónica. El movimiento principal es realizado por un sistema XY que desplaza la ECU dentro de la cámara de lavado, permitiendo además la rotación de la pieza para garantizar la limpieza automática de ambas caras sin intervención manual. El nivel de automatización es total: una vez iniciado el ciclo, el proceso se ejecuta sin asistencia del operador. El fluido de limpieza utilizado es alcohol isopropílico al 99%, lo que impone restricciones adicionales en términos de compatibilidad química y seguridad. El diseño contempla integración física y funcional con el robot SCARA, aunque puede operar como módulo independiente. Desde el punto de vista de diseño proyectado, se plantea un grado de protección IP68; sin embargo, el prototipo actual no incorpora juntas industriales ni sellos certificados, por lo que este valor debe considerarse como objetivo de diseño futuro.

7.2 Diseño del sistema de movimiento XY

El sistema de posicionamiento está compuesto por:

- Guías lineales con varillas lisas.
 - Carro desplazable impreso en 3D.
 - Transmisión mediante tornillo helicoidal tipo CNC.
 - Accionamiento directo mediante motores paso a paso NEMA17.
- El tornillo helicoidal convierte el movimiento rotacional del motor en desplazamiento lineal del carro, garantizando precisión y control posicional. La masa total desplazada por el sistema (estructura móvil + ECU + motores + mecanismo de rotación) es aproximadamente:
- $$m = 1,2 \text{ kg}$$
- La precisión objetivo del sistema es del orden de $\pm 1 \text{ mm}$, coherente con la resolución del motor paso a paso y el paso del tornillo helicoidal. El movimiento vertical (Z) permite una elevación de 50 mm hacia arriba y 50 mm hacia abajo, controlado mediante finales de carrera mecánicos.

7.3 Sistema de sujeción de la ECU

La ECU se sujeta por los bordes mediante un sistema de grippers actuados. La presión se aplica lateralmente evitando esfuerzos sobre componentes electrónicos.

Características:

- Sujeción por bordes.
- Sistema de volteo automático.
- Recubrimiento de contacto con goma.
- Capacidad para soportar vibraciones del sistema ultrasónico.
- Accionamiento mediante servomotores MG946.

El diseño busca aplicar la mínima fuerza necesaria para garantizar fijación sin comprometer la integridad estructural del PCB.

7.4 Sistema de aspersión

El sistema de aspersión está compuesto por:

- Dos inyectores de nafta automotrices adaptados.
- Presión de trabajo: 3 bar.
- Distancia boquilla–placa: entre 20 mm y 40 mm.
- Tiempo mínimo de exposición: 5 minutos.
- Tanque abierto de 5 litros.
- Tuberías de poliamida.
- Bomba eléctrica automotriz de 12 V.

La presión de 3 bar permite generar un chorro atomizado suficiente para remover contaminantes adheridos sin dañar componentes electrónicos.

El funcionamiento es intermitente, ya que no se dispone de regulador de presión.

7.5 Sistema de limpieza ultrasónica

El diseño proyectado contempla un sistema ultrasónico con:

- Frecuencia nominal: 40 kHz.
- Volumen de batea: 2 litros.
- Material: chapa común.
- Tiempo de inmersión teórico: 15 minutos.
- Inmersión total de la ECU.

No se dispone actualmente de control de temperatura ni sistema de secado posterior.

Desde el punto de vista técnico, 40 kHz constituye una frecuencia adecuada para limpieza fina en placas electrónicas, ya que genera cavitación controlada sin agresividad excesiva.

Para el dimensionamiento conceptual se adopta una potencia estimada de 100 W, coherente con el volumen de trabajo.

7.6 Diseño estructural y hermetización

La estructura principal está fabricada en:

- Chapa galvanizada de 2 mm.
 - Paneles de acrílico transparente para inspección visual.
 - Tapa superior.
 - Sellado mediante adhesivo tipo parabrisas.
- La estructura es soldada con estaño.

No posee:

- Juntas industriales.
- Bandeja de contención.
- Aislamiento acústico.

Las dimensiones totales quedan pendientes de definición.

7.7 Consideraciones de seguridad

El prototipo actual no incorpora:

- Ventilación forzada.
- Sensores de tapa.
- Sensor de nivel.
- Sensor de temperatura.
- Protección anti-chispa.
- Normativa ATEX.

Dado que se emplea alcohol isopropílico al 99%, inflamable, la implementación futura deberá incluir sistemas de ventilación, separación de compartimentos húmedo/seco y protecciones eléctricas certificadas.

8. CÁLCULOS MECÁNICOS

8.1 Cálculo de fuerzas en el sistema XY

Masa desplazada:

$$m = 1,2 \text{ kg}$$

Si consideramos una aceleración moderada de:

$$a = 0,5 \text{ m/s}^2$$

La fuerza necesaria es:

$$F = m \cdot a$$

$$F = 1,2 \times 0,5$$

$$F = 0,6 \text{ N}$$

Incluso considerando fricción y pérdidas, suponiendo un coeficiente de fricción equivalente a 0,2:

$$F_{\text{total}} \approx 1 \text{ N}$$

Este valor es extremadamente bajo comparado con la capacidad de un NEMA17 ($\approx 0,4 \text{ N}\cdot\text{m}$ de torque).

Si el tornillo helicoidal tiene un radio efectivo de 5 mm:

$$r = 0,005 \text{ m}$$

El torque requerido es:

$$T = F \times r$$

$$T = 1 \times 0,005$$

$$T = 0,005 \text{ N}\cdot\text{m}$$

Comparado con:

$T_{\text{motor}} \approx 0,4 \text{ N}\cdot\text{m}$

Existe un margen de seguridad superior a 80 veces.

Conclusión: el sistema está sobredimensionado desde el punto de vista de torque.

8.2 Cálculo de presión de aspersión

Presión nominal:

$P = 3 \text{ bar} = 300.000 \text{ Pa}$

La velocidad teórica de salida (Bernoulli):

$$v = \sqrt{(2P/\rho)}$$

Densidad alcohol isopropílico $\approx 785 \text{ kg/m}^3$

$$v = \sqrt{(2 \times 300.000 / 785)}$$

$$v \approx 27,6 \text{ m/s}$$

Esta velocidad garantiza energía cinética suficiente para desprender contaminantes superficiales.

8.3 Selección de bomba

Bomba automotriz:

- Tensión: 12 V
- Corriente estimada: 10 A
- Funcionamiento intermitente
- Presión nominal: 3 bar

Potencia eléctrica:

$$P = V \times I$$

$$P = 12 \times 10$$

$$P = 120 \text{ W}$$

8.4 Cálculo de potencia mecánica del sistema XY

Trabajo para elevar 1,2 kg 50 mm:

$$W = m \cdot g \cdot h$$

$$W = 1,2 \times 9,81 \times 0,05$$

$$W \approx 0,59 \text{ J}$$

Incluso repitiendo el ciclo varias veces, la potencia requerida es insignificante frente a la bomba.

Conclusión: la bomba es el principal consumidor energético del sistema.

8.5 Análisis de vibraciones

El sistema ultrasónico (40 kHz) opera a frecuencias muy superiores a la frecuencia natural estructural estimada (<100 Hz), por lo que no existe riesgo de resonancia estructural con el sistema XY.

El desacople actual se limita al adhesivo de montaje, lo cual no constituye un aislamiento vibratorio formal.

9. DISEÑO ELECTRÓNICO DE LA LAVADORA

9.1 Arquitectura electrónica

La lavadora de ECUs posee una arquitectura electrónica independiente del robot SCARA, basada en:

- Microcontrolador: Arduino Mega 2560
- Shield de potencia: RAMPS 1.4
- Drivers para motores paso a paso
- Fuente principal de 12 V
- Fuente secundaria de 5 V para servomotores
- Etapas de conmutación para bomba e inyectores

El sistema se divide en tres bloques principales:

1 Bloque de control

- Arduino Mega
- Comunicación con PC vía USB
- Lógica programada de seguridad
- Gestión de secuencia automática

2 Bloque de potencia 12 V

- Bomba automotriz (3 bar)
- Dos inyectores de nafta
- Dos motores paso a paso NEMA17
- Electrónica auxiliar de 12 V

3 Bloque de potencia 5 V

- Cuatro servomotores MG946
- Lógica de control de actuadores

La comunicación con la PC permite iniciar ciclos, supervisar estado y enviar comandos.

9.2 Cálculo de consumo total del sistema

Se consideran los siguientes consumos estimados:

Motores paso a paso

$2 \times \text{NEMA17}$

Corriente nominal: 1,7 A por motor

Total stepper = $2 \times 1,7$

Total stepper = 3,4 A

Bomba automotriz

Corriente estimada:

bomba = 10 A

Inyectores de nafta

Corriente típica ≈ 1 A cada uno

inyectores = 2×1

inyectores = 2 A

Servomotores MG946

Consumo pico $\approx 2,5$ A cada uno

Sin embargo, definiste que nunca funcionan más de 2 simultáneamente:

Servos max = $2 \times 2,5$

servos max = 5 A

Consumo electrónica (Arduino + lógica)

Aproximadamente:

control $\approx 0,3$ A

Corriente total en línea de 12 V

Motores + bomba + inyectores:

Corriente 12V = $3,4 + 10 + 2$

Corriente 12V = $15,4$ A

Aplicando margen de seguridad del 30%:

Corriente 12V_diseño = $15,4 \times 1,3$

Corriente 12V_diseño ≈ 20 A

Cabe destacar que no funciona todo al mismo tiempo, este es un hipotético caso.

Corriente total en línea de 5 V

Servos + lógica:

Corriente 5V = $5 + 0,3$

Corriente 5V = $5,3$ A

Aplicando margen del 30%:

Corriente 5V diseño ≈ 7 A

9.3 Dimensionamiento de fuentes de alimentación

Fuente 12 V

Corriente mínima requerida:

20 A

Potencia:

$P = V \times I$

$P = 12 \times 20$

$P = 240$ W

Se recomienda una fuente de:

12 V – 25 A (300 W)

Esto permite margen térmico y evita operación al límite.

Fuente 5 V para servos

Corriente requerida:

7 A

Potencia:

$P = 5 \times 7$

$P = 35$ W

Se recomienda una fuente externa de:

5 V – 10 A

No se recomienda alimentar los servos desde el regulador del Arduino debido a las corrientes elevadas y picos transitorios

9.4 Sistema de protección contra sobrecorriente

Actualmente el sistema incorpora:

- Fusible dedicado para la bomba de nafta.
Desde el punto de vista de ingeniería, deberían incorporarse:
 1. Fusible general en la entrada de 12 V.
 2. Fusible independiente para motores.
 3. Protección contra inversión de polaridad.
 4. Diodos flyback en cargas inductivas.
 5. Relé o MOSFET de potencia con disipación térmica adecuada.

La ausencia de estas protecciones constituye una limitación del prototipo y se propone como mejora futura.

9.5 Sensores de fin de carrera y lógica de seguridad

El sistema dispone de:

- 2 finales de carrera mecánicos
 - Uno para límite superior.
 - Uno para límite inferior.
- Lógica de seguridad implementada por software.

Funcionamiento:

Si el sistema detecta activación de final de carrera, detiene el movimiento del eje correspondiente.

No se dispone de:

- Sensor de tapa.
- Sensor de nivel.
- Sensor de temperatura.
- Detección de sobrecorriente programada.

Desde el punto de vista de seguridad industrial, estos sensores deberían incorporarse en una versión industrializada del sistema.

9.6 Esquema eléctrico general (descripción conceptual)

El esquema eléctrico general se estructura de la siguiente manera:

Entrada 220 V AC

→ Fuente switching 12 V

→ Distribución a:

- Drivers NEMA17
- Bomba
- Inyectores
- Regulador / fuente 5 V externa

Fuente 5 V

→ Servomotores

→ Lógica auxiliar

Arduino Mega

→ Señales de control a:

- Drivers paso a paso
- Servos
- Relé/MOSFET bomba
- Inyectores

-
- Lectura finales de carrera
Se recomienda mantener separación física entre:
 - Zona de potencia
 - Zona de control
 - Zona húmeda
En el prototipo actual esta separación no es formal, lo que constituye otra mejora futura.

Conclusión Técnica del Bloque II

Desde el punto de vista de ingeniería:

- El sistema mecánico está sobredimensionado en torque.
- La bomba representa el mayor consumo energético.
- El sistema eléctrico requiere mejoras en protección.
- La arquitectura es funcional para prototipo académico.
- Existen oportunidades claras de mejora en seguridad industrial.

9.7 Lavadora de ECUs / Anexos

Los Esquemas electrónicos y códigos de la lavadora de ECUs se han incluido en los anexos:

- Anexo 9.A — [Esquema Lavadora.png](#)
- Anexo 9.B — [Comunicacion.ino](#)
- Anexo 9.C — [Lavadora ECU.ino](#)

BLOQUE III – SISTEMA DE VISIÓN E INTELIGENCIA ARTIFICIAL

10. Sistema de Visión Artificial

10.1 Selección de cámara

La cámara que se utilizó para este proyecto es una Logitech C270 de resolución 720p, la se optó por una cámara de 720p por el presupuesto que involucraba comprar una cámara de más alta calidad (las cámaras de 1080p estaban más del triple), se eligió el modelo Logitech C270 por sus buenas reseñas y relación precio calidad.

10.2 Flujo de procesamiento de imagen

El sistema procesa cada fotograma mediante un pipeline determinista orientado a estimación de pose y clasificación tipo ECU. El flujo implementado en el prototipo es:

1. **Adquisición** — captura directo con OpenCV (VideoCapture(0)).
2. **Preprocesado** — conversión a gris para la detección de marcadores ArUco.
3. **Detección ArUco** — ArucoDetector.detectMarkers obtiene esquinas y IDs.
4. **Construcción correspondencias WORLD↔IMG** — para cada marcador fijo conocido (tabla known_markers_world) se construyen puntos 3D en WORLD y sus proyecciones en píxeles.

5. **Estimación pose cámara en WORLD** — solvePnP con los marcadores fijos → $rvec_cam$, $tvec_cam$, T_cam_world y su inversa T_world_cam . Esto define el transformador que permite pasar entre frames.
6. **Candidatos objetivo (target_id)** — se extraen centroides de los marcadores detectados con $id == target_id$.
7. **Detección YOLO (OBB/boxes)** — inferencia en RGB con el modelo entrenado (best.pt) en dos modos: **on_demand** (trigger por p o GET TCP) o **periodic** (cada YOLO_PERIOD frames). Se selecciona la detección de mayor confianza y se guarda en historial para voting.
8. **Asociación bbox↔marker y locking** — si existe bbox y candidatos marker se asocian por mínima distancia en píxeles (umbral MATCH_MAX_DIST_PIX); se mantiene un lock temporal con tolerancia LOCKED_MAX_MISSES.
9. **Estimación pose del objeto** — solvePnP sobre los corners del marker lockeado → T_cam_obj ; si T_world_cam es válido: $T_world_obj = T_world_cam @ T_cam_obj$.
10. **Suavizado** — filtrado anisotrópico en X/Y/Z (α_{xyz}) y SLERP para rotación (α_{rot}), con fase de convergencia rápida tras adquisición de lock (parámetros FAST_*).
11. **Salida/visualización** — ejes proyectados, bbox (si aplica), y etiqueta de tipo en esquina superior derecha con color por clase.
12. **Interfaz** — servidor TCP (GET/EXIT) que puede disparar una corrida YOLO on-demand y devuelve JSON {type, X, Y, Z, Yaw} con OFFSET aplicado por tipo.

10.3 Modelo de inteligencia artificial utilizado

Arquitectura y pesos

- **Backbone / arquitectura:** YOLOv8m-obbb (modelo base yolov8m-obbb.pt usado como pre-entrenado y adaptado por transfer learning). El resumen del modelo en el log indica 320 capas y ≈ 26.4 M parámetros; la cabeza final es de tipo **OBB** para cajas orientadas.
- **Framework:** Ultralytics.
- **Formato de salida:** soporta boxes (xyxy) y obbb (oriented bounding boxes). El código toma tanto r.boxes como r.obbb y elige la detección de mayor confianza.

Modo de uso

- Inferencia vía model.predict.
- Filtrado: MIN_CONF_ACCEPT = 0.30 en el pipeline.

- El sistema guarda la mejor detección por frame y mantiene un historial de etiquetas (LABEL_HISTORY_LEN = 7) para voting (acepta etiqueta si \geq LABEL_VOTE_MIN).

Rendimiento medido

- Precisión (BBOX): **0.994**
- Recall (BBOX): **0.994**
- mAP50: **0.992**
- mAP50-95: **0.990**
- Velocidad promedio por imagen: **0.538 ms preprocess + 10.401 ms inference + 3.867 ms postprocess = 14.805 ms / imagen $\rightarrow \approx 67.5$ FPS.**

10.4 Entrenamiento y validación

Plataforma / entorno

- Entrenamiento realizado en Google Colab (Tesla T4 GPU, ~15 GB memoria GPU). Ultralytics v8.3.30 y PyTorch con CUDA:0.

Dataset

- Estructura: entrenamiento en /content/drive/MyDrive/Entrenamiento9/train y validación en /content/drive/MyDrive/Entrenamiento9/valid.
- Número de imágenes: train: 2703 imágenes, valid: 256 imágenes.
- Distribución por clase: Ecu-1 = 43 instancias, Ecu-2 = 181 instancias (total 224).

Hiperparámetros y configuración

- epochs=120, imgsz=640, batch=16, device='cuda', workers=2.
- Optimizer: auto \rightarrow Ultralytics seleccionó **AdamW** con $lr \approx 0.001667$ y momentum 0.9.
- Augmentations destacadas: Blur, MedianBlur, ToGray, CLAHE; mosaic=1.0; translate=0.1, scale=0.5, fliplr=0.5, entre otros parámetros estándar de la configuración.
- Loss head: box/cls/df/pose/... configurados por defecto del head OBB (DFL activo).

Curvas y convergencia

- El entrenamiento muestra reducción sostenida de box_loss, cls_loss y df_loss durante las 120 epochs.
- El log final muestra box_loss ≈ 0.1546 , cls_loss ≈ 0.1346 , df_loss ≈ 1.373 en epoch final.
- Tiempo total entrenamiento: **120 epochs en 2.112 horas $\rightarrow 2$ h 6 min 43.2 s.**

Resultados de validación final

- **All:** precision 0.994 — recall 0.994 — **mAP50 = 0.992** — **mAP50-95 = 0.990**.
- **Por clase:**
 - Ecu-1 (43 inst.): P=0.999, R=1.000, mAP50=0.995, mAP50-95=0.992
 - Ecu-2 (181 inst.): P=0.989, R=0.989, mAP50=0.989, mAP50-95=0.987

Inference speed (métricas de tiempo)

- Preprocess 0.538 ms | Inference 10.401 ms | Postprocess 3.867 ms → **≈14.805 ms / imagen (≈67.5 FPS)**. Esto es medido en el entorno Colab con batch per image y configura un buen margen para uso en tiempo real en CPU/GPU según despliegue.

Archivos generados

- Pesos: last.pt y best.pt (≈53.2 MB).
- Carpetas con logs, labels.jpg (plot labels), TensorBoard logs y curvas de metrics en project/ecu_yolo_obb.

10.5 Conversión de coordenadas píxel a coordenadas reales

- La calibración intrínseca se carga desde calibracion_camara.npz produciendo K_y dist.
- Para estimar la pose de la cámara en el sistema WORLD se construyen correspondencias 3D (puntos world de marcadores fijos) ↔ 2D (corners detectados), y se aplica $cv2.solvePnP \rightarrow rvec_cam, tvec_cam$. Esto define $T_{cam \leftarrow world}$. Su inversa $T_{world \leftarrow cam}$ se calcula por inversión de la rotación y traslación.
- Para la pose del objeto (marcador objetivo) se utiliza solvePnP con marker_corners_local (4 puntos en 3D definidas por markerLength = 37.0 mm) y los corners detectados en imagen; esto da $T_{cam \leftarrow obj}$. Entonces:

$$T_{world \leftarrow obj} = T_{world \leftarrow cam} T_{cam \leftarrow obj}$$

y la posición en WORLD es la traslación de $T_{world \leftarrow obj}$.

- El pipeline aplica offsets por tipo rotados sólo en Z por el yaw estimado:

$$\mathbf{p}_{ecu} = \mathbf{p}_{obj} + R_z(\text{yaw}) \text{ OFFSET}$$

Filtrado y robustez

- Se aplica filtrado anisotrópico en X,Y,Z con alpha_xyz y SLERP para rotaciones con alpha_rot. Al adquirir un lock se dispone una fase de convergencia rápida (FAST_ALPHA_*) o una inicialización instantánea (INSTANT_INIT_ON_LOCK).

10.6 Comunicación entre PC y microcontrolador

Protocolo implementado

- **Servidor TCP** en 0.0.0.0:5000 (socket TCP, single-client loop, SO_REUSEADDR).
- **Comandos:**
 - GET — dispara corrida YOLO on-demand (si el modelo está cargado), espera hasta YOLO_WAIT_TIMEOUT segundos por los frames solicitados y devuelve JSON con {"type":..., "X":..., "Y":..., "Z":..., "Yaw":...} o {"error":"no detection"}.
 - EXIT — responde {"status":"bye"} y cierra la sesión.
 - Otros comandos → {"error":"unknown command"}.
- **Formato:** ASCII/UTF-8 JSON simple, fácil de parsear en MCU/PLC.
- **Comportamiento y robustez:** el servidor utiliza sincronización (yolo_done_event) para coordinar la ejecución on-demand; se utiliza force_yolo_frames para asegurar que el bucle principal ejecute YOLO los siguientes N frames. Se usan timeouts y accept() con timeout para un apagado limpio.

10.7 Códigos Vision/Anexos

- Anexo 10.A — Código/Vision Artificial [VisionArtificial.py](#)

BLOQUE IV – INTEGRACIÓN DEL SISTEMA

11. Integración del sistema

Esta sección describe cómo se conectan y coordinan entre sí los subsistemas del proyecto: visión (PC + cámara), manipulador SCARA, y la “caja” lavadora. El texto es descriptivo y refleja la implementación actual del prototipo (arquitectura, protocolos, topología física, flujo operativo, manejo de errores y limitaciones conocidas).

11.1 Arquitectura general

La arquitectura es en topología estrella con la PC como nodo central. Los subsistemas y su relación son:

- PC (nodo maestro): ejecuta el software de visión y la interfaz de usuario, recibe imágenes de la cámara, decide qué placa procesar y envía coordenadas/comandos al robot.
- Robot SCARA: controlado por una placa **Arduino** + RAMPS (firmware propio). Recibe coordenadas y comandos de la PC y ejecuta movimientos.
- Caja lavadora: controlada por una placa (Arduino + RAMPS) con lógica local para maniobrar sus grippers, bombas y servos; se comunica con la PC/robot mediante un ESP32 (nodo de I/O).
- Cámara: conecta con la PC y ejecuta el código de detección (YOLO v8m).
La PC orquesta el proceso; los microcontroladores ejecutan la lógica local mínima (control de motores, drivers, lectura de sensores) y actúan a nivel de actuadores cuando reciben órdenes.

11.2 Protocolos y señales

Resumen de los protocolos y buses utilizados:

- TCP / UART: la comunicación entre la PC y el firmware del robot se establece vía USB–Serial (conexión serie sobre USB). Asimismo se utiliza TCP para la comunicación con el código de la cámara según la implementación actual.
- I²C: bus local usado internamente en el robot para encoders / sensores (AS5600 y multiplexor TCA9548A).
- Señales GPIO: entre el ESP32 y el Arduino de la caja se usan pines digitales directos para comandos sencillos (p. ej. “cerrar gripper”, “iniciar ciclo”).
- Sin handshake: actualmente no existe protocolo de ACK/handshake formal entre SCARA ↔ lavadora; la comunicación es mayoritariamente unidireccional y basada en comandos simples.

Nota: cuando se menciona “UART/TCP” se refiere a los canales usados en la práctica: comunicación serie sobre USB para microcontroladores y TCP/Socket para servicios de visión si están desplegados en procesos distintos.

11.3 Topología física del cableado

La topología física implementada es la siguiente:

- PC ↔ Robot: cable USB (USB–Serial) conectado a la placa Arduino del robot (longitud típica en bancada).
- PC ↔ Cámara: conexión directa de la cámara a la PC (USB o flujo de red según la configuración del software de visión).
- ESP32 ↔ Caja lavadora (Arduino): conexión por pines (GPIO) entre ESP32 y Arduino de la lavadora.
- Cableado de potencia: cada subsistema dispone de su fuente (SCARA: 12 V 15 A; lavadora: 12 V 20 A y 5 V 20 A) y líneas de alimentación separadas; el control de señales usa las conexiones descritas arriba.

11.4 Flujo operativo (secuencia normal de trabajo)

Flujo ideal (paso a paso) tal como se ejecuta en el prototipo:

1. El operario pulsa Play en la interfaz de la PC.
2. La PC inicia el programa de visión; la cámara escanea las zonas predefinidas.
3. Si la cámara detecta QRs / placas, la PC selecciona una (prioridad según criterio) y envía las coordenadas de pick al robot.
4. El robot recibe las coordenadas y ejecuta la rutina de desplazamiento para agarrar la placa.
5. El robot traslada la pieza hasta la zona de transferencia frente a la lavadora y se posiciona en el punto de entrega.
6. El robot espera unos segundos y envía señal a la lavadora (vía ESP32) para que cierre su gripper y asegure la pieza.
7. Una vez confirmado el agarre (por tiempo/retardo en el prototipo), el robot abre su gripper y se retira.
8. La PC envía la orden de inicio de ciclo a la lavadora; la lavadora ejecuta el ciclo configurado (aspersión, rotación, inmersión).
9. Al finalizar el ciclo, la lavadora envía una señal de finalización a la PC/robot.
10. El robot regresa, recoge la pieza y la deposita en la ubicación final predefinida.
11. El sistema vuelve a modo escaneo; si no se detectan más placas en el tiempo límite la rutina finaliza.

11.5 Lógica de orquestación

La PC actúa como maestro: toma decisiones de alto nivel (qué pieza procesar, cuándo iniciar ciclo de lavado).

- Los Arduinos ejecutan lógica de control local (mover ejes, homing, control de servos) y

rutinas sencillas de seguridad.

- El ESP32 actúa como puente de señal entre la PC/robot y la electrónica de la lavadora (pines digitales para comandos).

En la implementación actual la “lógica de orquestación” está centralizada en la PC; los microcontroladores no implementan una máquina de estados distribuida con confirmaciones formales (ACK).

11.6 Tiempos y tolerancias temporales

- La integración está diseñada para tolerar variaciones de segundos en la sincronización de eventos.
- Latencias de red/puerto serie no impactan el proceso si son del orden de decenas a cientos de milisegundos.
- El comportamiento del sistema es robusto frente a pequeñas desincronizaciones: una orden iniciada “unos segundos antes o después” no perjudica la secuencia global.

11.7 Manejo de errores y políticas operativas

Actualmente:

- Detección de ausencia de detecciones (visión): si la cámara no detecta objetos durante un tiempo prefijado la rutina finaliza.
- Fallo de agarre: no existe sensor que confirme agarre en el prototipo, por lo que se asume que el diseño del gripper elimina la posibilidad de fallo; si ocurriera, el sistema no lo detectaría automáticamente.
- Bomba sin presión: la lavadora debe detener su rutina si la bomba no alcanza presión; esto provoca parada del sistema de lavado y la rutina global.
- Reintentos y timeouts: algunos subsistemas incorporan reintentos automáticos y timeouts locales; no existe un framework centralizado de manejo de excepciones ni registro de eventos.

11.8 Handshake, ACK y trazabilidad

- En el estado actual no existe protocolo de handshake ni confirmaciones formales entre SCARA ↔ lavadora ↔ PC.
- No se registran logs de operación ni de eventos (no hay trazabilidad automatizada de acciones, IDs de pieza ni timestamps).

11.9 Interfaz de usuario y control

- Interfaz principal: GUI en PC con botones y controles para Home, Start/Play, Stop, y jog manual.
- Existe un botón de parada en la interfaz del robot y de la caja en la GUI.
- Hay un control físico general (zapatilla) que corta alimentación a todos los subsistemas (interruptor de alimentación física).

11.10 Calibración y parámetros configurables

- Calibración visión ↔ robot: se usa calibración manual por offsets (se alinean QRs y se ajustan offsets en software).
- Parámetros configurables online: tiempos de aspersion, tiempo de inmersión, velocidad y aceleración de motores, perfiles de movimiento; estos se ajustan desde la interfaz y se aplican en tiempo de ejecución.
- No existe, por el momento, un registro persistente de versiones de firmware; la última versión del código está corriendo en el sistema actual.

11.11 Logging, trazabilidad y versiones

- Logging: actualmente no se registran logs estructurados.

- Trazabilidad de piezas: no se documenta en sistema (por ejemplo: id de ECU, QR y timestamp por ciclo).
- Versionado: no hay control formal de versiones de firmware; se sugiere dejar un campo “versión” para registro (pendiente en el prototipo).

11.12 Diagrama de secuencia (texto)

A continuación se presenta una representación textual del diagrama de secuencia entre los actores principales:

1. Operario → PC : Start() (pula Play)
2. PC → Cámara : scanArea()
3. Cámara → PC : detections[] (lista de QRs/contenedores detectados)
4. PC → Robot (Arduino) : goto_pick(x,y,z) (coordenadas objetivo)
5. Robot → PC : (no hay ACK formal; movimiento ejecutado localmente)
6. Robot → Lavadora (via ESP32) : positioned_for_transfer (señal indicativa, en la práctica un pin HIGH)
7. Lavadora → Robot : (no hay ACK) — espera tiempo fijo y cierra gripper local
8. Robot : open_gripper(); retreat()
9. PC → Lavadora (via ESP32) : start_wash(cycle_params)
10. Lavadora → PC : done (señal de fin de ciclo, en la práctica pin HIGH o mensaje simple)
11. PC → Robot : goto_pick_return()
12. Robot : recoge y deposita la pieza.

Observación: la ausencia de ACKs formales implica que las “señales” descritas pueden implementarse como pulsos/pines o como mensajes simples, pero no hay confirmación de estado completa entre nodos.

BLOQUE V – GESTIÓN DEL PROYECTO

12. Planificación del Proyecto

12.1 Diagrama de Gantt

Se elaboró un diagrama de Gantt que refleja las tareas principales del proyecto (diseño, modelado CAD, fabricación, integración visión, pruebas, documentación y cierre). El diagrama incluye hitos, dependencias y entregables asociados a cada etapa.

[Proyecto Final - Diagrama de Gantt.xlsx](#)

12.2 Horas estimadas por etapa

Antes de comenzar la ejecución se registraron las horas estimadas por etapa (diseño mecánico, electrónica, visión, integración, ensayos y documentación). La hoja de planificación muestra el reparto de horas por rol y por sprint.

Durante la ejecución se llevó un registro de horas reales invertidas por persona y etapa; los datos se recogieron en formato hoja de cálculo para facilitar el cruce con lo estimado.

Se realizó la comparación entre estimado y real por etapa y por rol, con columnas de variación absoluta y relativa. El resumen muestra acumulados por módulo (mecánica, electrónica, visión, integración y documentación).

[Tabla de Horas.xlsx](#)

12.3 Análisis de desvíos

El análisis identifica las causas principales de desvío (iteraciones de diseño, reemplazos, tiempo en debug de visión) y cuantifica su impacto en horas. Las observaciones permiten priorizar medidas correctivas en futuros proyectos.

Se consolidaron las lecciones aprendidas relacionadas con estimación de tiempos, verificación temprana de componentes críticos y gestión de repuestos. Este documento se incorpora al cierre del proyecto como referencia para próximas iteraciones.

Se registró una estimación inicial de costos agrupada por categorías (mecánica, electrónica, sensores, actuación, mano de obra y contingencias). Esa estimación sirvió de línea base para el control presupuestario.

Todos los gastos reales (facturas, compras, reposiciones y compras de repuesto) se consolidaron en una hoja de control de costos. El archivo incluye notas sobre partidas atípicas (rodamientos, drivers extra, piezas reemplazadas).

[Gestión y Control de Costos del Proyecto.pdf](#)

BLOQUE VI – RESULTADOS Y CONCLUSIONES

15. Ensayos y Pruebas Realizadas

En esta sección se listan y resumen los ensayos ejecutados para validar los requisitos del sistema. El protocolo completo, los procedimientos paso a paso, las tablas de captura de datos y los criterios de aceptación se encuentran en el documento de ensayos del proyecto.

Ensayos realizados (resumen):

- 15.1 Ensayo de repetibilidad del SCARA — prueba posicional repetida en puntos representativos del workspace, registro de desviaciones en X, Y y Z. Ver procedimiento y criterios en el anexo de ensayos.
- 15.2 Ensayo de carga máxima — verificación de integridad estructural y funcionamiento de los actuadores bajo cargas crecientes hasta el rango objetivo de diseño.
- 15.3 Ensayo del sistema de aspersión — activación de bomba, control on/off y evaluación de cobertura y tiempos de aspersión. Presión nominal de la bomba: 2.7–4.1 bar (bomba de inyectores).
- 15.4 Ensayo del sistema ultrasónico — pruebas demostradoras con batea + motor desbalanceado; no se alcanzaron condiciones ultrasónicas activas con el arreglo usado (se documenta como demostrador mecánico).
- 15.5 Ensayo de integración completa — ciclo completo desde detección por visión → pick & place → transferencia a lavadora → ciclo de lavado → devolución. Registro de tiempos de ciclo y tasa de éxito por pieza.

Los registros detallados, planillas de medición y las hojas con los timestamps y resultados crudos están en el fichero [Ensayos.pdf](#)

16. Resultados y Análisis

16.1 Resultados obtenidos

- Repetibilidad del SCARA: las pruebas realizadas muestran que el manipulador cumple con la repetibilidad de referencia requerida en el diseño (orden de 1 mm en condiciones de prueba), lo que permite la operación de pick & place con la geometría autocentrante del gripper. (Datos crudos y protocolo: Ensayos.docx).
- Carga y estructura: las verificaciones estáticas y funcionales confirmaron que la estructura y las transmisiones (con las reducciones definidas) sostienen de forma confiable la carga objetivo (diseño sobre-dimensionado para dar margen). Detalles de cálculos y supuestos están en la sección mecánica del informe principal.
- Sistema de aspersión: la bomba (on/off) y los inyectores funcionaron conforme a lo esperado dentro del rango de presión medido (2.7–4.1 bar). Se validó la secuencia de aspersión en ambas caras mediante rotación del soporte.
- Ultrasonido: la implementación prototipo es demostradora: el motor desbalanceado no alcanza frecuencias ultrasónicas útiles; por tanto, la limpieza por ultrasonido no fue verificada en condiciones reales de cavitación. Se documenta la necesidad de transductores y electrónica dedicada para una implementación efectiva.

16.2 Comparación con objetivos iniciales

- En cuanto a repetibilidad y manipulación, el sistema cumple con el objetivo de repetibilidad mínima (≈ 1 mm), por lo que el requisito de manipulación de ECUs se considera alcanzado bajo las condiciones de prueba.
- En limpieza por aspersión, el sistema funciona funcionalmente y cumple la parte de aspersión del proceso; sin embargo, el objetivo de limpieza por ultrasonido no se alcanzó en el prototipo por limitaciones de hardware (transductores/driver).

17. Conclusiones y Mejoras Futuras

17.1 Conclusiones técnicas

El presente proyecto permitió diseñar, desarrollar e integrar una célula automatizada compuesta por un manipulador tipo SCARA, un sistema de visión artificial y un módulo de lavado orientado a la manipulación y limpieza de placas electrónicas (ECUs). El trabajo abarcó las etapas de concepción, modelado mecánico, selección y montaje de componentes electrónicos, programación de control, integración de visión con inteligencia artificial y validación experimental mediante ensayos específicos.

Desde el punto de vista mecánico, el manipulador desarrollado cumplió con los requerimientos de repetibilidad planteados para la aplicación, logrando una precisión adecuada para operaciones de pick & place sobre piezas de dimensiones reducidas. La arquitectura adoptada, junto con el sistema de reducción por poleas y la compensación de masa en el eje vertical, permitió un desempeño estable y coherente con los cálculos de diseño. El efector final diseñado facilitó la manipulación consistente de las placas, demostrando la correcta correspondencia entre diseño teórico y comportamiento real.

En cuanto al sistema de visión, la implementación basada en detección mediante modelo YOLO permitió identificar y localizar las piezas dentro del campo de trabajo, integrando el procesamiento en una arquitectura centralizada en PC. La calibración por offsets y la comunicación con el controlador del robot posibilitaron cerrar el lazo entre detección y acción, consolidando el flujo automático completo del sistema.

El módulo de lavado validó el funcionamiento del sistema de aspersión dentro de los parámetros definidos, integrándose correctamente con la secuencia general del proceso. La interacción entre robot y caja de lavado, coordinada desde la PC, demostró la viabilidad del esquema de control propuesto y la correcta sincronización entre subsistemas.

En términos de gestión, el proyecto incluyó planificación mediante diagrama de Gantt, estimación y seguimiento de horas de trabajo, así como análisis comparativo entre costos estimados y reales. Este enfoque permitió no solo materializar el desarrollo técnico, sino también evaluar el desempeño del proyecto desde una perspectiva organizacional y económica, integrando criterios de ingeniería y gestión en un mismo marco de trabajo.

En conjunto, el proyecto logró validar el concepto de una célula automatizada para manipulación y lavado de ECUs, integrando mecánica, electrónica, control y visión artificial en una solución funcional de prototipo. La experiencia adquirida evidencia la coherencia entre diseño, implementación y ensayo, y consolida una base técnica sólida para desarrollos posteriores dentro del ámbito de la automatización industrial aplicada.

17.2 Propuestas de Mejora

LAVADORA DE ECUS

1. Incorporación de sistema de seguridad para solventes inflamables

Dado que el sistema opera con alcohol isopropílico al 99%, resulta imprescindible, en una versión industrial, incorporar:

- Ventilación forzada interna con extracción controlada.
- Separación física entre compartimento húmedo y compartimento eléctrico.
- Sensores de apertura de tapa con enclavamiento de seguridad.
- Sistema de corte automático ante sobrecorriente o falla.

Esto incrementaría significativamente el nivel de seguridad operativa.

2. Implementación de recirculación y filtrado del fluido

El sistema actual utiliza tanque abierto sin filtrado. Se propone:

- Sistema cerrado con retorno.
- Filtro de partículas en línea.
- Control de nivel de líquido.
- Regulador de presión para estabilizar los 3 bar.

Esto permitiría mejorar la eficiencia del consumo de solvente y la repetibilidad del proceso.

3. Mejora estructural y hermetización

Para aproximarse realmente a un grado de protección IP68, sería necesario:

- Incorporar juntas de goma perimetrales.
- Uso de prensaestopas industriales.
- Sellado mecánico con tornillería en lugar de adhesivo.
- Bandeja de contención para derrames.

4. Optimización del sistema ultrasónico

El diseño actual contempla 40 kHz como especificación proyectada. Para una versión mejorada se propone:

- Generador ultrasónico dedicado de potencia controlada ($\approx 100\text{--}150\text{ W}$).
- Control de temperatura del baño ($40\text{--}50\text{ }^{\circ}\text{C}$).
- Sensor térmico con corte automático.
- Sistema de secado posterior (aire caliente o soplado forzado).

5. Sensores adicionales y monitoreo

Se propone incorporar:

- Sensor de nivel de líquido.
- Sensor de temperatura.
- Sensor de corriente en bomba.
- Señalización luminosa de estado (ciclo en curso, error, finalización).

Esto permitiría convertir el prototipo en una máquina con características industriales.

6. Mejora del sistema de sujeción

El sistema actual funciona correctamente, pero podría optimizarse mediante:

- Control de fuerza de sujeción.
- Revestimiento con material antiestático.
- Sistema de drenaje en el soporte.
- Diseño modular intercambiable según tipo de ECU.

ROBOT SCARA

1. Incorporación de encoders absolutos

Actualmente el posicionamiento depende de pasos de motor y homing. Se propone:

- Incorporar encoders absolutos o incrementales en cada eje.
- Implementar lazo cerrado de control.

Esto permitiría mayor precisión y reducción de error acumulativo.

2. Optimización estructural

Si bien el sistema está sobredimensionado en torque, podrían mejorarse:

- Rigidez de los brazos.
- Uso de aluminio mecanizado en lugar de piezas impresas.
- Reducción de peso en eslabones para disminuir inercia.

Esto permitiría mayores velocidades sin comprometer precisión.

3. Mejora del sistema de visión

Se podría optimizar mediante:

- Cámara industrial con obturador global.
- Iluminación controlada LED.
- Uso de algoritmos de detección más robustos.

4. Implementación de seguridad industrial

El SCARA podría incorporar:

- Zona de trabajo delimitada.
- Sensores de colisión.
- Sistema de parada controlada.

18. Referencias Bibliográficas

- Códigos — [Código](#)
- Esquemas Electrónicos — [Esquemas Electronicos](#)
- Fotos y Videos — [Fotos y Videos](#)
- Piezas — [Piezas](#)
- Brief Document — [Brief SCARA + Lavadora de ECUs.pdf](#)
- Ensayos — [Ensayos.pdf](#)
- Control de cosos del proyecto — [Gestión y Control de Costos del Proyecto.pdf](#)
- Lista de ensamblaje y diseño — [Lista de Ensamblaje y diseño.xlsx](#)
- Lista de Materiales robot SCARA — [Lista de Materiales Robot SCARA.xlsx](#)
- Manual de Usuario — [MANUAL DE USUARIO.pdf](#)
- Diagrama de GANT — [Proyecto Final - Diagrama de Gant.xlsx](#)
- Propuesta Final — [Proyecto final - Propuesta final.pdf](#)
- Propuesta Inicial — [Proyecto final - Propuesta inicial.pdf](#)
- Tabla de Horas — [Tabla de Horas.xlsx](#)