Universidad de Oviedo

Normagrup

# ESCUELA POLITÉCNICA DE INGENIERÍA DE GIJÓN

**Grado en Ingeniería Electrónica Industrial y Automática**

**Área de Ingeniería de Sistemas y Automática**

**TRABAJO FIN DE GRADO**

# MEJORA EN EL PROCESO DE MONTAJE DE LUMINARIAS MEDIANTE AUTOMATIZACIÓN CON BRAZO ROBÓTICO Y OPTIMIZACIÓN DE LA DOSIFICACIÓN DE ADHESIVO

**D. Alfredo Rodríguez Magdalena**
**TUTOR/ES:**
**D. Tomás Castro Riera**
**D. Ignacio Díaz Blanco**
**D. José María Enguita González**

**FECHA: Mayo 2024**

# Resumen

El proyecto, liderado por el departamento de ingeniería de Normagrup Technology, tiene como objetivo optimizar la producción de las luminarias Luzerna Avant mediante la implementación de tecnologías de automatización industrial. Se plantea diseñar un sistema integrado y eficiente, donde un brazo robótico de ABB recorra diversas subestaciones de montaje para agilizar el proceso y mejorar el control de calidad en la fabricación de estos dispositivos.

Los subprocesos del sistema abarcan desde la carga de bandejas metálicas en el brazo robótico hasta la etapa de posicionamiento para el secado del adhesivo. Se presta especial atención a la inyección de adhesivo en las pistas de la bandeja, un paso crítico para asegurar la fijación adecuada de las tiras LED y otros componentes electrónicos. Este análisis minucioso garantizará un diseño óptimo y una integración fluida del sistema en la línea de fabricación existente.

El estudio también contempla la selección del adhesivo más adecuado, considerando la compatibilidad con los semiconductores y la morfología del recipiente. Además, se explorarán diferentes métodos de fijación para la integración de la circuitería electrónica en la carcasa del dispositivo, con el objetivo de maximizar la eficiencia y la calidad en todo el proceso de producción.

En suma, el proyecto se enfoca en mejorar la eficiencia y la calidad en la producción de luminarias Luzerna Avant, utilizando tecnologías avanzadas y prácticas de ingeniería eficientes. La implementación de un sistema integrado, automatizado e inteligente permitirá optimizar los procesos de montaje y garantizar un producto final de alta calidad en menos tiempo.

## Palabras clave

Servomotor, pistola/dispensador de adhesivo, PLC, HMI, robot, perfilometría, machine learning, métodos numéricos, modelado dinámico por medios empíricos, sistemas automatizados, red neuronal, optimización.

# Índice de contenido

# Índice de figuras

# Índice de tablas

# Índice de algoritmos

# Glosario

| Término | Descripción |
|---|---|
| PLC | Autómata programable (*Programmable Logic Controller*) |
| $\rho$ | Densidad de un fluido $[kg/m^3]$ |
| $g$ | Constante de gravedad en la Tierra $(9{,}8m/s^2)$ |
| VC | Volumen de control |
| SC | Superficie de control (Plano de corte perpendicular al volumen de control) |
| D | Diámetro $[m]$ |
| u | Velocidad de un fluido $[m/s]$ |
| Q | Flujo volumétrico de un fluido (Caudal) $[m^3/s]$ |
| F | Fuerza de empuje $[N]$ |
| $\tau$ | Esfuerzos cortantes de un fluido $[Pa]$ |
| $\phi$ | Densidad de flujo de cantidad de movimiento |
| $K$ | Índice de consistencia de un fluido |
| $n$ | Índice de comportamiento de un fluido de ley de potencia |
| $\dot{\gamma}$ | Gradiente de velocidad perpendicular al plano de corte |
| $\mathscr{P}$ | Presión modificada $(P+\rho gh)$ |
| RTI | Rutina de Tratamiento de Interrupciones |
| HMI | Interfaz Humano-Máquina |
| ADS | Protocolo de comunicación de TwinCat (*Automation Device Specification*) |
| TCP | Protocolo de Control de Transmisión (*Transmission Control Protocol*) |
| API | Interfaz de Programación de Aplicaciones (*Application Programming Interface*) |
| ML | Aprendizaje Automático (*Machine Learning*) |
| DL | Aprendizaje Profundo (*Deep Learning*) |
| FIFO | Primero en entrar, último en salir (*First In, First Out*) |
| FFT | Transformada rápida de Fourier (*Fast Fourier Transform*) |
| SVD | *Singular Value Decomposition* |
| PSD | Densidad Espectral de Potencia (*Power Spectrum Density*) |
| $\gamma_m$ | Frecuencia de muestreo [Hz] |
| SGD | Descenso del Gradiente Estocástico |
| RN | Red Neuronal |

# 1. Introducción

## 1.1.- Preámbulo del proyecto

Normagrup Technology es una empresa española con sede en Asturias que se dedica al diseño, fabricación y comercialización de soluciones tecnológicas para la gestión del entorno. Su enfoque principal se basa en el área de la iluminación, pues ofrecen una amplia gama de luminarias LED de alta eficiencia y soluciones de control inteligente para diversos sectores, incluyendo oficinas, industria, alumbrado público y espacios comerciales.

Esta empresa se caracteriza por su fuerte apuesta por la innovación, invirtiendo en investigación y desarrollo para ofrecer soluciones tecnológicas de vanguardia. A su vez, está comprometida con la eficiencia energética y la sostenibilidad, desarrollando productos y soluciones que optimizan el consumo de energía. Un ejemplo de esto se puede ver en la nave principal, en la que se encuentran instalados 100 kW de potencia proveniente de paneles solares fotovoltaicos, esa energía se destina directamente a las oficinas y a la maquinaria que se encuentra en las líneas de producción. A su vez, la empresa tiene un compromiso de 35 años basado en el mantenimiento y expansión de un bosque en el cual se plantan numerosas hectáreas todos los años con el fin de compensar la huella de carbono.

Junto a los ejemplos comentados, Normagrup sostiene las siguientes 5 claves de ecodiseño para producir sus luminarias:

- **Eficiencia energética**. El ecodiseño debe centrarse siempre en la creación de luminarias y sistemas de iluminación altamente eficientes que minimicen el consumo de energía.
- **Materiales sostenibles**. La selección de materiales de bajo impacto ambiental implica la reducción de sustancias tóxicas, el uso de materiales reciclados o reciclables y la consideración de su huella de carbono.
- **Durabilidad y vida útil prolongada**. Los productos de iluminación de nueva generación deben tener una vida útil prolongada. Esto reduce la necesidad de reemplazo frecuente y la generación de residuos. Además, se deben facilitar las reparaciones y actualizaciones.
- **Diseño modular y desmontable**. La capacidad de desmontar y reemplazar los componentes de las luminarias facilita la reparación y el mantenimiento, lo que a su vez prolonga la vida útil de los equipos y reduce la cantidad de residuos.

- **Reciclabilidad y circularidad**. El diseño de los nuevos productos de iluminación debe considerar la facilidad de desmontaje y reciclaje al final de su vida útil.

Para concienciar al lector de lo que se está comentando, con el simple hecho de cambiar el empaquetado de la *Luzerna Avant*, la luminaria de la que se pretende automatizar la producción en este proyecto, a uno más reciclable, se ha conseguido reducir el consumo de combustibles fósiles 56 toneladas al año, el consumo de agua en 2.9 millones de litros al año, la generación de 135 toneladas al año en emisiones de carbono, y la tala de 1059 árboles anuales.

Es por ello, que para poder seguir mejorando la sostenibilidad de este producto, se ha decidido crear un sistema de automatización para su montaje con el fin de ahorrar todos los costes y emisiones generados por el transporte, y por otra parte diseñar un sistema que sea capaz de optimizar el uso de todos los materiales necesarios para el montaje de la luminaria. Pero antes de empezar a hablar de la automatización, es imprescindible conocer lo que se quiere automatizar.

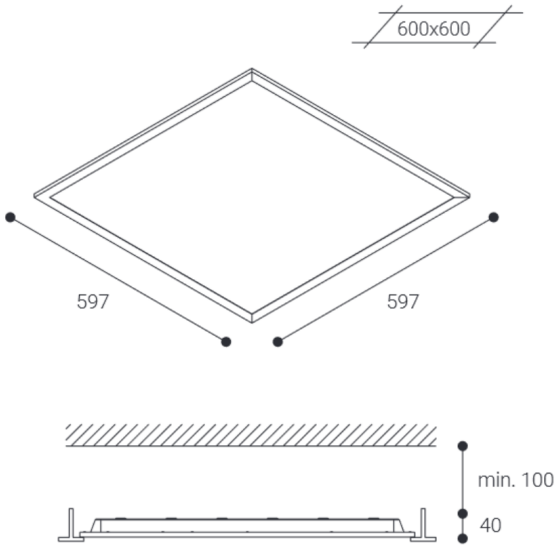



Figura 1.1: Dimensiones de la luminaria *Luzerna Avant*

Viendo la figura 1.1, se pueden comentar varios aspectos que determinarán la topología del sistema que las va a montar.

Para empezar, si se observa el área que ocupa, es de $597[mm] * 597[mm]$, esto significa que es lo suficientemente grande como para que un sistema basado en cintas sobre las que los diferentes subprocesos vayan montando la luminaria no sea la mejor opción. Por ello, se descarta la idea de emplear cintas transportadoras para este proyecto. En su defecto, la

mejor forma de poder tratar una bandeja de esas dimensiones, sería moviéndola a voluntad, y pudiendo operar con ella en 3 dimensiones. Por esa razón, se empleará un brazo robótico de ABB, que tendrá la función que se está comentando, manipular la bandeja metálica con un mayor grado de libertad. Una de las secciones de esta memoria tratará de explicar todo el procedimiento seguido para diseñar la parte robótica de la célula de montaje.

Si se vuelve a la figura 1.1, en específico, a la vista de abajo del todo, se puede observar que hay 6 pistas salientes por la parte superior, en esas ranuras se llevará a cabo la inyección de adhesivo, para posteriormente adherir las tiras LED a dichas superficies. La mayor parte de este documento consistirá en hacer un análisis de esta subestación. Pues tiene una gran complejidad debido al tipo de fluido con el que se está tratando, y a la gran precisión requerida para que no se desperdicie material, y que las tiras LED queden pegadas correctamente en la pista.

Todo el desarrollo presente en la memoria de este proyecto seguirá ordenadamente las siguientes pautas acerca del diseño de sistemas de control: [6]

1. Entender el proceso y sus requisitos de funcionamiento.
2. Seleccionar actuadores teniendo en cuenta su ubicación, tecnología, ruido y potencia.
3. Seleccionar sensores considerando ubicación, tecnología y ruido.
4. Construir un modelo lineal del proceso, actuador y sensor (si es posible).
5. Cerrar el lazo de control (Empleando control clásico, moderno, técnicas experimentales, ...). Si el resultado es satisfactorio, ir directamente al paso 8.
6. Considerar la modificación física de la planta para mejorar el comportamiento en lazo cerrado.
7. En caso de que sea posible, probar técnicas de control óptimo para mejorar las propiedades dinámicas del sistema.
8. Simular el diseño, incluyendo no-linealidades, ruido y variación de parámetros. Si el rendimiento no es el adecuado, volver al paso 1 y repetir.
9. Construir el prototipo y probarlo. Si el resultado no se corresponde con lo esperado, volver al paso 1 y repetir.

## 1.2.- Objetivos, cuestiones a resolver y delimitaciones

Como se ha comentado anteriormente, en este proyecto se pretende diseñar un sistema a medida capaz de llevar a cabo el montaje de las luminarias *Luzerna Avant* en un tiempo óptimo.

Para ello es necesario tener en cuenta que la etapa de inyección de adhesivo será la que más tiempo añadirá al tiempo unitario de fabricación, por esa razón, más de la mitad de esta memoria consistirá en buscar una solución óptima a esta subestación, de forma que se inyecte la cantidad necesaria de adhesivo en el menor tiempo posible.

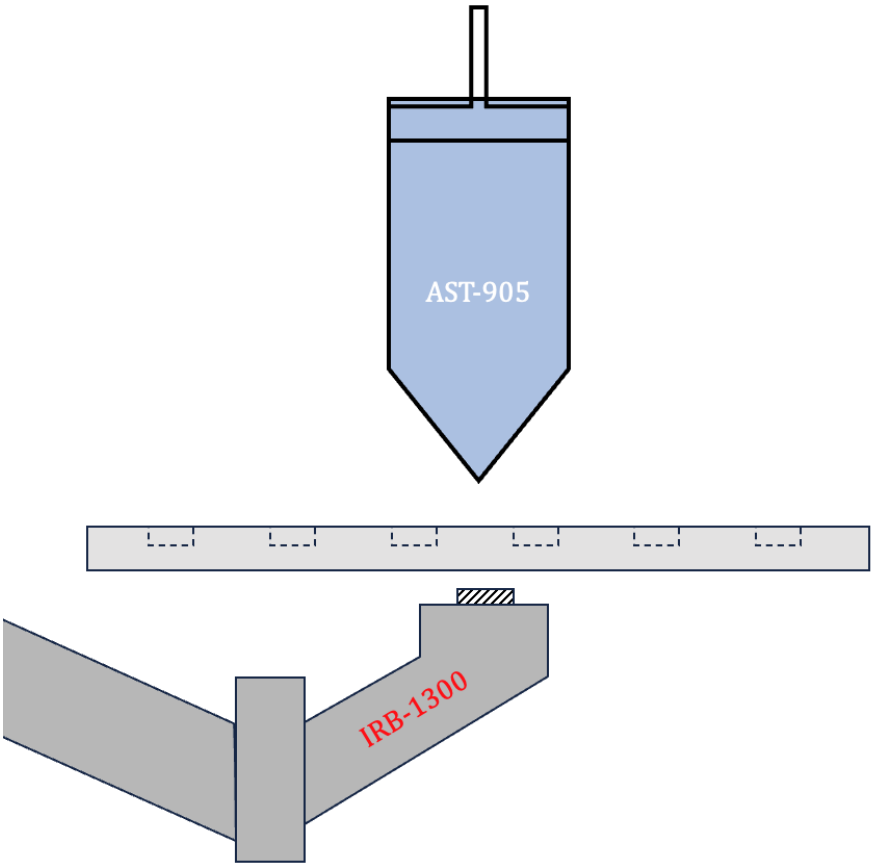



Figura 1.2: Esbozo del sistema a diseñar

Entonces, el principal objetivo de este proyecto, será llevar a cabo el diseño de un sistema de inyección de adhesivo en una bandeja de 6 pistas. Es importante mencionar que el bote de adhesivo estará fijado a una estructura, dejando como única parte móvil el brazo robótico, y por consiguiente, la bandeja. Siendo la máxima del proyecto el conseguir diseñar un sistema de control capaz de dispensar hilos de adhesivo con las propiedades deseadas (consignas).

Se debe remarcar que todas las soluciones propuestas se llevarán a cabo en mayor medida reutilizando componentes de maquinaria descatalogada. A excepción del robot y del sensor

del cual se hablará más adelante, todos los dispositivos involucrados en el desarrollo del sistema automatizado serán reutilizados.

A lo largo de todo el informe se irán valorando posibles opciones, con sus ventajas e inconvenientes, escogiendo la que se considere más correcta para la aplicación comentada. Con todo esto, al final de la memoria, se pretende dar respuesta a las siguientes cuestiones:

- ¿Cuál es el comportamiento del fluido en el sistema? ¿Es relevante?
- ¿Qué procedimientos se pueden seguir para llevar a cabo la programación de un robot de ABB?
- ¿Cómo funciona un servomotor de SMC?
- ¿Cómo se programa un PLC de Beckoff?
- Si se pretende monitorizar el proceso, ¿qué servicios pueden ser empleados para ese fin?
- ¿Cuál es el procedimiento a seguir para modelar empíricamente un sistema (*Data-Driven Learning Model*)?

A su vez se le recomienda al lector tener nociones básicas en áreas como mecánica de fluidos, programación en computadores y autómatas, protocolos de comunicación, métodos numéricos, álgebra lineal y cálculo.

Se recuerda que este informe hablará principalmente del desarrollo técnico de la célula de fabricación haciendo énfasis en la subestación de inyección adhesivo, desde el punto de vista de la automatización y la informática industrial. Y se remarca que el diseño mecánico de los elementos involucrados en la estación y los aspectos más relevantes en su montaje no entrarán dentro del ámbito de esta memoria.

Por último, se aconseja una lectura lineal del documento, pues cada capítulo está escrito asumiendo que los conceptos introducidos en sus predecesores han sido asimilados.

# 2. Estado del arte

La fabricación aditiva (FA), también conocida como impresión 3D, es un método que utiliza el diseño asistido por computador (CAD) y los sistemas de control numérico para crear componentes gradualmente, capa por capa. Esta industria ha experimentado un rápido desarrollo en los últimos dos décadas, integrándose en ámbitos como la industria aeroespacial, la educación, la electrónica, la fabricación de herramientas, la automoción y la medicina. La FA tiene ventajas únicas en comparación con otras tecnologías de fabricación. Está altamente digitalizada y se controla directamente a través de un modelo CAD para producir una pieza automáticamente. Este patrón requiere menos experiencia profesional y simplifica significativamente los procesos de fabricación, lo que puede acortar el ciclo de desarrollo del producto. Con la potente capacidad de fabricación, se pueden diseñar estructuras más complejas, sin un aumento aparente de costo por complejidad. Se espera que la FA modele el futuro de la fabricación distribuida, personalizada e individualizada. [7]

Una impresora 3D deseable debe tener las siguientes propiedades para satisfacer los requisitos de las vastas aplicaciones y desarrollar aún más las tecnologías de FA. Para controlar la precisión geométrica y la calidad mecánica y física de una pieza, primero debe realizar un proceso de FA confiable y repetible. En consecuencia, una pieza fabricada puede cumplir con las especificaciones del usuario. Segundo, una impresora 3D debe ser más eficiente o tener la capacidad de manejar más tareas de fabricación. Sin embargo, estos objetivos de rendimiento no se logran con las impresoras 3D actuales. La mayoría de las impresoras 3D utilizan un control en lazo abierto, lo que representa un problema importante para el aspecto controlable. Sus parámetros de material e impresión se determinan de manera empírica y luego se establecen antes de que comience un proceso de FA sin cambios adicionales. Este método de control en lazo abierto es propenso a defectos porque un proceso de FA enfrenta perturbaciones y reacciones físicas complejas. La promoción de la precisión dimensional, por ejemplo, al dividir una pieza en más capas, aumenta el tiempo de fabricación para la mayoría de las tecnologías de FA para el objetivo de alta productividad. [7]

En el caso de este proyecto se puede encontrar una relación altamente similar a lo que comentado sobre FA, pues a pesar de que el objetivo no sea la deposición de material por capas para fabricar piezas, sí se pretende diseñar un modelo empírico que represente las características fluido-dinámicas del adhesivo para poder hacer un control en lazo abierto. El objetivo de este proyecto de fin de grado será abrir una línea de investigación en la que a través del prototipo que se va a diseñar en los próximos capítulos se pueda cerrar un lazo de control mediante un modelo que sea capaz de adaptarse a las condiciones que se le impongan. Y de esta forma tratar de conseguir una solución extrapolable a otros campos de la industria.

# 3. Análisis teórico del bote de adhesivo

Como paso previo al diseño del sistema automatizado, es imprescindible conocer los conceptos teóricos que rigen el comportamiento de los componentes de este. En este caso toda la teoría se centra en el estudio del adhesivo. Para ello se necesitará recurrir a la mecánica de fluidos, la cual tendrá el papel de aportar la *intuición* necesaria para el correcto diseño de la célula de fabricación.

Se comenzará por hacer un breve recordatorio de una serie de magnitudes físicas que se verán presentes a lo largo de todo el modelado teórico del sistema dinámico. A continuación, se procederá a analizar el bote de adhesivo desde varios puntos de vista, y se sacarán conclusiones. Finalmente, se evaluará la opción de diseñar un sistema de control basado en toda la teoría comentada.

## 3.1.- Conceptos básicos

Antes de modelar el sistema fluido-dinámico presente en el bote de adhesivo, es necesario revisar una serie de definiciones básicas para el correcto entendimiento de dicho modelo.

### 3.1.1.- Densidad

Esta propiedad esencial para definir el comportamiento de un fluido se define como la masa concentrada en una unidad de volumen.

$$\rho = \frac{\mathrm{d}m}{\mathrm{d}V} \tag{3.1}$$

Dependiendo del campo de aplicación, sus unidades pueden ser varias. Pero en este proyecto se optará por usar siempre el sistema internacional, es decir, $[kg/m^3]$.

### 3.1.2.- Viscosidad

Se puede definir como una propiedad de los fluidos basada en la resistencia de ciertas sustancias a fluir o deformarse gradualmente debido a esfuerzos normales o cortantes. En este proyecto se hablará siempre de la viscosidad dinámica, medida en $[Pa \cdot s]$.

Para entender este concepto, se va a plantear un ejemplo. A la hora de comprar aceite de motor, se sabe que unos aceites son más *viscosos* que otros, y dicha magnitud varía con la

temperatura.

Tal y como se verá más adelante, existe una clasificación de los tipos de fluidos en función de la viscosidad ante el gradiente de velocidad.

### 3.1.3.- Cantidad de movimiento

Magnitud física capaz de describir el movimiento de un cuerpo en cualquier paradigma de la mecánica. En el caso que se está tratando en este proyecto se estaría hablando de medios continuos. Es decir, para un fluido regido por un campo de velocidades $\vec{v}$ la cantidad de movimiento será

$$\mathrm{p} = \int v \mathrm{d}m = \int v \rho \mathrm{d}V \tag{3.2}$$

La suma de la cantidad de movimiento de todas las partículas de dicho fluido.

### 3.1.4.- Fluido de ley de potencia

También denominados como aquellos que cumplen con la relación *Ostwald-de Waele*. Son modelos reológicos de fluidos no newtonianos que no tienen memoria temporal, para los cuales el esfuerzo cortante $\tau$ es

$$\tau = m \left( \frac{\partial u}{\partial y} \right)^n \tag{3.3}$$

Donde $\tau$ es el esfuerzo cortante, $\frac{\mathrm{d}u}{\mathrm{d}y}$ el gradiente de velocidad perpendicular al plano de corte, m el índice de consistencia del fluido, y n el índice de comportamiento de este. Nótese que si $m = \mu$ y $n = 1$, se estaría hablando de un fluido newtoniano.

A partir de esa expresión, si se hacen logaritmos y se calculan las pendientes de las curvas resultantes, se puede deducir la viscosidad [1] aparente del fluido en función del gradiente de velocidad. [8]

$$\eta_{eff} = m \left( \frac{\partial u}{\partial y} \right)^{n-1} \tag{3.4}$$

[1] En los fluidos newtonianos la viscosidad suele denotarse como $\mu$, mientras que en el resto se denota como $\eta$.

## 3.2.- Análisis cinemático del bote de adhesivo

En cuanto a cinemática se refiere, es decir, analizando única y exclusivamente el movimiento a lo largo del tiempo, despreciando toda fuerza que pueda darse. Es posible encontrar una relación entre la velocidad del émbolo del bote, y el caudal de salida. Para ello, hay que emplear volúmenes de control.

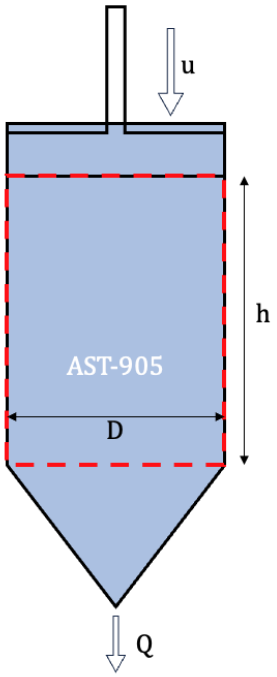



Figura 3.1: Modelado cinemático del bote con VC

Si se observa la figura 3.1, se puede ver representado un diagrama del bote de adhesivo sobre el cual se halla un volumen de control **deformable**. Para poder calcular el caudal de salida en función de la velocidad del fluido en la parte superior del bote, es necesario recurrir a la ecuación integral de la conservación de la masa. Que teniendo en cuenta la morfología del VC, devolverá la relación esperada entre ambas magnitudes físicas.

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{VC}\rho \mathrm{d}V + \oint_{SC}\rho(\vec{u_R}\cdot\vec{\mathrm{d}S}) = 0 \tag{3.5}$$

Para empezar, se debe suponer que la densidad será constante y uniforme en todo el bote. A su vez, se remarca el convenio de signos, en el cual todo flujo entrante será positivo, y todo saliente será negativo. A partir de ahí, se comenzará a particularizar la expresión al caso actual, empezando por la integral de superficie.

$$\oint_{SC}\rho(\vec{u_R}\cdot\vec{\mathrm{d}S}) = \rho\oint_{SC}(\vec{u_R}\cdot\vec{\mathrm{d}S}) = -\rho Q \tag{3.6}$$

Ahora, se desarrollará el término restante, la integral sobre el VC. Sabiendo que el volumen

se define como área transversal por longitud, se puede expresar la integral en función de la altura del bote.

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{VC}\rho \mathrm{d}V = \frac{\mathrm{d}}{\mathrm{d}t}\rho\int\frac{\pi}{4}D^2\mathrm{d}h = \frac{\mathrm{d}}{\mathrm{d}t}\rho\frac{\pi}{4}D^2\int\mathrm{d}h \tag{3.7}$$

Si se reordena la expresión, y uno se percata que la altura del bote irá cambiando a lo largo del tiempo, la derivada de la altura con respecto al tiempo se traducirá en la velocidad de avance del émbolo del bote, quedando el término de la siguiente forma.

$$\frac{\mathrm{d}h}{\mathrm{d}t}\rho\frac{\pi}{4}D^2 = \rho\frac{\pi}{4}D^2u \tag{3.8}$$

Finalmente, si se vuelven a unir los términos desarrollados siguiendo la estructura de la definición de la ecuación integral de la conservación de la masa...

$$\rho\frac{\pi}{4}D^2u - \rho Q = 0 \tag{3.9}$$

Se llegará a la ecuación mostrada en la línea posterior. Cabe mencionar que la densidad desaparece, lo cual sucede debido a que en este VC el único fluido presente es el adhesivo.

$$\boxed{Q = \frac{\pi}{4}D^2u} \tag{3.10}$$

## 3.3.- Análisis dinámico del bote de adhesivo

En este apartado del desarrollo teórico se hará un modelado del bote de adhesivo atendiendo a todas las fuerzas que generan el movimiento del fluido. Para ello se empleará la teoría de transporte de fluidos, que permitirá calcular la distribución de esfuerzos cortantes de un fluido de Stokes sobre una superficie concreta.

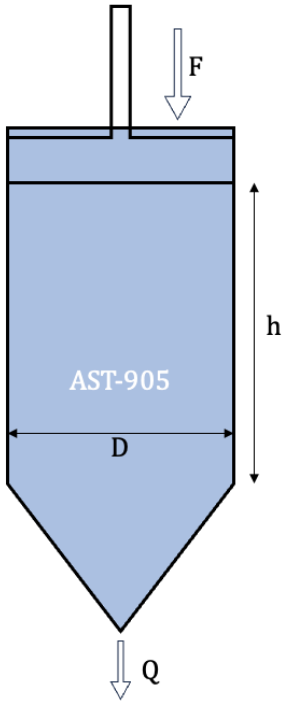



Figura 3.2: Modelado dinámico del bote

Para poder establecer la relación entre el caudal de salida y las fuerzas que generan dicho movimiento del fluido se emplea la ecuación de la energía (expresión 3.11) entre el punto donde se aplican las fuerzas, y el de salida del fluido. Pero al estar tratando con un flujo viscoso, y no con uno ideal, hay que tener en cuenta la reología del material, porque mermará el sistema a la hora de transferir la presión ejercida en caudal.

$$\frac{P_1}{\rho g} + \frac{u_1^2}{2g} + z_1 = \frac{P_2}{\rho g} + \frac{u_2^2}{2g} + z_2 + \text{pérdidas de carga} \tag{3.11}$$

En este caso, para estudiar la relevancia de esas pérdidas debidas a la viscosidad del fluido, se **aproximará** el cálculo del flujo a la envoltura de un tubo circular de radio R y longitud *L*. A su vez, también se comentará brevemente la reología del adhesivo, la cual dará una visión del comportamiento de la viscosidad de este. Y finalmente se pondrán en común las conclusiones de ambos procedimientos y se enunciará el modelo dinámico del bote.

### 3.3.1.- Cálculo de distribución de densidad de flujo

Se pretende hacer análisis de un fluido que circula por un tubo circular de radio R y longitud L como resultado de la diferencia de presión y de la gravedad. Para ello, es necesario en primer lugar hacer las siguientes suposiciones:

- El flujo debe ser laminar.
$$Re = \frac{4\rho Q}{\pi D \mu} \leq 2300 \tag{3.12}$$
- El flujo debe ser estacionario, es decir, la velocidad del fluido no debe depender del tiempo ($\vec{v} \neq f(t)$), e incompresible ($\vec{\nabla} \cdot \vec{v} = 0$).[2]
- La densidad del fluido debe ser constante.
- El fluido fluye hacia abajo por una diferencia de presión y por la gravedad.

Para poder deducir la densidad de flujo del fluido, es necesario hacer un balance de cantidad de movimiento en la envoltura del fluido y tener en cuenta las condiciones límite de este. [9]

$$\begin{bmatrix} \text{Variación de } \phi \\ \text{sobre el VC con} \\ \text{respecto al tiempo} \end{bmatrix} = \begin{bmatrix} \text{Incremento de } \phi \\ \text{por convección} \\ \text{sobre el VC} \end{bmatrix} + \begin{bmatrix} \text{Incremento de } \phi \\ \text{por difusión} \\ \text{sobre el VC} \end{bmatrix} + \begin{bmatrix} \text{Fuerzas que} \\ \text{actúan} \\ \text{sobre el VC} \end{bmatrix}$$

En condiciones de flujo estacionario y considerando a $\phi$ como una variable genérica, el balance de cantidad de movimiento depende de la velocidad de entrada y salida de cantidad de movimiento y de la suma de fuerzas que actúan sobre el sistema, es decir, la presión sobre la superficie y la gravedad que actúa sobre el volumen.

Este planteamiento proviene de la aplicación de la ley de conservación de la cantidad de movimiento. Entonces, el procedimiento a seguir para resolver este problema de flujo viscoso será: [8] [10]

- Identificar la componente de velocidad que no se elimina y la variable espacial de la que depende.
- Plantear el balance de cantidad de movimiento sobre una delgada envoltura perpendicular a la variable espacial relevante.

[2] Considérese un campo de velocidad $\vec{v}$ definido desde un punto de vista euleriano, expresado en coordenadas cartesianas de la siguiente forma: $\vec{v} = u\hat{i} + v\hat{j} + w\hat{k} \ \forall \ u,v,w = f(x,y,z,t)$

- Hacer que dicho espesor tienda a 0 y usar la definición de la derivada para obtener la ecuación diferencial correspondiente para la cantidad de flujo de cantidad de movimiento.

- Integrar esa ecuación para obtener la distribución de densidad de flujo de cantidad de movimiento.

Hay que tener en cuenta que a la hora de integrar, aparecerán constantes de integración. Estas se evaluarán mediante las *condiciones límite*. En este caso vendrán dadas por la *condición de no deslizamiento*, la cual dice que si todo fluido que esté en contacto con un objeto sólido tendrá una velocidad nula. Todo ello asumiendo que en la superficie entre las dos fases no hay absorción, adsorción, disolución, evaporación, fusión o cualquier reacción química.

Como se ha comentado anteriormente, el objetivo es obtener una expresión que defina los esfuerzos cortantes para una tubería de radio $R$ y longitud $L$, donde el fluido se mueva a causa de diferencia de presión y/o de gravedad. Pues debido a la morfología de la envoltura del fluido, lo mejor será emplear un sistema de coordenadas cilíndrico $(r, \theta, z)$. [8] [10]

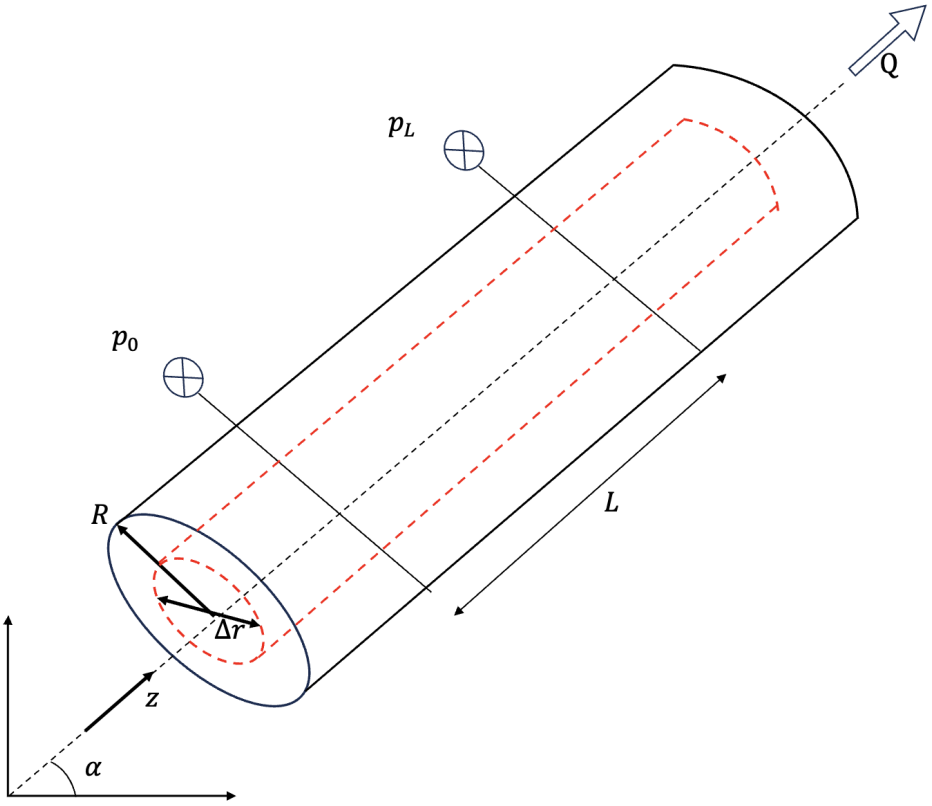



Figura 3.3: Flujo laminar en tubería

Escogiendo como sistema uno con una envoltura cilíndrica de espesor $\Delta r$ y longitud $L$,

el balance de cantidad de movimiento se vería expresado de la siguiente forma. [8] [10]

$$(2\pi rL\phi_{rz})|_r - (2\pi rL\phi_{rz})|_{r+\Delta r} + (2\pi r\Delta r)(\phi_{zz})|_{z=0} - (2\pi r\Delta r)(\phi_{zz})|_{z=L} + (2\pi r\Delta rL)\rho g = 0 \tag{3.13}$$

A partir de la expresión 3.13, si se postula que $u_z = u_z(r)$, $u_r = 0$ y $P = P(z)$, es decir, que el fluido se moverá única y exclusivamente hacia abajo, se particulariza la ley de viscosidad de Newton y se reordena convenientemente, se puede deducir la siguiente ecuación.

$$\frac{\partial}{\partial r}(r\tau_{rz}) = \left(\frac{(P_0 - \rho g z_0) - (P_L - \rho g z_L)}{L}\right) r \equiv \left(\frac{\mathscr{P}_0 - \mathscr{P}_L}{L}\right) r \tag{3.14}$$

Finalmente, si esta es integrada con respecto a $r$, se obtendrá la siguiente ecuación.

$$\tau_{rz} = \left(\frac{\mathscr{P}_0 - \mathscr{P}_L}{2L}\right) r + \frac{C_1}{r} \tag{3.15}$$

La constante $C_1$ se evalúa usando la condición límite, para la cual en $r = 0$, $\tau_{rz}$ es finito. Por ello, $C_1$ debe ser 0, ya que en caso contrario la densidad de flujo de cantidad de movimiento sería infinita en el eje del tubo. Quedando la distribución de esfuerzos cortantes de la siguiente forma: [8] [10]

$$\boxed{\tau_{rz} = \left(\frac{\mathscr{P}_0 - \mathscr{P}_L}{2L}\right) r} \tag{3.16}$$

### 3.3.2.- Deducción de reología del adhesivo

La segunda cuestión a plantear es, ¿qué naturaleza tiene el fluido con el que se está tratando? Claramente, no es un fluido newtoniano, como puede ser el agua. En este caso se está tratando con un pseudoplástico, el cual pertenece al grupo de los fluidos de ley de potencia.

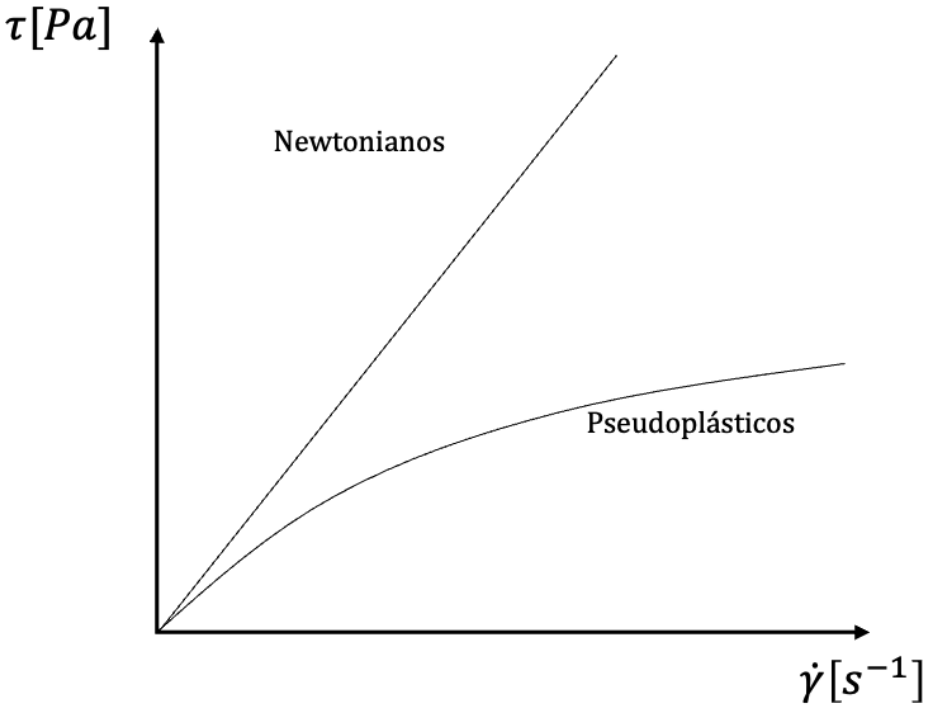



Figura 3.4: Curvas reológicas de fluidos newtonianos y pseudoplásticos

Tal y como se puede observar en la figura 3.4, los esfuerzos cortantes de un fluido newtoniano son lineales con respecto al gradiente de velocidad. Sin embargo, en los pseudoplásticos, esta relación es exponencial, definida por la siguiente expresión.

$$\boxed{\tau = K\left(\frac{\mathrm{d}u}{\mathrm{d}y}\right)^{n} = K\dot{\gamma}^{n}} \tag{3.17}$$

Donde $\tau$ es el esfuerzo cortante, $\frac{\mathrm{d}u}{\mathrm{d}y}$ el gradiente de velocidad perpendicular al plano de corte, K el índice de consistencia del fluido, y n el índice de comportamiento de este. Para pseudoplásticos $(1-n)$ recibe el nombre de índice de pseudoplasticidad, y será menor que 1.

Este comportamiento puede verse fácilmente en aquellos fluidos que al dejarlos caer por fuerza de la gravedad tardan mucho tiempo en empezar a moverse, por ejemplo, la miel.

### 3.3.3.- Obtención del modelo fluido-dinámico del sistema

Tal y como se comentó en las secciones anteriores, para obtener una relación entre el caudal de salida de adhesivo del bote y la fuerza que hay que ejercer en el émbolo, es necesario modelar la naturaleza del fluido (su reología) (3.17), y la morfología del medio por el que se va a mover, en este caso, es un flujo a través de un tubo circular de radio R y longitud L (3.16).

Para obtener el modelo fluido-dinámico, es necesario igualar dichas expresiones y operar con ellas.

$$\tau = K\left(\frac{\mathrm{d}u}{\mathrm{d}y}\right)^n = \left(\frac{\mathscr{P}_0 - \mathscr{P}_L}{2L}\right) r \tag{3.18}$$

Una vez se despeja la raíz n-ésima, se debe integrar la ecuación aplicando la condición de no deslizamiento en r=R (la velocidad del fluido en contacto con las paredes del bote es nula), se obtiene la distribución de velocidad.

$$u_z = \left(\frac{(\mathscr{P}_0 - \mathscr{P}_L)R}{2KL}\right)^{1/n} \frac{R}{(1/n)+1}\left[1 - \left(\frac{r}{R}\right)^{(1/n)+1}\right] \tag{3.19}$$

Si la expresión anterior se integra nuevamente sobre la sección transversal del tubo circular se obtiene el caudal en función de la diferencia de presiones.[3] [8]

$$Q = \frac{\pi R^3}{\frac{1}{n}+3}\left(\frac{(\mathscr{P}_0 - \mathscr{P}_L)R}{2KL}\right)^{\frac{1}{n}} \tag{3.20}$$

Para poder calcular el caudal de salida también se puede usar la ecuación 3.20 expresada como las pérdidas de carga en función de la viscosidad del fluido junto a un caudal de operación. Para ello hay que recordar que estas son directamente proporcionales al gradiente de presiones y a la longitud del conducto por donde este circula, e inversamente proporcionales a la gravedad y a la densidad de este. Si se igualan las diferencias de presiones de ambas expresiones, se puede llegar a la siguiente conclusión.

$$\boxed{h_{pL} = \frac{2KL}{\rho g R}\left(\frac{Q\left(\frac{1}{n}+3\right)}{\pi R^3}\right)^n} \tag{3.21}$$

Nótese que si el único elemento del circuito de deposición de material fuera el bote de adhesivo, estas pérdidas de carga se podrían despreciar debido a que la sección del bote es considerablemente grande en comparación con su altura. Si se añadieran tuberías habría que tenerlas en cuenta.

[3] Nótese que si $n = 1$ y $K = \mu$, se estaría hablando de la ecuación de *Hagen-Poiseuille*. La cual definiría el comportamiento de fluidos newtonianos bajo las mismas condiciones.


Figura 3.5: Bote de adhesivo real

Es necesario tener en cuenta que la ecuación asume condiciones ideales, es decir, el rozamiento del émbolo del bote con las paredes del mismo no está considerado, además, en muchos casos, pequeñas cantidades de pegamento se escapan por el espacio entre el émbolo y las paredes del bote, incrementando el rozamiento a posteriori. A su vez, con que haya una ligera entrada de aire en el circuito de deposición se generaría una estructura parcialmente solidificada que produciría microobtrucciones en el conducto, reduciendo el caudal de salida.

Observando la figura 3.6, se puede identificar una cámara presurizada, una tubería y una válvula, pero debido a la falta de información de parámetros reológicos del fluido no se puede comprobar si representan una pérdida de carga lo suficientemente relevante como para despreciarlas y calcular el caudal de salida asumiendo flujo ideal empleando la ecuación de *Bernoulli*. Si se conocieran los parámetros que definen el tipo de fluido con el que se está tratando, se podría hacer uso de la expresión 3.21 para estimar las pérdidas de carga y con estas emplear la ecuación de la energía para calcular el caudal de salida de material.

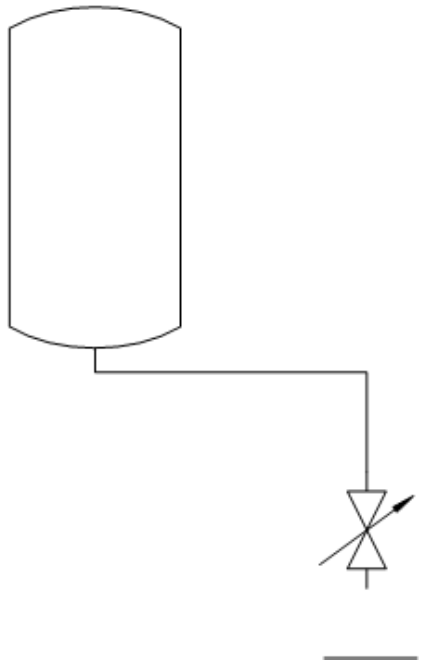


Figura 3.6: Esquemático del circuito de deposición de adhesivo

Pero volviendo la figura 3.6 se puede identificar que las pérdidas de carga se dividen en 2 grupos. Por un lado estarían las lineales, las cuales se asocian a la morfología de la tubería, y variarán en función de su longitud y su sección[4]. En este caso la tubería empleada tiene una longitud aproximada de 30 a 40 cm, y un diámetro de 8 mm. Por el otro lado se encuentran las pérdidas específicas, que se corresponderían a 2 codos de 90º, a la propia válvula y a la boquilla de salida del material al exterior. Una vez más, como se desconocen las propiedades del fluido más allá de su comportamiento, no se puede saber si el sistema se ve más mermado por un tipo de pérdidas u otro. Por todo ello la única solución factible para encontrar una relación entre la presión ejercida en la cámara presurizada y el caudal de salida del material es hacerlo de forma experimental. [11]

$$\Delta P = a \cdot Q^b \quad \forall \quad b \in (0,2] \tag{3.22}$$

Mediante la obtención de los parámetros $a$ y $b$ no solo se obtendrá la relación presión-caudal del sistema como conjunto, si no que comprobando si b es menor que 1, se puede decir que el flujo es laminar, mientras que si por el contrario es mayor, el flujo será turbulento. [12]

Para calcular estos valores se pueden tomar logaritmos de la expresión 3.22 con el fin de que quede una relación lineal.

$$ln(\Delta P) = A + b \cdot ln(Q) \quad \forall \quad A = ln(a),\ b \in (0,2] \tag{3.23}$$

[4]Por norma general estas pérdidas se ven reducidas a menor longitud de tubería y mayor sección transversal de esta.

# 4. Diseño del sistema automatizado

Si se vuelve a la figura 1.2, es posible identificar 3 elementos principales. Un brazo robótico, una bandeja como la que aparece en la figura 1.1 y un bote de adhesivo (analizado en detalle en el capítulo anterior). En este capítulo se hablará principalmente de la realización física del sistema automatizado, es decir, se detallará la estrategia de programación del robot para **todas las subestaciones**. A su vez, se valorarán los posibles actuadores necesarios para ejercer presión sobre el émbolo del bote en la etapa de inyección de adhesivo, escogiendo al final la mejor opción. Por último, se comentarán todas las herramientas valoradas para el diseño de las comunicaciones entre los distintos elementos del sistema, pero sólo se detallará la solución final.

## 4.1.- Estrategia de programación del robot

Un sistema robotizado generalmente está compuesto por tres componentes principales:

- El brazo robótico. Es un conjunto de piezas articuladas, normalmente metálicas, con capacidad de movimiento gracias a máquinas eléctricas. Para este proyecto, se ha decidido encargar un robot de 6 ejes de ABB, capaz de cargar con aproximadamente 10$kg$ y con un alcance de 1,15$m$, cuyo modelo es el *IRB 1300*.

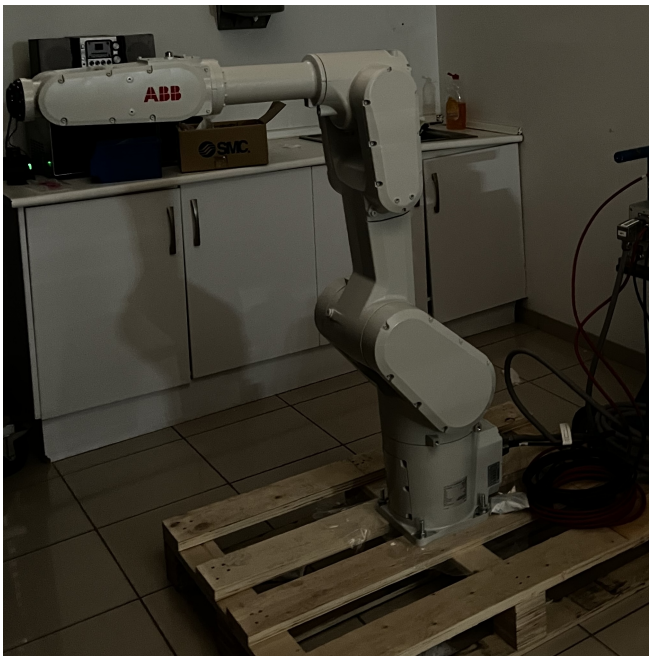



Figura 4.1: Robot encargado para el proyecto

- La controladora. Se podría decir que es un "PLC a gran escala", pues está encargada de llevar a cabo el control del brazo robótico, y puede ser programada para que el robot siga las trayectorias deseadas por el técnico. Puede tener tarjetas de E/S y comunicaciones por Ethernet o buses de campo. En pocas palabras, es el **dispositivo maestro** del sistema robotizado. Y en específico, es el modelo *C30* de la saga *Omnicore*.

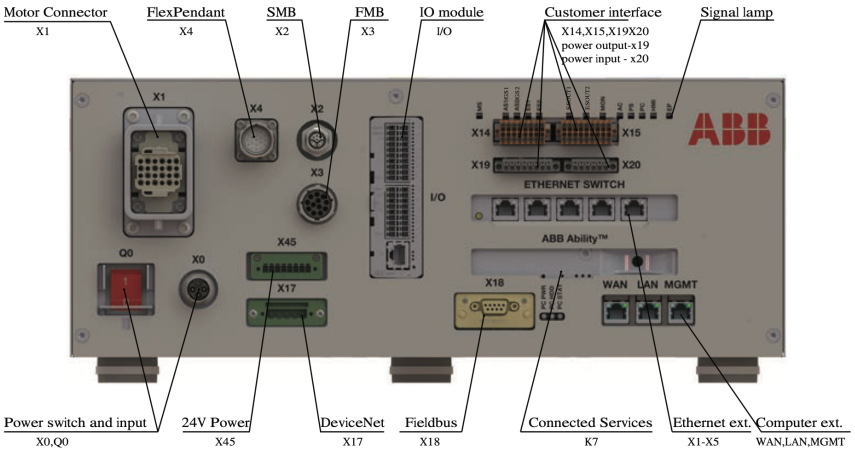



Figura 4.2: Controladora del robot [1]

- La pantalla de operador. Es básicamente un dispositivo que permite al técnico usar el robot con mayor facilidad, pues como se explicará en secciones posteriores, a parte de moverlo a posiciones concretas con un *joystick*, operar con el estado de las variables asociadas a las tarjetas E/S y modificar muchos parámetros de configuración. Permite programar el robot desde su interfaz de programación. En el caso de este proyecto, la pantalla recibe el nombre de *FlexPendant*, pues el robot que se ha encargado es de la marca ABB.

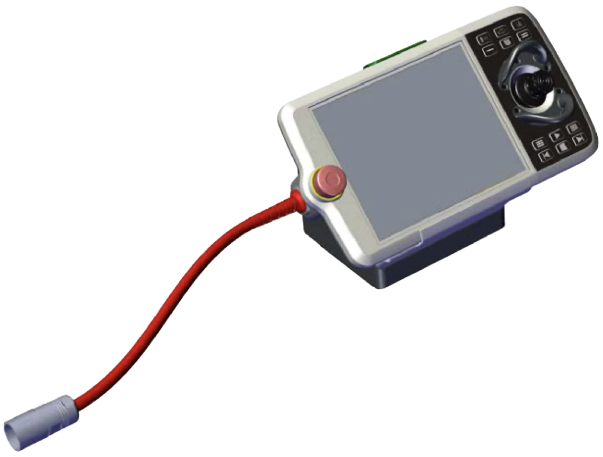


Figura 4.3: Pantalla de operador del sistema robotizado (FlexPendant) [1]

### 4.1.1.- Metodologías de programación

Los robots de ABB permiten ser programados de diferentes formas. Para establecer las trayectorias del robot, los programas de control, y cualquier otro tipo de configuración, la firma proporciona la aplicación *RobotStudio*. En esta memoria se comentarán los 2 principales métodos que se ofrecen a los técnicos para diseñar las rutinas de estos.

#### 4.1.1.1.- Programación mediante entorno virtual

La aplicación de *RobotStudio* permite la creación de un gemelo digital del robot a programar. Que además de simular un conjunto de trayectorias, facilita la construcción de un entorno virtual que se asemeje a la estación robotizada que se desea crear en la realidad. A lo largo de esta subsección se mostrará el procedimiento seguido para crear una estación virtual y un programa sencillo, con algunas de las funciones más relevantes a la hora del diseño del programa de la estación final.

En primer lugar, una vez se abre la aplicación, y se crea una nueva estación, lo que hay que hacer es escoger el brazo robótico que se va a emplear. Para ello, tal y como se muestra en la figura 4.4, en el menú Posición Inicial pulse *Biblioteca ABB*, se debe seleccionar el modelo a utilizar, en este caso el *IRB 1300*.

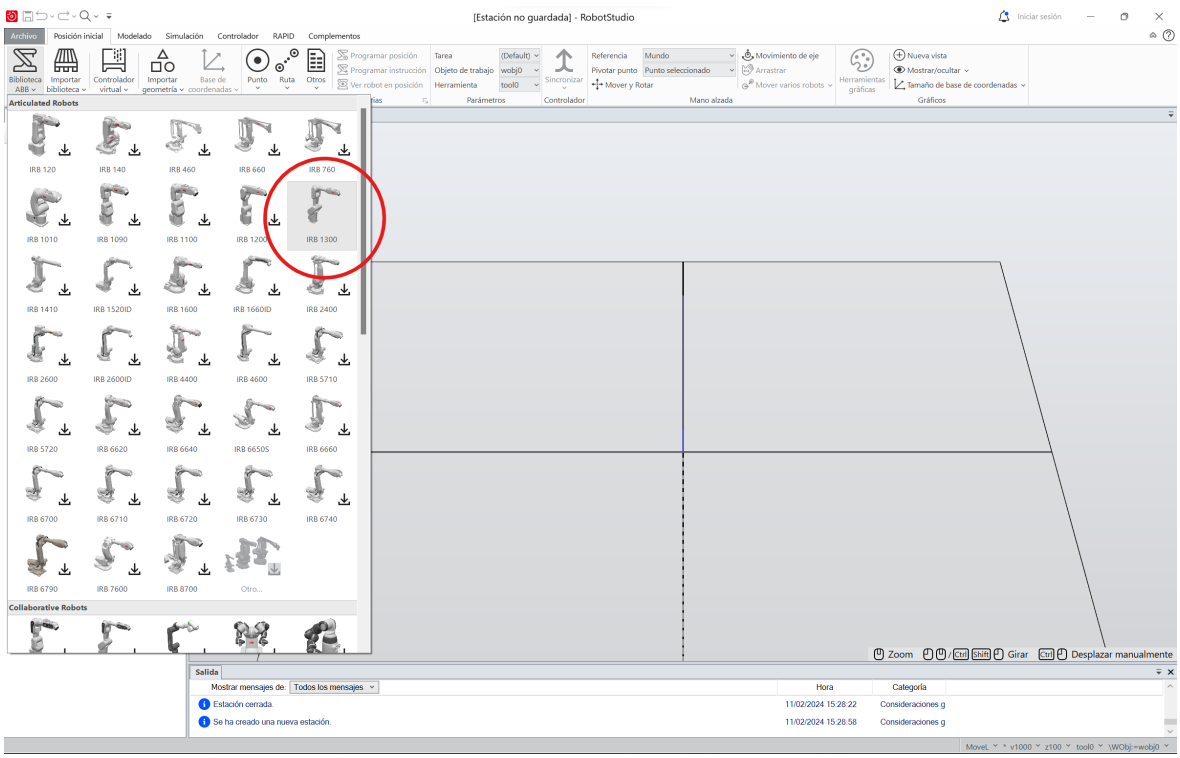



Figura 4.4: Selección de robot en estación virtual de *RobotStudio*

Después de escoger el brazo, es necesario seleccionar la controladora. Para ello, en el mismo menú de Posición Inicial, abra el desplegable *Controlador Virtual*, y pulse *Nuevo Controlador*. En las opciones de configuración de la controladora, se debe especificar el tipo, en este proyecto será la *Omnicore C30*.

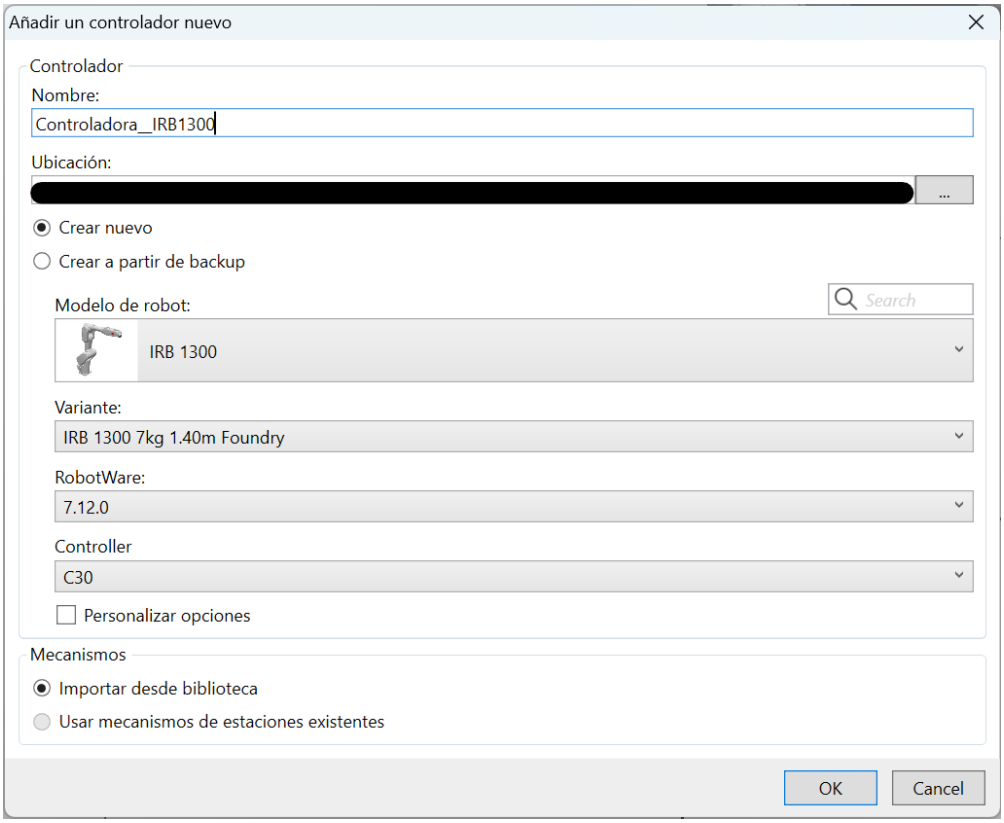



Figura 4.5: Selección de controladora en estación virtual de *RobotStudio*

Una vez está todo configurado, se pueden empezar a programar las trayectorias de puntos a seguir por el robot, siendo el primer paso la creación de los respectivos puntos que las conformarán. Para crearlos se debe ir al menú *Pasición Inicial* y pulsar en *Punto*.

Existen diversas formas de crear puntos, la primera de ellas es usando el menú que sale a la izquierda de la figura 4.6, donde se pueden establecer directamente las coordenadas en caso de que se requiera tal precisión. También es posible crear los puntos de forma directa en el espacio de trabajo, moviendo el robot a la posición deseada y presionando *Programar posición*. Esto es buena opción cuando en el entorno virtual hay objetos, como mesas, o moldes de cualquier tipo, pues el programa ya detecta automáticamente los puntos de interés, como las esquinas, los centros, etc. Una vez los puntos hayan sido creados, se pulsará en *Crear*, y se guardarán en los directorios de abajo asociados a su objeto de trabajo.

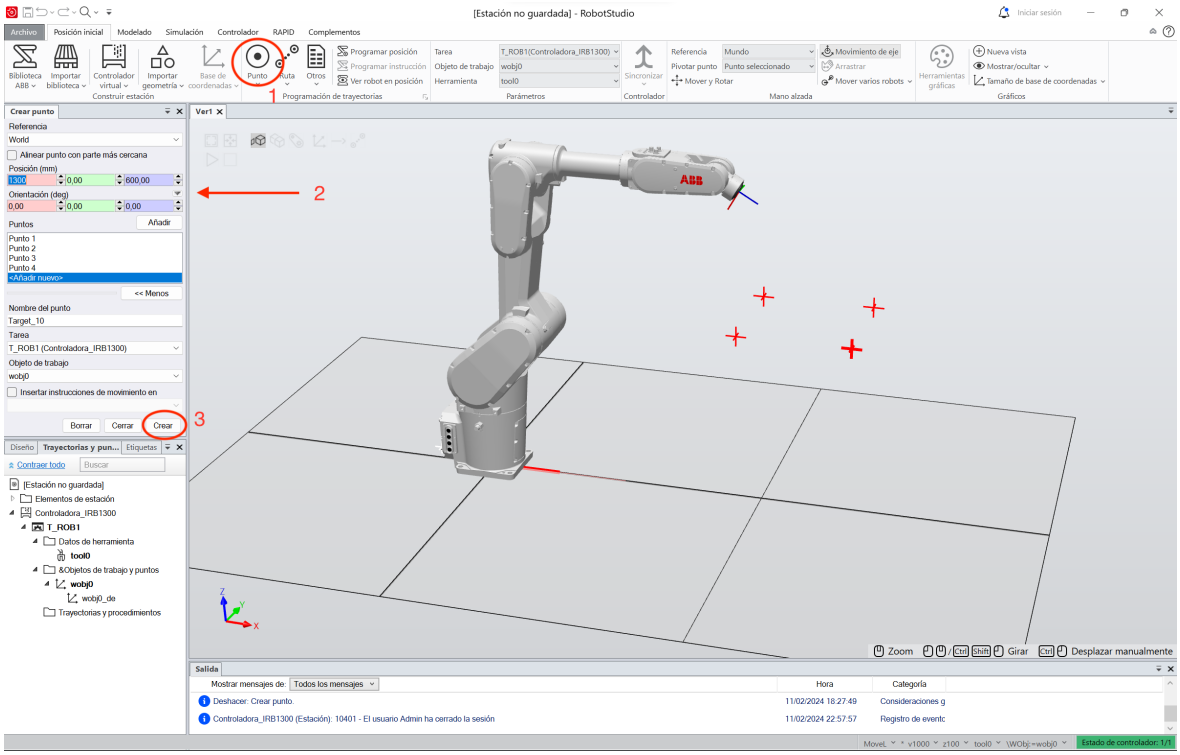



Figura 4.6: Menú de creación de puntos de *RobotStudio*

Una vez los puntos han sido creados, la creación de trayectorias es prácticamente trivial. Pues lo único que hay que hacer es click izquierdo en *Trayectorias y Posiciones* y pulsar *Crear trayectoria*. A partir de ahí, hay que ir arrastrando los puntos deseados en orden, para crear la ruta.

Llegado este punto sólo quedaría hacer un ajuste fino de las trayectorias para asegurar que el robot pueda llegar correctamente a cada punto sin exceder los límites de giro de las articulaciones. Para ello es necesario ir a cada una de las instrucciones de movimiento en cada trayectoria, pulsar click izquierdo, y en el desplegable *Modificar una instrucción*, pulsar en *Configuraciones*. Eso desplegará un menú que mostrará al usuario todas las posibles posiciones del robot al llegar a ese punto. Esta parte del desarrollo es iterativa, pues al principio siempre surgirán problemas de movimiento, pero a mayor experiencia en este campo, menor tiempo se tardará en ajustarlo todo correctamente.

En la figura 4.7 se muestran 2 trayectorias creadas a modo de ejemplo, donde las líneas amarillas representan el movimiento que haría el objeto de trabajo del robot.

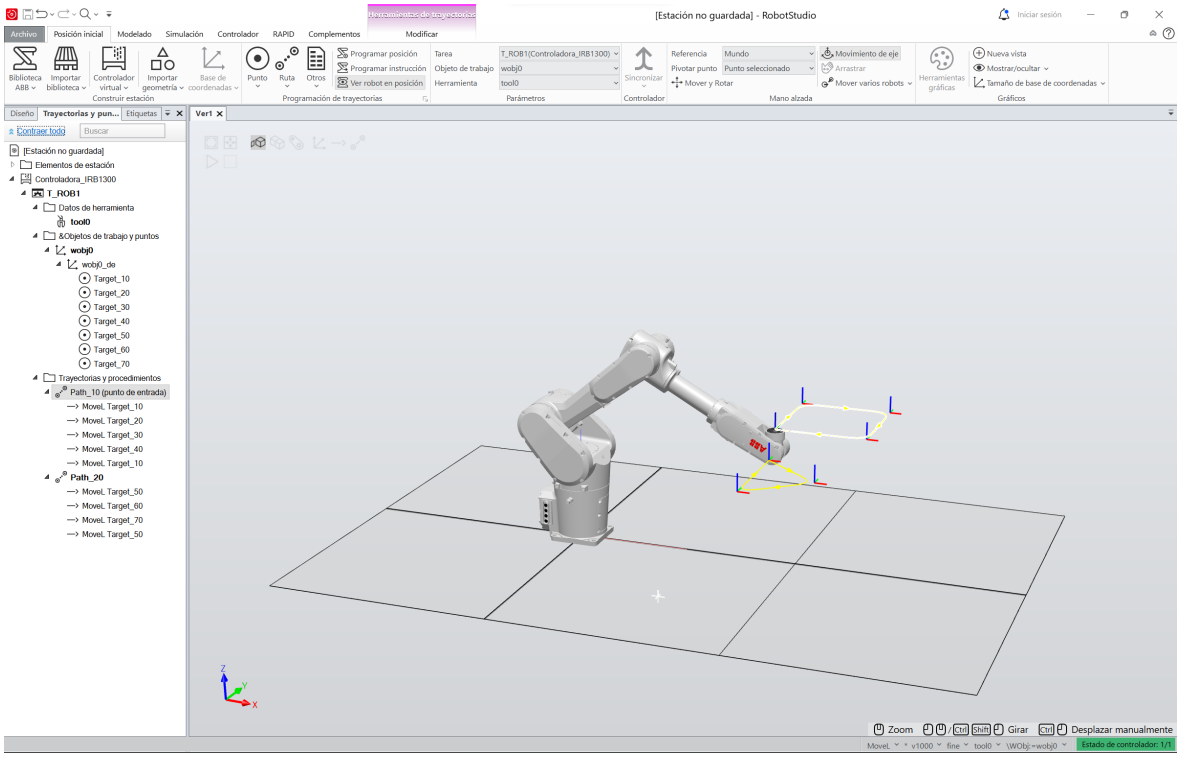



Figura 4.7: Creación de trayectorias en *RobotStudio*

A su vez, si uno se fija en los nombres de las componentes de cada trayectoria, se puede percatar de que todas son precedidas por un nombre. Este es el identificador del tipo de instrucción de movimiento, y en el desarrollo de este proyecto se han empleado principalmente 2. [13]

- *MoveL*. Mueve el robot siguiendo una trayectoria lineal.
- *MoveJ*. Mueve el robot mediante un movimiento de ejes.

La principal diferencia entre estos dos tipos de instrucciones de movimiento se da en que el movimiento lineal es más preciso, pero en contraparte *enreda* las articulaciones del motor, pudiendo hacer que aparezca un error en el movimiento porque alguno de los motores haya alcanzado el límite de giro. La manera de evitar esto es mediante la otra instrución, *MoveJ*, que sacrifica precisión a la hora de seguir la trayectoria con linealidad, pero hace la labor de *desenredar* los ejes.

Por último, se va a presentar el entorno de programación de estos robots. Es llamado *RAPID*, tal y como se muestra en la figura 4.8, su estructura es como la de cualquier lenguaje de programación convencional, y permite añadir todo tipo de operaciones lógicas a las rutinas de los robots. Otra ventaja que añade es que se puede sincronizar con las trayectorias defi-

nidas desde el espacio de trabajo, ahorrando el trabajo de definir los puntos desde el propio IDE. Toda la programación del robot en este proyecto se ha realizado en este lenguaje.

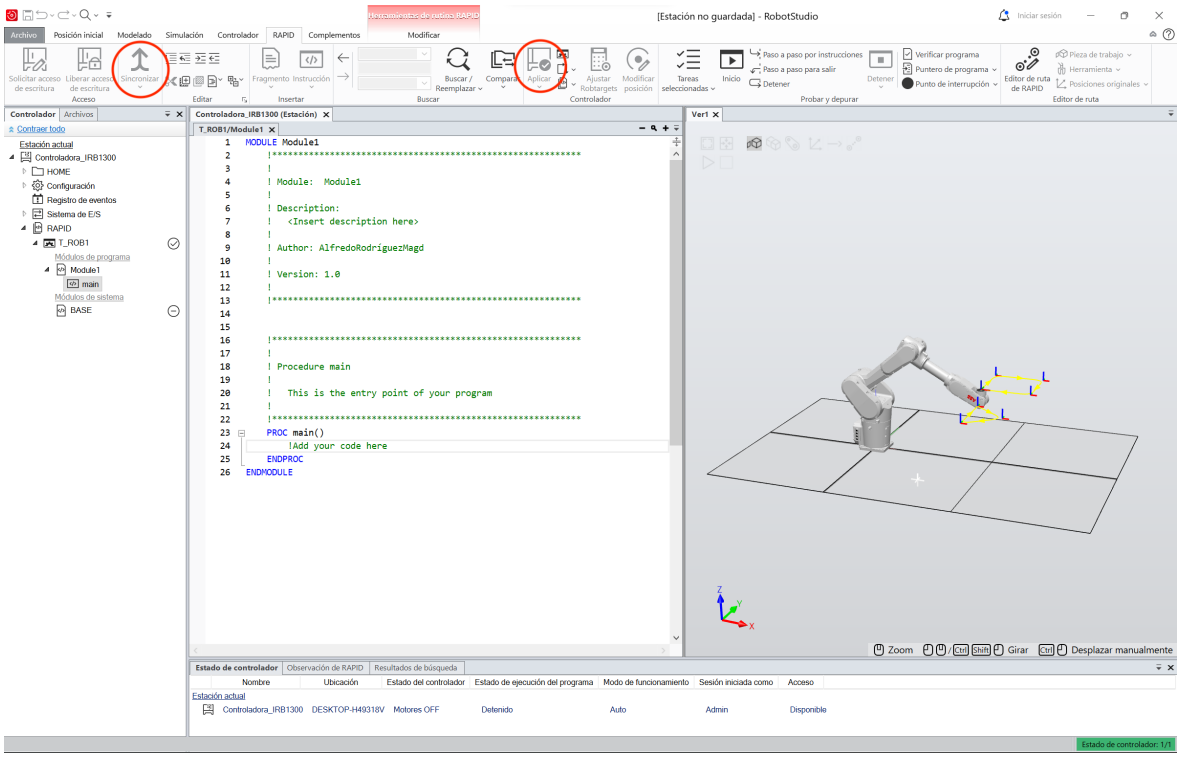



Figura 4.8: Programación con *RAPID* en *RobotStudio*

Para sincronizar la rutina del espacio de trabajo y la del módulo de *RAPID* es necesario emplear el pulsador *Sincronizar*. Y en caso de que se quiera transferir el programa a la controladora, se debe pulsar en *Aplicar*.

Cabe mencionar que no se ha hablado de los ejes de coordenadas, pues a parte del global, se pueden crear todos los necesarios para facilitar el trabajo en determinadas situaciones. Además, tampoco se ha comentado nada del objeto de trabajo, pues como se puede suponer, el robot debe tener acoplada una herramienta para poder hacer la función para la cual se ha comprado. Esto último se comentará más adelante, pues se ha desarrollado con otras herramientas de diseño. La ventaja que aporta el *RobotStudio* sobre esto se ve a la hora de simular, pues es capaz de calcular las inercias, y todos los datos necesarios para determinar al operario si con esa herramienta se pueden realizar las trayectorias deseadas.

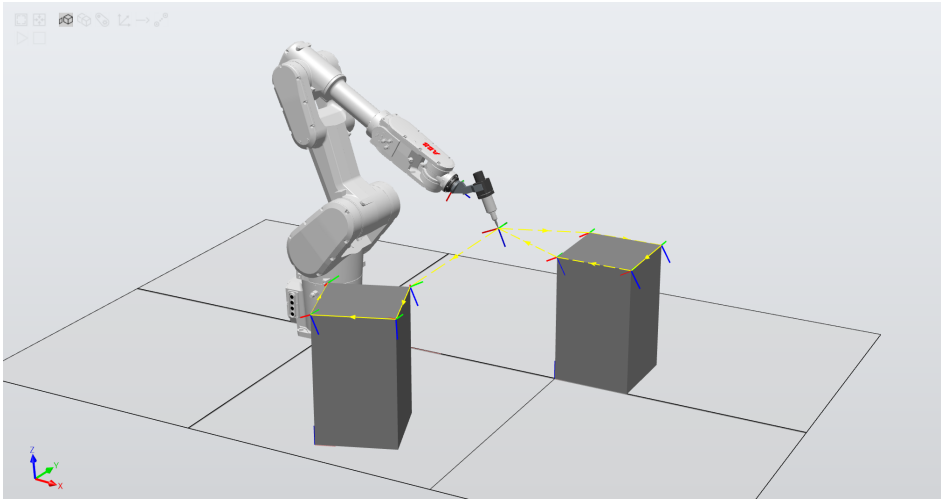


Figura 4.9: Estación de ejemplo con herramienta de trabajo y trayectorias sobre objetos en *RobotStudio*

#### 4.1.1.2.- Programación *in situ* mediante FlexPendant

Otro medio que proporciona *ABB* para programar sus robots es mediante el *FlexPendant*. Tal y como se comentó antes, es una pantalla de operador que permite operar y programar el robot de una forma muy sencilla y **presencial**.[1]

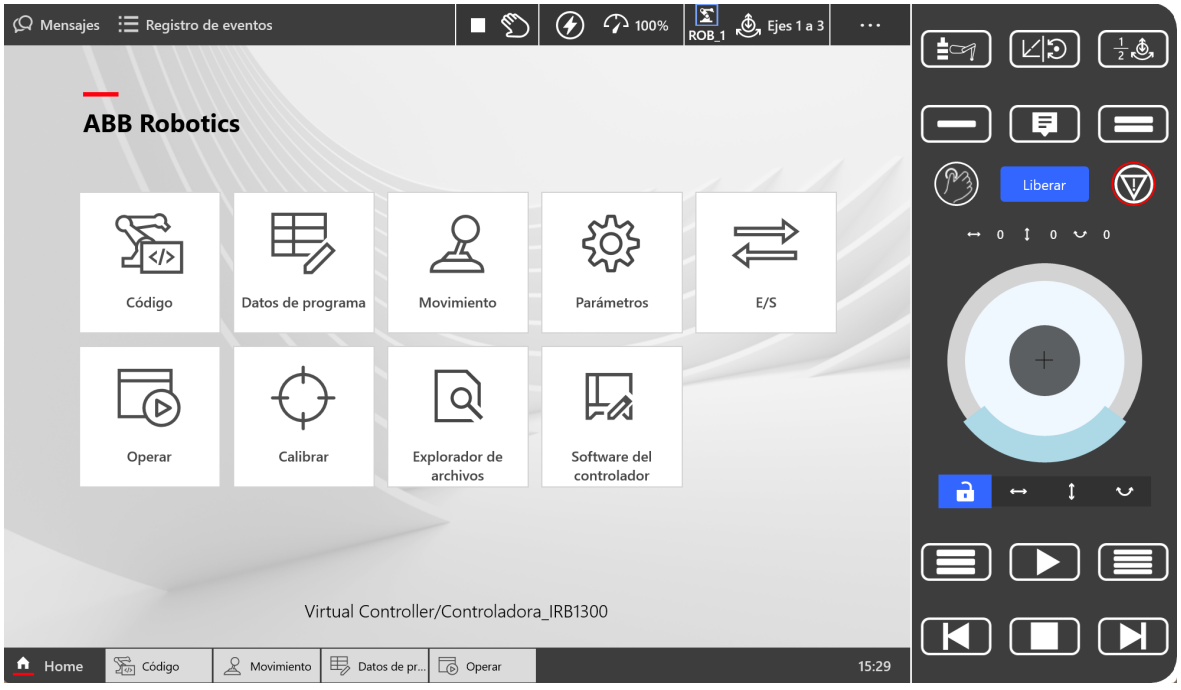



Figura 4.10: Pantalla de inicio del *FlexPendant*

Antes de detallar las aplicaciones presentes en este dispositivo, se hablará del panel de

[1] A lo largo de esta sección se emplearán imágenes de la aplicación *ABB Robotics FlexPendant* para computadores con sistema operativo Windows, pero es exactamente igual que la interfaz mostrada en el dispositivo físico. Con la única diferencia de que controla el robot de la estación virtual, no de la real.

control.

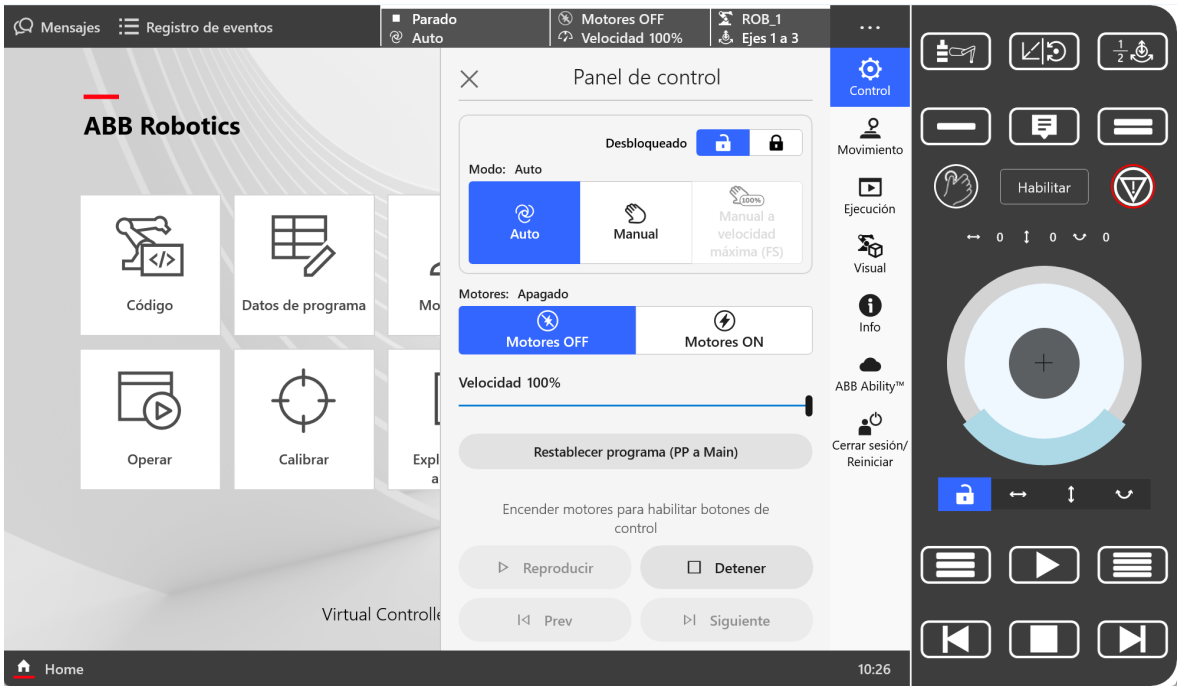



Figura 4.11: Panel de control del *FlexPendant*

Esta interfaz será la más usada del dispositivo, pues permite poner en movimiento al robot en todos sus modos de funcionamiento. Existen tres modos de funcionamiento:

- Modo automático. Ejecuta el programa de control de forma indefinida y a máxima velocidad. Para poder usar este modo de funcionamiento es obligatorio conectar los dispositivos de seguridad al *Customer Interface* de la controladora (véase la figura 4.2).
- Modo manual. Usado para llevar a cabo la labor de configuración y programación del brazo robótico. El robot se moverá a una velocidad reducida.
- Modo manual a máxima velocidad. Ofrece las mismas funcionalidades que en el modo manual pero sin reducir la velocidad.

Si se selecciona el modo manual y se mantiene presionado un interruptor ubicado en la parte trasera del dispositivo. Se cerrará un contactor para encender los motores y se podrá controlar el brazo mediante el *joystick* y los botones físicos de la parte superior. Además de estos, existen 4 botones con acciones configurables por el usuario y una seta de emergencia (en la aplicación aparece como un botón rodeado de color rojo). Por último, abajo del todo, hay 4 botones para controlar el flujo de ejecución de instrucciones del programa de control.

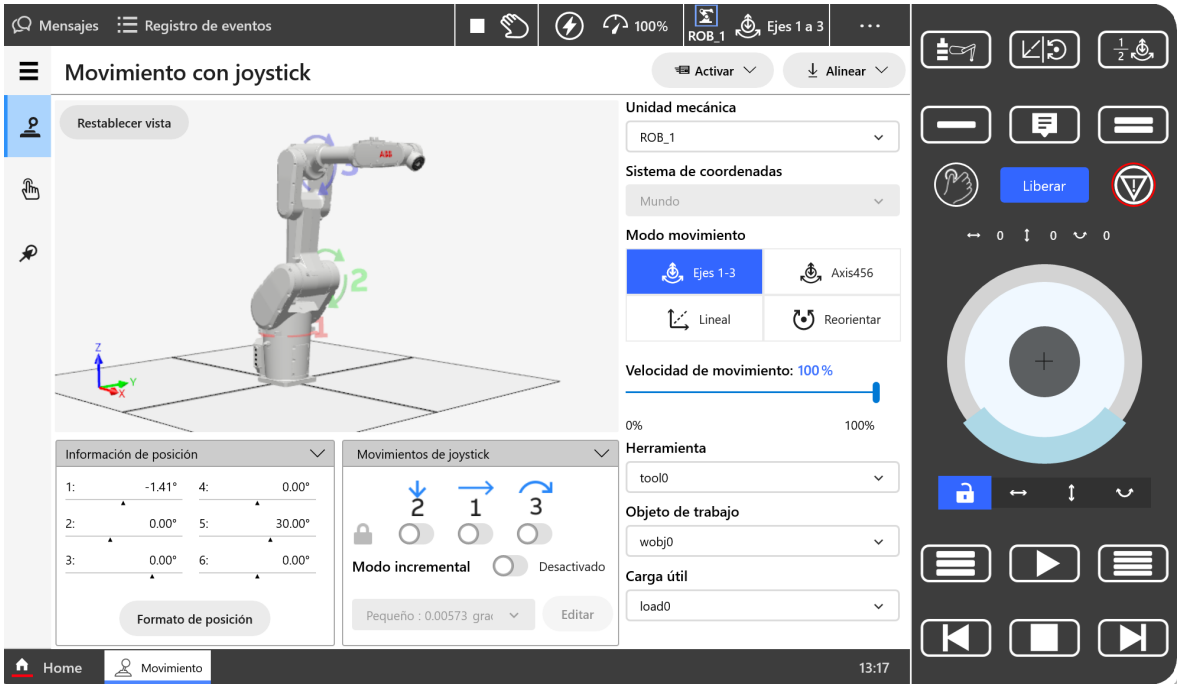



Figura 4.12: Interfaz de movimiento del *FlexPendant*

Se comenzará por comentar el menú de movimiento. Desde este es posible controlar de una forma precisa la posición del robot. Para ello, es preciso saber que estos dispositivos no sólo funcionan con las tres coordenadas lineales habituales (x,y,z), si no que también tienen coordenadas angulares para cada uno de los motores que mueven las articulaciones. Un detalle muy importante a tener en cuenta es que los motores tienen **límites de giro**, pues si aparecen errores a la hora de crear trayectorias de movimiento, la mayor parte de las ocasiones será debido a este hecho. Para poder usar este menú hay que poner el robot en modo manual y encender los motores.

En el menú *Código* se puede programar el robot mediante el uso de instrucciones básicas, esta sección está orientada a diseñar trayectorias con operaciones lógicas. Para cualquier otra función que se desee implementar se debería requerir a un computador con *RobotStudio*. La gran ventaja que ofrece esta interfaz se da en que se pueden guardar puntos en función de la posición real del robot.

Para poder programar el robot usando el *FlexPendant* y el IDE de *RAPID*, se debe conectar un computador al puerto *MGMT* de la controladora (véase la imagen 4.2) mediante un cable RJ-45, abrir el *RobotStudio*, conectarse a la controladora real, y mediante el uso del botón *Aplicar*, se puede modificar el programa de control del robot.

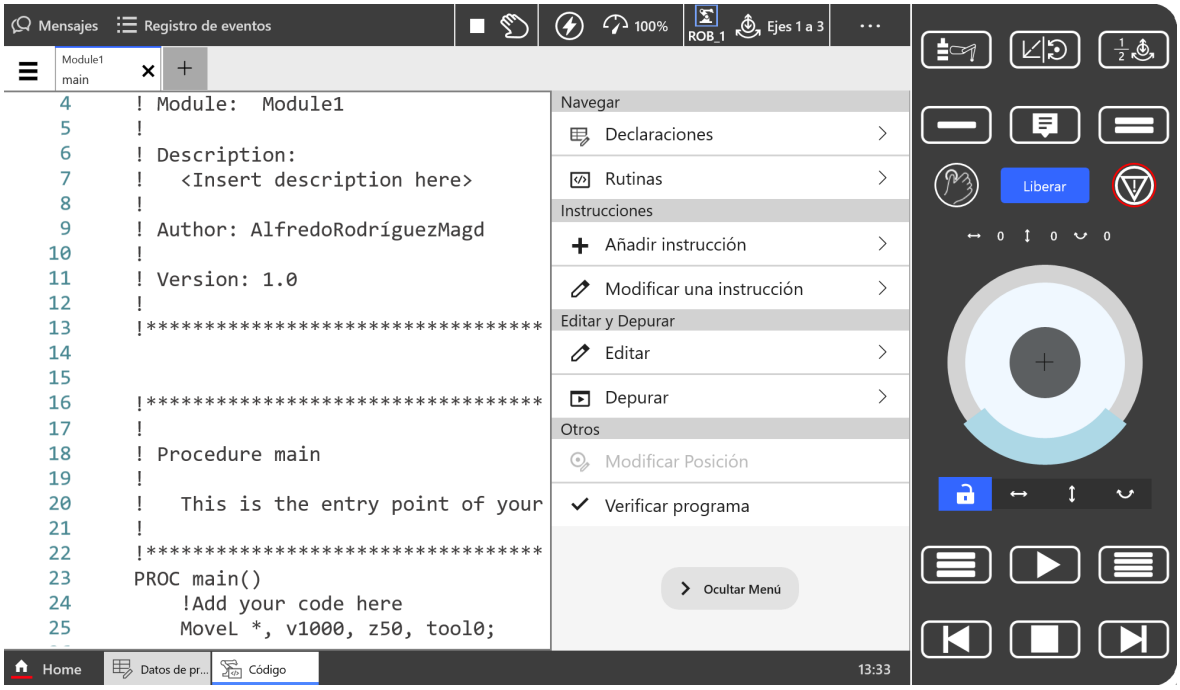



Figura 4.13: Interfaz de movimiento del *FlexPendant*

Por último, se comentará que en el menú *Operar* se puede tener acceso a la terminal de salida del robot, donde se pueden emitir mensajes desde los programas con instrucciones como *TPWrite*. También pueden diseñarse HMIs, pero no será el caso en este proyecto, pues el *FlexPendant* no estará destinado a ser usado por los operarios.

Entonces, una vez comentados los dos métodos para programar estos brazos robotizados, la metodología escogida es una híbrida entre las comentadas. Se hará uso de las estaciones virtuales para llevar a cabo pruebas de funcionamiento en aquellas trayectorias complejas (como en la subestación de adhesivo), y una vez se hayan optimizado se trasladarán al programa real. Y para el resto de operaciones se usará el *FlexPendant* en conjunto con un computador con *RobotStudio*. [14]

### 4.1.2.- Desarrollo genérico de la estación de montaje

Se mostrará una imagen sobre la que se identificarán todos los elementos pertinentes de la célula de montaje después de discutir las opciones potenciales para desarrollar el programa de control en la sección de control.


Figura 4.14: Prototipo de célula de montaje

Si se revisa la figura 4.14, se pueden identificar 5 etapas:

1. Etapa para cargar la bandeja en el brazo robótico.
2. Etapa para dispensar el adhesivo para componentes electrónicos en las pistas de la bandeja.
3. Etapa para pegar las tiras LED a la bandeja por presión.
4. Etapa de descarga de bandeja.
5. Etapa de almacenamiento de bandejas en estantería.

La dinámica de montaje de cada luminaria consiste en que el robot tome una bandeja y la mueva a lo largo de todas las subestaciones, agregando los diferentes elementos necesarios para que la luminaria quede finalmente montada. La herramienta de trabajo debe poder sostener la bandeja y moverla con el brazo sin que se caiga. Para ello, estará equipada con los siguientes dispositivos:

- Válvulas de vacío y ventosas para poder sostener la bandeja una vez se levante.
- Sensor inductivo (cable naranja en figura 4.15) para tener constancia de que la bandeja está correctamente sujeta a la herramienta.
- Ruedas que se usarán en la etapa de adhesión de LED para ejercer presión sobre la bandeja y el molde con el fin de que el pegamento haga correctamente su función.

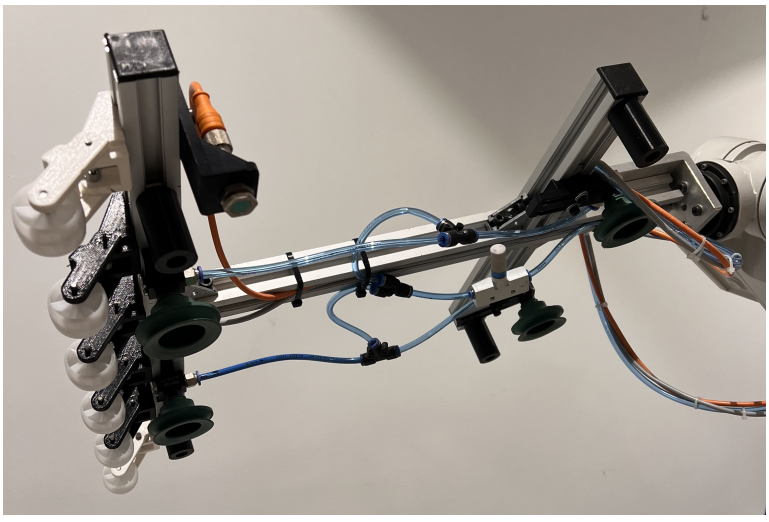


Figura 4.15: Herramienta de trabajo del brazo robótico

Esta herramienta ha sido diseñada con *SolidWorks* por los diseñadores mecánicos y construida por los técnicos del departamento.

A diferencia de los autómatas programables estandarizados, el lenguaje de programación del robot es **secuencial**, es decir, las instrucciones se ejecutan en orden. A partir de esta consideración, se ha optado por diseñar el programa de control con la intención de implementar un sistema comparable al que se sugiere en la guía GEMMA, pero adaptado a lo mencionado anteriormente.

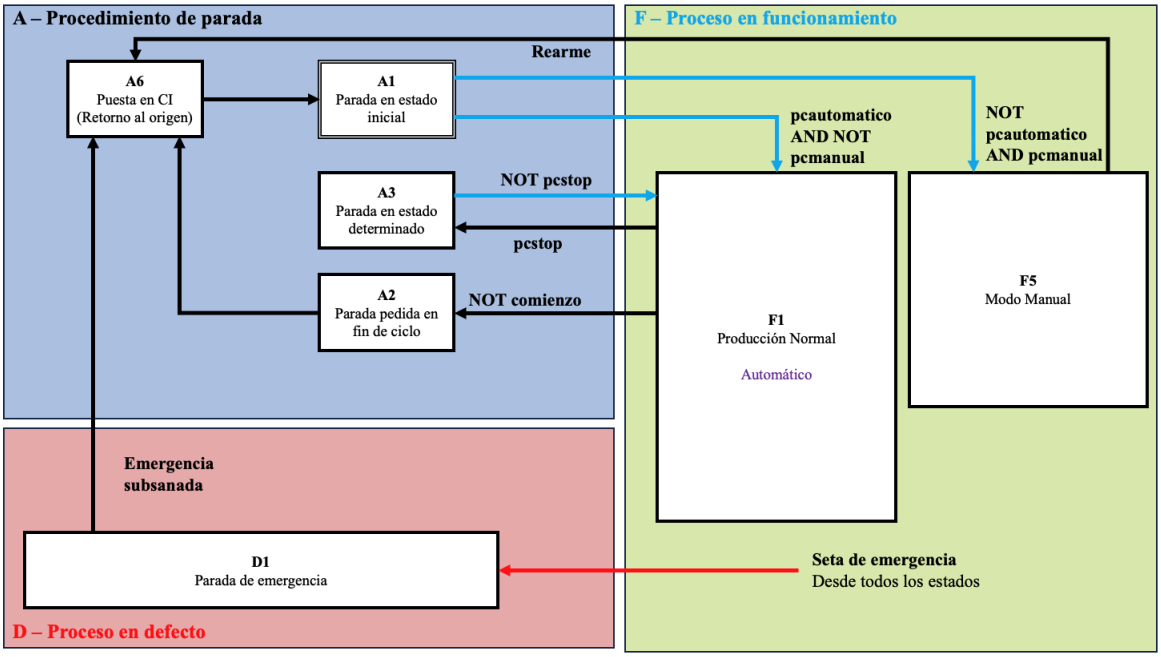



Figura 4.16: Diagrama de estados del robot

Viendo el diagrama de estados de la figura 4.16, se puede ver el funcionamiento del sistema en su nivel más generalista. El cual debe cumplir con las siguientes especificaciones:

- Con la señal *comienzo* se debe de poder poner el sistema en funcionamiento en cualquiera de sus modos. Mientras esta se encuentre en estado alto, el sistema seguirá produciendo. Si se decide poner en estado bajo, se esperará a finalizar el ciclo de montaje y se volverá a condiciones iniciales, es decir, se efectuará un retorno al origen.

- La señal *pcstop* permitirá al operario parar el sistema en cualquier momento de su producción. Pudiendo retomar la producción si se vuelve a poner en estado bajo.

- En caso de emergencia, se deberá actuar sobre la tarjeta E/S del controlador. Ya que incluye entradas específicas para las situaciones de emergencia, que paran todo el sistema mediante contactos normalmente cerrados.

#### 4.1.2.1.- Modo automático

En la gran mayoría de las ocasiones, sobretodo en producción, el sistema funcionará en este modo. El cual ejecutará secuencialmente las operaciones de forma que se vayan llenando las estanterías de bandejas con las tiras LED recién adheridas a sus pistas.

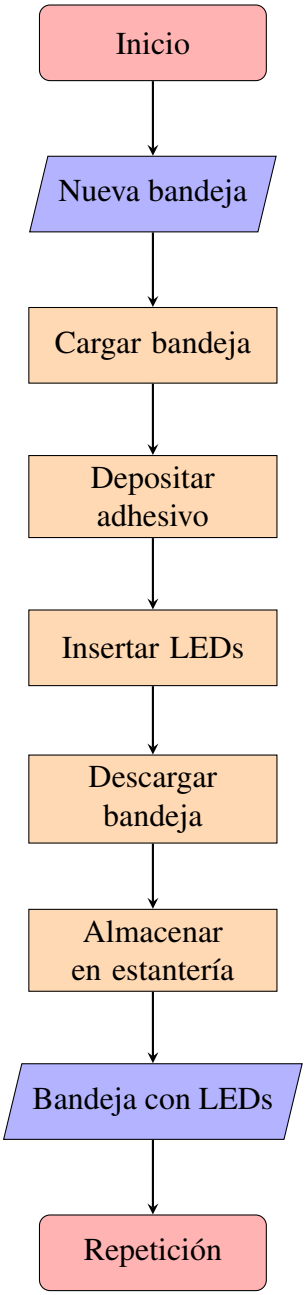



Figura 4.17: Diagrama de flujo del modo automático

#### 4.1.2.2.- Modo manual

En las fases de desarrollo y/o optimización del sistema, será necesario no solo seguir el orden de montaje establecido, sino también poder seleccionar la estación por la que se quiere pasar en todo momento. Este modo de funcionamiento debe utilizarse para ello.

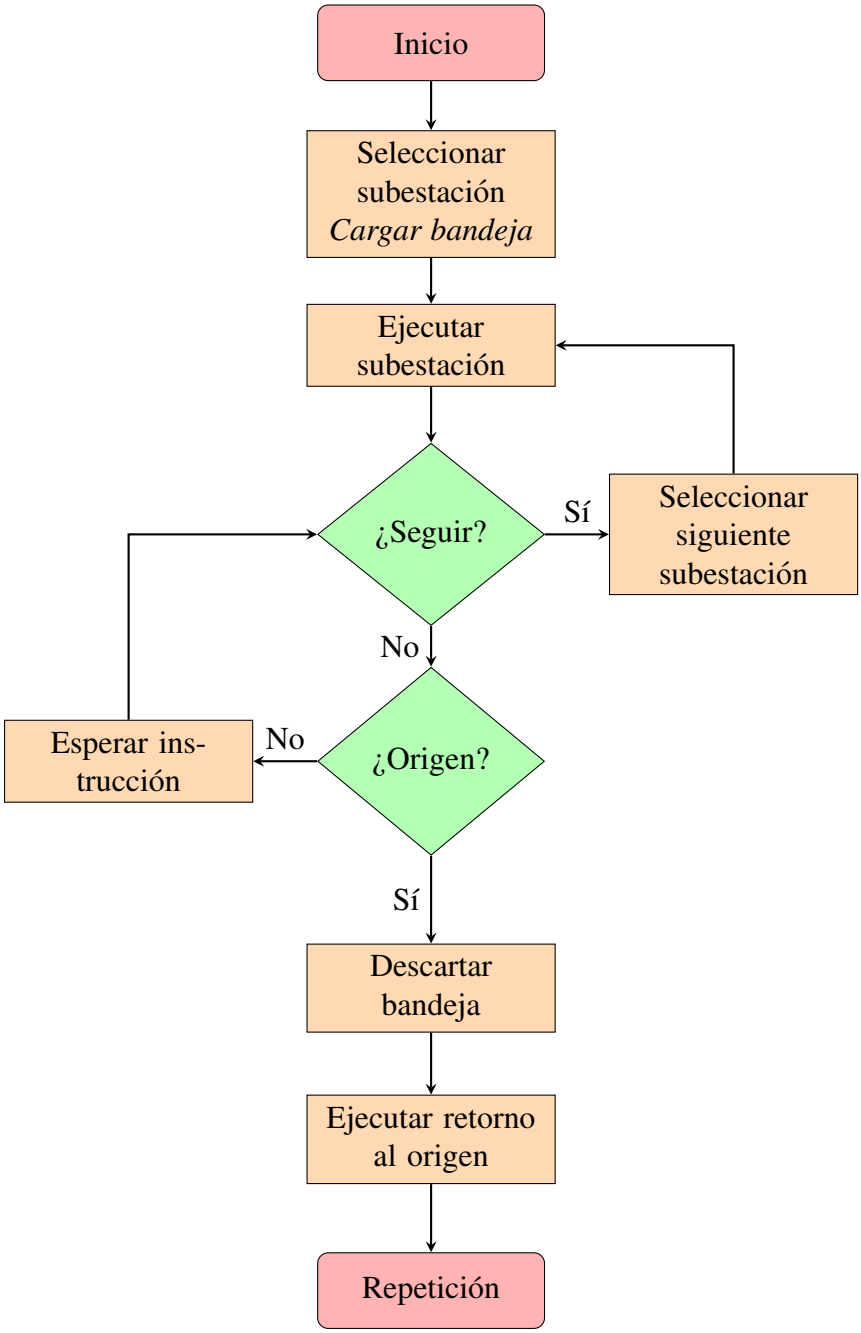



Figura 4.18: Diagrama de flujo del modo manual

### 4.1.3.- Detalle de las subestaciones de la célula

Una vez se ha dado una explicación general del programa de control creado para el robot, se discutirán con más detalle las instrucciones necesarias para realizar las operaciones de cada una de las subestaciones de montaje de la célula automatizada. Es importante recordar que el diseño de este sistema se llevó a cabo en un laboratorio y será necesario trasladar todo a la zona de producción. Por lo tanto, la programación estará enfocada en permitir que el robot se adapte de manera independiente a su ubicación.

#### 4.1.3.1.- Subestación de carga

Esta es la operación inicial a ejecutar. Las bandejas sin componentes se encontrarán ubicadas en una pila de almacenaje *FIFO*, y el brazo robótico cogerá la bandeja ubicada en la parte superior mediante un sistema de vacío.

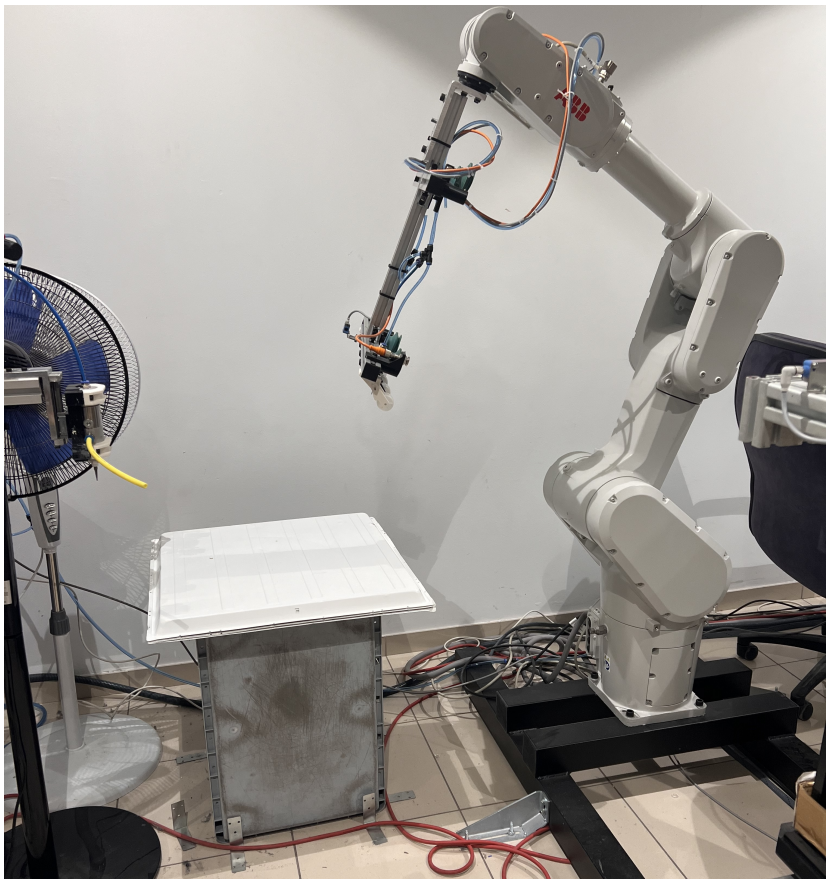


Figura 4.19: Subestación de carga de bandejas en brazo robótico

**Algorithm 1** Proceso de carga de bandeja en brazo robótico

```
1: procedure CARGA
2:     posRobot := pilaBandejas          ▷ El robot se situará encima de la pila de bandejas
3:     valvulaVacio := True              ▷ Se hace vacío para coger una bandeja
4:     while (vacuostato ≠ True) or (sensorInductivo ≠ True) do
5:     end while                         ▷ Se espera a que la bandeja esté correctamente sujeta
6: end procedure
```

#### 4.1.3.2.- Subestación de deposición de adhesivo

A diferencia de la etapa anterior, esta requerirá un nivel de diseño más avanzado, dentro del cual hay que tener en cuenta que la acción de control del lazo que se pretende cerrar será el movimiento del brazo robótico bajo la boca de deposición de adhesivo. En esta etapa de desarrollo del prototipo se decidió emplear un modelo digital, como se mencionó en las metodologías de programación. Se utilizó una versión anterior de *RobotStudio* para crear un gemelo de la planta a controlar. La decisión de realizar las primeras aproximaciones a la solución del problema en un entorno virtual se tomó por razones de seguridad, ya que se iba a operar con velocidades de movimiento que variarían y serían calculadas por algoritmos de aprendizaje automático. Para ver la estación virtual en funcionamiento abra el anexo I.

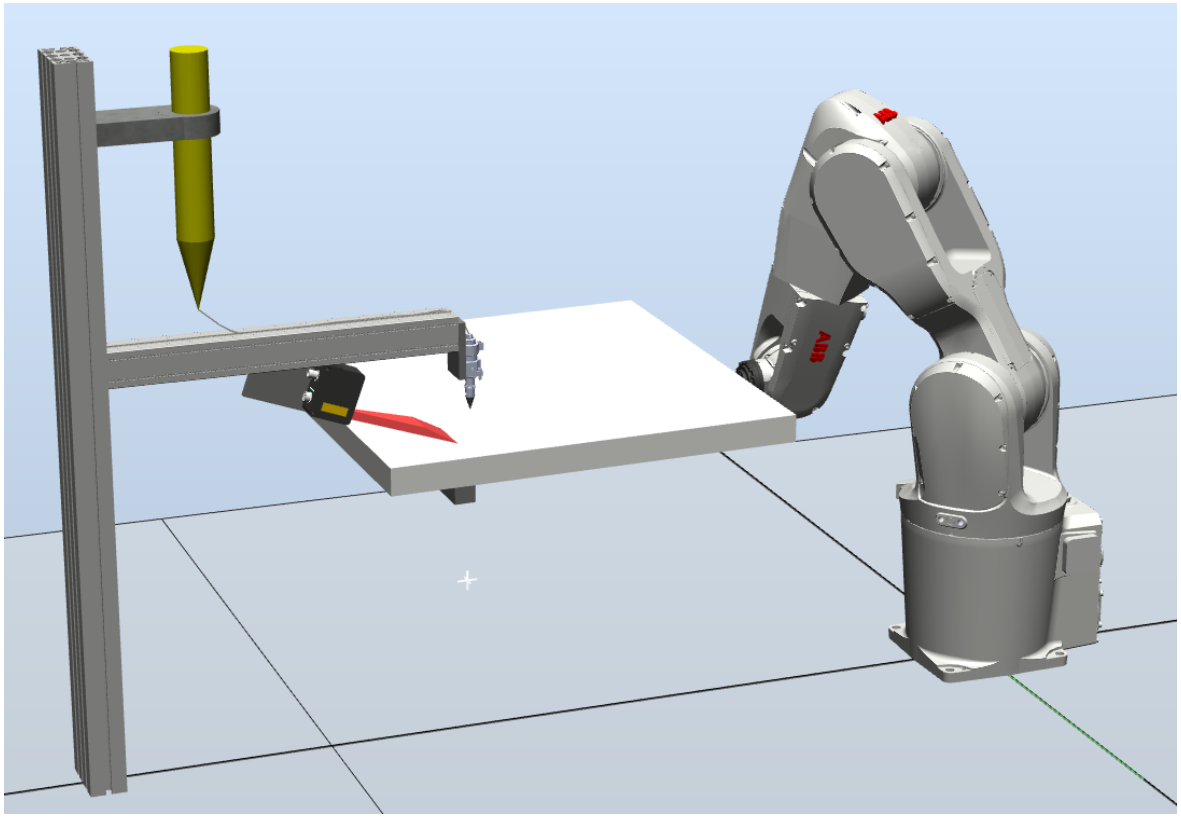


Figura 4.20: Modelo digital de la subestación de deposición de adhesivo

Teniendo en cuenta las consideraciones previas, el desarrollo técnico de esta subestación constará de 3 módulos.

- Para cumplir con el requisito de *adaptación al emplazamiento* del sistema propuesto, deberá existir una función que genere una matriz de puntos que representan los extremos de las pistas de adhesivo (véase la figura 1.1), y deben estar ordenados para que el robot se mueva secuencialmente a través de ellos.
- Un procedimiento que haga que el brazo se mueva por los puntos generados por la función comentada en el punto anterior, y que se encargue de gestionar la deposición de adhesivo en las zonas correspondientes.
- Una tarea que debe de actualizar la velocidad del brazo robótico en tiempo real.

Comenzando por el primer punto, para que el robot recorra las 6 pistas, hay que crear una matriz de 12 posiciones, siendo cada par de ellas las correspondientes al inicio de una pista y al final de esta, respectivamente. Para conservar el carácter *móvil* del sistema, el técnico tendrá que aportar el primer par de puntos, y sobre estos se generará la matriz que se está comentando. [2]

**Algorithm 2** Función para creación de matriz de ubicaciones de pistas de adhesivo

```
1:  function CREAMATRIZPISTAS(pInitPista,pEndPista)
2:      matrizPuntos := [pInitPista,pEndPista]          ▷ Se crea la matriz de puntos
3:      auxInit := pInitPista                           ▷ Se declaran puntos auxiliares
4:      auxEnd := pEndPista
5:      for (int k=1; k<numPistas; k++) do
6:          auxInit.x := distEntrePistas * k        ▷ Se recalculan los puntos para cada pista
7:          auxEnd.x := distEntrePistas * k
                                          ▷ Los puntos se almacenan de forma ordenada
8:          matrizPuntos := [matrizPuntos, auxInit, auxEnd]
9:      end for
10:     return matrizPuntos
11: end function
```

Sobre el algoritmo 2, se debe matizar que las entradas de la función son puntos, es decir, *arrays* con las coordenadas cartesianas, angulares para los ejes del motor, y demás detalles requeridos. A su vez, la función retornará una matriz con 12 puntos de estas características.

[2] Para asegurar la correcta comprensión de los algoritmos, siempre que se asigne un nuevo valor a la variable *posRobot*, se refiere que el robot se moverá a la posición designada

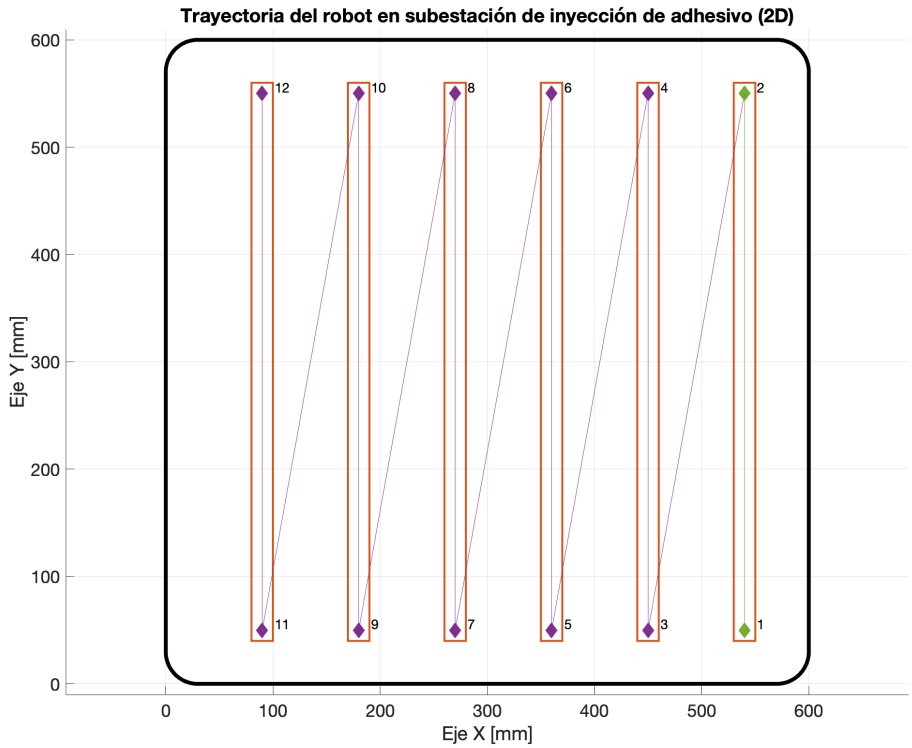



Figura 4.21: Representación en 2D de la trayectoria de la bandeja bajo la boca de deposición de adhesivo

Si se observa la figura superior, se pueden diferenciar dos tipos de puntos, unos de color verde, y otros de color morado. Los primeros son los especificados por el técnico, y los segundos, son los generados por la función. La trayectoria mostrada tiene esta apariencia por una razón en concreto. Debido a que el perfilómetro estará situado justo detrás de la boca de inyección, sólo podrá medir perfiles si el robot avanza en la dirección mostrada en la figura. A su vez, se remarca que se empezará a echar pegamento en un punto impar, y se terminará la operación en uno par.

Si el planteamiento se ciñe a un punto de vista estrictamente ideal, la trayectoria mostrada en la imagen 4.21 sería la adecuada. Pero hay que tener en cuenta que en la realidad, cuando se está dispensando adhesivo con las pistolas convencionales, una vez se deja de ejercer presión, sale siempre una *rebaba* de este, es decir, siempre hay un excedente de adhesivo que sale del bote cuando se corta la presión. Para ello, la solución propuesta es volver a la pista de adhesivo recién depositada, añadiendo esta *rebaba* a la tirada de pegamento, y a partir de ahí pasar a la siguiente pista. La corrección comentada se puede ver en la imagen 4.22.

**Algorithm 3** Procedimiento para llevar a cabo la deposición de adhesivo en las pistas de la bandeja

```
Require: matrizPuntos.size = 12
 1: procedure SUBESTACIONADHESIVO
 2:     posRobot := matrizPuntos[1]                    ▷ Mover robot a posición inicial
 3:     for (int k = 1;k<12;k++) do                    ▷ Se recorre la matriz de puntos
 4:         posActual := matrizPuntos[k]
 5:         posSiguiente := matrizPuntos[k+1]
 6:         if k es impar then
 7:             inyectarPegamento := True              ▷ Abrir válvula para depositar adhesivo
 8:             posRobot := posSiguiente               ▷ Ejecutar movimiento a fin de pista
 9:             inyectarPegamento := False             ▷ Cerrar válvula de adhesivo
10:             posRobot := posSiguiente - margenRebaba
11:         else if k es par then
12:             posRobot := posSiguiente               ▷ Ejecutar movimiento a inicio de pista
13:         end if
14:     end for
15: end procedure
```

Como última anotación a considerar en el desarrollo de esta subestación, queda por comentar el protocolo para actualizar la velocidad lineal del brazo en tiempo real. Para ello, el lenguaje *RAPID* permite el uso de interrupciones dentro de una tarea, y dentro de una de estas se empleará la instrucción *SpeedRefresh*, que actualiza la velocidad del brazo en un porcentaje sobre la programada, p.ej., si se tiene configurada una velocidad de $1000mm/s$, y se ejecuta esta instrucción pasándole como parámetro 30, la nueva velocidad del brazo será $300mm/s$. Entonces, cuando se ejecute el procedimiento encargado de mover el brazo robótico bajo la boca de deposición (3), se activará una rutina que tendrá el siguiente aspecto.

**Algorithm 4** Rutina de tratamiento de interrupción generada por cambio de velocidad

```
 1: function RTIVELOCIDAD
 2:     SpeedRefresh(porcentajeVelocidad)
 3:     if ERROR then                    ▷ Si hubo errores en la ejecución de la instrucción
 4:         if porcentajeVelocidad > 100 then
 5:             porcentajeVelocidad := 100
 6:         else if porcentajeVelocidad < 0 then
 7:             porcentajeVelocidad := 0
 8:         end if
 9:         Repetir rutina
10:     end if
11: end function
```

Hay que tener en cuenta que la interrupción será generada cada vez que la variable *porcentajeVelocidad* cambie, y su valor será actualizado por un módulo de comunicaciones que será comentado en el siguiente capítulo.

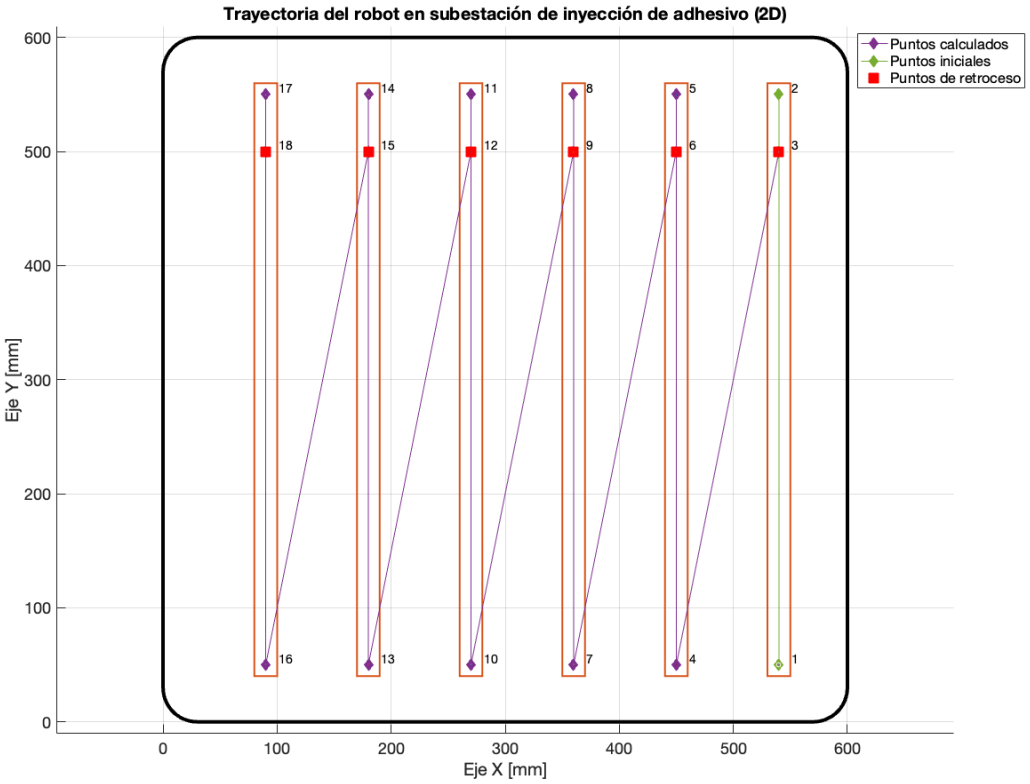



Figura 4.22: Representación en 2D de la trayectoria real de la bandeja

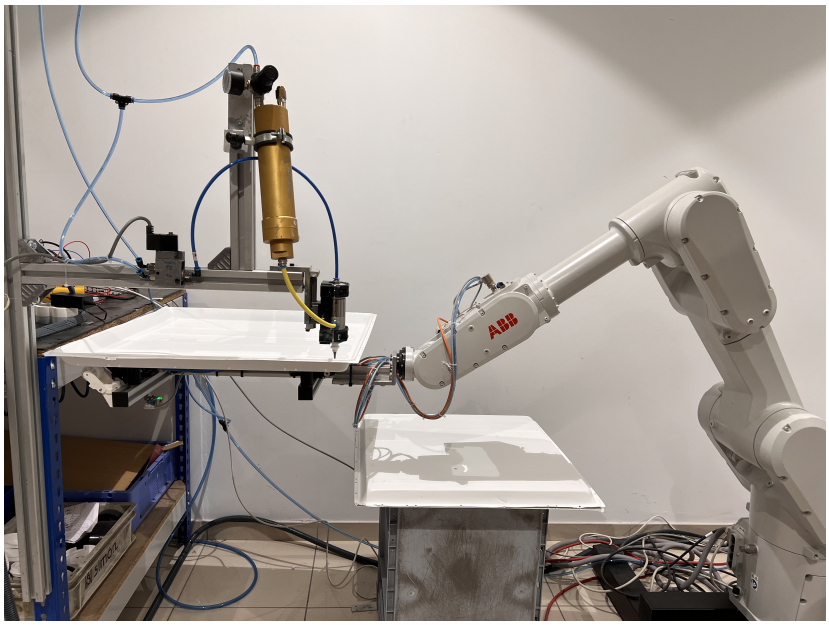



Figura 4.23: Montaje real de la subestación de adhesivo

#### 4.1.3.3.- Subestación de inserción de tiras LED

Una vez el adhesivo ha sido depositado en las pistas de la bandeja, hay que proceder a colocar las tiras LED para su posterior cableado. Para ello el brazo robótico colocará la bandeja boca a bajo en un molde donde estarán estos dispositivos electrónicos colocados de manera que nada más las ventosas dejen de hacer vacío, entren en contacto con el adhesivo, y queden adheridas.

**Algorithm 5** Procedimiento para la inserción de LEDs en las pistas con adhesivo (Parte I)

```
1: procedure INSERTARLEDS
2:     posRobot := moldeLEDs        ▷ El robot se sitúa encima del molde con las tiras LED
3:     valvulaVacio := False                          ▷ Se suelta la bandeja encima del molde
4:     while (vacuostato ≠ False) or (sensorInductivo ≠ False) do
5:     end while                  ▷ Se espera a que la bandeja esté correctamente depositada
```

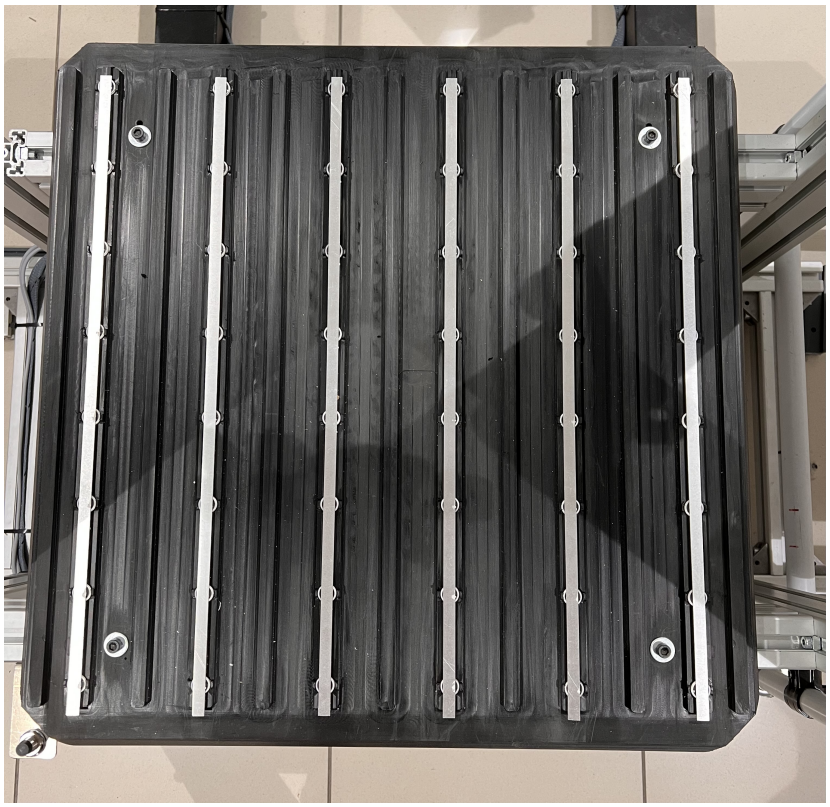


Figura 4.24: Molde de inserción de tiras LED

Para asegurar que las tiras queden correctamente pegadas a sus respectivas pistas se emplearán los rodillos del objeto de trabajo (véase la figura 4.15), inclinando la herramienta y haciéndolos pasar por encima de la bandeja, y el molde.

**Algorithm 6** Procedimiento para la inserción de LEDs en las pistas con adhesivo (Parte II)

```
6:    workObject.rotar(45)                          ▷ Se rota la herramienta 45º
7:    posRobot := moldeLEDs + longitudMolde         ▷ Se pasan los rodillos por la bandeja
8:    workObject.rotar(0)      ▷ Se devuelve la posición angular original a la herramienta
9:    posRobot := moldeLEDs
10:   valvulaVacio := True                          ▷ Se hace vacío para coger una bandeja
11:   while (vacuostato ≠ True) or (sensorInductivo ≠ True) do
12:   end while                ▷ Se espera a que la bandeja esté correctamente sujeta
13: end procedure
```

#### 4.1.3.4.- Subestaciones de descarga y almacenaje

Habiendo una bandeja lista para almacenar, por motivos de ahorro de espacio, se va a apilar con otras ya montadas antes de su almacenaje. Serán depositadas encima del molde de inserción de los LEDs (zona 4 en figura 4.14). Y una vez ya se han apilado unas 2 o 3 bandejas, se procederá a almacenarlas en la estantería de secado.

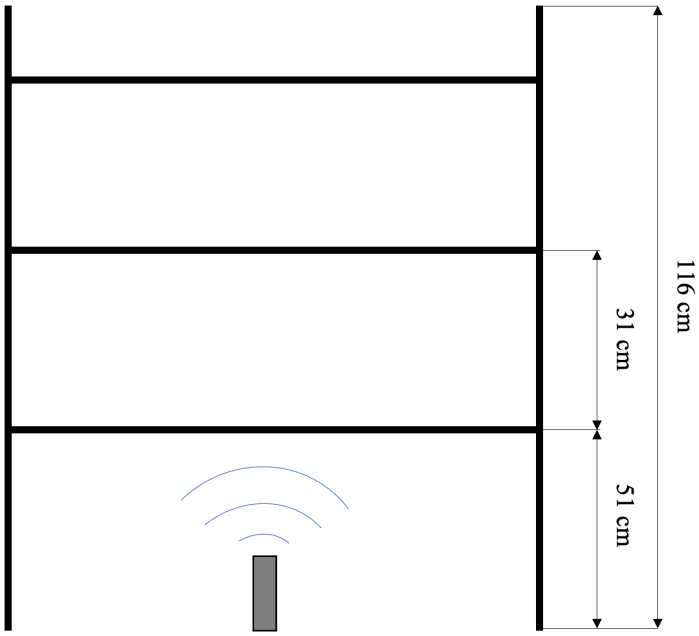



Figura 4.25: Medidas de un soporte de secado de bandejas (Sujeto a cambios)

Teniendo en cuenta que el robot empezará a almacenar las bandejas por las baldas superiores, para decidir en qué balda se va a depositar la pila de bandejas se empleará un sensor de ultrasonidos, determinando la distancia a la balda llena más lejana y traduciéndola en un valor discreto que indicará la balda a la que debe acudir directamente el brazo.

## 4.2.- Selección de actuador para dispensar adhesivo

Hasta este punto de la memoria, se ha hablado del bote de adhesivo, a su vez, se ha desarrollado minuciosamente el procedimiento que seguirá el brazo robotizado a la hora de pasar por debajo de la boca de dispensado de este material. Pero no se ha comentado el **actuador** que se encargará de ejercer la presión necesaria para que el adhesivo circule y acabe en las pistas de las bandejas.

En esta memoria se comentarán dos alternativas, las cuales tienen ventajas e inconvenientes. Se escogerá una, y con ella se llevará a cabo la labor de investigación y desarrollo del sistema en lazo cerrado de dispensación de adhesivo.

### 4.2.1.- Uso de servomotor como actuador

Una de las opciones valoradas en este proyecto es el uso de un servomotor de SMC como actuador, ya que este dispositivo se encontraba en desuso por una línea de montaje parada.

Se trata de un *LEFS25A* de SMC. Tiene una carrera de 700 mm, su motor es uno paso a paso de $24V_{DC}$, gobernado por la controladora *JXCE-1* mediante el bus de campo *ethercat*. Para el problema propuesto, hay dos datos muy importantes a tener en cuenta:

- Velocidad de empuje (Varía entre 12 y 30 $mm/s$)
- Carga de trabajo (Fuerza máxima disponible para empujar)

Dichos parámetros serán fundamentales para saber cuánto adhesivo se podrá dispensar por unidad de tiempo.

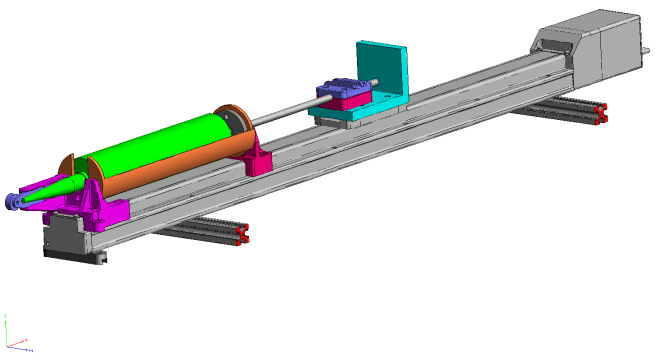
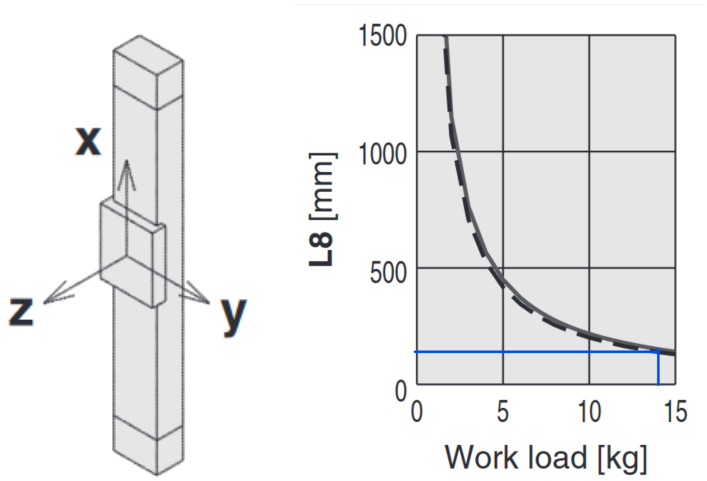


Figura 4.26: Modelado 3D del dispensador de adhesivo. (Diseñado por Jorge González Lastra, especialista en diseño mecánico del departamento)

Para saber el caudal que podrá proporcionar el sistema, es necesario saber la fuerza que puede aportar el servomotor. Para ello, será necesario acudir al catálogo del fabricante, en específico, a la página 35, donde se puede ver el procedimiento a seguir para calcular el dato.

Con respecto al eje de coordenadas mostrado (4.27a), se ubicará la posición de la carga.

$$(L_x, L_y, L_z) = (0, 0, 100)mm$$




(a) Referencia de coordenadas del fabricante para uso en vertical.

(b) Gráfica del fabricante

Figura 4.27: Estimación de carga máxima de trabajo[2]

Si se emplea la figura 4.27b, se puede deducir que la carga máxima de trabajo es de $14Kg$. Entonces...

$$\text{máx}(F_{servo}) = 14[Kg] \times 9{,}81[m/s^2] \approx 140[N] \tag{4.1}$$

Para controlar el servomotor, se hará uso de un PLC como dispositivo maestro (en específico, el modelo CX9020 de Beckoff), el cual le dará las instrucciones necesarias para su correcto funcionamiento dentro del contexto del proyecto.

Para poder conectar el PLC al servomotor es necesario emplear una controladora que disponga de un bus de campo compatible con el del autómata, siendo en este caso **Ethercat**. Un bus de campo basado en Ethernet, pero adaptado para cumplir con requisitos de tiempo real.

A su vez, el autómata tendrá la doble funcionalidad de coordinar el robot con el servo-

motor, ya que lo que se pretende es que el robot decida cuándo se debe echar pegamento y cuando no. Para ello se empleará un convenio de señales Ready-Fail-Pass que servirá para indicarle en todo momento al robot la situación de la unidad de inyección.

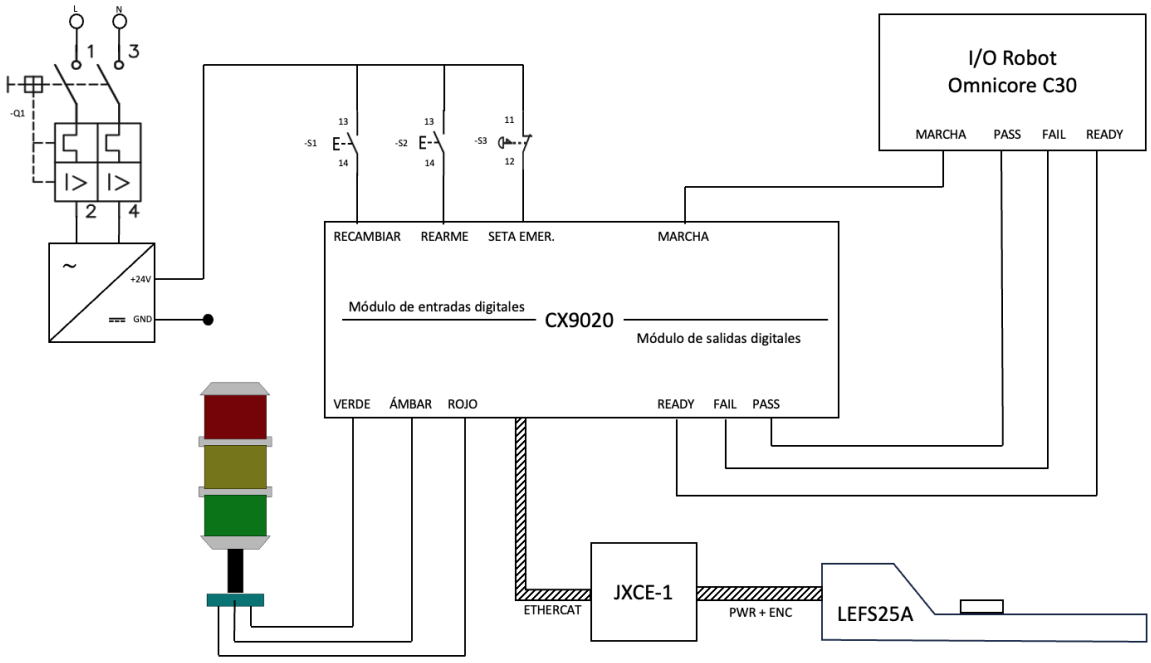



Figura 4.28: Esquema eléctrico de conexiones entre PLC, Robot y Servomotor

La figura superior muestra el conexionado entre todos los elementos del proyecto para que sea posible el control de la unidad de inyección desde el robot (No se ha mostrado la conexión de las masas por motivos de simplicidad). Para ello, el programa del PLC deberá seguir las siguientes especificaciones:

- Mediante la señal *MARCHA* se debe poder gobernar el servomotor. Si dicha señal lógica está en estado alto, el servomotor debe estar haciendo la operación *pushing* (para entender lo que significa esta operación, y cómo se programa, revisar el anexo II). Si se produce un flanco de bajada, la operación se deberá detener, y un retroceso de 3mm hacia atrás tendrá que realizarse para eliminar la fuerza residual del vástago sobre el émbolo, reduciendo así la rebaba del pegamento.

- Con la señal *RECAMBIAR*, el vástago debe volver al origen, permitiendo así el recambio del bote de adhesivo. Debe tener un pulsador asociado.

- Estará dotado de un pulsador de rearme que tendrá la doble función de indicar que el bote ha sido recambiado, y de gestionar defectos.

- Por normativa, tendrá una seta de emergencia que declarará el estado de emergencia.

- Habrá 3 señales digitales que se encargarán de informar al robot del estado de la subestación.
- Por motivos de ahorro de energía, si el servomotor está encendido más de 15 segundos y sin usarse, se apagará la etapa de potencia hasta que sea requerido su uso de nuevo.

Toda la programación del autómata será llevada a cabo siguiendo la guía GEMMA, dicha decisión de diseño se justifica por si se diera el caso de que otro técnico tuviera que reutilizar el código, de esta forma, siguiendo todo un estándar, es más sencillo de entender.

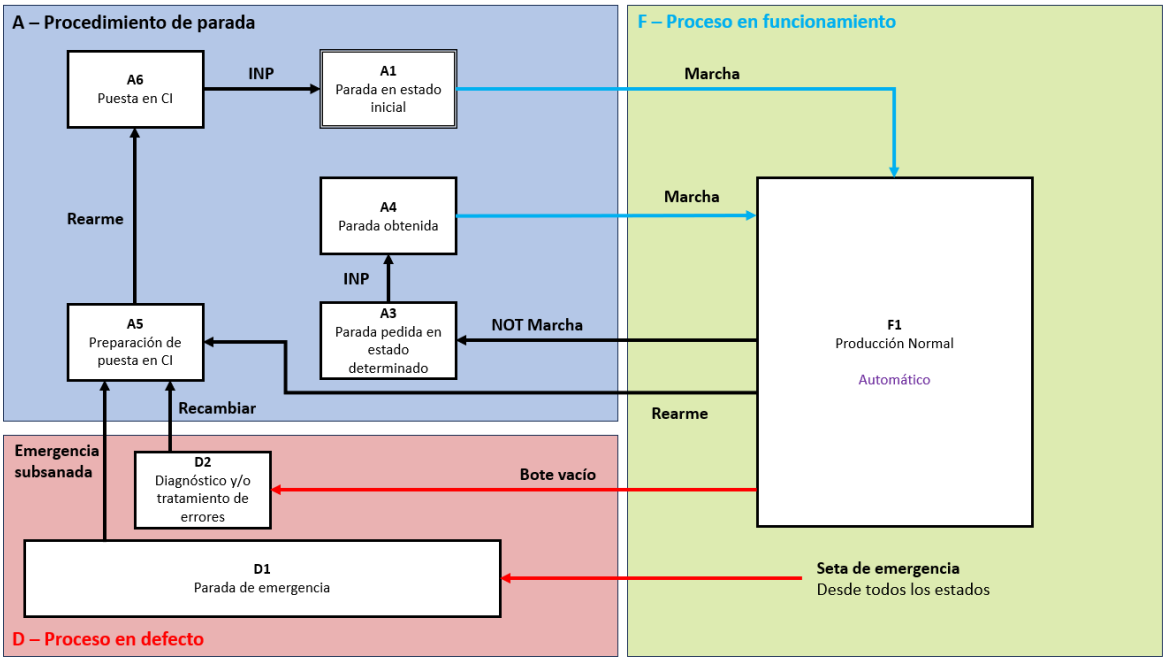



Figura 4.29: Diagrama de estados del servomotor siguiendo la guía GEMMA

La implementación de este diagrama de estados se puede subdividir en 2 programas principales:

- **Programa maestro**. En este bloque de código se gobernará todo el sistema. Aquí se vería implementada la guía GEMMA mediante programación por fases en ST.
- **Programa esclavo**. Este programa se encargará de cargar un movimiento en la controladora del servomotor. Dicho movimiento será el que decida el programa maestro en función del estado en el que se encuentre. Estará hecho en SFC.

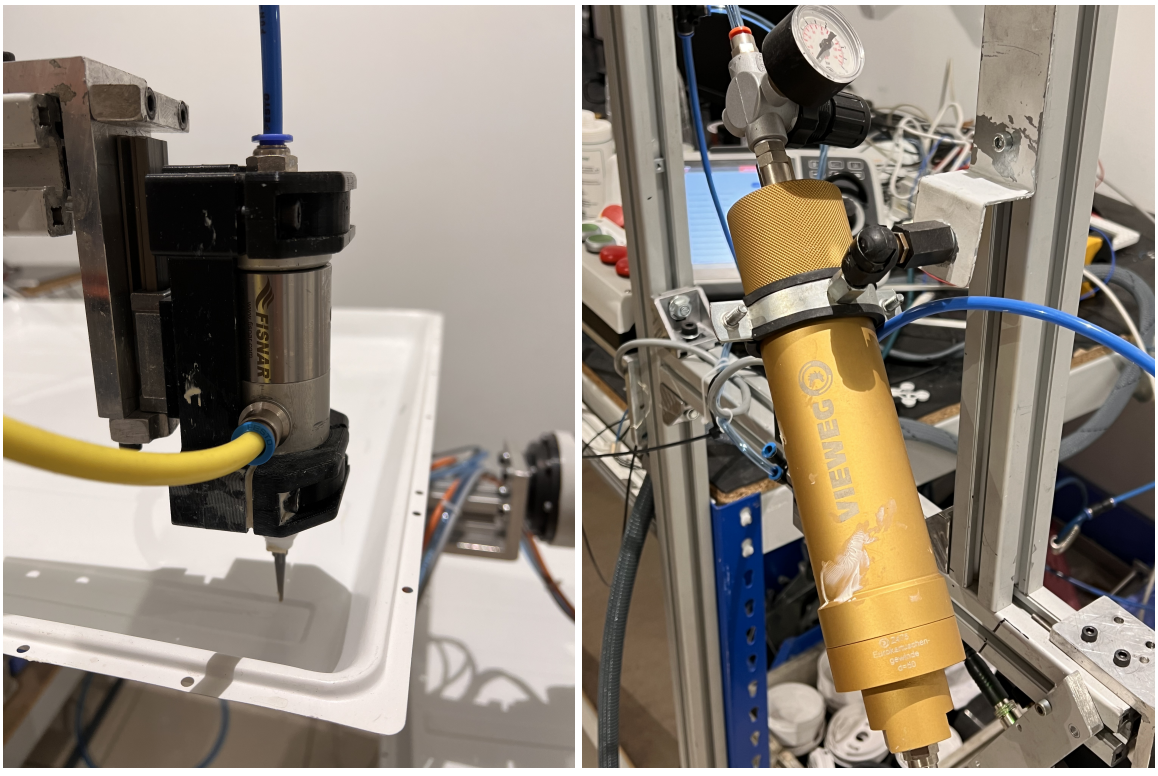
A su vez, existirán programas para gestionar las transiciones del SFC y subrutinas para resetear estados y movimientos. Para ver el programa en detalle, véase el anexo III (documento a parte).

Para programar el PLC es necesario descargar el entorno de TwinCat (en este proyecto se usó la versión 2, pero ya existe la versión 3 compatible con visual studio), el cual tiene 2 aplicaciones principales:

- **TwinCat System Manager**. Desde este portal se pueden definir todos los parámetros a nivel de comunicaciones del PLC, desde la red a la que está conectado, sus credenciales, la gestión de todas las tarjetas hardware que se introduzcan en el dispositivo, las comunicaciones mediante buses de campo, etc...
- **TwinCat Program Control**. Desde aquí se lleva a cabo la programación del autómata en su sentido más literal, se definen las tareas (POUs), y se programa en el lenguaje deseado. Cabe mencionar que la programación de estos PLCs es altamente similar a CodeSys.

### 4.2.2.- Uso de cámara presurizada como actuador

La segunda y última opción valorada para dispensar el adhesivo es un cámara presurizada a la presión de la línea (6 bares). Para eyectar el material se ha decidido emplear una válvula de control todo/nada de fluidos en estado líquido.




(a) Válvula empleada para dispensar el pegamento (b) Cámara presurizada

Figura 4.30: Elementos del circuito de adhesivo

Si se observa la figura 4.30a, se pueden ver dos conductos, uno amarillo y uno azul. El primero de estos será por el cual circulará el adhesivo. Por el segundo circulará aire comprimido a la propia presión de la línea. Teniendo esto en cuenta, la lógica a seguir será la siguiente:

- Si hay aire comprimido entrando en la válvula, este cerrará el paso al adhesivo, haciendo que no salga por la parte inferior.
- Si por el contrario no hay aire circulando, la válvula abrirá el paso al adhesivo para salir por la boca de deposición.

La circulación o no del aire comprimido será gestionado por una electroválvula 3/2 monoestable que estará conectada a la tarjeta de salidas digitales de la controladora del robot.


Figura 4.31: Electroválvula 3/2 monoestable usada para inyectar aire en la válvula de fluidos

### 4.2.3.- Actuador escogido para etapa de desarrollo

Tras la realización de varias pruebas de dispensado con ambos actuadores, se ha llegado a la conclusión de que el servomotor no es lo suficientemente potente. Pues una vez se consumía el 80 % del bote, se generaba una alarma de sobrecarga. Por ello se decidió hacer un estudio de la presión ideal para dispensar adhesivo.

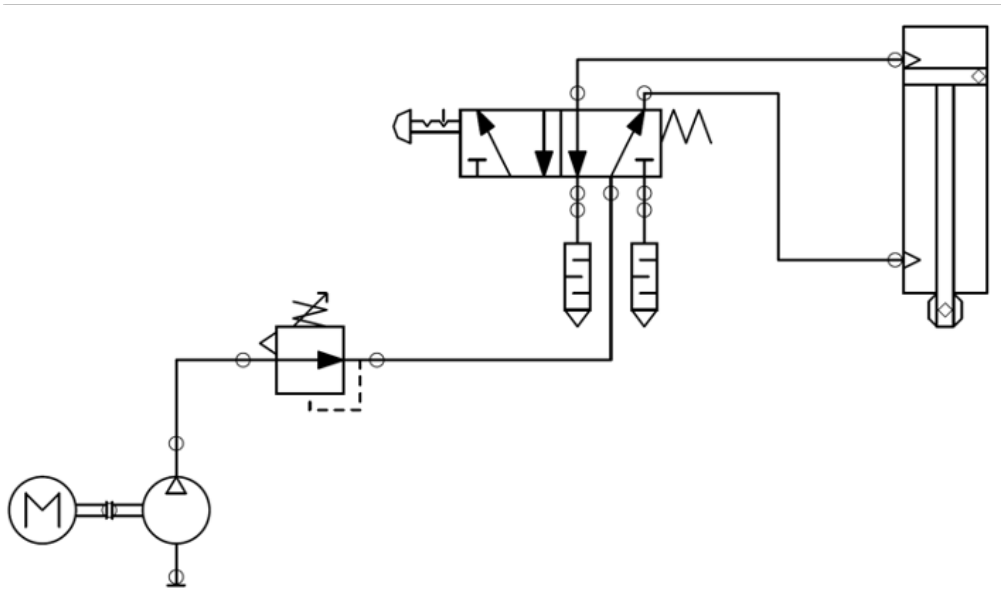


Figura 4.32: Circuito neumático empleado para el estudio del rango de presiones ideales para la deposición

Si se revisa el esquema mostrado en la figura 4.32, se puede ver el circuito neumático empleado para hacer el estudio mencionado anteriormente. La idea de este es ejercer presión con un cilindro de doble efecto a distintas presiones de alimentación, las cuales irán variando gracias a un regulador de presión. La decisión de si una presión es adecuada o no se remite a un criterio subjetivo.

Antes de mostrar los resultados del experimento, se recuerda al lector la relación entre presión y fuerza dada en el cilindro. Esta fue empleada para analizar los resultados.

$$F_{empuje} = P_{alimentacion} * \pi R^2 \tag{4.2}$$

Donde $R$ es el radio del émbolo del cilindro. Si se recuerda que la fuerza máxima del cilindro es de 140 N, y que el margen adecuado de presiones de funcionamiento es la banda de color rojo mostrada en la figura posterior (verde en caso de querer analizar las fuerzas), los resultados son los siguientes.

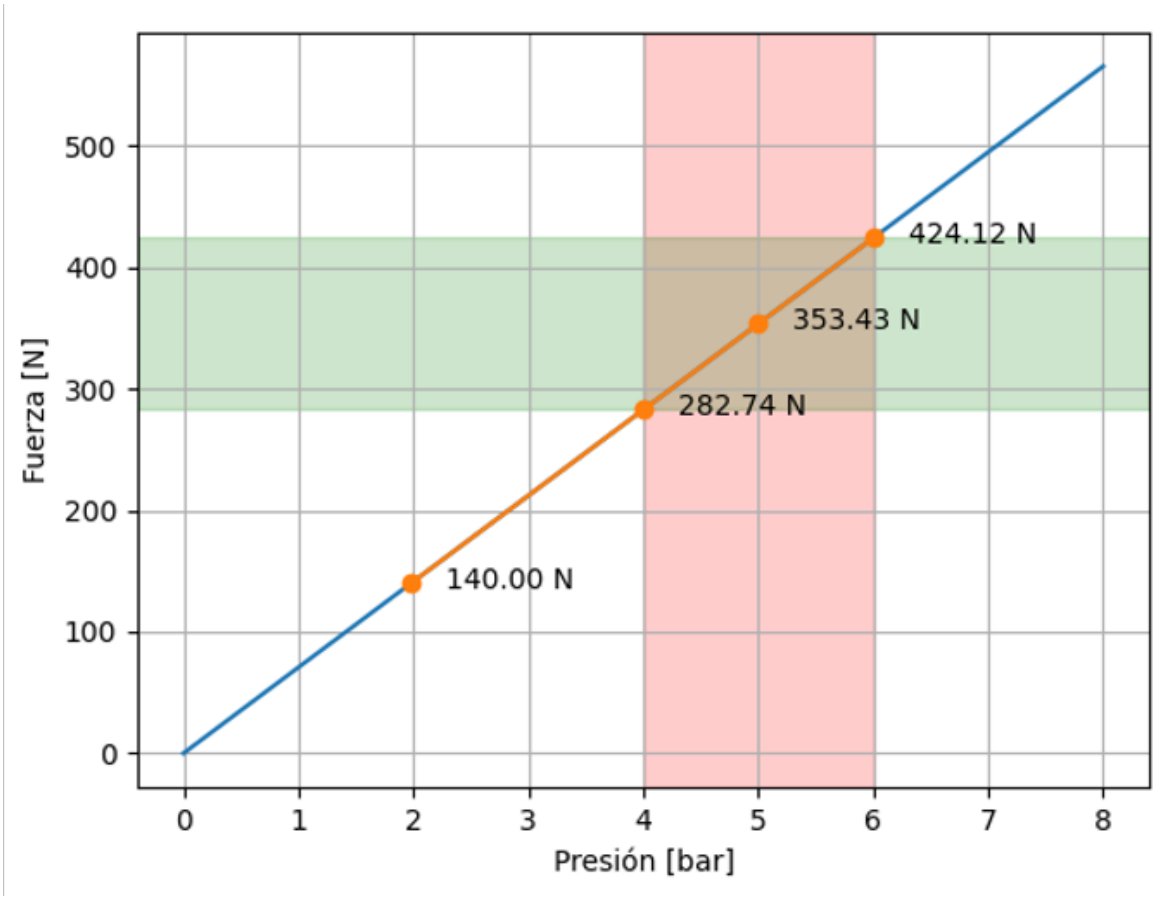



Figura 4.33: Análisis experimental de rango de presiones adecuadas

Entonces, dado que no se disponía de un servomotor más potente, la única opción válida restante acabó siendo el uso de la cámara presurizada. Aún así, el servomotor es realmente la mejor opción, pues permite actuar directamente sobre la velocidad del vástago, lo que facilita en gran medida la medición de caudales de salida, para ello simplemente bastaría con emplear la ecuación 3.10. A su vez, como la controladora del servomotor admite control por posición, se puede saber en todo momento la posición del vástago, y por consiguiente la altura restante del bote de adhesivo.

**Tabla 4.1** Cuadro resumen de posibles actuadores

| Actuador | Ventajas | Inconvenientes |
|---|---|---|
| Servomotor | Cálculo trivial de caudal (Q) | Mayor cantidad de dispositivos |
| | Cálculo trivial de altura (h) | Requiere programación |
| | Mayor facilidad para recambio | Significativamente más caro |
| | Pérdidas de carga despreciables | Boquilla de deposición ancha |
| Cámara presurizada | Sencillo de implementar | Mantenimiento muy deficiente |
| | Lógica muy básica | Dificultad para medir Q y h |
| | Solución más barata | Recambio más tedioso |
| | Boquilla de deposición estrecha | Pérdidas de carga a considerar |

Dicho esto, en la tabla 4.1 se mostrarán las ventajas y las contrapartidas de ambos actuadores.

En suma, aunque se vaya a emplear la cámara presurizada, si se consigue un servomotor más potente, se descartará la opción inicialmente escogida. Pues el desarrollo del programa de control en el autómata programable correspondiente ya se ha realizado, y es reutilizable.

# 5. Interacción Humano-Máquina

A estas alturas del informe, ya se conoce la planta sobre la que trabajar. Volviendo a la figura 1.2, ya se pueden identificar todos los elementos del sistema. Se tiene un brazo robótico que moverá una bandeja situada siempre bajo una boca de deposición de adhesivo, sobre la cual habrá una cámara presurizada encargada de garantizar un caudal de material. Pero una vez todo esto ya se ha decidido, es preciso hacerse la siguiente pregunta, ¿cómo se va a interactuar con el sistema? O aún más importante a este nivel de desarrollo, ¿de qué forma se va a extraer la información de lo que está sucediendo en el proceso de montaje?

Ambas cuestiones se van a solucionar en este capítulo. Pues el objetivo será diseñar unas librerías para poder leer datos de lo que está ocurriendo en tiempo real, y hacer las modificaciones pertinentes sobre este. Pero antes de empezar, se recordarán los potenciales dispositivos sujetos a la necesidad de enviar y/o recibir información.

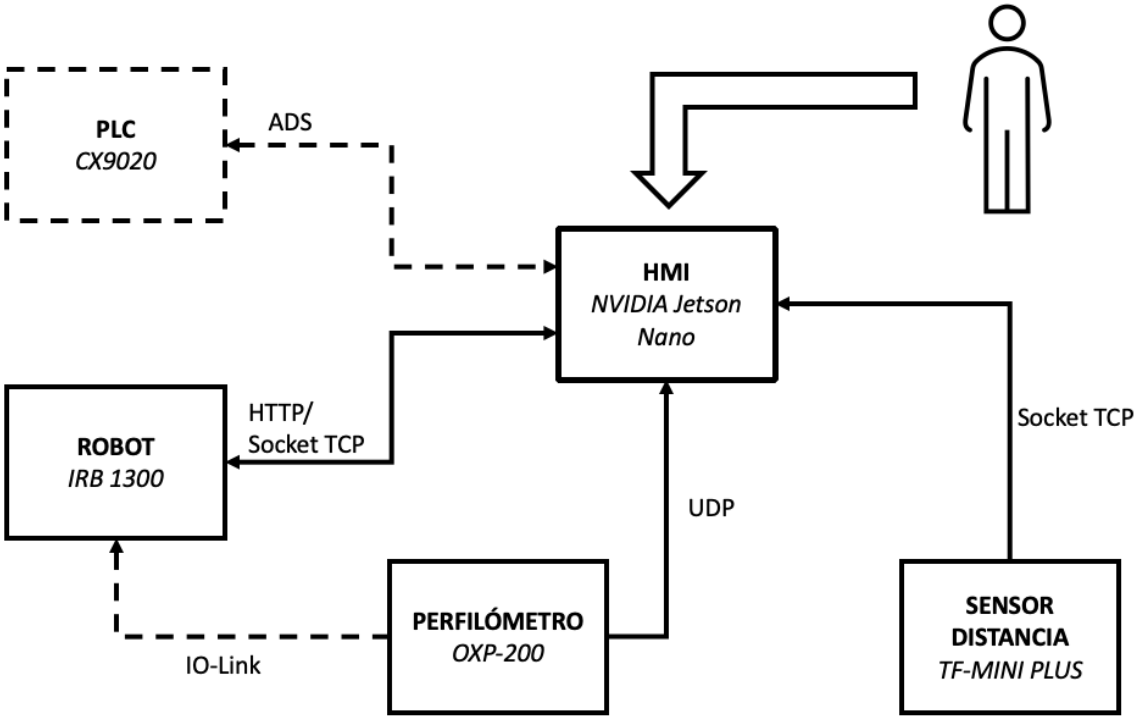



Figura 5.1: Esbozo ilustrativo de dispositivos y protocolos de comunicación presentes en el sistema

Si se observa la figura superior, se pueden identificar 4 dispositivos. El robot con su controladora, el sensor de tipo perfilómetro, del cual se hablará en el siguiente capítulo, el sensor de distancia para la estantería de secado, y un dispositivo llamado *NVIDIA Jetson Nano*. Siguiendo la analogía *maestro-esclavo*, en este sistema el maestro sería la controladora del robot, pues es la que coordina todo lo que está sucediendo en la etapa de montaje. Y el resto

de dispositivos son los esclavos. El objetivo de estos es enviarle información a la controladora sobre lo que está sucediendo en los puntos críticos del proceso, siendo la *NVIDIA Jetson Nano* el intermediario de esas comunicaciones.

Como ya se debe suponer, ese dispositivo será el que alojará el HMI, entre muchas otras tareas de cómputo que va a tener que realizar, como tratar la información del perfilómetro, la interpretación de la información enviada por el sensor de distancia, o las variaciones que quiera hacer el usuario sobre el modo de funcionamiento del sistema.

Para poder realizar dichas tareas es necesario crear unas librerías a medida para lo que se quiera comunicar. A lo largo de este capítulo se irán comentando las soluciones empleadas para conseguir ese fin[1]. Por último, cabe mencionar que en la figura 5.1, las líneas discontinuas representan protocolos de comunicación que no entrarán en el ámbito de esta memoria.

## 5.1.- Comunicaciones con el robot

Tal y como se ha comentado en la introducción a este capítulo, la controladora del robot será el dispositivo maestro de la instalación, pues es la encargada de tomar las decisiones más relevantes del montaje. A la hora de diseñar el esquema de comunicaciones de la célula se ha decidido emplear una estructura en la que todos los dispositivos conectados le envíen diagnósticos de las etapas más críticas, y la controladora adopte decisiones en base a los datos recibidos.

Por el otro lado, para poder supervisar el estado del proceso, se necesita poder acceder a la memoria del robot. Entonces, las comunicaciones serán bidireccionales, es decir, se deben poder realizar operaciones de lectura y escritura.

| **Robot** |
| --- |
| - direccionIP : string<br>- puerto : int |
| + robot() : none<br>+ leerInformacion(variable : string) : string<br>+ enviarInformacion(variable : string, valor : string) : none |

Figura 5.2: Diagrama de clase *robot*

[1]Toda la programación desarrollada de ahora en adelante se hará en el lenguaje *Python* siempre que no se indique lo contrario.

### 5.1.1.- Operación de lectura

Una de las herramientas que proporciona *ABB* para interactuar con el robot a distancia sin uso del *Flex Pendant* es mediante los *ABB Robot Web Services*. Estos son una plataforma que permite a los desarrolladores crear sus propias aplicaciones a medida para interactuar con la controladora del robot. Para ese fin, se emplea una *API REST*, la cual es un conjunto de reglas que definen cómo las aplicaciones o los dispositivos pueden conectarse y comunicarse entre sí, cumpliendo en este caso con los principios de diseño del estilo de arquitectura de *transferencia de estado representacional*.

Estos servicios dotan al programador de una serie de instrucciones a realizar para poder interactuar con el sistema robotizado. En este caso interesará sólo el *GET*. Este se encarga de devolver la información recogida en una dirección dada por el usuario. La respuesta devuelta por el sistema se dará en un XML o en un JSON, a petición del desarrollador.

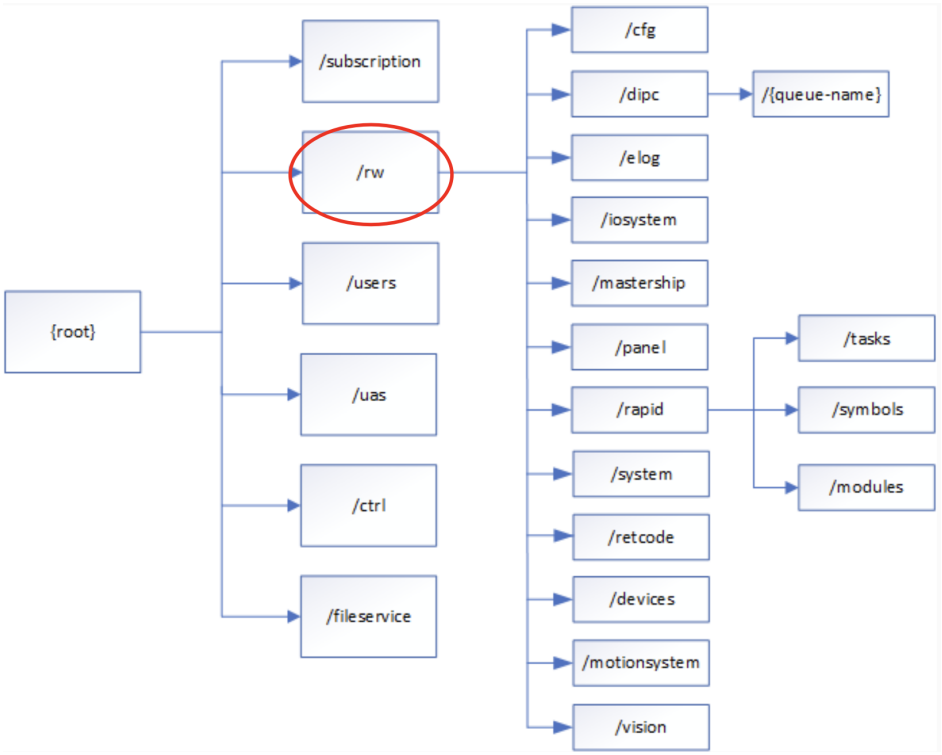



Figura 5.3: Esquema de servicios disponibles en *ABB Robot Web Services* [3]

De todos los servicios disponibles, vistos en la figura superior, sólo interesará llevar a cabo 2 tareas. La lectura de variables durante la ejecución del programa de control, y la supervisión de la velocidad del brazo robótico. Ambas tareas se pueden hacer si se accede al servicio *rw*. Para ello hay que seguir el esquema hasta llegar a la tarea donde se aloja la variable, resultando en la url mostrada a continuación. A su vez, hay que definir unas cabeceras en las que se puede configurar el formato de la respuesta, entre otras cosas. Y por

último, hay que configurar la autenticación para poder acceder al servicio.

```
url = "https://<direccionIP>:443/rw/rapid/symbol/RAPID/<IDTarea>/<NombreVariable>/data" # Direccion de la variable a leer

headers = {"accept": "application/hal+json;v=2.0"} # Formato de respuesta en JSON

auth = HTTPBasicAuth("<NombreUsuario>","<Password>") # Credenciales de acceso a la controladora
```

Una vez estos parámetros han sido definidos, hay que ejecutar la petición de información mediante la siguiente instrucción.

```
req = requests.get(url, headers=headers, auth=auth,verify=False)
```

En la variable *req* se guardará el resultado de la respuesta. En caso de que no hayan surgido errores, se generará un archivo de respuesta con el valor de la variable de la que se pidió la información. Si por el contrario hubo errores de sintaxis, timeout, permisos, etc. Se devolverá un archivo de respuesta con el código de error correspondiente.

```
{
"_links": {
    "base": {
        "href": "https://x.x.x.x:443/rw/rapid/"
    },
    "self": {
        "href": "symbol/RAPID/T_ROB1/etapaInyeccion/data"
    }
},
"state": [
    {
        "_type": "rap-data",
        "_title": "RAPID/T_ROB1/etapaInyeccion",
        "value": "FALSE"
    },
    {
        "pgmname_ret2": {
            "_links": {
                "error": {
                    "href": "https://x.x.x.x:443/rw/retcode?code=-1073414145"
                }
            }
        }
    }
```

```
25      ]
26  }
```

Fijándose en el ejemplo de respuesta superior, en específico en la línea 14. Se puede ver que la variable *etapaInyeccion* está en estado bajo (*FALSE*).

El objetivo será implementar este procedimiento en el método *leerInformacion* de la clase mostrada en la figura 5.2 de forma que dándole el nombre de la variable en cuestión, devuelva directamente el valor tras procesar el archivo de respuesta.

Dado que en esta memoria no se pretende profundizar en la explicación de esta tecnología, se recomienda al lector revisar las entradas bibliográficas relacionadas con este tema para tener una explicación más detallada. [3] [15]

### 5.1.2.- Operación de escritura

Al igual que con las operaciones de lectura, es posible actualizar los valores de las variables mediante procedimientos POST. Pero debido a que para ello es necesario tener una licencia, se decidió bajar a la capa de red implementando un socket TCP.

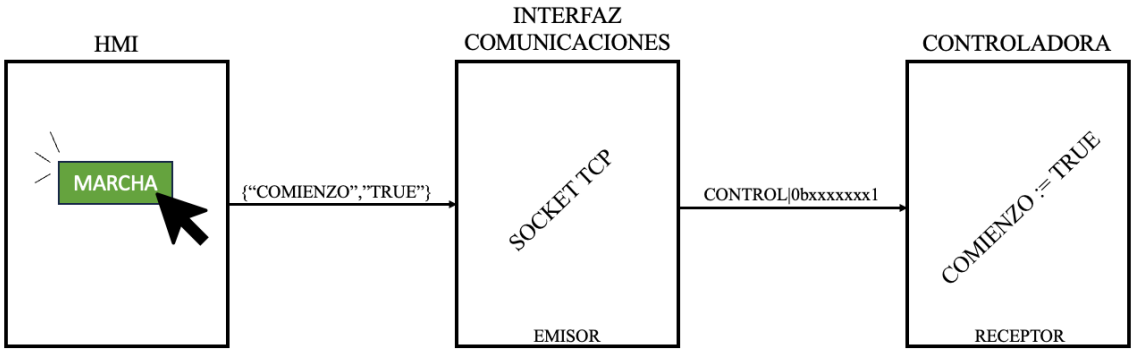



Figura 5.4: Flujo de operaciones necesarias para escribir sobre una variable del robot

En la ilustración superior se muestra una secuencia de sucesos que va a ser muy frecuente, la puesta en marcha del sistema. Hay que tener en cuenta que a la hora de bajar a la capa de red para implementar las operaciones de escritura, hay que desarrollar un protocolo para interpretar la información, pues las tramas de información enviadas y recibidas por los sockets son secuencias de bytes. Teniendo eso en cuenta, se ha desarrollado un método para clasificar la información en función de su propósito en *palabras*.

- *CONTROL*. Contendrá las variables booleanas que permiten la transición entre estados de funcionamiento del robot. Debido a que son 6 variables en total, se ha decidido alojarlas todas en un byte, siendo cada una de ellas un bit específico.

**Tabla 5.1** Desglose de palabra de comunicación *CONTROL*

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| x | x | Origen | Seguir | pcmanual | pcautomatico | pcstop | COMIENZO |

Para entender los nombres de los bits de la tabla superior, revise los diagramas 4.16 y 4.18.

- *ETAPA*. Contendrá la etapa a cargar en modo manual.
- *VELOCIDAD*. Se empleará para cargar la nueva velocidad del brazo robótico en la etapa de inyección.

Para poder integrar todas las variables booleanas en un byte, se ha decidido crear una clase en *Python* que simula los campos de bits del lenguaje *C*.

Volviendo a la figura 5.4, cuando se pulsa el botón *MARCHA*, el callback de la interfaz humano-máquina envía al objeto encargado de gestionar las comunicaciones el nombre de la variable a editar, y su nuevo valor, tal y como se especifica en el diagrama 5.2. Llegado a ese punto, lo que debe hacer el método de la clase es clasificar la variable en función de la palabra a la que pertenece, en este caso, a la palabra *CONTROL*. Ayudándose de la clase campo de bits, traduciría el string generado por el callback de la interfaz a una trama que debe tener el siguiente aspecto.

```
1    tramaSocket = "<palabra>|<valor>"
```

En el ejemplo que se está usando para la explicación, el contenido de la trama será *CONTROL*|*0bxxxxxxx1*. Es importante mencionar que en el caso de la palabra *CONTROL*, el byte se debe actualizar bit a bit, para evitar que se sobreescriban variables no deseadas.

Finalmente, el cliente enviará la información a la controladora del robot, que tendrá un puerto abierto y estará permanentemente escuchando en una tarea independiente de la principal. Una vez reciba la información, aplicará el proceso inverso para decodificar la trama y actualizar los valores de las variables pertinentes.

Cabe mencionar que la gran ventaja que tiene este sistema es que se puede restringir el conjunto de variables a editar de una forma muy sencilla.

## 5.2.- Interfaz Humano-Máquina

Con el fin de realizar pequeñas modificaciones sobre el proceso durante las pruebas, se ha diseñado una interfaz de operador básica para poder operar con el prototipo de manera más sencilla.

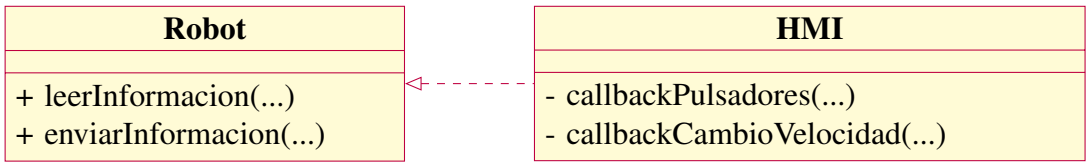



Figura 5.5: Diagrama de clases del HMI

Como se puede ver en el diagrama superior, el HMI implementa la clase robot, pues cada interacción que se haga con uno de sus elementos supondrá una variación en las variables de control establecidas en la controladora del robot.

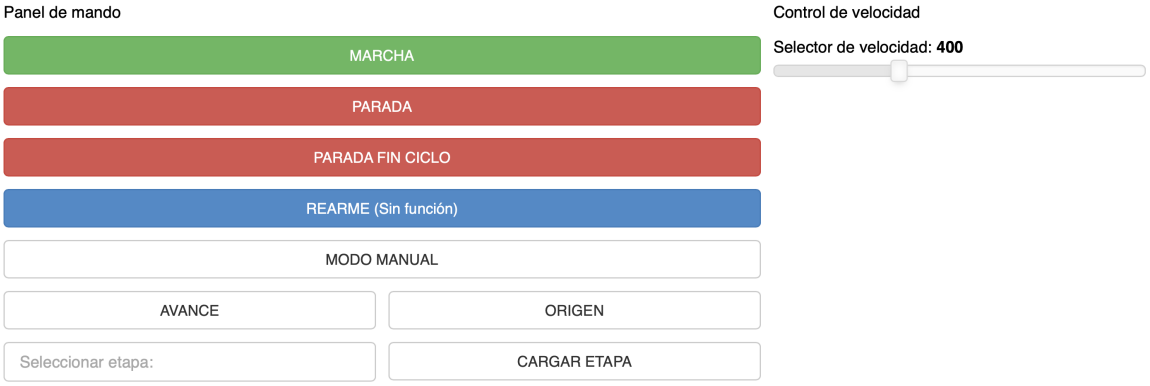



Figura 5.6: Interfaz de operador diseñada para etapa de desarrollo

En cuanto a diseño se refiere, debido a la simplicidad de la aplicación, se ha decidido usar la librería *Bokeh* de *Python*. Y tal y como se muestra en la figura superior, se pueden hacer las siguientes acciones.

- Poner en marcha al robot mientras la controladora está en modo automático (No confundir el modo automático de la controladora con el del programa de control).
- Parada instantánea o a fin de ciclo del brazo.
- Selección de modo de funcionamiento y operaciones exclusivas del modo manual.
- Selección de velocidad para la etapa de inyección de adhesivo.

# 6. Supervisión y control de la etapa de deposición de adhesivo

Este capítulo se enfocará en un componente fundamental de este sistema: el sensor. Pues juega un papel crucial al proporcionar datos precisos que alimentan los algoritmos de IA, permitiendo así la toma de decisiones inteligentes y adaptativas. Es a través de la sensorización que se logra capturar información detallada sobre el comportamiento del adhesivo en el momento de su deposición.

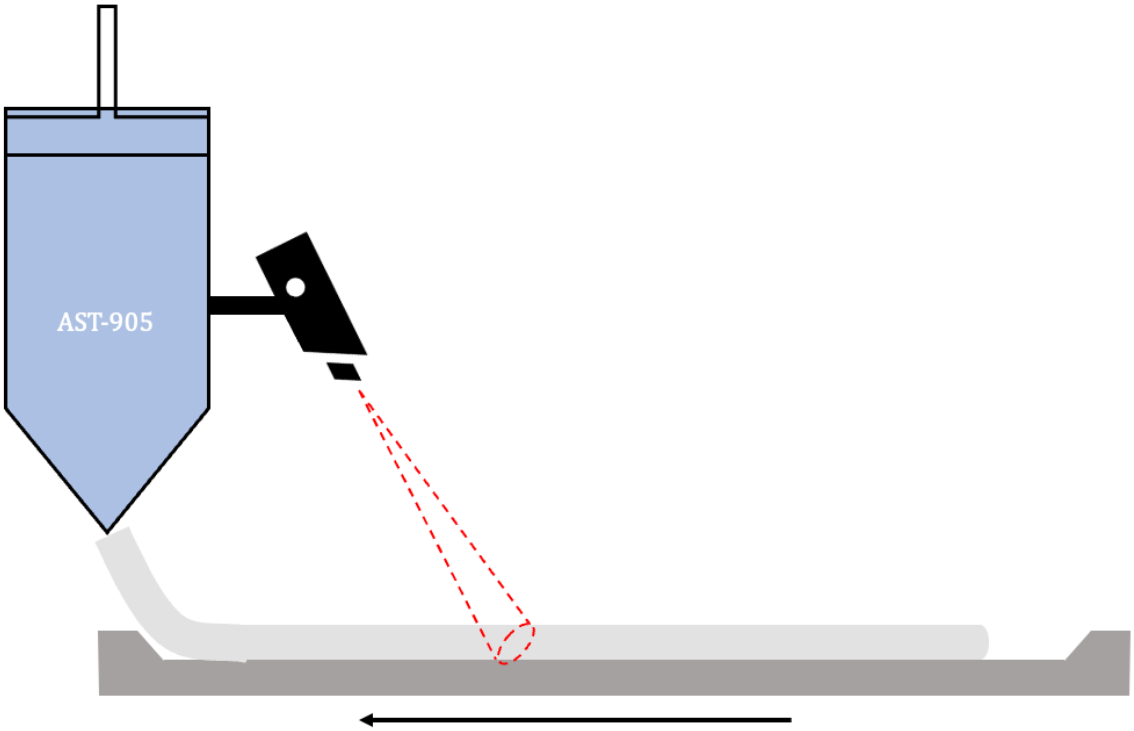



Figura 6.1: Esbozo ilustrativo de la disposición del sensor para la medición de perfiles

Tal y como se puede ver en la imagen superior, el sensor estará fijado a la misma estructura que el sistema de inyección de adhesivo, y la bandeja se moverá de forma que el sensor pueda medir el adhesivo que va dispensando la válvula de deposición, en caso contrario el sensor no sería capaz de medir nunca. Partiendo de esa premisa, para poder entender con precisión el proceso a seguir para minimizar el desperdicio de material, se tratarán principalmente dos temas:

- Parámetros constructivos del sensor a utilizar.
- Diseño del algoritmo basado en técnicas de inteligencia artificial para la optimización de la deposición del material.

## 6.1.- Sensorización del proceso

### 6.1.1.- Sensor empleado

Para llevar a cabo la medición de las propiedades del hilo de adhesivo se ha decido instalar un sensor de tipo perfilómetro, en específico el *OXP200-R05C.004* de *Baumer*. Este dispositivo permite medir perfiles de una superficie y devuelve una nube calibrada en 2 dimensiones preparada para procesamiento externo de señal.

Para su uso dispone de dos terminales, un conector eléctrico macho de 12 pines que tiene tanto señales de alimentación como de comunicación para el uso de buses de campo como *IO-Link*. Y dispone de un segundo conector para *ethernet*, el cual es hembra y de 8 pines, que permite llevar las comunicaciones e incluso alimentar el sensor mediante PoE[1].

Figura 6.2: Conectores del sensor [4]

Para la etapa en la que se encuentra el proyecto, se descarta el uso del conector eléctrico, pues se va a emplear un switch con capacidad de distribuir la alimentación.

A nivel operativo, es conveniente tener en cuenta ciertos parámetros de uso del sensor. Este tiene una frecuencia de muestreo comprendida entre 200 y 800 Hz, y cada nube de puntos tiene **aproximadamente** 580 valores. Es preciso remarcar que los puntos de cada perfil **no están equiespaciados** entre sí, lo cual afecta directamente a la etapa de extracción de características, que se comentará más adelante.

[1]Power over Ethernet (PoE) es una tecnología que permite la transmisión de datos y alimentación a través del mismo cable Ethernet

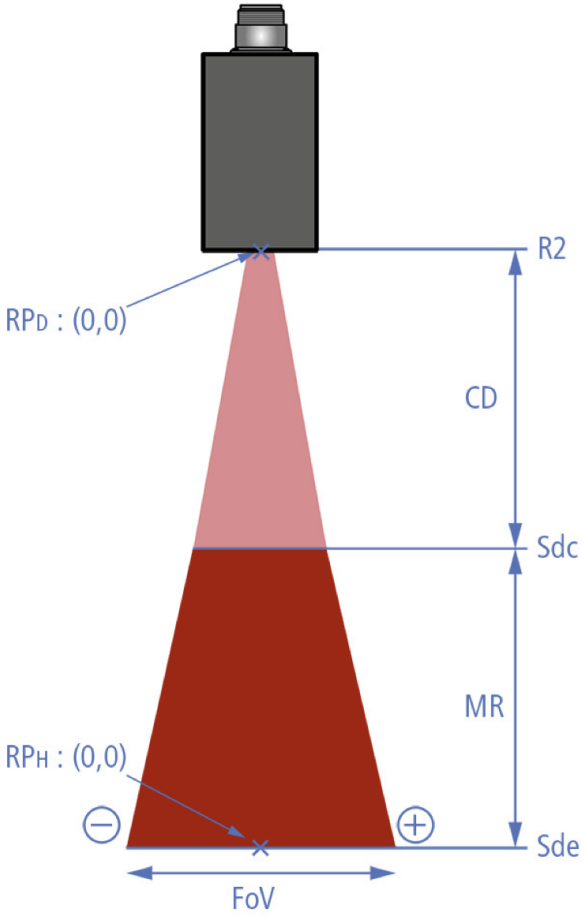



Figura 6.3: Campo de visión del sensor [4]

Tal y como se puede ver en la figura superior, existen varias zonas de interés a la hora de usar el sensor, las cuales definirán a posteriori la localización de este a la hora de efectuar estas mediciones. La región de color rojo claro (CD) se define como la zona ciega, tiene una longitud de 100 mm y si se pone un objeto delante, el sensor no será capaz de obtener información. Por el contrario, la zona rojo oscuro (MR), es la zona de medida, ubicada a una distancia de 100 a 150 mm de la lente del sensor. Para poder leer los perfiles es necesario que la zona a medir se encuentre en ese área. Por último, aparece otro concepto llamado campo de visión (FoV), el cual varía entre 48 y 72 mm en función de la distancia a la que se encuentre de la lente. Debido a que el ancho de la pista de adhesivo es de 25 mm, cualquier nivel de la zona de medida es válido.

A su vez, el propio sensor tiene un servicio web integrado que permite acceder a un portal de configuración desde el cual se pueden cambiar ciertos parámetros del dispositivo, como su IP, máscara de subred, frecuencia de muestreo, salidas digitales, alarmas, etc.

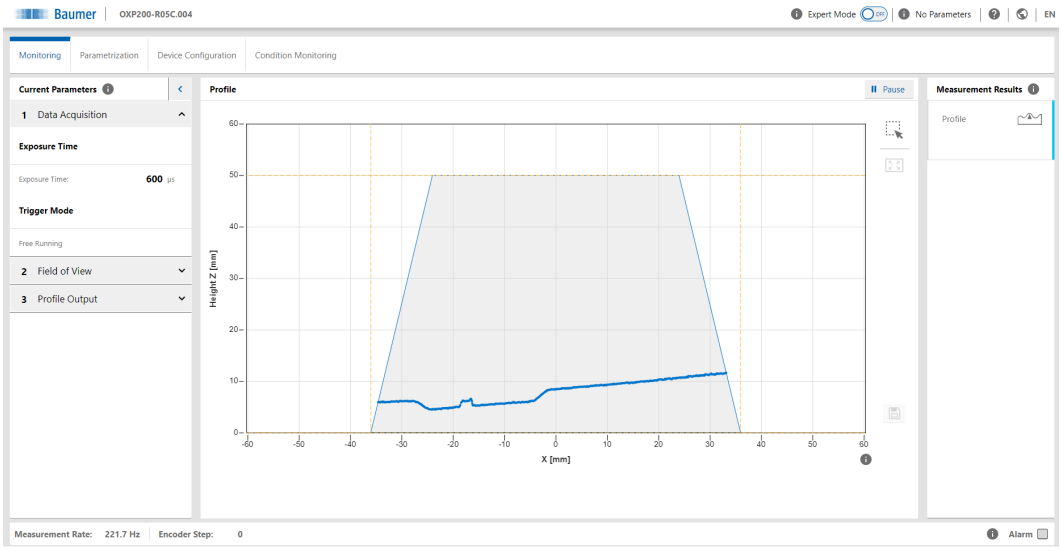



Figura 6.4: Portal de configuración del sensor

Nótese que el área sombreada de la figura superior se corresponde con la zona de medida del sensor. Para interactuar con este, el fabricante ofrece unas librerías propias que facilitarán la obtención de la información en diversos lenguajes (C/C++, C# y Python).

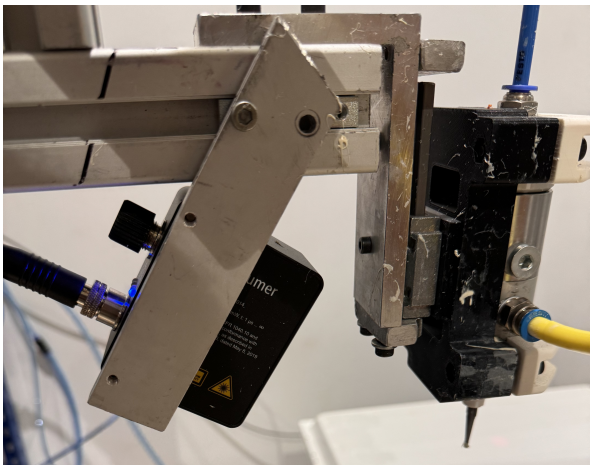


Figura 6.5: Montaje del sensor en prototipo célula de montaje

Como se puede ver en la imagen superior, el sensor se colocó detrás de la boca de deposición con un ligero ángulo de inclinación para poder medir correctamente. Esta medida se ha tomado para asegurar la integridad del dispositivo ante posibles golpes que se puedan producir debido al entorno industrial.

### 6.1.2.- Programa de adquisición de datos

Para obtener las medidas de perfilometría del sensor el fabricante ofrece una librería que facilita la obtención de la información mediante el protocolo UDP. Como se mencionó antes, permite ser implementada en varios lenguajes como C/C++, C# y Python. Debido a que todo el proyecto está siendo implementado en este último lenguaje, seguirá siendo utilizado por motivos de homogeneidad en el código.

A su vez, existen 2 modos de obtener la información. En el primero el usuario pide un perfil mediante un método de la clase y recibe, entre otros valores, dos arrays con los datos medidos por el sensor. Uno representando a los datos del eje x y otro a los del eje z. El segundo método consiste en que existe una clase derivada de la del sensor que permite tratar las mediciones mediante un buffer, transformando la aplicación a desarrollar en el clásico problema del productor y el consumidor.

Teniendo en cuenta que el lenguaje que se está utilizando no cumple con las condiciones necesarias para ser de tiempo real, la única opción plausible se reduce a usar el buffer del sensor. Pues este se puede configurar para que introduzca un perfil cada cierto tiempo en la pila de almacenamiento.

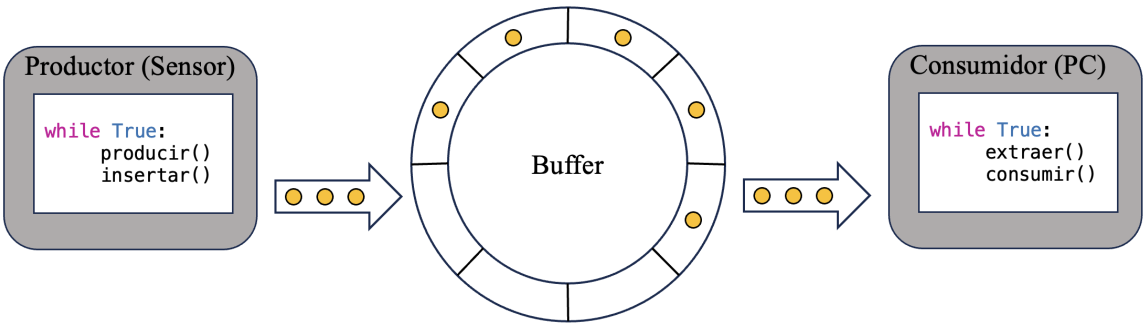



Figura 6.6: Esbozo ilustrativo del algoritmo a emplear para la adquisición de datos

El programa a implementar para **la recogida de datos de entrenamiento** será muy sencillo. Pues mediante la librería implementada en la figura 5.2 se podrá saber en todo momento cuando se está dispensando pegamento. De esta forma se le pedirá al sensor que empiece a introducir perfiles muestreados en el buffer, y nada más termine de dispensar el material de la última pista se dejará de producir información. A partir de ese momento se abandona la sección crítica del programa ya que el tiempo deja de ser algo a tener en cuenta. Llegado ese punto se recolectará toda la información vaciando el buffer, se formateará, y se enviará a las etapas de extracción de características para el posterior entrenamiento de la inteligencia artificial.

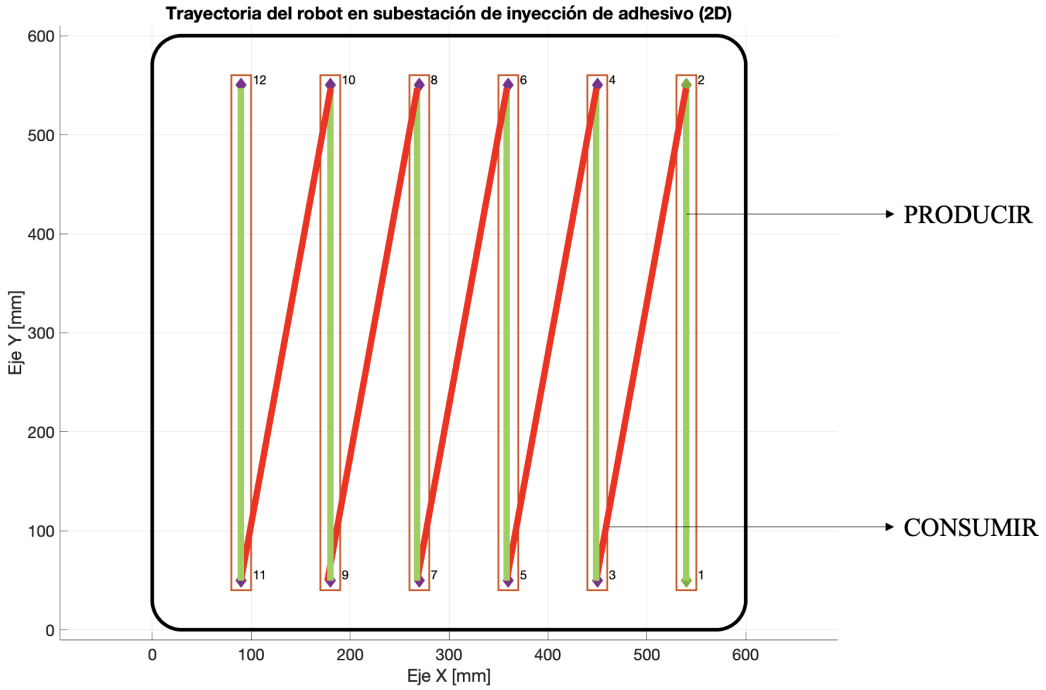



Figura 6.7: Etapas de generación y consumo de datos durante etapa de deposición de adhesivo

## 6.2.- Diseño de algoritmo de optimización de dosificación de adhesivo

El principal objetivo de este algoritmo es identificar la calidad de los hilos de adhesivo dispensados por la válvula, emitir un veredicto que defina cómo de óptimo es, y en función de este hacer que el brazo robótico responda en consecuencia. Para cumplir ese fin se va a recurrir a modelos basados en el aprendizaje con datos (*Data-Driven Learning Models*), en específico, a modelos de aprendizaje automático (*Machine Learning*). Esto se debe a que, como se comentó en el primer capítulo de la memoria, no se dispone de medios suficientes para elaborar un modelo dinámico fiable del sistema basado en ecuaciones diferenciales. Por ello, mediante el uso de estas técnicas se puede emplear un enfoque experimental *(data-driven)*, usando la estadística en lugar de los modelos físicos para analizar los patrones de comportamiento del fluido bajo las condiciones propuestas. En suma, el aprendizaje automático permitirá enseñar a una máquina a hacer una tarea que debido a la falta de información por parte del fabricante no se sabe programar.

Para el diseño de este algoritmo se pueden emplear varias filosofías de trabajo, y distintas herramientas. En este proyecto se optará por seguir el planteamiento que ofrecen los sistemas basados en el aprendizaje automático.

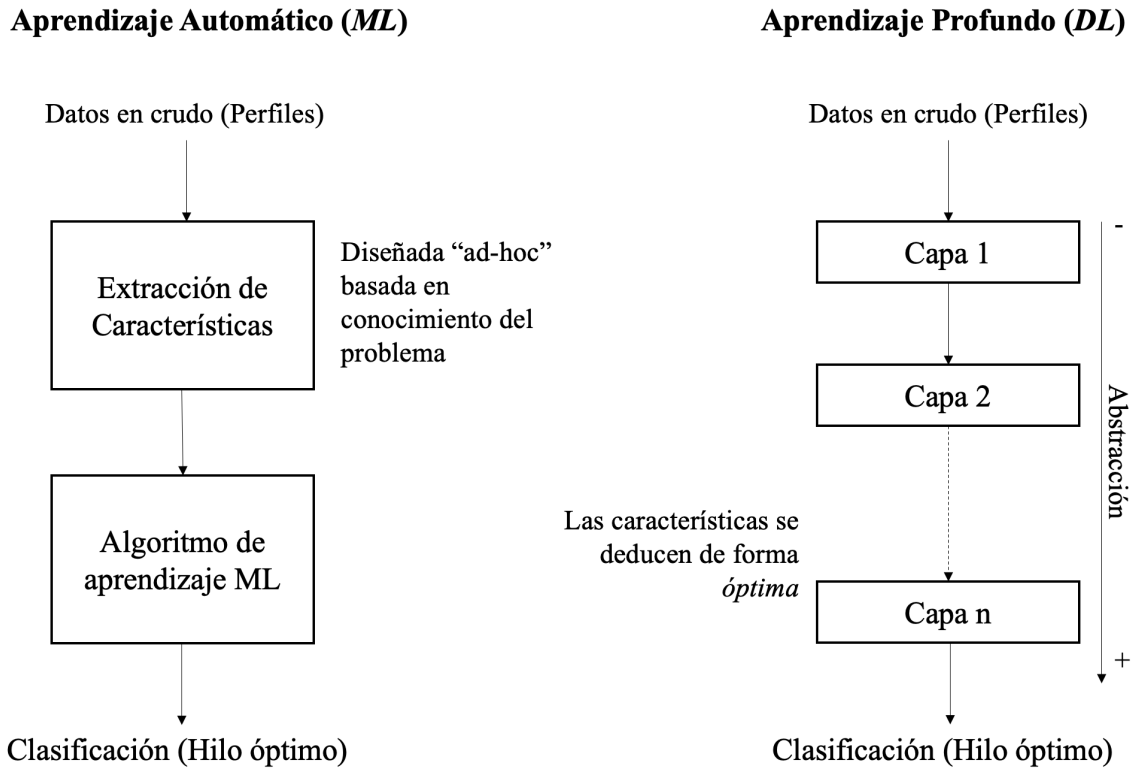



Figura 6.8: Tratamiento de información en sistemas de Aprendizaje Automático y Aprendizaje Profundo[5]

En la actualidad, a la hora de resolver problemas de aprendizaje automático se encuentran dos opciones a la hora del diseño del modelo de inteligencia artificial, y la principal diferencia que se ve entre ambos radica en la programación de la etapa de extracción de características. Esta se define como un programa que hace un pre-tratamiento de los datos con el fin de amenizar la labor de entrenamiento y predicción del modelo de ML aprovechando el conocimiento que posee el ingeniero del sistema.

Pero, si se ha remarcado que se va a seguir un planteamiento basado en aprendizaje automático, ¿por qué hablar del aprendizaje profundo? O mejor dicho, ¿por qué usar redes neuronales en un proyecto de *machine learning*? La respuesta a estas cuestiones es sencilla, pues la mayoría de las librerías usadas en estos ámbitos emplean por dentro redes neuronales para obtener mejores resultados. Entonces, lo que se va a obtener es un modelo que calculará características óptimas a partir de la información que el programador quiera que use para aprender deducida a partir de los datos en crudo.

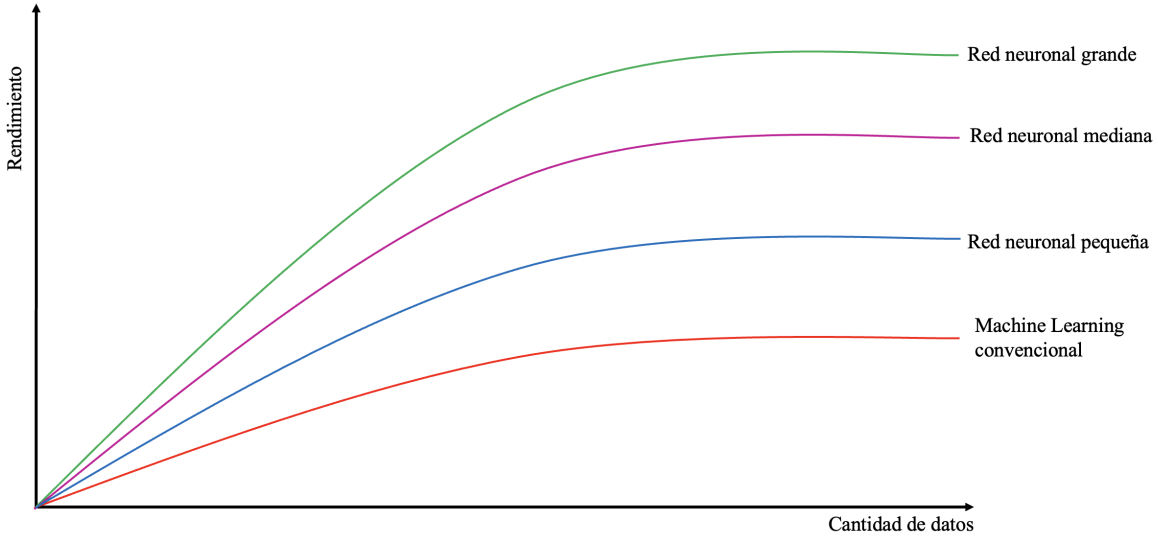



Figura 6.9: Rendimiento de los algoritmos de IA en función de la cantidad de datos usada

A lo largo de la historia de estos algoritmos se ha demostrado que a mayor número de capas y neuronas, y mayor cantidad de datos se manejen, serán capaces de identificar patrones con mucho mayor grado de complejidad. Esto es debido a un fenómeno denominado **comportamiento emergente**, el cual se puede definir como los patrones, estructuras o funciones espontáneos e impredecibles que surgen en sistemas complejos debido a las interacciones entre sus componentes más simples. Es un fenómeno donde el todo es mayor que la suma de sus partes, que se caracteriza por el comportamiento no lineal, complejo y auto-organizado de las neuronas [16]. Este tipo de comportamientos se pueden ver análogamente en las bandadas de pájaros o en los bancos de peces al intentar defenderse de un depredador o hacer

movimientos migratorios.

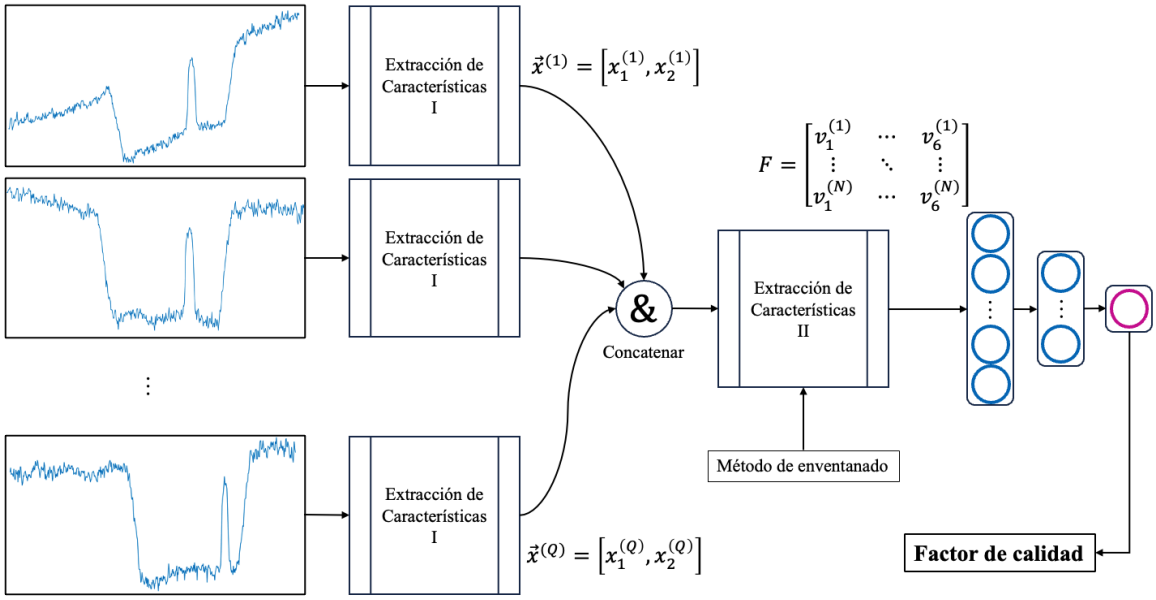



Figura 6.10: Esquema de etapas del sistema de optimización de la dosificación de adhesivo

Si se revisa la figura 6.10 se podrá ver un esquema más detallado del procedimiento para diseñar el algoritmo de aprendizaje automático. A la izquierda del todo se pueden ver ejemplos de medidas efectuadas por el sensor, a continuación esos perfiles serían pre-procesados en la etapa de extracción de características, y finalmente pasarían por una red neuronal que dará como resultado el veredicto del que se ha hablado al principio de la sección.

Cabe mencionar que en este proyecto hay 2 etapas de extracción de características. Esto se debe a que el sensor devuelve un perfil en 2 dimensiones, del cual se pueden extraer medidas de posición o área, pero para poder llegar al concepto de volumen, es necesario concatenar varios perfiles en una matriz, y hacer otra extracción de características que sí podrá efectuar medidas volumétricas, como se verá más adelante. Es por ello que el desarrollo técnico de esta sección se dividirá en tres apartados:

1. Extracción de características superficial (2 dimensiones)
2. Extracción de características volumétrica (3 dimensiones)
3. Diseño de red neuronal para clasificar hilos de adhesivo

### 6.2.1.- Etapa de extracción de características superficial

Tal y como se comentó en la introducción a esta sección, las etapas de extracción de características son secuencias de operaciones matemáticas efectuadas vía métodos numéricos para obtener información relevante de la nube de puntos a tratar. Es por ello que en esta primera etapa de extracción de características se tratará de calcular la sección del hilo de adhesivo, y la posición de este sobre la pista. Pero para poder calcular ambos parámetros es necesario poder ubicar el perfil de pegamento sobre la pista.

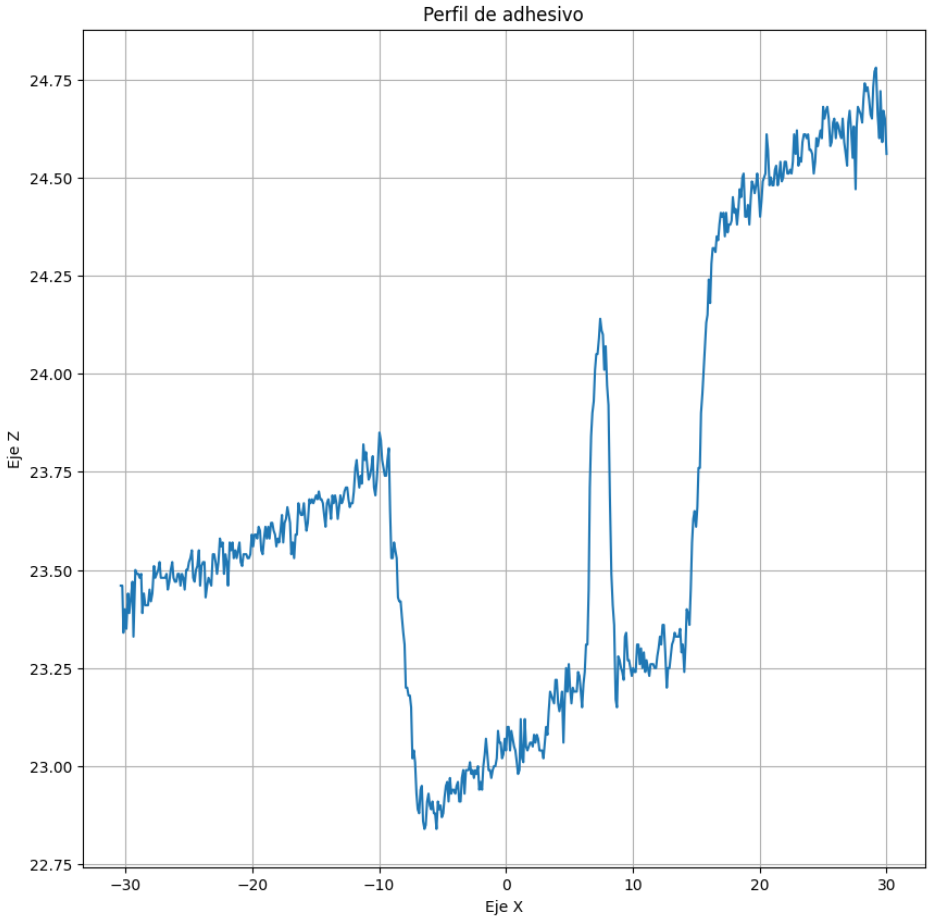



Figura 6.11: Perfil en crudo devuelto por el sensor

Lo que se ve en la figura superior es un perfil que perfectamente puede leer el sensor en una operación normal de inyección de adhesivo. Sobre este se puede ver a simple vista dónde se ubica el perfil de adhesivo (el saliente en medio del valle), y el inicio y fin de la pista.

#### 6.2.1.1.- Acondicionamiento y filtrado de la señal

Para poder trabajar con mayor comodidad sobre esta nube de puntos sería conveniente eliminar los offsets en ambas componentes para tratar la información correctamente.

$$\vec{x} = \begin{bmatrix} x_1 - \bar{x} \\ x_2 - \bar{x} \\ \vdots \\ x_n - \bar{x} \end{bmatrix} \qquad \vec{z} = \begin{bmatrix} z_1 - \text{mín}\, z \\ z_2 - \text{mín}\, z \\ \vdots \\ z_n - \text{mín}\, z \end{bmatrix} \tag{6.1}$$

A su vez es conveniente filtrarla para reducir el ruido presente en toda la figura. Es por ello que se utilizará un filtro de media móvil con una ventana de trabajo de 5 valores.

$$\bar{z}[n] = \frac{1}{5} \sum_{k=n-2}^{n+2} z[k] \tag{6.2}$$

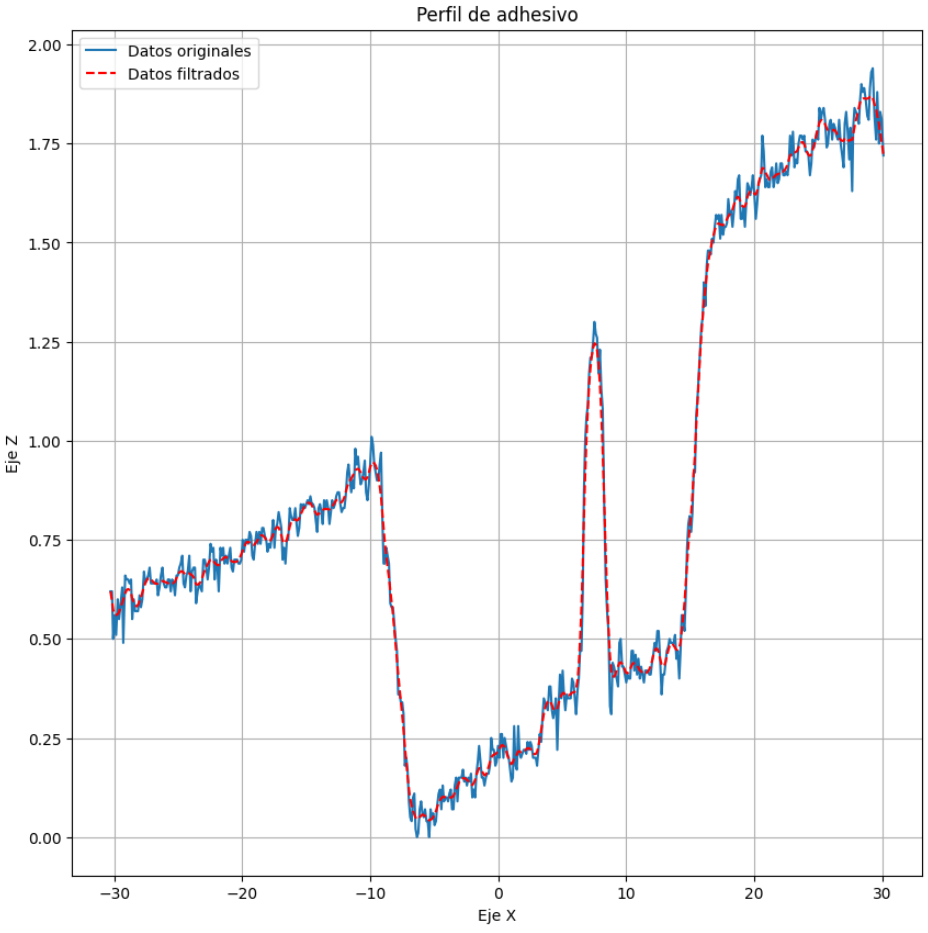



Figura 6.12: Perfil acondicionado y filtrado

#### 6.2.1.2.- Detección de hilo mediante derivación numérica

Si se revisa el perfil a tratar, uno de los aspectos a notar es que existen zonas en las que la pendiente aumenta o disminuye drásticamente. Eso se traduce en que si se calcula la derivada de la señal, se pueden identificar los puntos de interés, y con ellos, ubicar el hilo de adhesivo sobre la pista.

Para ello se deberán emplear métodos de derivación numérica. Como se mencionó antes hablando del sensor, la información deducida empíricamente, o en otras palabras, los datos obtenidos de los experimentos, suelen estar organizados de forma no equiespaciada. El control del equiespaciado de los puntos normalmente sólo suele poder hacerse cuando se puede usar una función que genere una tabla de valores.

Un forma de lidiar con la información empírica de estas características es mediante el uso de un polinomio de interpolación de Lagrange de segundo orden. Ya que este no requiere que todos los puntos estén equiespaciados. Este se puede definir como una simple reformulación del polinomio de Newton que evita el cálculo de las diferencias divididas. Y se representa como [17]

$$f_n(x) = \sum_{i=0}^{n} f(x_i) \cdot \ell_i(x) \tag{6.3}$$

Donde

$$\ell_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j} \tag{6.4}$$

Si se calcula el polinomio de orden 2 se obtendrá lo siguiente

$$f_2(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} f(x_0) + \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} f(x_1) + \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} f(x_2) \tag{6.5}$$

Y si finalmente se deriva analíticamente con respecto a x, se obtendrá la expresión a utilizar para derivar correctamente la nube de puntos.

$$\begin{aligned} \frac{\partial}{\partial x} f(x) \approx & f(x_{i-1}) \frac{2x - x_i - x_{i+1}}{(x_{i-1} - x_i)(x_{i-1} - x_{i+1})} \\ & + f(x_i) \frac{2x - x_{i-1} - x_{i+1}}{(x_i - x_{i-1})(x_i - x_{i+1})} \\ & + f(x_{i+1}) \frac{2x - x_i - x_{i-1}}{(x_{i+1} - x_i)(x_{i+1} - x_{i-1})} \end{aligned} \tag{6.6}$$

Donde x es el valor donde se quiere estimar la derivada. A pesar de que esta expresión pueda parecer más complicada que las más usuales de primer orden, tiene ciertas ventajas:[17]

- Puede ser usada para derivar en cualquier punto dentro de un rango prescrito por tres puntos.
- Los puntos en si mismos no tienen que estar equiespaciados.
- La estimación de la derivada tiene la misma precisión que la derivada central de primer orden.

Entonces, si se emplea la expresión 6.6 para derivar el perfil se obtendrá lo siguiente.

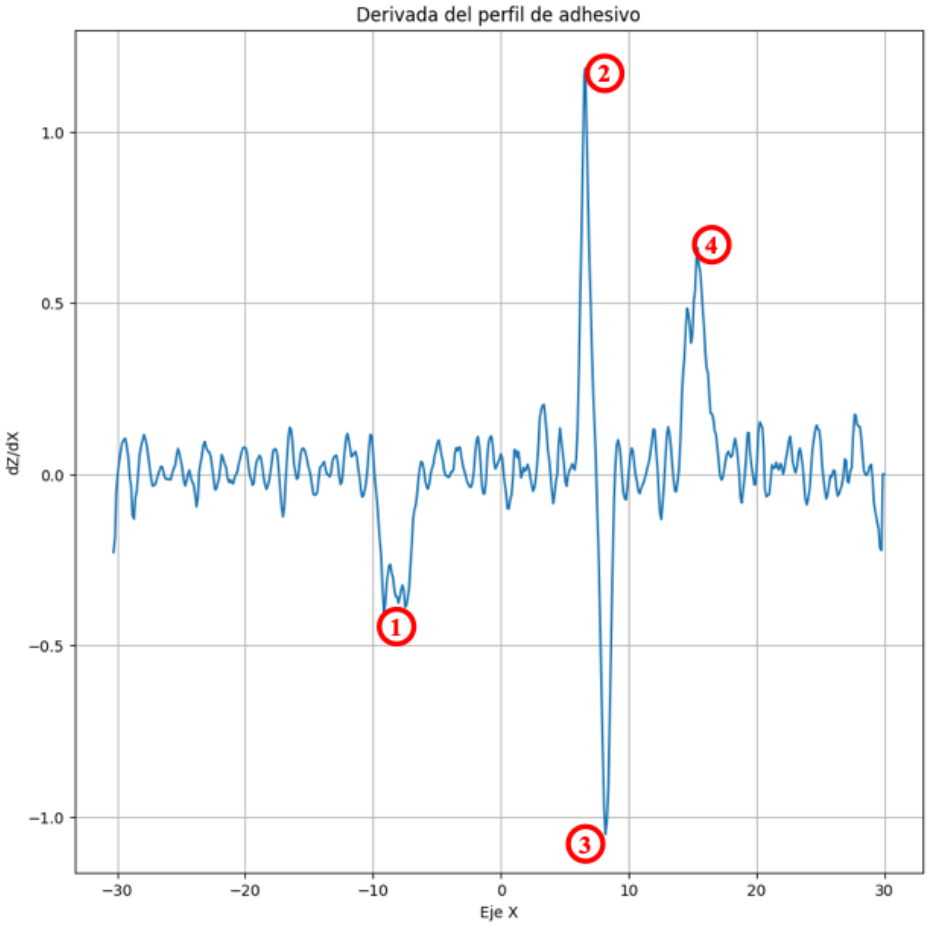



Figura 6.13: Derivada del perfil

Revisando la figura superior, es posible identificar los puntos críticos que se mencionaron en el paso anterior. Siendo el primer punto el inicio de la pista, el segundo el inicio del hilo de adhesivo, el tercero el fin del hilo de adhesivo y el cuarto el fin de la pista. Como la tarea que se va a hacer es la búsqueda de máximos y mínimos sobre la derivada del perfil, se puede decir que la búsqueda del hilo de adhesivo se hace mediante **detección de cambios de curvatura** en el perfil.

La tarea a realizar en sí misma aparenta ser muy sencilla, pero en el mundo de la computación la búsqueda de máximos y mínimos en conjuntos de datos siempre ha sido un reto. El problema que se produce con el tipo de datos que se están manejando es que existen *armónicos* que generan ruido y hacen que los algoritmos de búsqueda cometan errores con mayor facilidad. Es por ello que para facilitarles la labor, se va a emplear un filtro espectral basado en la transformada rápida de Fourier para literalmente truncar el espectro de la señal y eliminar armónicos, dejando de esta forma una señal mucho más agradable de tratar.

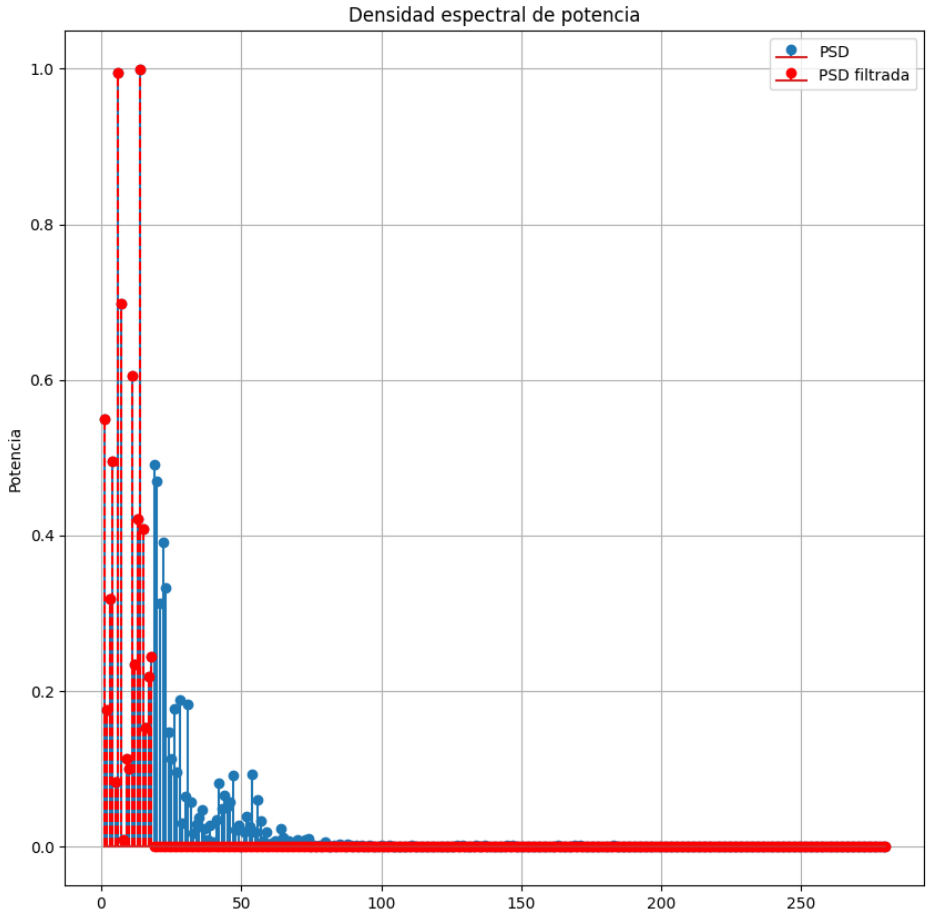



Figura 6.14: FFT sobre la derivada del perfil

Este algoritmo es muy relevante en la ingeniería, y es una versión de la Transformada Discreta de Fourier, el cual descompone una señal periódica en una suma de senoides discretas, se puede definir como

$$Y_n = \sum_{k=0}^{N-1} y_k e^{-jn\theta_0 k} \Longleftrightarrow y_k = \frac{1}{N} \sum_{n=0}^{N-1} Y_n e^{jn\theta_0 k} \tag{6.7}$$

Donde N es el número de muestras en un periodo y $\theta_0 = 2\pi/N$ es la frecuencia fundamental normalizada.

Debido a la gran eficiencia de la FFT sobre la DFT, su aplicación se ha generalizado a cualquier tipo de tareas de análisis de armónicos, e incluso se puede usar para lidiar con señales que ni siquiera son periódicas. Cabe mencionar que la señal que se está tratando no es periódica, pero el objetivo no es descomponerla en senoides, lo que se pretende es eliminar armónicos para reducir el ruido. De esta forma se puede facilitar la tarea de los algoritmos de búsqueda de extremos relativos, ya que con este filtro se verá la misma señal "suavizada".

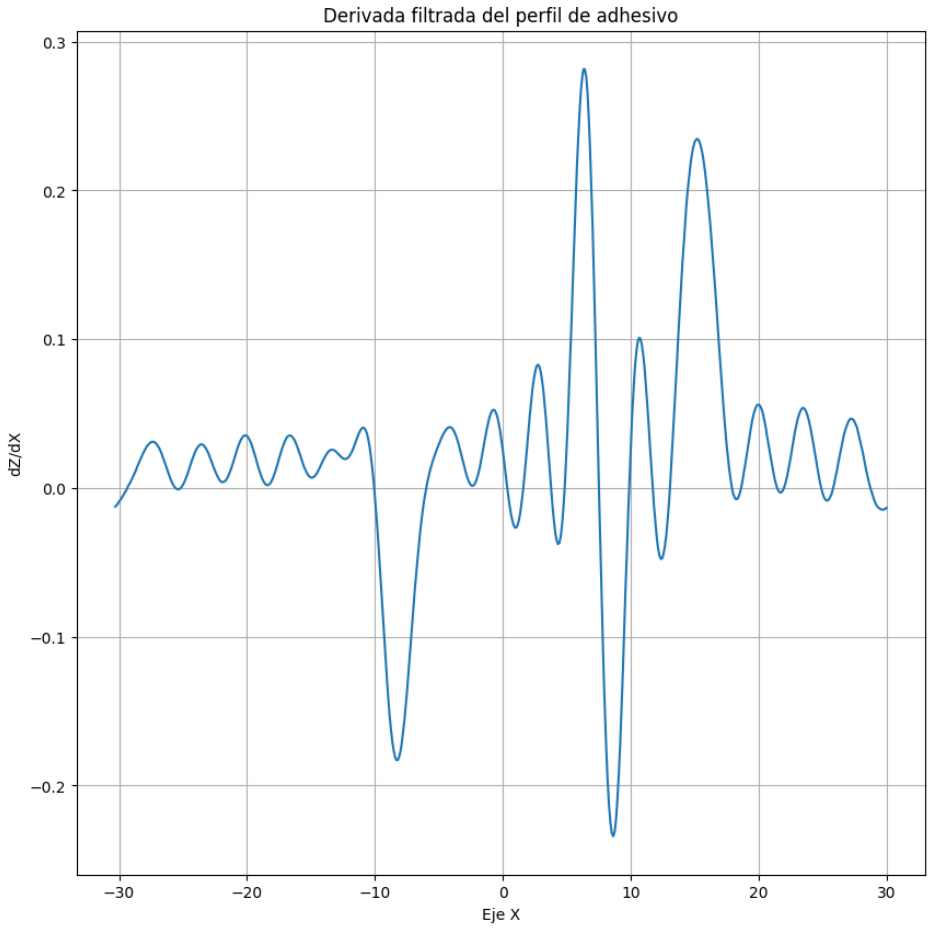



Figura 6.15: Filtrado espectral de derivada del perfil

A partir de esa señal tratada ya se pueden emplear los algoritmos de búsqueda de extremos locales sin ningún tipo de miedo a que fallen. Siendo el resultado la figura mostrada a continuación.

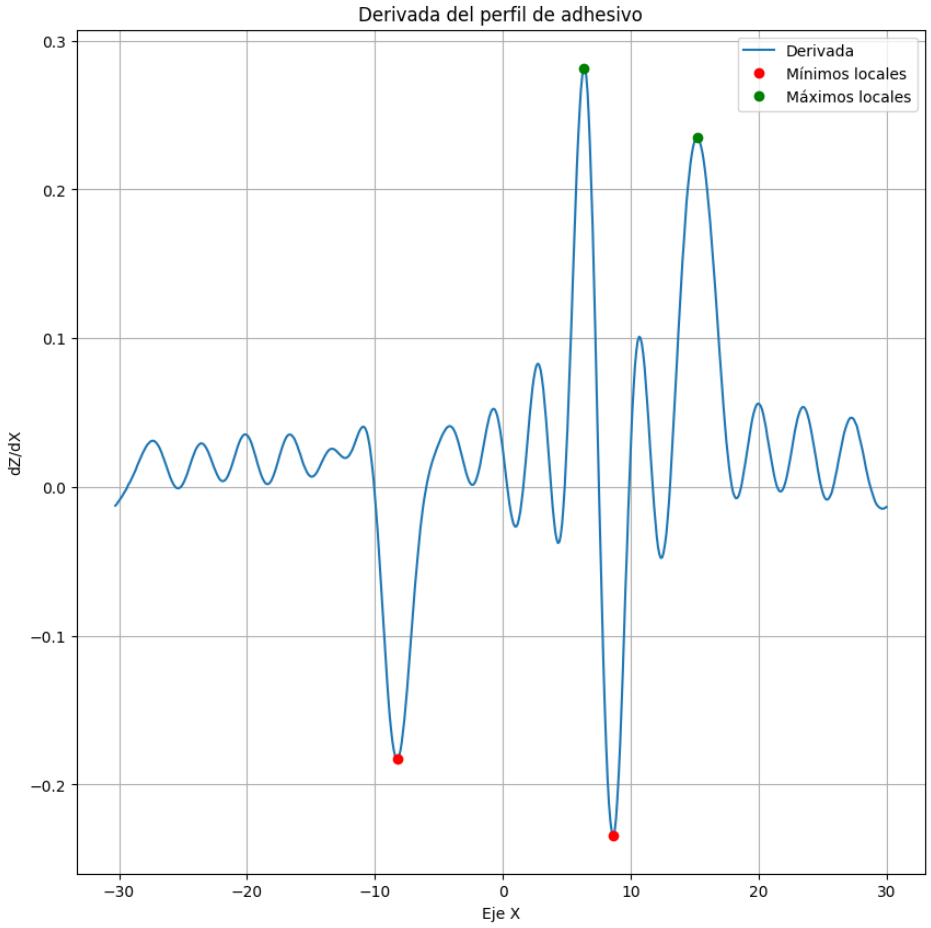



Figura 6.16: Ubicación de puntos críticos del perfil

#### 6.2.1.3.- Orientación de los datos

Actualmente ya se tiene la información suficiente como para calcular la sección del adhesivo, pero si se revisa el perfil (figura 6.11), se puede ver que tiene una inclinación sobre el eje x. Esto afectará negativamente al cálculo de la integral, pues habrá un área no deseada que se incluirá en el cálculo. Para ello la solución propuesta es rotar la nube de puntos hasta que quede alineada con el eje dominante.

Para ello se llevará a cabo una SVD (*Singular Value Decomposition*), la cual es un método de factorización matricial muy presente en todos los algoritmos de procesamiento de datos. Es usada para obtener aproximaciones óptimas de bajo orden de matrices y para llevar a cabo *pseudo-inversiones* de matrices no cuadradas para encontrar soluciones a los sistemas $Ax = b$.

Este método numérico permite llevar a cabo una descomposición de una matriz con la siguiente estructura

$$X = U \cdot S \cdot V \tag{6.8}$$

Donde U es la proyección de los datos **ortonormalizados** sobre el eje dominante, S es una matriz que contiene la dimensión de los datos, y es diagonal. Y V es una matriz de rotación. Entonces, para poder rotar los datos sobre el eje x, basta con hacer la siguiente operación. [18]

$$X_{proj} = U * S \qquad o \qquad X_{proj} = X * V \tag{6.9}$$

Finalmente el perfil quedaría rotado tal y como se muestra en la figura posterior, reduciendo en gran medida el error generado por una posición no deseada del perfil sobre los ejes de coordenadas.

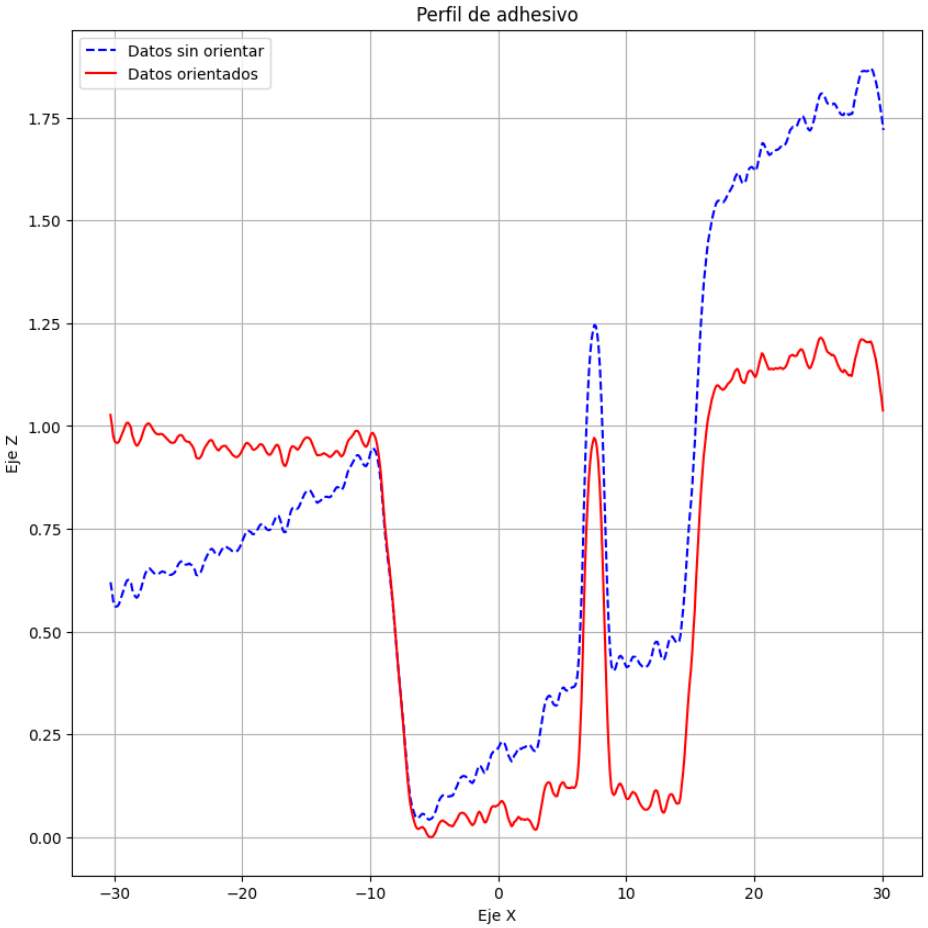



Figura 6.17: Rotación del perfil mediante SVD

#### 6.2.1.4.- Cálculo de características

Una de las principales operaciones que se pueden hacer con los perfiles medidos por el sensor, es calcular secciones. Para ello la principal herramienta es la integración.

Sabiendo que el sensor emite una nube de puntos, la única forma plausible de obtener la sección del objeto detectado es mediante una integración numérica. En este caso, se em-

pleará la integral trapezoidal compleja, ya que emplear recursos de computación en el uso de métodos numéricos más complejos como la regla de Simpson no tendría sentido.

La regla trapezoidal lleva a cabo una interpolación polinomial de grado 1 en dos nodos $x_0 = a$ y $x_1 = b$. Esta integral se hará en todos los pares de puntos que se han etiquetado dentro de lo que se considera como "hilo de adhesivo", las áreas se sumarán, y así se obtendrá el valor deseado. Siendo $n$ el número de puntos a integrar.

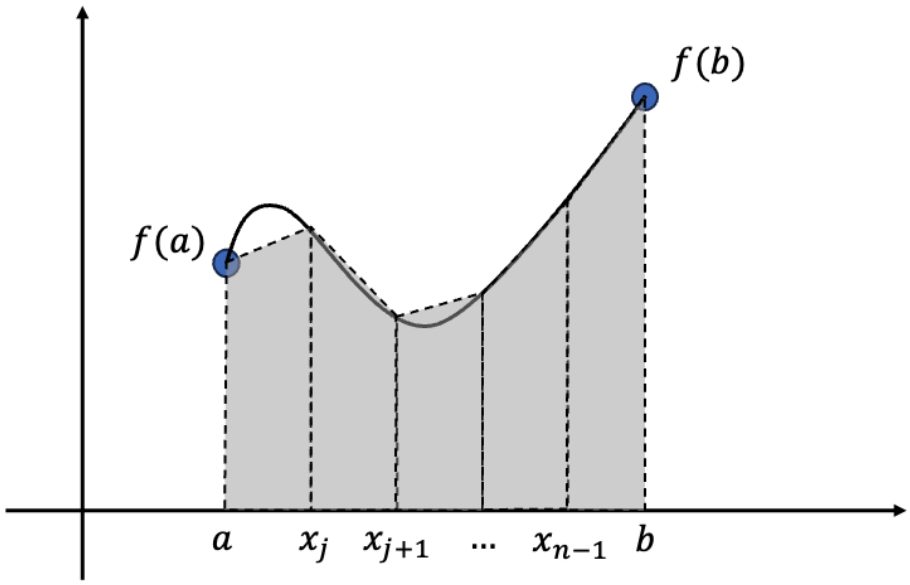



Figura 6.18: Integración numérica mediante la regla trapezoidal

Siendo la integral numérica igual a la siguiente expresión:

$$\int_a^b f(x)dx \approx \frac{b-a}{2n}\left[f(a)+2\sum_{j=1}^{n-1} f(x_j)+f(b)\right] \tag{6.10}$$

Otra medida importante a realizar es el cálculo de la posición relativa del hilo en pista. Este cálculo se reduce a obtener la diferencia entre la posición en el eje x del inicio de la pista de dispensado y un punto significativo del hilo de adhesivo, siendo en este caso la mediana de los puntos que se consideren como el hilo de adhesivo.

Es preciso anotar que esta etapa de extracción de características efectúa una reducción de la dimensión de los datos de entrada, pues vienen vectores de 580 valores y estos se reducen a uno de 2, en el que el primer valor será la sección y el segundo la posición.

$$\vec{X} \in \mathbb{R}^{580} \longrightarrow \vec{x} \in \mathbb{R}^2 \tag{6.11}$$

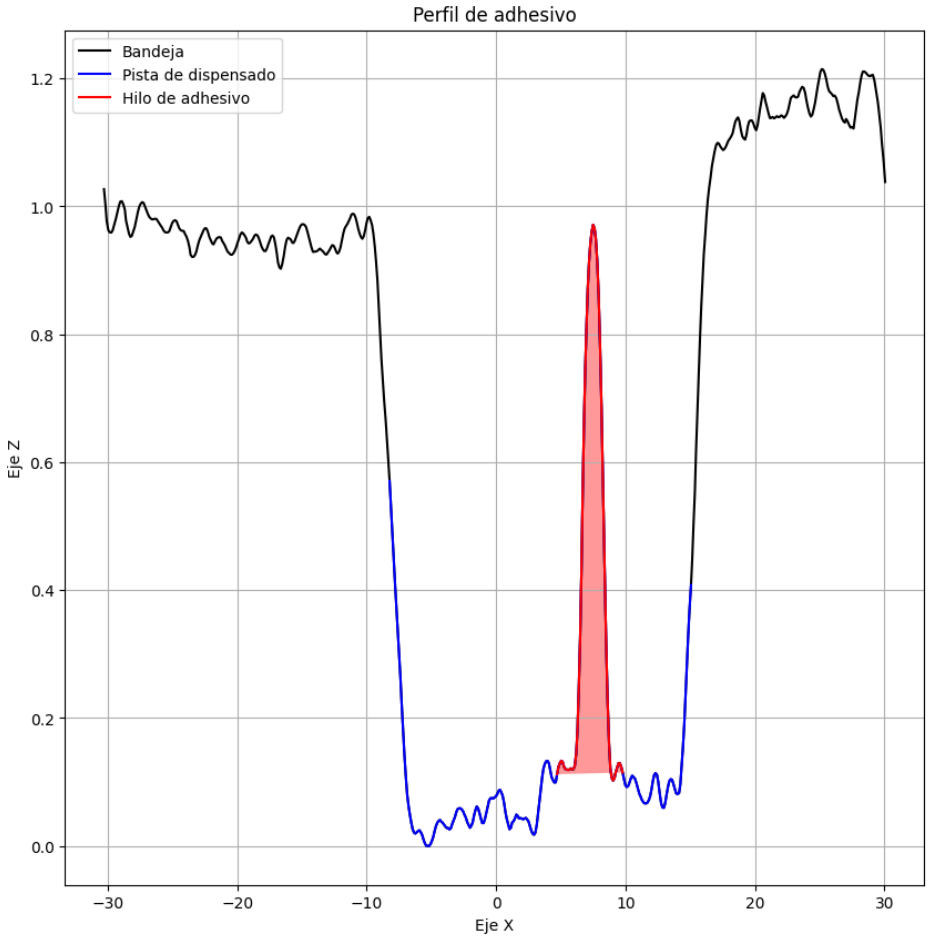



Figura 6.19: Perfil de adhesivo procesado

Para ver en detalle el desarrollo de esta etapa de extracción de características, revise el *Anexo IV*.

### 6.2.2.- Etapa de extracción de características volumétrica

Llegado este punto en el flujo de procesamiento de los datos, ya se tiene para cada perfil la sección de adhesivo y la posición del hilo sobre la pista, y será en esta etapa cuando se podrá tratar el concepto de volumen.

Al concatenar la información recogida de los perfiles se puede diseñar una *macro nube* de puntos que permite al programador ver con todo detalle el hilo de adhesivo e incluso las imperfecciones de la bandeja. Tal y como se muestra en la imagen posterior.

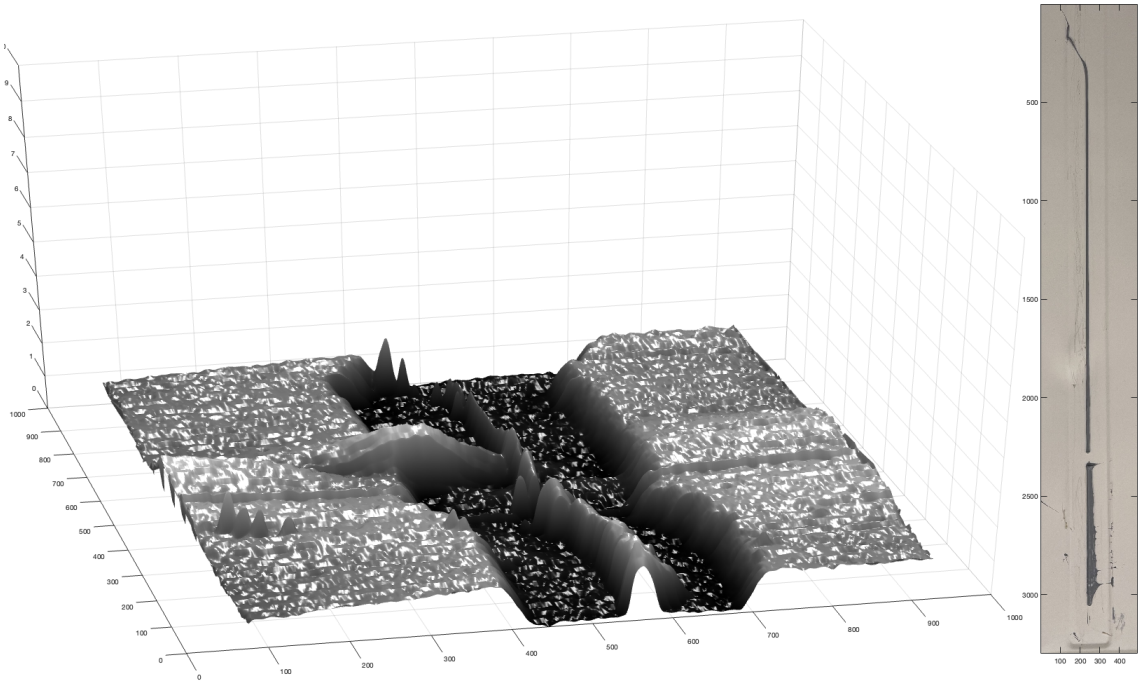


Figura 6.20: Representación en 3 dimensiones de un hilo de adhesivo sobre una pista de dispensado

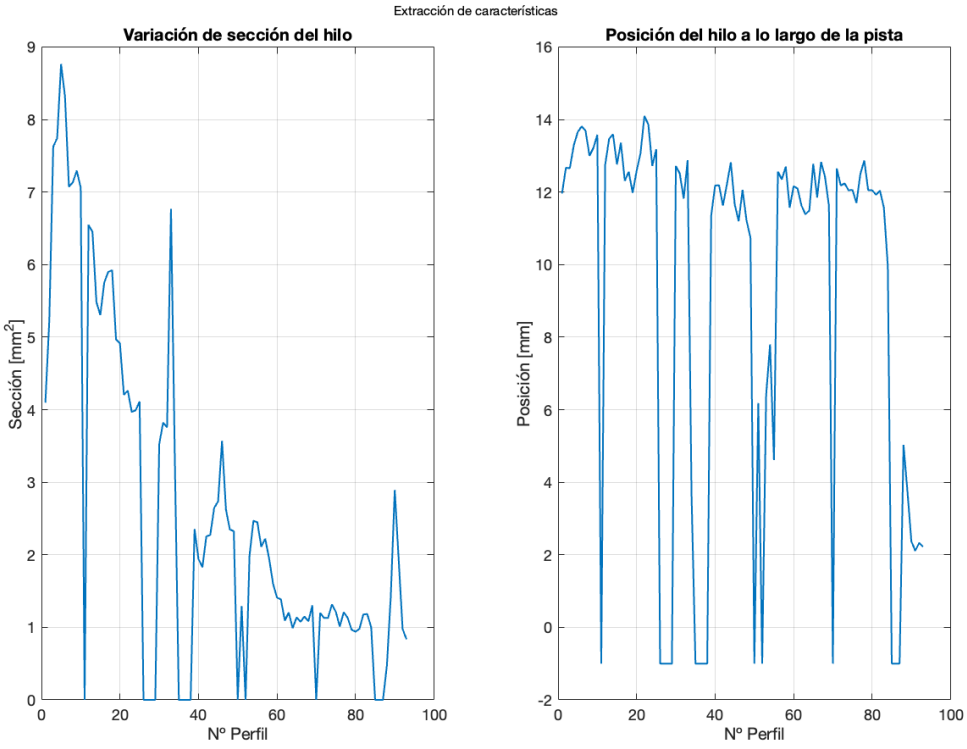



Figura 6.21: Seguimiento de características superficiales de figura 6.20

Pero el objetivo de esta etapa de extracción de características no es visualizar la infor-

mación, si no tratarla para encontrar patrones de comportamiento y poder corregirlos lo más rápido posible. Para ello, la idea propuesta es ir captando perfiles y segmentar el hilo de adhesivo en *micro-hilos*, que serán tratados consecutivamente para ir analizando la tendencia de sus características. Aún así, antes de comentar el procedimiento a seguir para llevar a cabo esta segmentación, se va a hablar de cómo se comportan este tipo de fluidos ante las condiciones a los que se les va a someter.

#### 6.2.2.1.- Comportamiento esperado del adhesivo

Para ganar una intuición sobre el sistema a modelar, es preciso encontrar una relación entre el caudal de salida de material, la velocidad del brazo robótico, la sección de adhesivo depositado y su posición sobre la pista. Y antes de usar el perfilómetro para obtener conclusiones, es conveniente hacer las siguientes consideraciones.

- A caudal constante ($Q \equiv cte$), si se aumenta la velocidad del robot, la sección disminuirá debido a que tendrá menos tiempo para adherirse a la pista. En este caso la posición del hilo vería reducida su dispersión.
- A velocidad constante del robot, si se aumenta el caudal, la sección aumentará, debido a que habrá más adhesivo a distribuir en el mismo tiempo. Eso debería generar ciertas ondulaciones en el fluido derivando en armónicos tanto en la sección como en la posición.
- A caudal constante, si se reduce la velocidad del robot, la sección aumentará, debido a que se depositará más volumen de material por el incremento de tiempo de pasada por pista. Haciendo que se generen imperfecciones como bultos o formas irregulares, traducidos en armónicos.
- A velocidad constante del robot, si se reduce el caudal, se verán los mismos efectos que en el primer punto. Reducción de sección y de dispersión en pista.

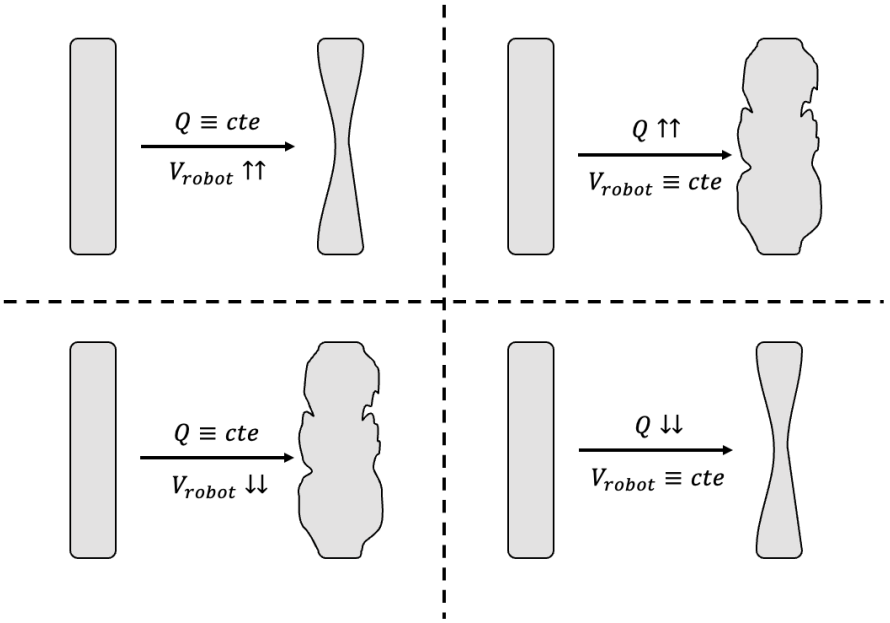



Figura 6.22: Comportamiento del adhesivo en función de los parámetros de operación

#### 6.2.2.2.- Tratamiento de la información

Como se dijo al principio de la explicación de esta etapa de extracción de características, se pretende enseñar al algoritmo de aprendizaje el concepto de volumen. Para ello se va a subdividir un hilo de adhesivo en micro hilos, que se irán tratando secuencialmente para extraer la información.

Estos micro hilos también se pueden concebir como ventanas de trabajo que recogen datos de varios perfiles a la vez, extrayendo de ellos descriptores estadísticos, frecuenciales y otras magnitudes físicas (como el volumen). A ello se le llama **método de enventanado**.

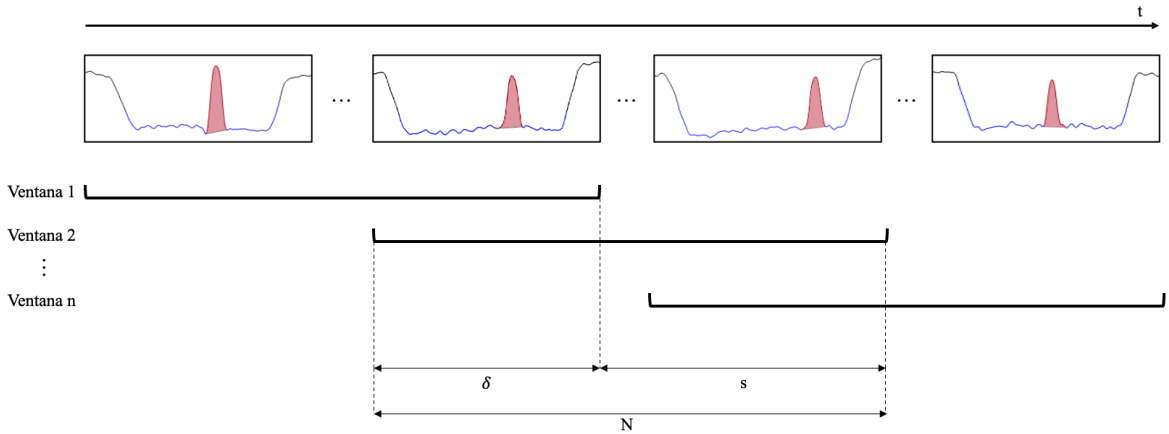



Figura 6.23: Esquema ilustrativo del método de enventanado

Tal y como se puede ver en la figura superior, este método consiste en crear subconjuntos de datos a partir de otros mayores y encapsularlos en ventanas de trabajo que pueden estar solapadas entre sí. Donde $N$ es el tamaño de la ventana, $\delta$ el solapamiento y $s$ el desplazamiento de la ventana por cada iteración. Esto traducido a un algoritmo se vería de la siguiente manera.

**Algorithm 7** Implementación del método de enventanado

**Require:** datosCrudos.size = Q
**Require:** ventana.Size = N AND N<Q
1: **for** (int $k$ = 1;$k$<Q-N+1;$k$+=$\delta$) **do**
2: ventana := datosCrudos[k:k+N-1] ▷ Se crea un array con el subconjunto de datos
3: ... ▷ Calcular descriptores
4: ... ▷ Almacenar descriptores
5: **end for**

#### 6.2.2.3.- Cálculo de características

Una vez aplicado el método de enventanado, para cada iteración del bucle se llevarán a cabo una serie de cálculos para deducir las características que definirán la morfología del adhesivo que se está dispensando. Se comenzará por calcular algunos descriptores estadísticos, después se hará uso de la FFT para calcular descriptores frecuenciales y por último se calcularán descriptores físicos haciendo uso de la integración numérica.

Pero antes de comenzar a enunciar los descriptores a calcular, se remarca que cada ventana de trabajo tendrá los datos a tratar estructurados de la siguiente forma

$$X = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} \\ \vdots & \vdots \\ x_1^{(Q)} & x_2^{(Q)} \end{bmatrix} \tag{6.12}$$

Donde la primera columna corresponde a las secciones de adhesivo, la segunda a las posiciones del adhesivo en pista y $Q$ al número total de muestras recogidas. Por cada iteración del método de enventanado se generarán subconjuntos de datos de N muestras con la misma estructura que la matriz 6.12, pero para diferenciarlas, se denominará V.

Se comenzará por definir los **descriptores estadísticos**. En este caso sólo interesan los valores medios, por ello las primeras características serán la sección media y la posición

media.

$$f_1 = \frac{1}{N}\sum_{j=1}^{N} v_1^{(j)} \tag{6.13}$$

$$f_2 = \frac{1}{N}\sum_{j=1}^{N} v_2^{(j)} \tag{6.14}$$

A continuación se calcularán los **descriptores frecuenciales**, para los cuales será necesario emplear la FFT. Teniendo siempre presente el tiempo de muestreo del sistema, se escogerá una banda de frecuencia sobre la que calcular el valor eficaz. En el caso que ocupa este proyecto se escogió una banda de 0,5 a 8 Hz.

$$f_3 = \frac{1}{N}\sqrt{\sum 2 * \left(V_1^{(j)}\right)^2} \tag{6.15}$$

$$f_4 = \frac{1}{N}\sqrt{\sum 2 * \left(V_2^{(j)}\right)^2} \tag{6.16}$$

Sobre las expresiones superiores se harán dos anotaciones. En la primera se insiste en que el rango de valores que conforma el sumatorio son todos aquellos en los que la frecuencia se encuentre entre 0,5 y 8 Hz, el número de valores depende exclusivamente de la frecuencia de muestreo y del número de valores por ventana.

```
1    tm = T/Q; % Periodo de muestreo
2    fm = 1/tm; % Frecuencia de muestreo
3
4    f = (0:N-1)*(fm/N); % Vector de frecuencias para analisis de
     armonicos
5    idx = find((f>=0.5)&(f<=8.5)); % Banda de frecuencia de interes
```

Como segunda anotación se quiere enfatizar en el producto de 2 por el cuadrado del armónico correspondiente en la suma. Pues hay que tener en cuenta que para hacer el cálculo del valor eficaz se hace el valor absoluto de la FFT, y esta devuelve valores complejos con módulo y fase. Haciendo así que sea necesario tener en cuenta las frecuencias negativas.

Finalmente se calcularán los **descriptores físicos**, que en este caso serán el caudal de deposición y el volumen depositado. Para ambos será necesario recurrir a la integración numérica empleada en la etapa de extracción de características anterior. Véase la figura 6.18 y la expresión 6.10.

Es sabido que el volumen se define como la integral de la sección

$$V = \int S \cdot \mathrm{d}x = \int S \cdot \mathrm{d}x \cdot \frac{\mathrm{d}t}{\mathrm{d}t} = v_{robot} \int S \cdot \mathrm{d}t \tag{6.17}$$

Donde $S$ es la sección transversal del hilo de adhesivo y $x$ el eje sobre el que se mueve la bandeja. Si se calcula dicho volumen integrando la sección...

$$f_5 = V \approx \frac{v_{robot}}{2} \cdot t_m \cdot \left[ v_1^{(1)} + 2 \sum_{j=1}^{N-1} v_1^{(j)} + v_1^{(N)} \right] \tag{6.18}$$

Y la última característica sería sencillamente el caudal de deposición.

$$f_6 = Q = \frac{\mathrm{d}V}{\mathrm{d}t} \approx \frac{V}{t_m \cdot \delta} \tag{6.19}$$

Tras el cálculo de los descriptores se generará una matriz para el entrenamiento del modelo de inteligencia artificial que tendrá el siguiente aspecto.

$$F = \begin{bmatrix} f_1^{(1)} & f_2^{(1)} & \cdots & f_6^{(1)} \\ f_1^{(2)} & f_2^{(2)} & \cdots & f_6^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ f_1^{(m)} & f_2^{(m)} & \cdots & f_6^{(m)} \end{bmatrix} \tag{6.20}$$

### 6.2.3.- Diseño y entrenamiento de la red neuronal

Esta sección del proyecto será el trabajo más iterativo de todos. Pues se irán diseñando redes neuronales cada vez más extensas con el fin de modelar con el mayor grado de precisión las relaciones que se ven entre los datos de entrada, y por tanto, del comportamiento del adhesivo. Pero se comenzará por algo sencillo, como es el entrenamiento de un algoritmo de clasificación binaria para determinar si el hilo que se está dispensando se considera *óptimo* o no.

Tal y como se comentó anteriormente, la condición indispensable para que los sistemas basados en aprendizaje automático supervisado funcionen correctamente, es mediante un conjunto de datos correctamente dimensionado. A la hora de tratarlos en este proyecto se seguirá la siguiente terminología:

- Etiqueta $(y)$: Valor que se quiere predecir.
- Atributos[2] $(\vec{x} = [x_1, x_2, \ldots, x_n])$: Variables de entrada que describen los datos.
- Ejemplo $\vec{x}^{(i)}$: Instancia de datos de entrada en particular (El superíndice se corresponde al número de la muestra tomada).

[2]Los atributos empleados para entrenar la red neuronal son los provenientes de la matriz de características F, pero para respetar el convenio se denominarán con la letra X.

- Ejemplo etiquetado $[\vec{x}^{(i)}, y^{(i)}]$: Se usan para entrenar el modelo.
- Ejemplo sin etiquetar $[\vec{x}^{(i)}, ?]$: Se usan para hacer predicciones sobre datos nuevos.
- Modelo $(f_{\vec{w},b}(\vec{x}))$: Asigna ejemplos a etiquetas predichas y permite ser evaluado para datos nuevos. Un modelo genera predicciones $(\hat{y})$.
- Número de muestras recogidas $(m)$
- Número de características por muestra $(n)$

Entonces, el modelo que se va a construir tendrá el siguiente aspecto. Remarcando que tiene múltiples entradas, y una salida (MISO).

$$\hat{y} = f_{\vec{w},b}(\vec{x}) \quad \setminus \quad \forall \vec{x} \in \mathbb{R}^6 \tag{6.21}$$

Para diseñar un modelo de aprendizaje automático supervisado hay que llevar a cabo 4 decisiones de diseño:

1. Definir función de coste en términos del tipo de salida que se espera que tenga.
2. Decidir qué optimizador usar para entrenar el modelo de inteligencia artificial.
3. Diseñar la arquitectura de la red neuronal
4. Etiquetar los datos de entrenamiento.

Estos puntos se irán comentando en detalle a lo largo de las próximas páginas. Y se remarca que a pesar de que las librerías de entrenamiento y despliegue de estos modelos ofrezcan la posibilidad de usar estos algoritmos sin un marco teórico extenso, en esta memoria se comentarán los métodos matemáticos involucrados para enfatizar la idea de que para hacer una aplicación a medida, hay que conocer la teoría que hay por detrás.

#### 6.2.3.1.- Función de coste

Es sabido que el entrenamiento de una inteligencia artificial es un proceso iterativo, pues partiendo de unos datos de entrenamiento y de unos parámetros del modelo dados por el programador, el optimizador va buscando la mejor forma de encajar el modelo con los datos proporcionados.

Pero para que el optimizador haga su trabajo correctamente, debe tener un *feedback*. Para entender esto, se anima al lector a pensar en un dicho que dice así, *todos los errores tienen su coste*. Pues la función de coste sigue exactamente ese planteamiento.

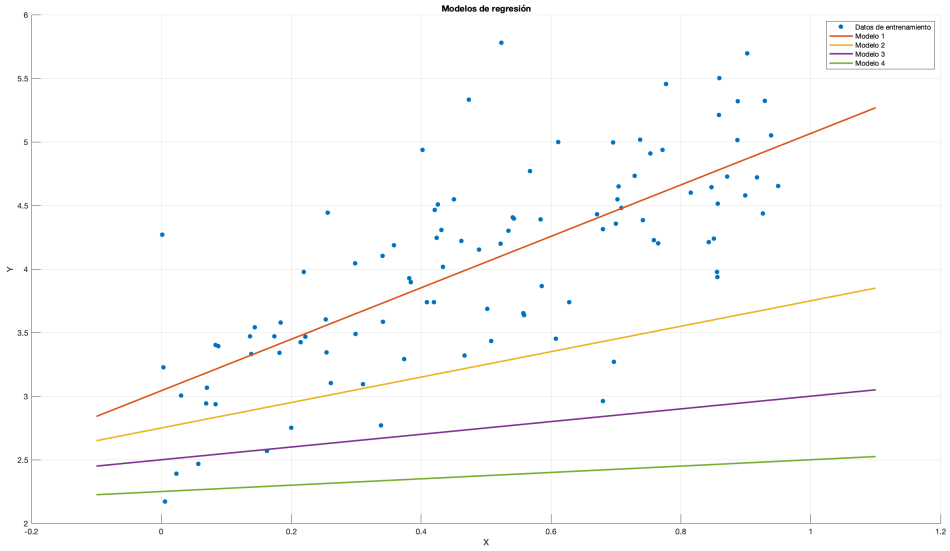



Figura 6.24: Modelos de ejemplo de regresión

Si se observa la figura superior se pueden ver distintos modelos de regresión lineal sobre un conjunto de datos, y es apreciable que hay modelos que encajan mejor con la información que otros. Ahora bien, ¿cómo se puede determinar lo bien que representa cada modelo al conjunto de datos? O mejor dicho, ¿cómo se puede calcular su coste?

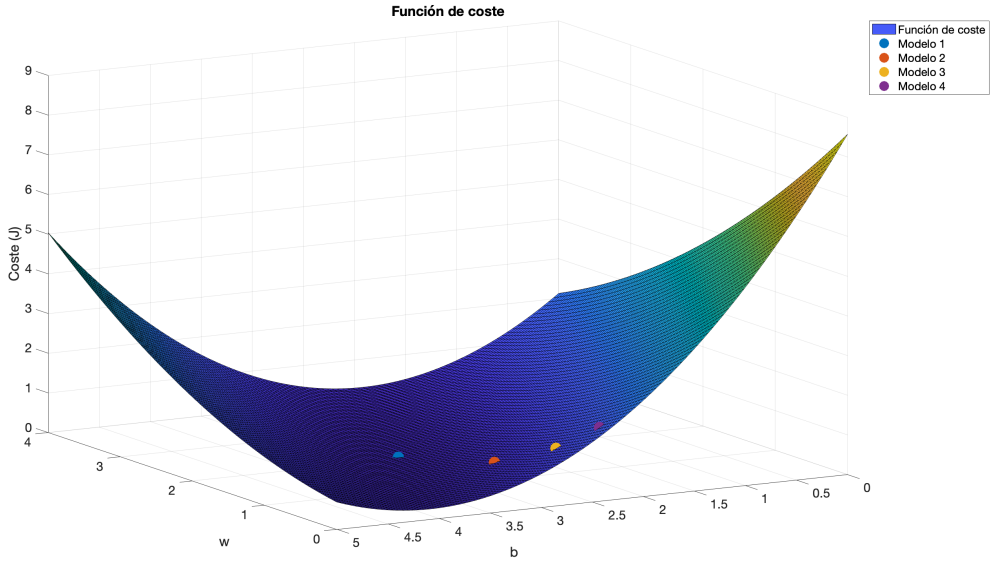



Figura 6.25: Función de coste del ejemplo propuesto en la figura 6.24

$$J(\vec{w},b) = \frac{1}{2m}\sqrt{\sum_{i=1}^{m}(\hat{y}^{(i)} - y^{(i)})^2} = \frac{1}{2m}\sqrt{\sum_{i=1}^{m}(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)})^2} \tag{6.22}$$

Lo que se hace es determinar el coste mediante la función de mínimos cuadrados, que calcula la raíz cuadrada del sumatorio de errores absolutos entre predicción y valor de entrenamiento elevados al cuadrado, sabiendo que la predicción en este ejemplo viene dada por un modelo del tipo $f_{w,b}(x) = w \cdot x + b$.

Pero si se recuerda lo que se comentó al principio de la sección, lo que se pretende hacer es un clasificador binario, no un modelo de regresión. En estos modelos sólo hay un conjunto limitado de resultados, y en los binarios sólo hay 2 posibles, 1 y 0. Y el objetivo es crear un modelo que permita establecer un margen de decisión para determinar si una muestra se clasifica en una clase u otra.

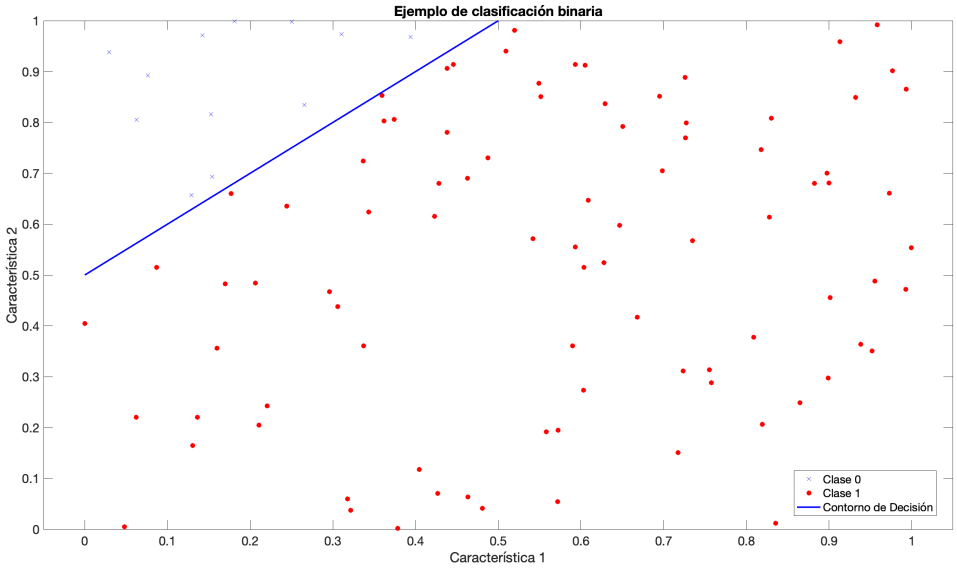



Figura 6.26: Modelo de ejemplo de clasificación binaria

La función de coste se vería afectada debido a que el modelo no sería polinomial. Ya que lo que va a dar como salida es la probabilidad de que la muestra pertenezca a una clase o a otra. Esto es posible gracias a la función sigmoide, que es el resultado del cálculo de probabilidades sobre una función de densidad de una distribución normal.

$$f_{\vec{w},b}(\vec{x}) = \frac{1}{1 + e^{-(\vec{w}\cdot\vec{x}+b)}} \tag{6.23}$$

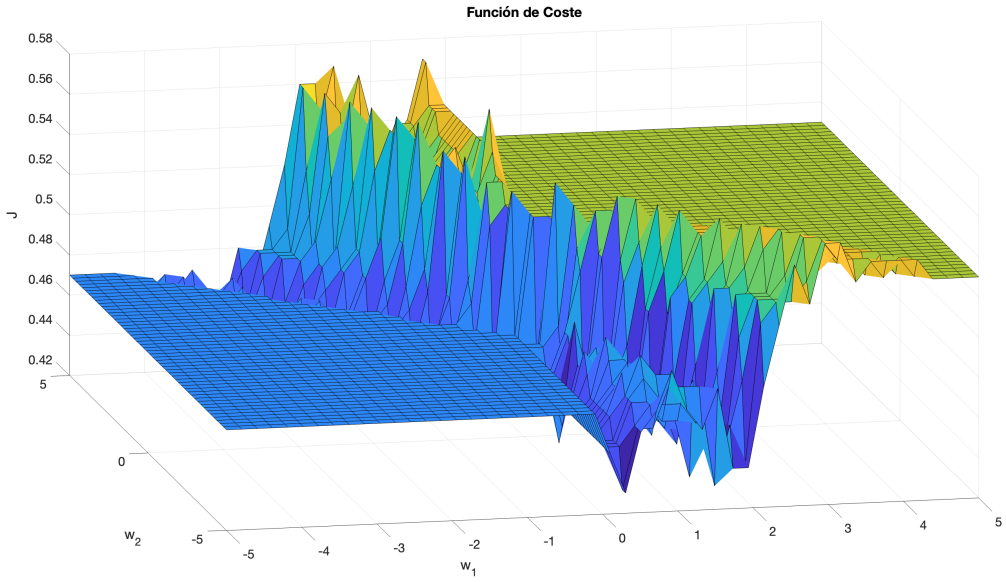



Figura 6.27: Función de coste del ejemplo propuesto en la figura 6.26

Viendo la figura superior, se puede ver que la función de coste presenta una morfología con muchos mínimos locales. Lo cual es un problema para el optimizador. Es por ello que para este tipo de problemas se ha implementado la función de pérdida o entropía cruzada, que pretende convertir a esta función de coste en una convexa.

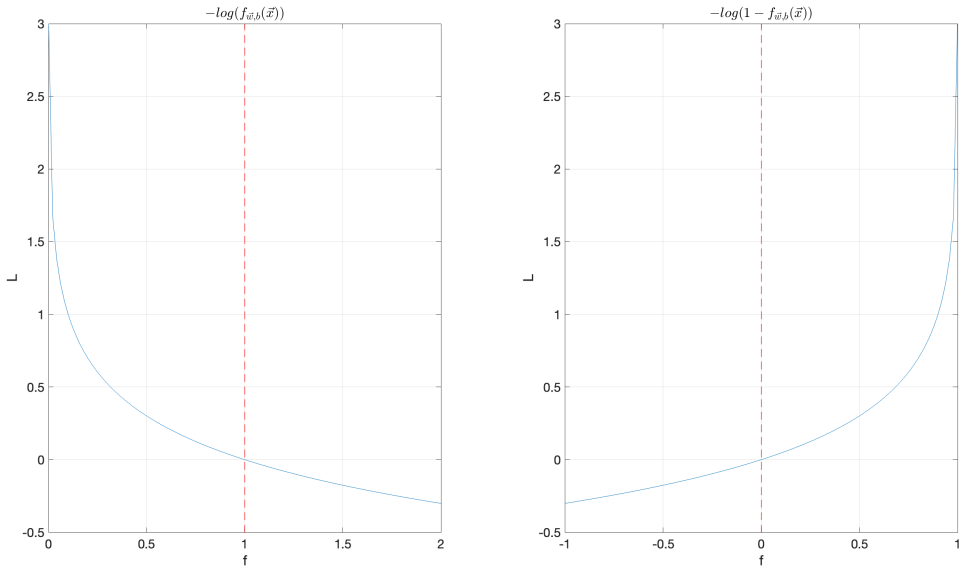



Figura 6.28: Función de entropía cruzada

La función estará en su mínimo mientras la predicción del modelo se encuentre más cerca de una etiqueta. Es decir, cuanto más lejos esté la predicción de la etiqueta, mayor será la

pérdida. Y puede ser expresada de la siguiente forma

$$L\left(f_{\vec{w},b}(\vec{x}^{(i)}),y^{(i)}\right) = \begin{cases} -log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right) & IF \quad y^{(i)}=1 \\ -log\left(1-f_{\vec{w},b}(\vec{x}^{(i)})\right) & IF \quad y^{(i)}=0 \end{cases} \tag{6.24}$$

O de esta otra,

$$L\left(f_{\vec{w},b}(\vec{x}^{(i)}),y^{(i)}\right) = \left(-y^{(i)}log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right) - \left(1-y^{(i)}\right)log\left(1-f_{\vec{w},b}(\vec{x}^{(i)})\right)\right) \tag{6.25}$$

Entonces la función de coste a utilizar será

$$\begin{aligned} J(\vec{w},b) &= \frac{1}{2m}\sum_{i=1}^{m} L\left(f_{\vec{w},b}(\vec{x}^{(i)}),y^{(i)}\right) \\ &= -\frac{1}{2m}\sum_{i=1}^{m}\left(y^{(i)}log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right) + \left(1-y^{(i)}\right)log\left(1-f_{\vec{w},b}(\vec{x}^{(i)})\right)\right) \end{aligned} \tag{6.26}$$

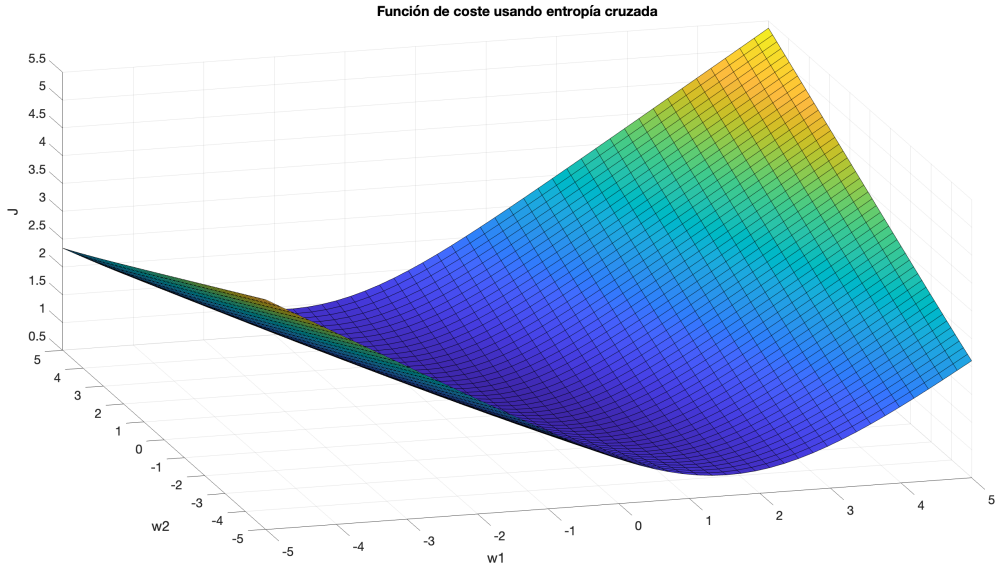



Figura 6.29: Función de coste del ejemplo 6.26 usando la entropía cruzada

Con esto queda determinada la función de coste a usar para entrenar el modelo. En la librería que se empleará en este proyecto es llamada *Binary Cross-Entropy Cost Function*. Y para emplearla en un modelo hay que poner la siguiente línea de código.

```
model.compile(
    loss=BinaryCrossentropy(...),
    ...
)
```

#### 6.2.3.2.- Selección de optimizador

Tal y como se mencionó antes, mediante la función de coste se puede determinar lo bien que encaja un modelo con sus parámetros en un conjunto de datos, tanto en problemas de regresión como en los de clasificación. Pero para poder obtener el mejor modelo hay que saber moverse por el plano, es decir, hay que buscar el mínimo de la función de coste, porque eso implicará un error mínimo, y por lo tanto un modelo preciso dentro de las posibilidades.

El algoritmo por excelencia para encontrar un modelo óptimo es llamado **descenso del gradiente**, en el que asumiendo que en los sistemas de alta dimensionalidad pueden estar dotados de extremos en los cuales el gradiente debe ser 0 (también denominados puntos de silla)

$$\nabla f(x) = 0 \tag{6.27}$$

Se usa la información dada por la derivada como base de un proceso iterativo que progresivamente converge en un mínimo local de la función sobre la que se aplica.[18]

**Algorithm 8** Implementación del descenso del gradiente

1: **repeat**
2: $w_j := w_j - \alpha \cdot \frac{\partial(J(\vec{w},b))}{\partial w_j}$
3: $b := b - \alpha \cdot \frac{\partial(J(\vec{w},b))}{\partial b}$
4: **until** convergence with simultaneous updates

El procedimiento consiste en ir actualizando los parámetros del modelo para ir acercándose cada vez más al mínimo de la función de coste. Es importante anotar la aparición de un nuevo parámetro, que es otro grado de libertad a la hora de entrenar estos sistemas, y es $\alpha$. Este es el ratio de aprendizaje, y básicamente se emplea para preestablecer la *longitud de los pasos* que da el algoritmo en cada iteración. Hay que tener especial cuidado con esto, ya que si es demasiado grande el algoritmo no convergerá, y si es demasiado pequeño tardará demasiado en converger y consumirá demasiados recursos. La selección de este parámetro se reduce a un criterio meramente basado en la experiencia del programador.

Para redes neuronales se usa el **descenso del gradiente estocástico** (SGD), que actualiza los pesos de las neuronas según la función de coste. A diferencia del descenso del gradiente tradicional, el SGD no usa todas las muestras debido a que el número de parámetros y muestras es muy grande, lo que haría el cálculo del gradiente una labor computacionalmente muy costosa. En su lugar, selecciona aleatoriamente un punto o un subconjunto de datos para aproximar el gradiente en cada iteración, una técnica conocida como *mini-batching*. Un algoritmo derivado de estos métodos es *Adam* (*Adaptive Moment Estimation*), que ajusta el ratio

de aprendizaje de manera variable para cada característica y fue utilizado en este proyecto.

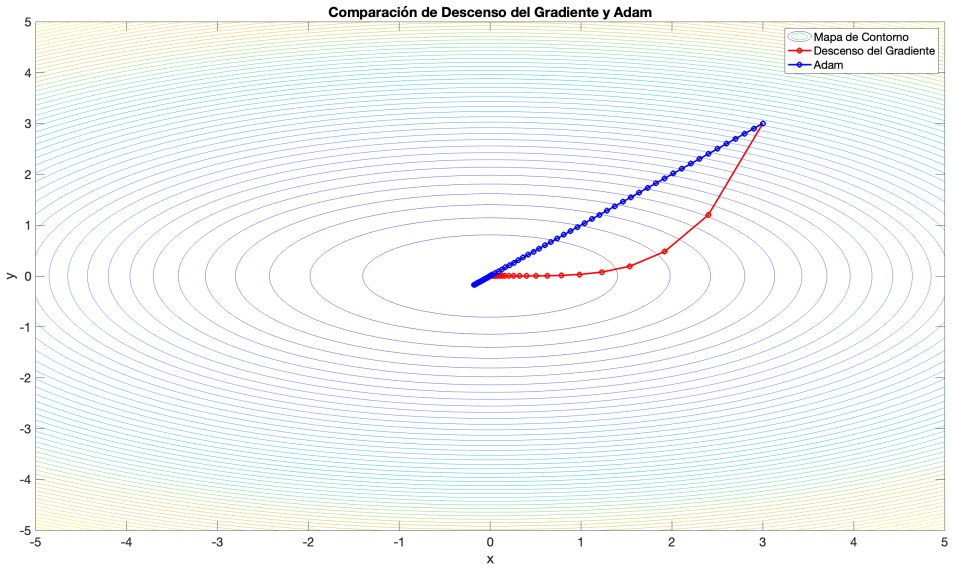



Figura 6.30: Comparativa de descenso del gradiente contra Adam

Para usar este algoritmo en el programa de entrenamiento del modelo habría que escribir la siguiente línea de código.

```
    model.compile(
        optimizer=Adam(learning_rate=1e-4),
        ...
    )
```

#### 6.2.3.3.- Arquitectura de la red neuronal

En el paso anterior del diseño se hablaron de distintos algoritmos usados para entrenar estos modelos basados en redes neuronales, pero en ningún momento se ha llegado a definir lo que son. Para ello se empezará por lo básico, ¿qué es una neurona?

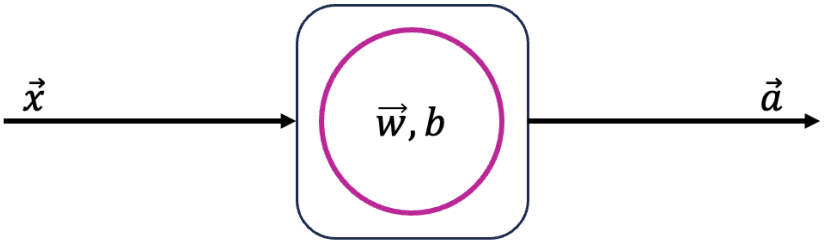



Figura 6.31: Modelo analítico de una neurona

Estos modelos matemáticos están inspirados en las neuronas reales, pues reciben la información por las dendritas, la procesan en un núcleo y la transmiten por el axón. En el caso de las neuronas que conforman las redes neuronales, un vector de entrada $\vec{x}$ entra en la neurona y en un primer lugar se calcula el producto con sus pesos $z = \vec{w} \cdot \vec{x} + b$, y el valor calculado es pasado a una función de activación $g(z)$. El resultado de esa transformación es transmitido a la siguiente neurona o directamente puede ser el resultado de la red neuronal.

La clave de estos modelos consiste en que las neuronas pueden ser almacenadas en capas y estas pueden interactuar entre sí. A mayor número de capas, mayor número de parámetros a optimizar, y mayores relaciones se pueden establecer entre los datos.

Las funciones de activación son importantes porque permiten introducir no-linealidades en el modelo, permitiendo así favorecer la aparición del comportamiento emergente. Los tipos que se van a emplear en este modelo son los siguientes.

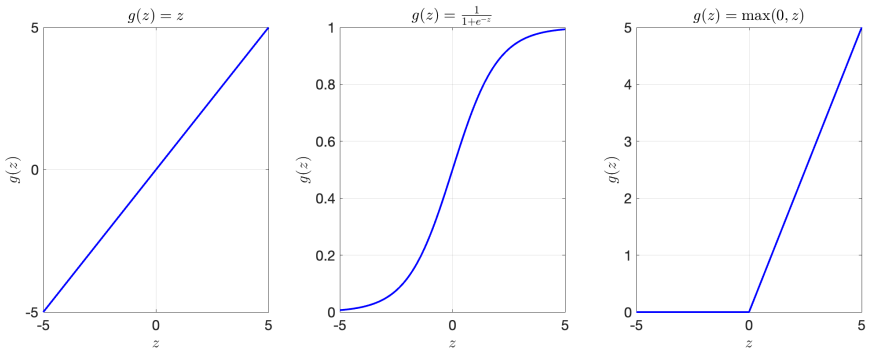



Figura 6.32: Funciones de activación

A la izquierda se puede ver la función de activación lineal, en mucha documentación se dice que esta función de activación es equivalente a no tenerla. En el medio se ve la función sigmoide, se utilizará en la neurona de salida del modelo para llevar a cabo el clasificador binario. Antaño se utilizaba en las capas interiores de neuronas pero se descubrió que la función de activación de la derecha, la ReLU (*Rectified Linear Unit*) facilitaba mucho la labor de entrenamiento ya que generaba una función de coste más convexa.

Para decidir qué arquitectura (número de capas con número de neuronas en cada una) tendrá la RN, hay que seguir una serie de pasos que conforman el desarrollo iterativo de un sistema de ML.

1. Escoger arquitectura (modelo, datos, ...)
2. Entrenamiento del modelo (función de coste, optimizador)
3. Diagnóstico (sesgo, varianza y análisis de error)

La idea es escoger una arquitectura determinada de una RN, entrenarla y evaluarla. Pero, ¿qué es evaluar una red neuronal?

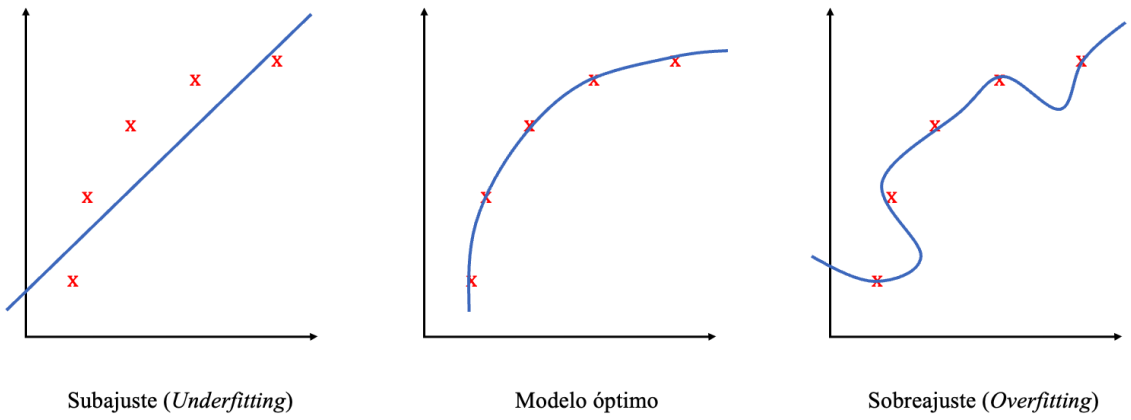



Figura 6.33: Análisis de ajuste de modelos

Si se revisa la figura 6.33, se puede ver a su izquierda un modelo que no encaja correctamente los datos debido a que el entrenamiento no ha sido suficiente. Este fenómeno es denominado subajuste o *underfitting* se considera peligroso debido a que puede generar predicciones con un error tan severo que pueda dar lugar a engaños. El caso de la derecha es lo contrario, se llama sobreajuste o *overfitting*, en este se le asignaron tantos recursos al modelo que se ajustó “demasiado bien” a los datos, haciendo que si se emiten predicciones

no sean fieles a la idea que se quiere modelar. Finalmente quedaría el modelo del medio, que representa fielmente a los datos.

Una de las razones por las que se escogió el uso de redes neuronales es porque son consideradas como *máquinas de bajo sesgo*, es decir, se ajustan tan bien a los datos que no suelen generar problemas de subajuste. El problema es que para poder ver si se están produciendo estos fenómenos no se pueden mostrar los datos como en la figura superior. Para ello existen unas técnicas de diagnóstico que consisten en subdividir el conjunto de datos en 3 subconjuntos:

- Conjunto de entrenamiento $(X_{train}, y_{train})$: Este se empleará para entrenar las distintas arquitecturas de redes neuronales.
- Conjunto de validación cruzada $(X_{cv}, y_{cv})$: Este se empleará para comparar los distintos modelos que se vayan creando.
- Conjunto de prueba $(X_{test}, y_{test})$: Para afirmar que los datos usados para el entrenamiento son correctos, se reservará una parte de la información con el fin de comprobar que no hay desajustes. Estos datos no se emplearán para tomar decisiones de diseño.

Si se calculan las fracciones en los que las predicciones del modelo fueron erróneas sobre las de los datos etiquetados, se pueden sacar conclusiones. Para ver en detalle el procedimiento seguido para escoger la arquitectura de la RN y el diseño de la etapa de extracción de características volumétrica, revise el *Anexo V*.

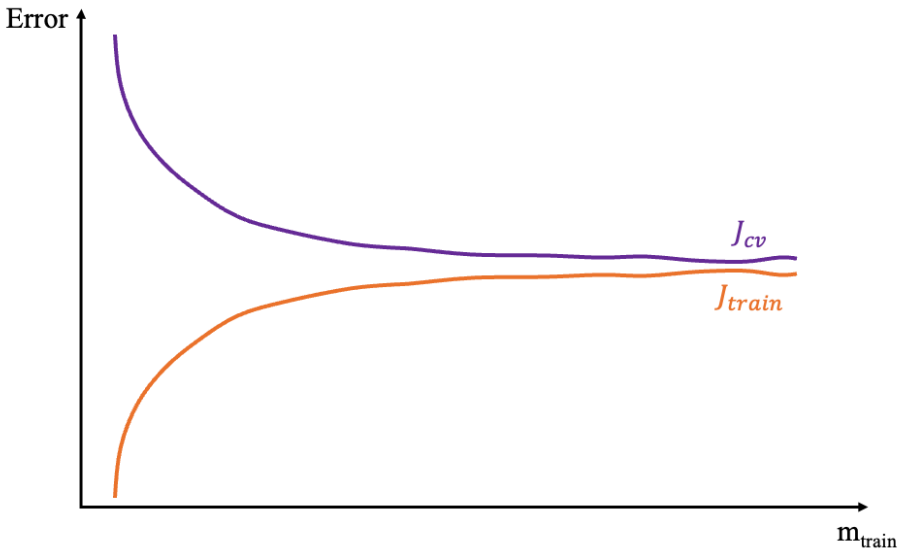



Figura 6.34: Diagnóstico del modelo en función de la cantidad de datos de entrenamiento

Si se calculan dichos errores y estos son parecidos se puede decir que el modelo es correcto y que no sufre de estos problemas.

Tras algunas iteraciones, se ha tomado de decisión de emplear una RN con dos capas internas de 25 y 15 neuronas respectivamente y una neurona de salida con el resultado de la predicción.

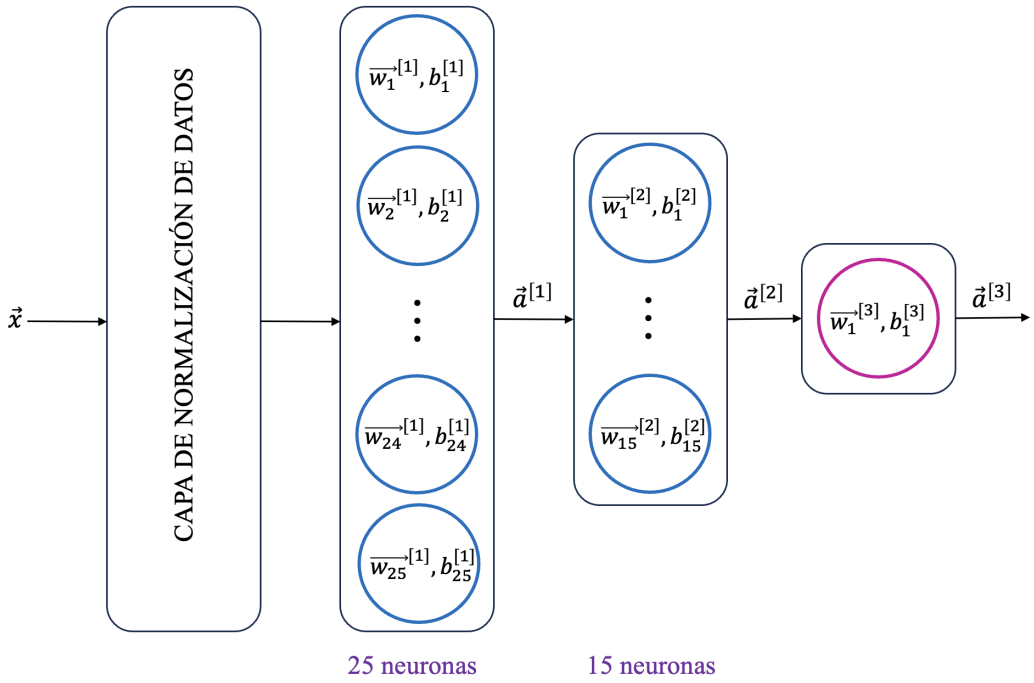



Figura 6.35: Arquitectura de la red neuronal

Sobre el modelo de la figura 6.35 se quieren hacer 3 anotaciones. En primer lugar, se remarca el uso de la capa de normalización de los datos. Esto se considera una buena práctica cuando las características de un modelo toman rangos de valores distintos, como es el caso, ya que la posición del hilo de adhesivo en pista sólo puede variar entre 0 y 20mm, mientras que los descriptores frecuenciales o los físicos pueden ser mucho mayores. Es por ello que al normalizarlos se consigue que la función de coste tenga una morfología más sencilla de tratar por parte el optimizador.

$$x_i = \frac{x_i - \mu}{\sigma} \tag{6.28}$$

A su vez, remarcar que las funciones de activación de las capas ocultas serán todas ReLU, mientras que la de la neurona de salida será una sigmoide para poder hacer el clasificador binario.

Por último, se recuerda que este tipo de neuronas son denominadas *Densas*, pues cada neurona de una capa toma los valores de todas las neuronas que le preceden en la capa

anterior. A diferencia de las convolucionales, que toman valores por un método similar al de enventanado.

Una vez remarcados estos aspectos, se mostrará el código necesario para definir este modelo de RN en Python con la librería tensorflow.

```
    model = Sequential([
        Input(shape=(6,)),
        BatchNormalization(),
        Dense(units = 25, activation = 'relu'),
        Dense(units = 15, activation = 'relu'),
        Dense(units = 1, activation = 'linear')
    ],name = 'Modelo')
```

Nótese que en la neurona de salida se ha empleado una función de activación lineal. Esto se considera una buena práctica debido a que ayuda a que los errores de redondeo por computación numérica se reduzcan en las predicciones. Pero es imprescindible tener en cuenta que cada vez que se haga una predicción de este modelo, habrá que pasar la salida por la función sigmoide para tener el resultado verdadero. A su vez, habrá que especificar la siguiente opción en la declaración de la función de coste.

```
    model.compile(
        loss=BinaryCrossentropy(from_logits=True),
        optimizer=Adam(learning_rate=1e-4),
    )
```

# 7. Resultados y Discusión

Para el entrenamiento del modelo de ML se decidió hacer varios experimentos de deposición de adhesivo a distintas velocidades, dentro del cual se echaba el material a la misma velocidad en todas las pistas de una bandeja. Esto dio lugar a los siguientes resultados.

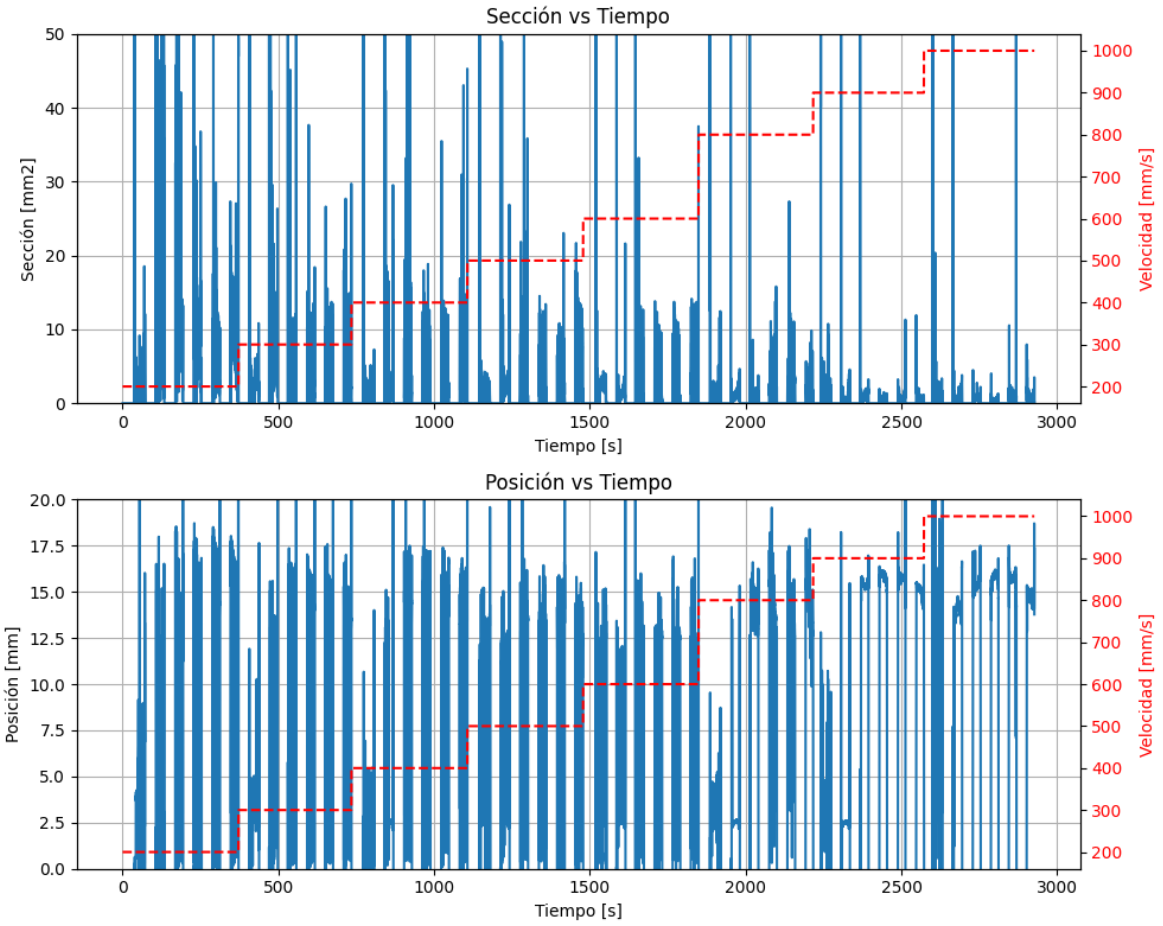



Figura 7.1: Características superficiales calculadas del experimento realizado

Se puede apreciar que a menores velocidades aparecen armónicos, esto se debe a lo que se comentó anteriormente. Al ir despacio el material se tiene que ir distribuyendo por donde le es posible, aumentando de esta forma la zona considerada como hilo de adhesivo dentro del cálculo de secciones de la primera etapa de extracción de características.

Lo mismo sucede con la posición. A menor velocidad, mayor dispersión. Y al aumentar se estabiliza. A este fenómeno se le llama **flujo inestable y/o serpenteante**. Este es una manifestación de la inestabilidad inherente en los fluidos pseudoplásticos debido a la interacción compleja entre la gravedad, la viscosidad variable y las fluctuaciones iniciales en el flujo. Esta inestabilidad es lo que da lugar a las oscilaciones y patrones ondulantes observados cuando el fluido cae desde una altura. Este suceso depende directamente de la distancia

entre la boquilla de salida de la válvula y la pista. También cabe mencionar que medida que se aumenta la velocidad del brazo robótico se puede apreciar que los lazos se “deshacen”, tal y como se muestra en la figura 7.2.

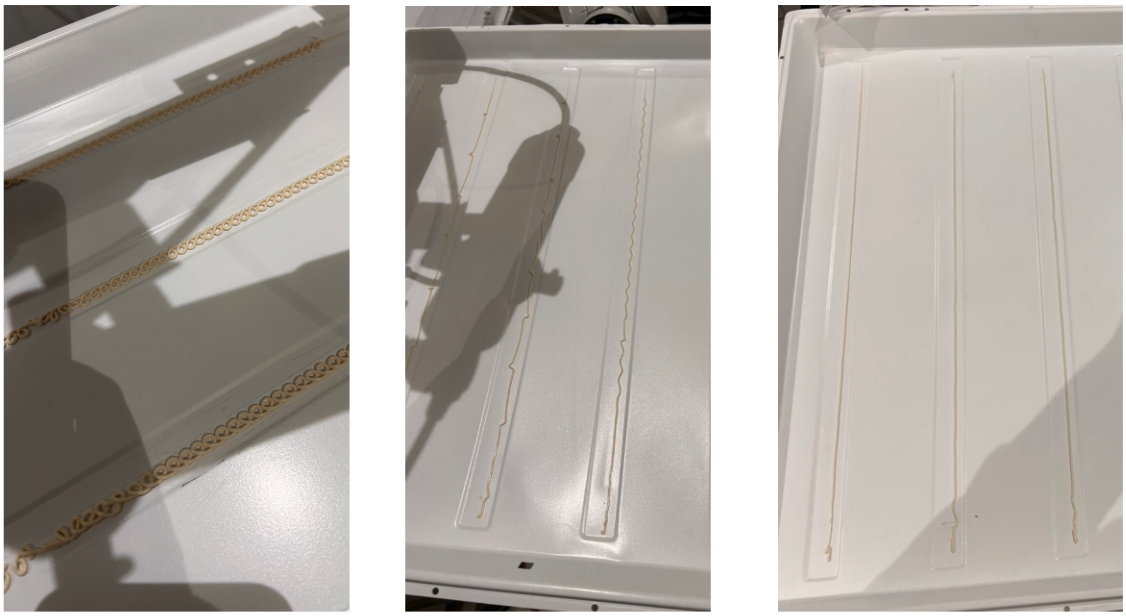


Figura 7.2: Fenómeno de *rizado* en los hilos de adhesivo

Si se calculan las características de la etapa de extracción volumétrica se puede ver con mayor facilidad todos los fenómenos que se están comentando.

Nótese que tanto en la figura 7.1 como en la 7.3 se muestra la velocidad con un color distinto al del resto de los datos. La razón de esto es que el programador ha impuesto esta característica en el sistema; por lo tanto, aunque se considere en los cálculos, no se mide en ningún momento.

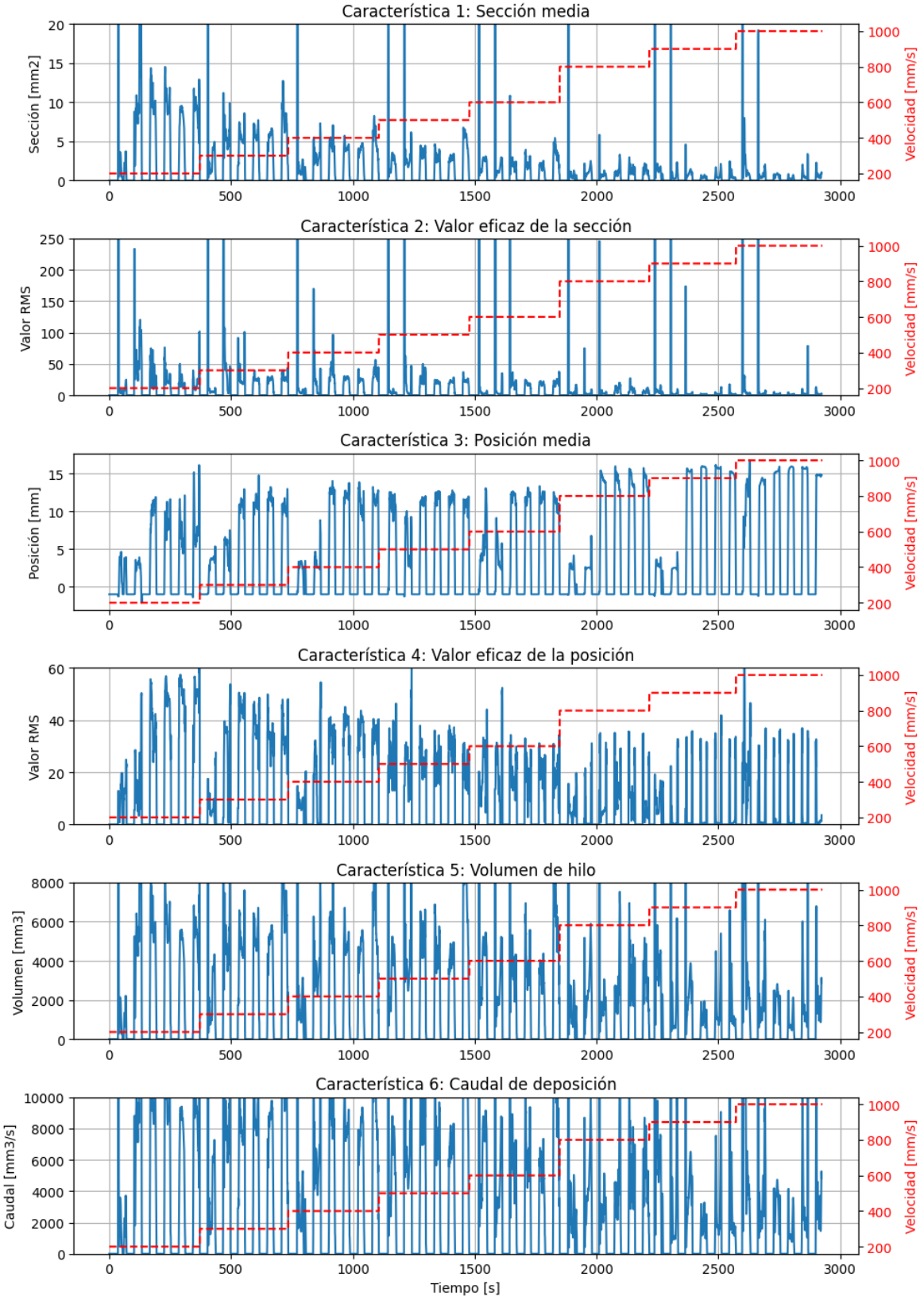



Figura 7.3: Características usadas para entrenar la red neuronal

En la figura 7.3 se encuentran las características extraídas de la etapa de extracción volumétrica. Se puede de esta forma confirmar la hipótesis mostrada en la figura 6.22 acerca del comportamiento del fluido bajo las condiciones impuestas por el sistema robotizado. Pues existe una relación inversamente proporcional entre velocidad de la bandeja y sección depositada de adhesivo por ventana de trabajo, es decir, a mayor velocidad del brazo robótico, menor sección de material depositado.

Es preciso tener en cuenta que si se sigue estrictamente la definición de caudal, este debería ser constante en todos los experimentos. Pero si se revisan los experimentos, a bajas velocidades se puede llegar a considerar que hay un caudal similar en todos ellos, mientras que a partir de $900 mm/s$ el caudal baja drásticamente. Esto da lugar a pensar que el sensor no es capaz de generar esta medida fielmente a velocidades altas, pues los hilos son tan finos que en muchos casos la etapa de extracción de características superficial no llega a identificar un hilo sobre la pista cuando realmente sí lo hay.

Gracias a la medición del caudal también es posible crear un modelo dinámico del sistema siguiendo el planteamiento propuesto en la expresión 3.22. Para ello se han calculado los caudales de deposición a una velocidad constante de $600 mm/s$ a distintas presiones.

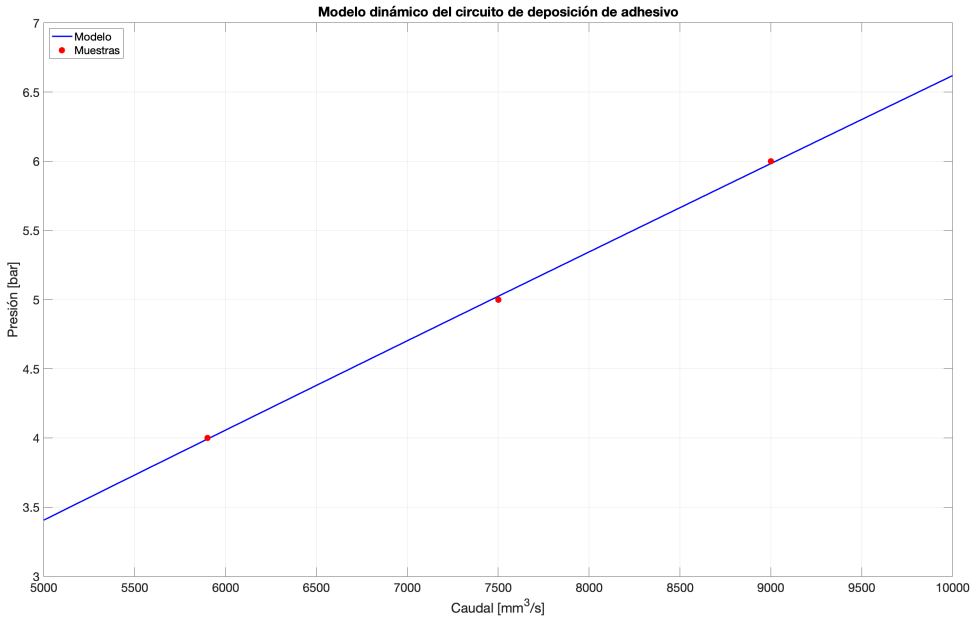



Figura 7.4: Modelo dinámico del sistema de deposición de adhesivo

En la figura 7.4 se puede ver el modelo dinámico obtenido tras seguir el planteamiento comentado al final del capítulo de antecedentes teóricos. Cuya expresión analítica sería la siguiente.

$$\boxed{\Delta P = 9{,}7 \cdot 10^{-4} \cdot Q^{0{,}96}} \tag{7.1}$$

Nótese que el exponente al ser menor que 1, da lugar a deducir que el flujo es laminar, pero debido a que está muy cerca del límite, no se debería descartar que en ciertas situaciones pueda llegar a alcanzarse el flujo turbulento[12]. Además, denotar que la utilidad de este modelo en el entorno de producción se reduce únicamente a la detección de bajadas de presión en la línea de aire comprimido de la instalación. También es preciso anotar que los parámetros pueden variar de un bote a otro, o por las microobstrucciones generadas por entradas no previstas de aire en el sistema. Por lo que el modelo enunciado debería hacerse teniendo en cuenta todas casuísticas comentadas.

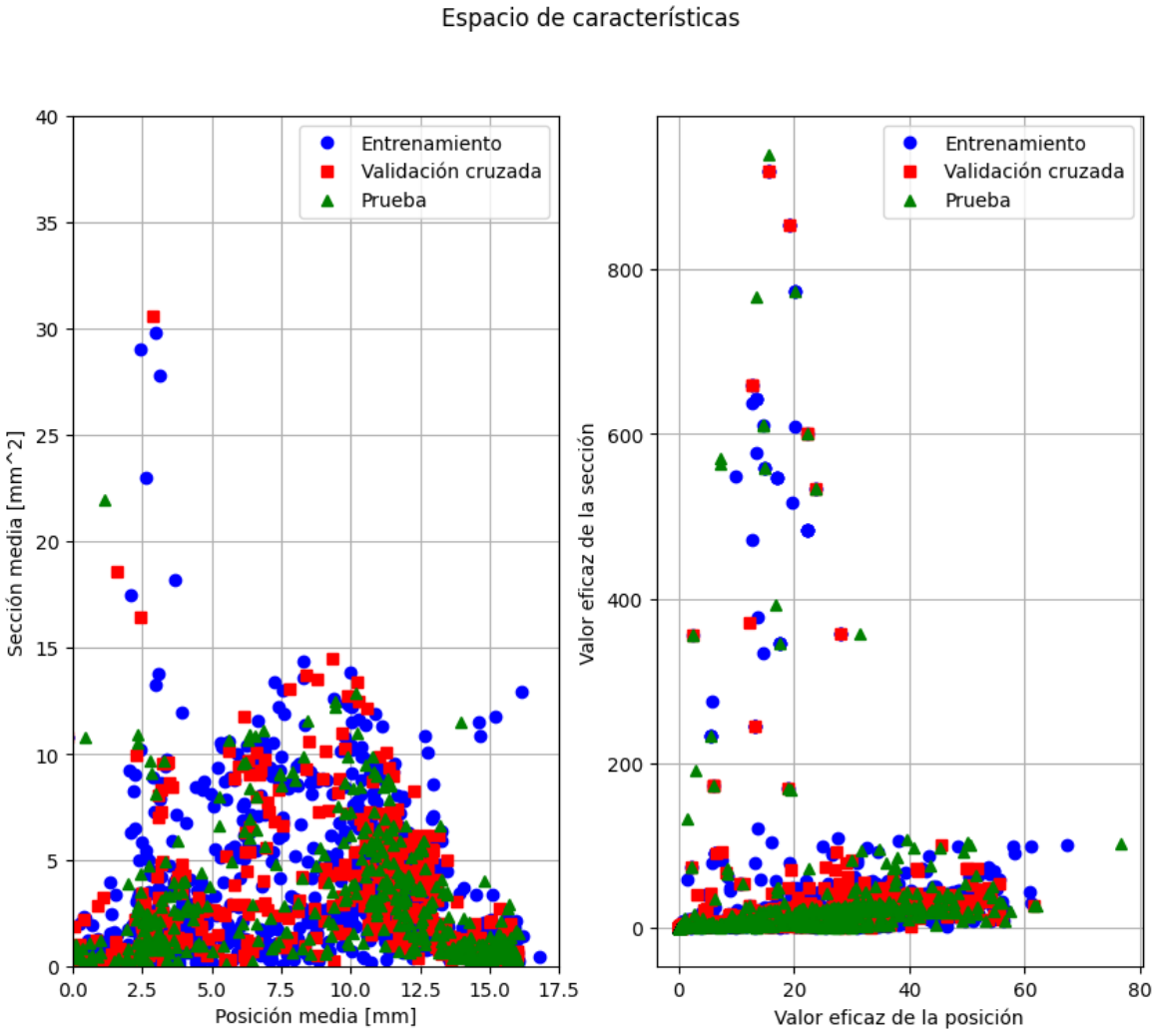



Figura 7.5: Datos de entrenamiento sobre el espacio de características

Volviendo sobre la figura 7.3, se puede observar que hay una relación directa entre los armónicos de posición y el fenómeno de flujo inestable. Por esa razón interesará que dicha

característica tienda a 0. Para el entrenamiento de la red neuronal es necesario etiquetar la información, siendo un 1 lo que se considere un hilo óptimo y un 0 lo que no. Una vez hecho eso, el conjunto de datos debe ser subdividido en los grupos de entrenamiento, validación y test como se comentó en el capítulo 6, y como se muestra en la figura 7.5.

Partiendo de esos subconjuntos de información, se procederán a entrenar los distintos modelos candidatos a ser desplegados en el sistema de detección de hilos óptimos.

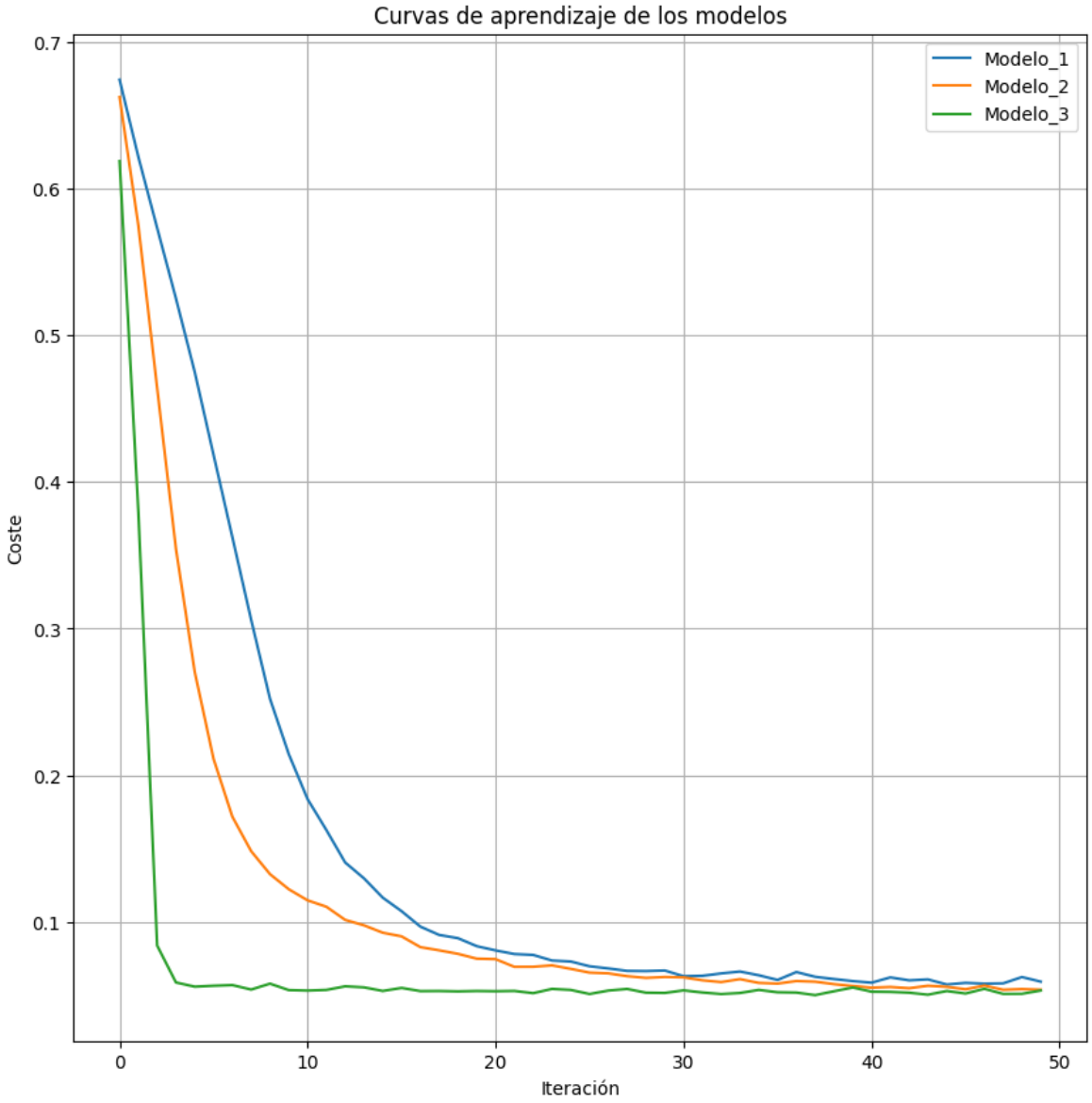



Figura 7.6: Curvas de aprendizaje de diferentes modelos

Como se puede observar en la figura 7.6, no hubo problemas a la hora de entrenar, pues el coste de todos los modelos ha ido reduciéndose con las iteraciones hasta que convergió. Descartando problemas generados por escoger un incorrecto ratio de aprendizaje, o un mal

conjunto de datos de entrenamiento.

Finalmente, se han calculado los errores de entrenamiento y de validación cruzada para todos los modelos y resultaron ser muy parecidos en los tres, eso significa que todos ellos se situarían en la zona de la derecha de la gráfica 6.34. Debido a esto se ha escogido el modelo con menos neuronas, pues eso ayudará a reducir los tiempos de cómputo de predicciones.

Dicho eso, sólo queda usar el conjunto de datos de prueba para confirmar que el modelo se comporta correctamente ante nuevos datos de origen similar. Y como se puede comprobar en la figura 7.7, las predicciones son más que correctas.

Nótese que debido a la alta dimensionalidad de los vectores de características no pueden ser visualizados directamente sobre los ejes cartesianos de 2 y 3 dimensiones. Para ello se pueden emplear técnicas de reducción de la dimensión, cuyo objetivo es encontrar nuevos ejes sobre los que proyectar la información y que al mismo tiempo retengan toda la varianza posible de los datos. En la figura 7.7 se ha empleado la técnica de *Análisis de Componentes Principales (PCA)* que transforma los vectores de 6 dimensiones en otros de 2.

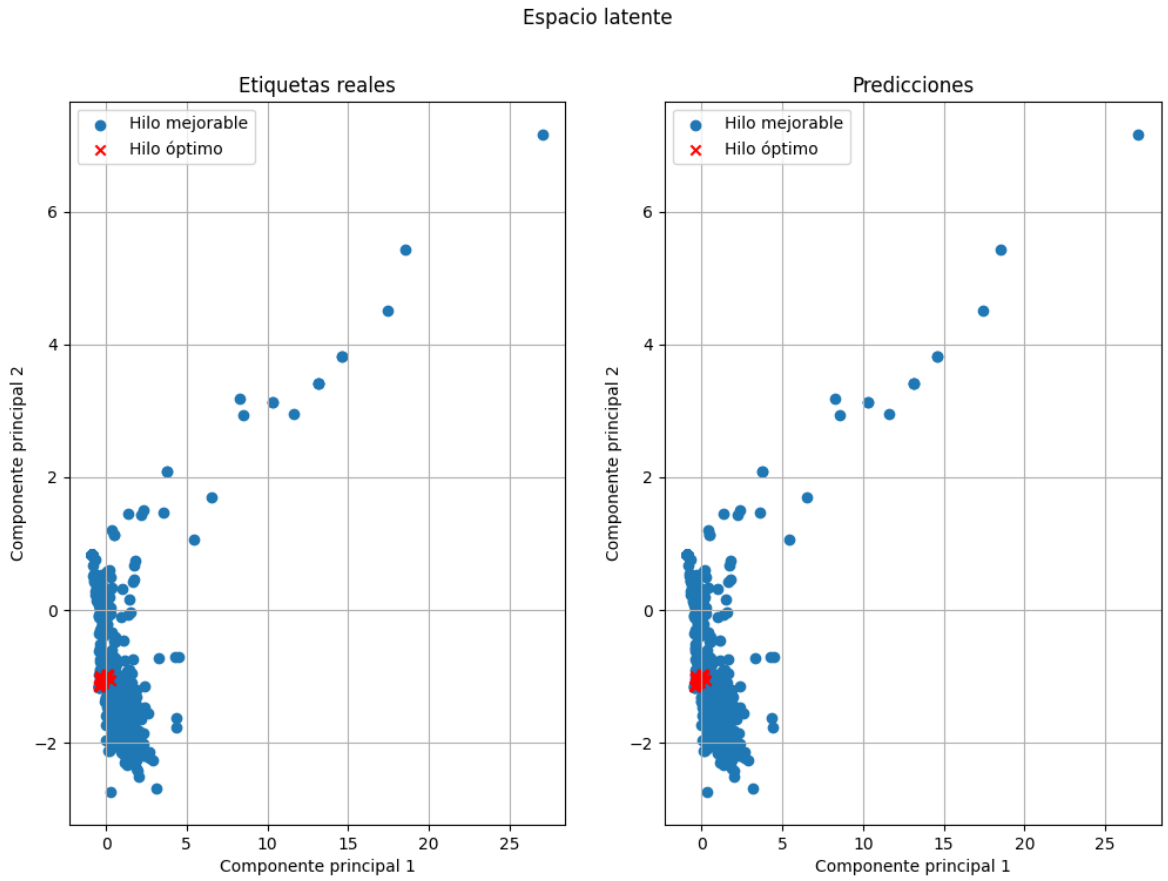



Figura 7.7: A la izquierda, conjunto de test etiquetado por programador. A la derecha, conjunto de test clasificado por RN

Es importante destacar que un diseño basado en el buen conocimiento de las propiedades del fenómeno a estudiar ayudará en gran medida a reducir las iteraciones en el desarrollo del modelo de la inteligencia artificial, además de hacerla mucho más eficiente gracias a las etapas de extracción de características.

Gracias a la metodología de trabajo seguida, la red neuronal ha sido capaz de deducir características óptimas a partir de la información física más relevante del proceso. Es por ello que se ha logrado un error nulo en las predicciones, sujeto a más pruebas. De esta forma se concluye el desarrollo técnico del presente trabajo de fin de grado insistiendo en que el conocimiento a nivel industrial del fenómeno a modelar ayuda enormemente al desarrollo de modelos de inteligencia artificial desplegados en este campo.

# 8. Conclusiones

## 8.1.- Discusión general

En el apartado de conclusiones de este trabajo de fin de grado, se analizarán los resultados obtenidos a partir del prototipo implementado y los experimentos realizados. Se destacarán los logros alcanzados en la aplicación de la inteligencia artificial a la optimización del uso de material en la robótica industrial, subrayando los beneficios observados en términos de eficiencia y sostenibilidad del sistema de montaje. Se evaluará la efectividad de la aplicación de algoritmos de aprendizaje automático para minimizar el desperdicio de material, y se discutirá cómo estos resultados contribuyen al campo de la robótica industrial y la ingeniería de sistemas. Finalmente, se plantearán posibles mejoras y líneas de investigación futuras para optimizar aún más el sistema y ampliar su aplicabilidad en entornos industriales reales.

## 8.2.- Logros obtenidos

Durante la realización del proyecto, se han alcanzado una serie de logros significativos que han contribuido al avance y éxito del mismo. A continuación, se presentan los principales logros obtenidos:

1. **Diseño y Desarrollo del Sistema de Inyección de Adhesivo:** Se logró diseñar y desarrollar un sistema de inyección de adhesivo para una bandeja de 6 pistas. Este sistema incorpora un brazo robótico y un conjunto de dispositivos supervisados por un sensor de perfilometría para dispensar hilos de adhesivo con precisión y eficiencia.

2. **Integración de Componentes Reutilizados:** Se logró integrar componentes de maquinaria descatalogada en el sistema automatizado, permitiendo reducir costos y aprovechar recursos disponibles en el entorno de trabajo.

3. **Selección de la Mejor Solución:** Se evaluaron diversas opciones para cada componente del sistema, considerando sus ventajas e inconvenientes. Se seleccionaron las soluciones más adecuadas para cada caso, asegurando un funcionamiento óptimo del sistema.

4. **Optimización del Proceso de Fabricación:** Se implementaron medidas para optimizar el proceso de fabricación, incluyendo la mejora de la precisión y la eficiencia del sistema de inyección de adhesivo gracias al sensor de perfiles.

5. **Documentación Detallada:** Se elaboró una documentación detallada que describe el diseño, desarrollo e implementación del sistema, facilitando su comprensión y futuras actualizaciones vinculando conceptos académicos con problemáticas recurrentes en el entorno industrial.

6. **Capacitación y Conocimiento Adquirido:** Se adquirió conocimiento en áreas como mecánica de fluidos, programación de robots, funcionamiento de servomotores y PLCs, así como en métodos de control, monitorización de procesos industriales e inteligencia artificial.

7. **Contribución al Avance Tecnológico:** Este proyecto ha contribuido al avance tecnológico en el campo de la automatización industrial, proporcionando una solución innovadora y eficiente para la dispensación de adhesivo en entornos industriales.

8. **Reproducibilidad:** Se elaboró un sistema de supervisión dotado de capacidad para medir volúmenes a partir de nubes de puntos que es extrapolable a cualquier aplicación con condiciones similares.

La integración del sensor no solo impulsa la eficiencia operativa, sino que también contribuye de manera significativa a la sostenibilidad ambiental. Al minimizar el desperdicio de material, se reduce el impacto negativo en el medio ambiente y se promueve un enfoque más responsable hacia la producción industrial. Este enfoque no solo beneficia a la empresa en términos de costos y competitividad, sino que también fortalece su compromiso con la preservación del entorno. Se considera oportuno mencionar que dispensando siempre la cantidad de adhesivo óptima por bandeja en lugar de una arbitraria que pueda variar por factores meramente humanos al usar pistolas de adhesivo convencionales, a la velocidad óptima de dispensado (900 mm/s) y en continuo durante un año, se podrían ahorrar aproximadamente **110000 L** de material, o lo que es lo mismo, **140000** botes de adhesivo, con todas las emisiones que supone su transporte.

## 8.3.- Aportaciones y mejoras futuras

Precediendo a las mejoras futuras se enunciarán las aportaciones realizadas al ámbito de la ingeniería industrial clasificadas por áreas de conocimiento:

1. **Mecánica de fluidos:** Se han obtenido modelos cinemáticos y dinámicos de pseudoplásticos en los botes de adhesivo.
2. **Automatización y robótica industrial:** Se ha programado un robot *IRB 1300* de *ABB* y un autómata programable *CX9020* de *Beckhoff* para el diseño de la célula de montaje de luminarias usando técnicas estandarizadas de programación como la guía GEMMA.
3. **Comunicaciones Industriales:** Se ha implementado una librería para efectuar comunicaciones con la controladora del robot desde cualquier dispositivo que se encuentre conectado a su misma red.
4. **Inteligencia Artificial:** Se ha diseñado y entrenado un modelo de aprendizaje automático supervisado para labores de identificación de hilos de adhesivo óptimos en función de medidas superficiales, frecuenciales y volumétricas deducidas a partir del uso de un sensor capaz de medir perfiles.

Para visualizar el resultado de este proyecto se anima al lector a acceder mediante el código QR mostrado en la figura 8.1 al contenido audiovisual que sirve como demostración del sistema en funcionamiento.[1]

Figura 8.1: Código QR para acceder al vídeo demostración del proyecto

[1]Normagrup Technology se reserva todos los derechos de autor sobre el contenido audiovisual presentado en este vídeo.

Si se recuerda lo comentado en el estado del arte, el problema de la deposición de este tipo de materiales sólo se sabe abordar en lazo abierto, pero si existen variaciones en los parámetros del sistema, este no va a saber adaptarse, dándose la casuística de que suceden ciertos fenómenos que precisamente varían estos parámetros:

- A mayor distancia desde la boca de deposición a la pista, mayor inestabilidad se generará en la deposición del hilo de adhesivo.
- Al cambiar el bote pueden cambiar los parámetros reológicos del fluido, haciendo que cambie su velocidad óptima.
- Al entrar en contacto pequeñas cantidades de aire en el circuito se generan microobstrucciones no medibles que aumentan las pérdidas de carga en el sistema.

Esto desemboca en una problemática recurrente en el paradigma de la fabricación aditiva, pues estos sistemas están diseñados en lazo abierto. Por ello como mejora futura se propone diseñar un sistema en lazo cerrado adaptativo que esté continuamente aprendiendo de la reología del material para buscar en todo momento la velocidad óptima de deposición. Para conseguir ese fin, dentro de la disciplina del aprendizaje automático existe un área que consiste en entrenar al sistema mediante un sistema de recompensas virtuales.

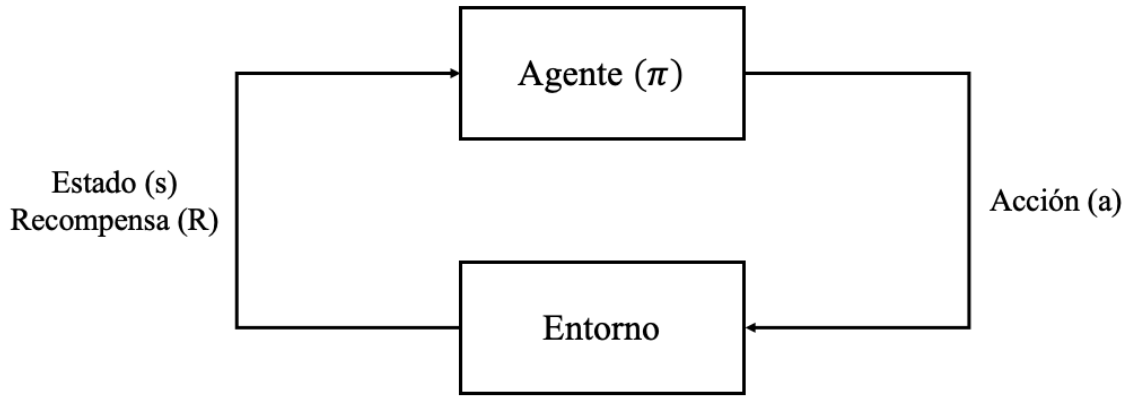



Figura 8.2: Relación Agente-Entorno según Teorema de Decisión de Markov

El diagrama presente en la figura 8.2 es llamado el Teorema de Decisión de Markov, y enuncia lo siguiente: *"El futuro depende únicamente de donde estás ahora, no de donde vienes"*. Pretende decir que si el agente puede actuar sobre el entorno y esas acciones son medibles, pueden ser recompensadas. Esta técnica llamada aprendizaje por refuerzo emplea **espacio de estados**, y si se reestructura todo lo comentado siguiendo la filosofía empleada en la ingeniería de control, el problema a resolver tendría un aspecto similar al mostrado en la figura 8.3.

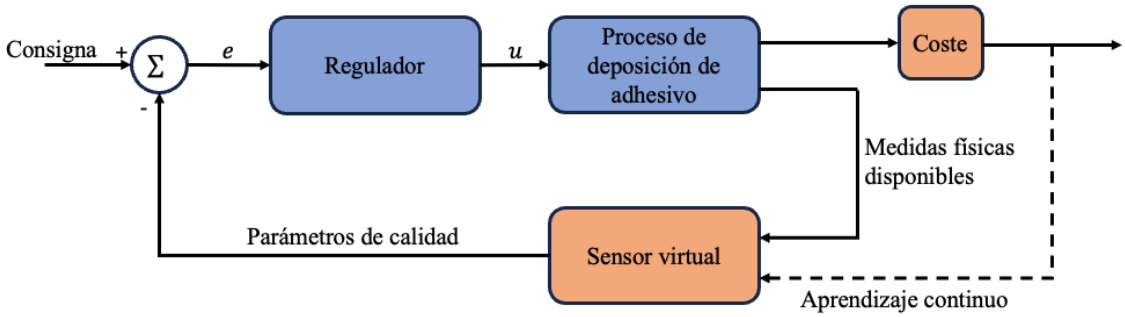



Figura 8.3: Diagrama de bloques de sistema de control basado en experimentos

Es por ello que con este trabajo de fin de grado se pretende abrir una línea de investigación basada en el modelado dinámico de sistemas de deposición de adhesivo, que en un futuro pueda ser extrapolable a cualquier sistema de características similares. En próximas publicaciones se propondrán algoritmos más avanzados de aprendizaje profundo usados para cerrar lazos de control basados en experimentos (*Data-Driven Control Loops*).

# Bibliografía

[1] ABB, "Especificaciones del producto. Línea C de Omnicore," 6 2022.

[2] SMC, "Electric Actuator."

[3] ABB, "Robot Web Services."

[4] Baumer, "Catálogo - OXP200-R05C.004," 8 2023.

[5] Ignacio Díaz Blanco, "Introducción al Deep Learning," 2023.

[6] Gene F. Franklin, J. David Powell, and Abbas Emami-Naeini, *Feedback Control of Dynamic Systems*. Pearson, 8 ed., 2020.

[7] Q. Fang, G. Xiong, M. Zhou, T. S. Tamir, C.-B. Yan, H. Wu, Z. Shen, and F.-Y. Wang, "Process Monitoring, Diagnosis and Control of Additive Manufacturing," *IEEE Transactions on Automation Science and Engineering*, vol. 21, no. 1, pp. 1041–1067, 2024.

[8] R. B. Bird, W. E. Stewart, and E. N. Lightfoot, *Fenomenos de transporte*. Limusa Wiley, 2006.

[9] H K Versteeg and W Malalasekera, *An Introduction to Computational Fluid Dynamics*. Pearson, 2 ed., 2007.

[10] M. G. Ordorica Morales, "Fenomenos de Transporte-Cantidad de Movimiento,"

[11] R.C. Hibbeler, *Fluid Mechanics in SI Units*. Pearson, 2 ed., 2021.

[12] A. Okon and F. Udoh, "Pressure Drop-Flow Rate Profile of Some Locally Formulated Drilling Fluids Using Bingham Plastic and Power Law Rheological Models," *Indian Journal of Engineering*, vol. Volume 2, pp. 10–17, 6 2013.

[13] ABB, "Manual de referencia técnica. Instrucciones, funciones y tipos de datos de RAPID," 11 2023.

[14] ABB, "Manual del operador. RobotStudio," 2 2024.

[15] IBM, "¿Qué es una API REST?."

[16] Genaro Cuofano, "Comportamiento Emergente," 2 2024.

[17] S. C. Chapra and R. P. Canale, *Numerical methods for engineers*. McGraw-Hill Higher Education, 2010.

[18] Steven L. Brunton and J. Nathan Kutz, *Data-Driven Science and Engineering*. Washington: Cambridge University Press, 2 ed., 2022.