



Horizontalización y automatización de robot SCARA paralelo

de Lajonquiere, Iván

Universidad Nacional de Rosario
Facultad de Ciencias Exactas, Ingeniería y Agrimensura
Escuela de Ingeniería Mecánica
Rosario, Argentina
28 de octubre de 2025

Horizontalización y automatización de robot SCARA paralelo

de Lajonquiere, Iván

Proyecto Final de Ingeniería Mecánica presentado como requisito parcial para optar al título de:
Ingeniero Mecánico

Directores:

Ing. Guillermo Rodríguez

Ing. Pedro Sismondi

Ing. Juan Sismondi

Universidad Nacional de Rosario
Facultad de Ciencias Exactas, Ingeniería y Agrimensura
Escuela de Ingeniería Mecánica
Rosario, Argentina
28 de octubre de 2025

Agradecimientos

Quiero agradecer al profesor Ing. Guillermo Rodriguez por brindarme la oportunidad de realizar un proyecto que me permitió integrar todos los conocimientos aprendidos a lo largo de este camino, pudiendo devolverle a la facultad una pequeña parte de todo lo que me brindó.

A los profesores Ing. Pablo Demartini e Ing. Daniel Ades por el acompañamiento durante todo el proyecto.

A mis padres, por apoyarme de forma incondicional desde el día en que tomé la decisión de seguir esta maravillosa carrera, celebrando cada logro alcanzado y acompañandome en los días malos.

A mis compañeros de carrera, que más que compañeros son amigos. Con los cuales compartimos horas y horas de estudio, alegrías y tristezas, para llegar al final del camino.

A mi novia, quien me acompañó durante todo este proceso, apoyandome en los días que sentía que no podía y festejando cada pequeño avance logrado.

A mis amigos de toda la vida.

Por último, quiero hacer una mención en la memoria del Ing. Dario Carlovich, maestro particular cuando realicé la secundaria. Ya que el fue uno de los primeros que me incentivó y creyó en mi para que siguiera esta increíble carrera.

Gracias a la universidad pública, por darme la oportunidad de formarme como profesional.

Resumen

Hoy en día en la industria es muy común la utilización de automatización y robots para realizar diferentes tareas donde las mismas van desde operaciones de corte, soldadura, ensamblaje hasta transporte de diferentes objetos. Suelen ser muy utilizados los conocidos robots SCARA de cadena abierta pero a su vez también los robots SCARA paralelos los cuales conforman una cadena cerrada formando un mecanismo de 5 barras. Estos últimos, poseen un gran potencial para ser utilizados en aplicaciones de "pick and place", pero además pueden ser utilizados para realizar trayectorias determinadas permitiendo hacer dibujos o hasta incluso realizar cortes por láser implementando un cabezal en su punto efector si se quisiese.

En el presente proyecto, se logró la operatividad del robot SCARA paralelo que se encuentra en el Fablab de la EIM mostrando el potencial del mismo para determinadas aplicaciones de manera didáctica. Comenzando por el diseño de una estructura portante que sostiene el conjunto completo, verificando la resistencia estructural del robot y ampliando la automatización, logrando realizar diferentes trayectorias. Por último, se concluye este informe con un análisis de las trayectorias realizadas y cómo las mismas se relacionan con la forma de control de movimiento del robot.

Abstract

Nowadays, automation and robotics are widely used in industry to perform a variety of tasks, ranging from cutting, welding, and assembly operations to the transportation of different objects. While open-chain SCARA robots are commonly employed, parallel SCARA robots are also of great relevance, consisting of a closed-chain mechanism with five bars. These robots are particularly well suited for pick-and-place applications, but they can also be programmed to execute predefined trajectories, which enables them to draw patterns or even perform laser cutting when equipped with an appropriate end-effector.

In this project, the parallel SCARA robot located in the EIM FabLab was successfully brought into operation, demonstrating its potential for specific applications. The work began with the design of a supporting frame to hold the entire assembly and continued with the expansion of its automation capabilities, allowing the execution of different trajectories. Finally, this report concludes with an analysis of the performed trajectories and their relation to the robot's motion control approach.

Índice

Resumen	I
Abstract	II
Introducción	1
1. Introducción	1
1.1. Estado del arte	1
1.2. Problemática y estado inicial del proyecto	7
1.3. Objetivos y propósitos	8
2. Marco Teórico	10
2.1. Definiciones	10
2.2. Características robot SCARA paralelo	11
2.3. Espacio de trabajo.	12
2.4. Singularidades.	13
2.4.1. Singularidad serie	13
2.4.2. Singularidad paralela	14
2.5. Modos de trabajo	15
2.6. Cinemática	15
2.6.1. Cinemática inversa	16
2.6.2. Cinemática directa	18
3. Desarrollo	21
3.1. Optimización tablero eléctrico	21
3.2. Soporte	23
3.3. Simulaciones	30
3.3.1. Simulación estática	30
3.3.2. Análisis de vibraciones	39
3.3.3. Pandeo	40
3.4. Cálculos tensiones Von Mises soporte	40
3.5. Cálculo carga máxima brazos	45
3.6. Simulación de movimientos	53
3.6.1. Selección de trayectorias	53

3.6.2. Pasaje de ángulos en grados a pulsos	56
3.6.3. Gráficas trayectorias y evolución del movimiento	57
3.7. Automatización	63
3.7.1. Función trayectoria	65
3.7.2. Función logopoli	66
3.7.3. Función homming	68
3.7.4. Programa principal	69
3.7.5. Menú	74
3.8. Análisis velocidades angulares.	75
3.9. Construcción y montaje.	79
4. Pruebas experimentales	80
4.1. Pruebas en vertical	81
4.2. Pruebas en horizontal	87
4.3. Análisis resultados experimentales	93
5. Conclusiones finales.	94
Referencias Bibliográficas	96
Índice de figuras	97
Índice de Tablas	102
6. Anexos	103
6.1. Anexo A: Programas Matlab	103
6.2. Anexo B: Programas IPOSPlus	103
6.3. Anexo C: Planimetría	103

1. Introducción

Los automatismos con robots conforman una parte muy importante en el campo industrial, ya que nos permiten realizar tareas repetitivas y en un menor tiempo en que lo haría una persona. Se reemplazan los errores que pueden llegar a cometer los humanos, y además, pueden realizar actividades donde se podría poner en riesgo su seguridad. Por esto último, los mismos son de gran ayuda en el ambiente industrial.

Los robots de tipo paralelo han sido foco de estudio y se han desarrollado ampliamente en los últimos años, desde un punto de vista tanto teórico como de aplicaciones prácticas.

1.1. Estado del arte

En 1934, Willard Pollard Jr. presentó una patente para una máquina de pintura en aerosol, destinada a la industria automotriz. El mismo nunca fue construido. El dispositivo cuenta con 5 grados de libertad, en donde la plataforma móvil se une a una plataforma fija mediante 3 cadenas cinemáticas o piernas. [[3]]

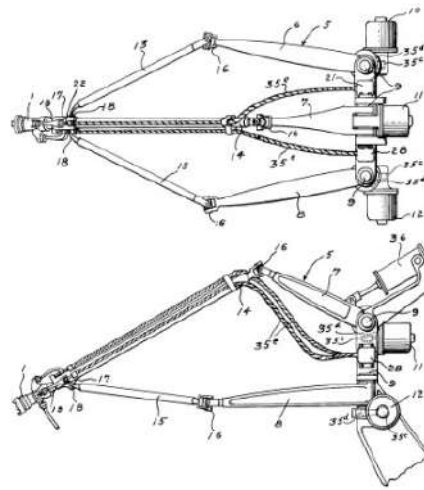


Figura 1: Robot paralelo para pintura [3]

En la década de los 50 Eric Gough un ingeniero automotriz trabajando en la fábrica de neumáticos de la Ford Dunlop en Birmingham, Inglaterra, desarrolla una máquina Universal de pruebas de neumáticos. La plataforma cumplía la función de probar mecánicamente neumáticos mediante la aplicación de cargas combinadas. La Figura (2) muestra una imagen del dispositivo utilizado por la empresa. Su diseño presenta una forma de octaedro donde cada cadena cinemática es impulsada por actuadores lineales. El mecanismo presenta 6 grados de libertad. [[3]]



Figura 2: Máquina de pruebas ideada por Eric Gough [3]

En 1965, D. Stewart introduce un robot paralelo de 6 grados de libertad basándose en la plataforma diseñada por Gough. Esta plataforma posee seis actuadores lineales que se fijan a una placa base y luego se vinculan en tres puntos de montaje a una placa superior, estos mismos se encuentran vinculados a las placas mediante juntas universales. Es de destacar que la plataforma Stewart fue propuesta para aplicaciones de simulador de vuelo. La figura (3) muestra el robot basado en el concepto de Stewart desarrollado para un simulador de vuelo de Lufthansa. [[3]]



Figura 3: Simulador de vuelo Lufthansa basado en plataforma de Stewart [[3]]

Otra de las aplicaciones prácticas de los robots paralelos se encuentra en el área de desarrollo de centros de mecanizado. Los primeros dispositivos diseñados para mecanizados utilizaron un robot paralelo del tipo plataforma Stewart (6 grados de libertad) para realizar operaciones de mecanizado en 5 ejes. Una de las grandes desventajas de estos centros de mecanizados es que poseen muy poco espacio de trabajo, además de ser costosos en componentes y dificultosos en control.

El centro de mecanizado comercial Giddings and Lewis' Variax Hexacenter posee como base una máquina del tipo hexápodo (6 grados de libertad) con un diseño particular donde las piernas que unen la plataforma móvil a la fija se cruzan. La figura (4) muestra una imagen del robot Variax, en ella se puede observar cómo la plataforma móvil va unida a la fija por medio de 6 piernas donde agrupadas en pares, sus piernas se cruzan. En esta misma línea de robots basados en la configuración Hexápodo se tiene el Okuma PM600 presentado en la figura (5) cuya diferencia respecto a la configuración del Variax subyace en que las piernas del robot están conectadas en configuración hexagonal sin cruzarse. [3]



Figura 4: Robot paralelo centro de mecanizado Variax [3]



Figura 5: Robot paralelo centro de mecanizado Okuma PM600 [3].

Los robots paralelos también tienen presencia en el sector automotriz que requiere equipos muy flexibles en cuanto a manufactura donde se busque un equilibrio entre desempeño y costos operacionales. Las estructuras paralelas juegan un papel importante en mecanizados muy complejos como el mecanizado de camisas para blocks de motor. [1]. En la figura (6) podemos visualizar un robot paralelo de 3 grados de libertad donde se agrega un cuarto grado de libertad que permite la rotación del cabezal que sostiene el portaherramienta.



Figura 6: Robot paralelo para mecanizado de blocks de motor [1].

El primer robot comercial y posiblemente uno de los más exitosos en implementación industrial lo constituye el robot Delta desarrollado a partir de la década de los 80 en la École Polytechnique Fédérale de Lausanne (Suiza) por Reymond Clavel. El robot fue desarrollado partiendo de la idea de desarrollar un robot para manipular objetos de bajo peso a altas velocidades. La particularidad del robot Delta es que la plataforma móvil va unida a la base mediante 3 piernas donde cada pierna presenta un mecanismo de paralelogramo que permite balancear el centro de masa de cada una de ellas. Las piernas se unen a la base fija mediante pares o juntas universales (U). La Figura (7) muestra la representación esquemática del robot Delta donde el movimiento de las piernas externas aporta 3 grados de libertad que permiten posicionar la plataforma móvil en el espacio. Generalmente, la plataforma superior va fija a la bancada, mientras que la plataforma inferior es la móvil donde va ubicado la herramienta. Puede también agregarse un cuarto grado de libertad utilizando una junta cardánica como se observa en la figura (7) o implementando directamente sobre la plataforma móvil un actuador giratorio. [3]. En la figura (8) puede verse un robot Delta comercial fabricado por Kawasaki el cual cumple la función de pick and place.

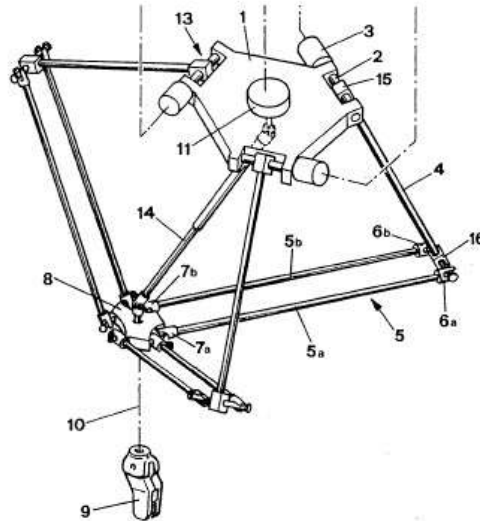


Figura 7: Robot delta desarrollado por Clavel [3].



Figura 8: Robot delta YF002N desarrollado por Kawasaki Robotics [12].

También, es de destacar que el robot serial equivalente para este tipo de operaciones lo constituye el robot SCARA (Selective Compliant Assembly Robot Arms) que presenta 4 grados de libertad. Un robot SCARA permite posicionar el objeto a trasladar o herramienta incorporada al mismo en el espacio cartesiano, además, también puede realizar una rotación [M. Díaz-Rodríguez and et al, Aplicación de los robot paralelos (2018), En Manipuladores paralelos: Síntesis, análisis y aplicaciones (Pereira: Universidad Tecnológica de Pereira)]. En la figura (14), puede verse un robot SCARA serie de la marca Mitsubishi Electric el cual se destina a dicha aplicación.



Figura 9: Robot SCARA serie [4].

Para el caso de los robots SCARA paralelos hoy en día no hay una aplicación industrial, pero si se comercializa este tipo de robots para aplicaciones a nivel didáctico. Existen varios desarrollos como el realizado por la empresa Medademic, también la empresa Mitsubishi Electric desarrolló este tipo de robot y muchos proyectos a nivel aficionado o de investigación en universidades.

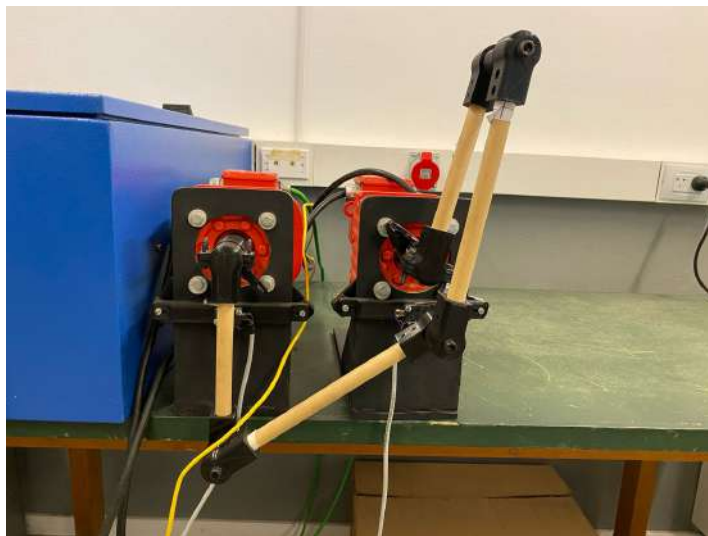


(a) SCARA paralelo Dexter desarrollado por Mecademic [8]. *(b) SCARA paralelo desarrollado por Mitsubishi Electric [8].*

Figura 10: Robots SCARA paralelos.

1.2. Problemática y estado inicial del proyecto

En el Fablab ubicado en la escuela de ingeniería mecánica correspondiente a la Facultad de Ciencias, Exactas, Ingeniería y Agrimensura de la Universidad Nacional de Rosario se encuentra un robot SCARA paralelo, como se puede observar en la figura (11a). El mismo es accionado por dos servomotores de la marca SEW Eurodrive los cuales cuentan con un tablero de control (figura (11b)), compuesto por interruptores automáticos y sus respectivos controladores. El control de los servomotores se logra a partir del Software MOVITOOLS perteneciente a SEW Eurodrive, el cual permite la creación de diferentes proyectos de automatización.



(a) SCARA paralelo EIM [Propia].



(b) Tablero de control del robot [Propia].

Figura 11

Como se mencionó en el resumen al inicio del presente informe, el robot está conformado por un mecanismo de cinco barras el cual posee dos grados de libertad, permitiendo el posicionamiento de la cadena cinemática con dos variables independientes, estas mismas se corresponden con los ángulos de giro de los servomotores.

Actualmente el robot se encuentra operativo y la operatividad del mismo se logró en una primera etapa la cual se desarrolló en [2]. En la misma se logró poner en marcha los servomotores, investigando el software necesario para automatizarlos. Se realizó un programa base con un menú interactivo en donde el usuario puede introducir cuatro combinaciones de ángulos y así posicionar el robot en cuatro puntos diferentes. Además, se puede introducir si los movimientos se realizan de forma automática o paso a paso. También indica la posición actual y el modo de trabajo del robot. Además de la puesta en marcha, se realizó el diseño del robot.

La problemática principal a resolver en este proyecto trata de que el robot no dispone de una

función específica, es decir, solamente podemos ver cómo se mueve. A raíz de esto, surge la necesidad de poder encontrarle una finalidad y poder demostrar de manera didáctica el gran potencial que tiene este tipo de robot para diferentes aplicaciones. Debido a que estos robots son muy utilizados para aplicaciones de pick and place, como ya se mencionó en secciones anteriores, se plantea que el mismo pueda ser adaptado para dicha tarea. Como primer paso para esto, en este proyecto la solución a dicha problemática se basa en realizar la horizontalización del robot mediante la elaboración de un soporte y la ampliación de la automatización, donde se buscará lograr con la misma que realice diferentes movimientos los cuales permitan generar diferentes trayectorias.

1.3. Objetivos y propósitos

En función de lo planteado en la sección anterior, se busca lograr los siguientes objetivos principales:

- Optimizar el tablero de comando de los servomotores para aumentar el espacio disponible en la mesa de trabajo.
- Lograr la disposición horizontal del robot para adecuarlo a su nueva función.
- Automatizar los movimientos del robot necesarios para moverse sobre el plano XY y poder transportar los objetos.
- Implementar un actuador en el punto efector para agregar un movimiento en el eje Z que permita recolectar los objetos.
- Combinar los movimientos automatizados del robot con el movimiento del actuador en el eje Z de manera tal que al ubicarse en el punto recolecte el objeto y lo lleve al punto de destino.

De los cuales se desprenden los siguientes propósitos:

- Lograr la optimización del tablero de comando de los servomotores para aumentar el espacio disponible en la mesa de trabajo.
- Lograr que el usuario tenga fácil acceso y visualización al tablero de comando.
- Lograr un óptimo enfriamiento de los controladores ubicados en el interior del gabinete.
- Lograr la disposición horizontal del robot para adecuarlo a su nueva función.
- Diseñar la estructura portante de manera tal que lugar de trabajo disponible permita al robot aprovechar la totalidad de su espacio de trabajo.
- Lograr que la estructura portante garantice la estabilidad del robot y no afecte la precisión del mismo durante el funcionamiento.
- Automatizar los movimientos del robot necesarios para moverse sobre el plano XY.

- Permitir que el usuario pueda determinar las trayectorias a realizar mediante la interfaz hombre-máquina.
- Lograr un transporte rápido y eficiente.
- Implementar un soporte que sostenga el fibrón que realizará los dibujos.
- Minimizar las interferencias mecánicas del robot para maximizar el aprovechamiento de su espacio de trabajo.
- Obtener un robot didáctico que permita a los estudiantes verlo en funcionamiento, mostrando la potencialidad del trabajo interdisciplinario en la industria y una de las tantas aplicaciones que pueden tener los robots paralelos.

2. Marco Teórico

2.1. Definiciones

Eslabón: es un cuerpo rígido (supuesto) que posee por lo menos dos nodos que son puntos de unión con otros eslabones. [10]

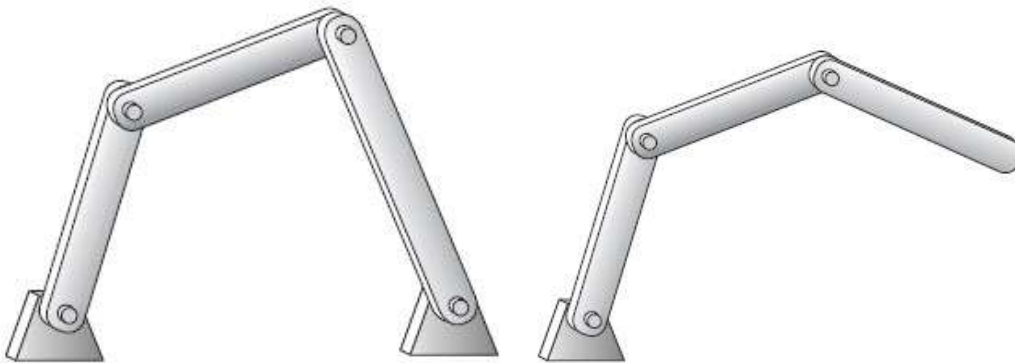
Junta: es una conexión entre dos o más eslabones (en sus nodos), la cual permite algún movimiento, o movimiento potencial, entre los eslabones conectados. [10]

Cadena cinemática: ensamble de eslabones y juntas interconectados de modo que produzcan un movimiento controlado en respuesta a un movimiento suministrado. [10]

Grado de libertad: número de entradas que se necesita proporcionar para crear una salida predecible o también número de coordenadas independientes requerido para definir su posición. [10]

Cadenas cinemáticas abiertas: son cadenas cinemáticas las cuales un eslabón que la compone posee un nodo libre. [10]

Cadenas cinemáticas cerrada: son cadenas cinemáticas las cuales ningún eslabón posee un nodo libre. [10]



(a) Cadena cinemática abierta [10]. (b) Cadena cinemática cerrada [10].

Figura 12

2.2. Características robot SCARA paralelo

Los robots poseen una gran cantidad de configuraciones interviniendo diferentes términos para describirlos como por ejemplo número de cadenas cinemáticas, tipos de cadenas cinemáticas, restricciones, etc., por lo cual es complejo obtener una clasificación sistemática para los mismos según su estructura. Sin embargo, estos robots pueden ser clasificados de forma general según como esté conformada su cadena cinemática: tipo serie y tipo paralelo. En este informe, se hará hincapié en el desarrollo del automatismo y horizontalización de un robot SCARA paralelo, por lo cual se presentarán fundamentos teóricos del mismo, destacando sus principales características.

El robot SCARA paralelo resulta de la unión de dos robots de tipo SCARA serie en el punto donde se ubica el efector final, se forma a partir de dos cadenas cinemáticas en serie abiertas. Esto forma un mecanismo de 5 barras (o eslabones), cuatro corresponden a los brazos y una a la bancada. El robot posee 2 grados de libertad, por lo cual se necesitan dos variables independientes para definir la posición final del robot. Los brazos que se unen entre sí para formar el punto efector se conocen como brazos distales, ya que se no se encuentran vinculados directamente a los actuadores, estos mismos poseen un movimiento de rototraslación. Los brazos acoplados a los actuadores son conocidos como brazos proximales y los mismos poseen un movimiento de rotación pura. Los brazos se unen entre sí mediante articulaciones las cuales se pueden considerar como juntas de revoluta o pasador. Al estar compuestos por varias cadenas cinemáticas se obtienen estructuras con mayor rigidez. Además, poseen una capacidad de carga mayor a las estructuras en serie ya que esta se reparte sobre las diferentes cadenas cinemáticas logrando así una mayor precisión en los trabajos realizados. Este tipo de robots hacen posible reducir el volumen de los eslabones y, como resultado, permite el uso de actuadores de menor capacidad, ya que se requerirá mover menor masa, además las fuerzas inerciales durante el movimiento también serán menores.

En síntesis, se obtienen las siguientes ventajas y desventajas asociadas a robots paralelos:

Ventajas:

- Gran rigidez la cual les permite realizar un trabajo más preciso comparado con un robot en serie.
- Mayor capacidad de carga, la carga ubicada en el punto efector se reparte entre las diferentes cadenas cinemáticas en serie que conforman al robot paralelo.

Desventajas:

- Cinemática mucho más compleja comparados con los robots en serie, los robots paralelos poseen singularidades que dificultan el control y poseen una construcción del modelo dinámico mucho más compleja.
- Poseen un espacio de trabajo más pequeño y difícil de definir.

2.3. Espacio de trabajo.

El espacio de trabajo de un robot se lo define como el conjunto de puntos que pueden ser alcanzados por el efector final del mismo. En general, los robots paralelos poseen un menor espacio de trabajo que un robot serie de tamaño similar (hacer ejemplo en solid), este espacio de trabajo puede ser segmentado por las singularidades que existen en el robot. Para el caso de un SCARA paralelo, el espacio de trabajo del mismo se puede determinar realizando la intersección entre dos circunferencias cuyos centros se ubican en la posición de los motores y poseen un radio equivalente a sumar las distancias de los brazos de cada cadena serie que lo conforma. Entonces, el espacio del robot SCARA paralelo quedará conformado por el área formada por la intersección de las dos circunferencias.

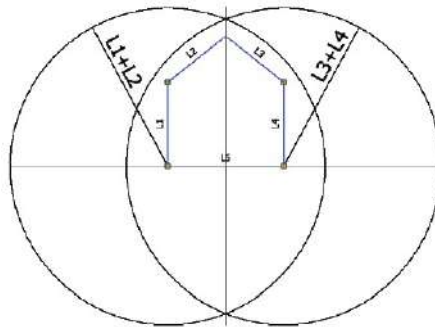


Figura 13: Determinación espacio de trabajo [9].

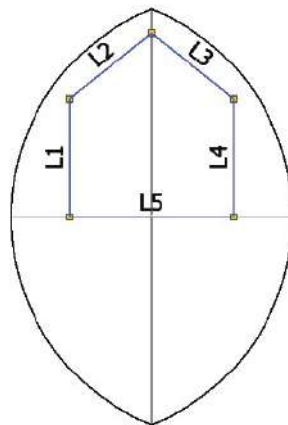


Figura 14: Espacio de trabajo [9].

2.4. Singularidades.

Las singularidades de un robot SCARA paralelo son configuraciones en donde la rigidez inherente del robot cambia repentinamente. El mismo posee dos tipos de singularidades.

2.4.1. Singularidad serie

La singularidad serie es una disposición que adopta el robot donde el efector final del mismo pierde un grado de libertad, es decir, con una sola variable independiente podemos posicionar el robot ya que una cadena “arrastrará” a la otra a posicionarse. Este tipo de singularidad se da al alcanzar la frontera del espacio de trabajo como lo ilustra la figura (24) o al cambiar entre modos de trabajo, ya que se produce cuando se alinean el brazo distal y proximal de la cadena cinemática en serie que lo conforman como puede observarse en la figura (15).

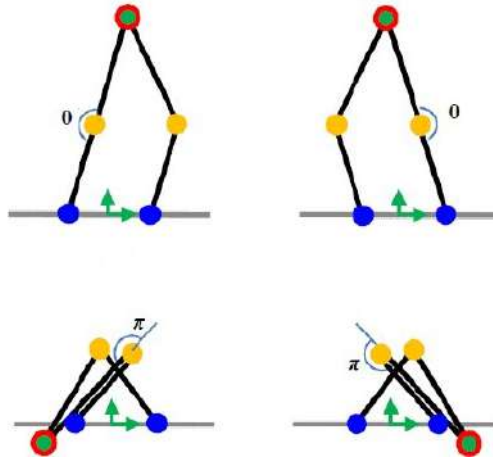


Figura 15: Singularidad serie [5].

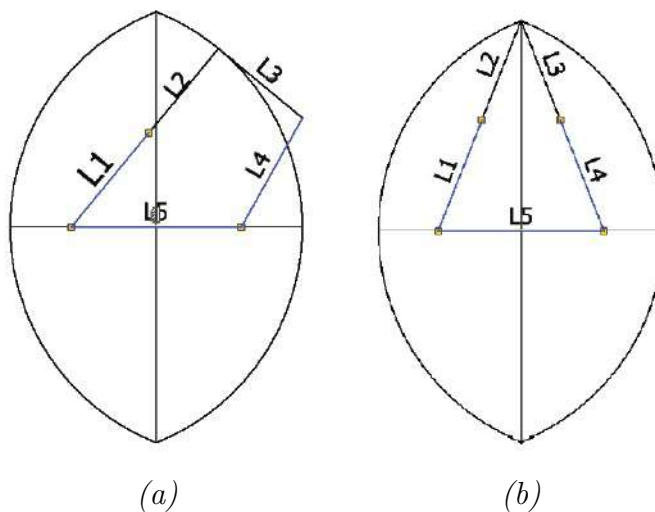


Figura 16: Singularidades serie al alcanzar la frontera del espacio de trabajo [9].

2.4.2. Singularidad paralela

La singularidad paralela es una disposición que adopta el robot donde el efector final gana un grado de libertad, haciendo que este pueda moverse en una dirección determinada por más que los brazos proximales se encuentren en una posición determinada. Cualquier fuerza y/o torque que se ejerza sobre el mismo hará que este se mueva. Este tipo de singularidad hace que se pierda rigidez y precisión en el efector final, perdiendo control sobre el mismo. La misma se encuentra dentro del espacio de trabajo del robot por lo que es de suma importancia tenerla en cuenta a la hora de diseñar el robot y pensar sus movimientos dentro del mismo. Estas se generan cuando se alinean los brazos distales, ya sea formando un ángulo de 0° o 180° entre sí, como puede observarse en la figura (17).

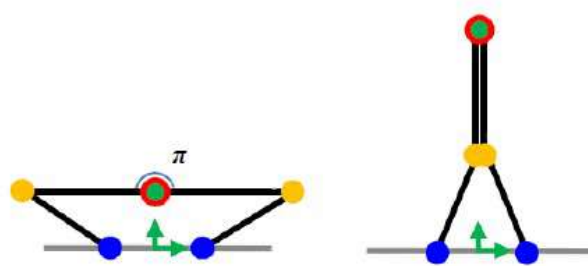


Figura 17: Singularidad paralela [5].

Podemos reducir la probabilidad de que se produzca este tipo de singularidad disminuyendo la distancia entre motores o trabajando en zonas alejadas de la misma. Una forma de evitarla es segmentar el espacio de trabajo haciendo que el robot ocupe determinados conjuntos de puntos con una configuración determinada, como puede observarse en la figura (24). Esto último se logra trabajando en diferentes modos de trabajo, los cuales son consecuencia del análisis cinemático inverso, como se desarrollará en la sección siguiente. El robot SCARA posee cuatro modos de trabajo, donde cada uno representa una solución de dicho análisis.

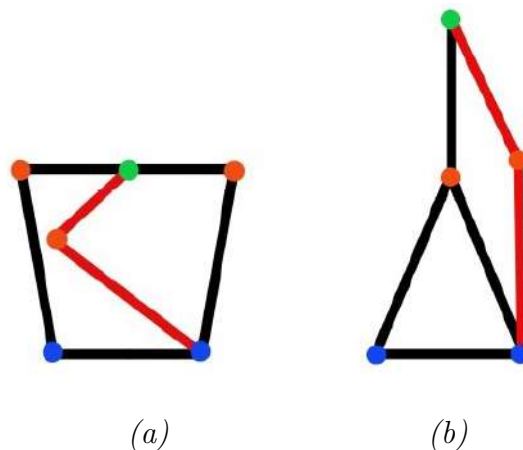


Figura 18: Evitar singularidades paralelas variando el modo de trabajo [5].

Observe que para el modo de trabajo ++ (color negro) indicado en la figura (18a) (se verá en la siguiente sección) al posicionar el robot en dicho punto se produce una singularidad y cuando se utiliza el otro modo de trabajo +- (color rojo) el robot no se ve afectado por dicha singularidad. Lo

mismo sucede con la situación que se observa en la figura (18b) para el modo de trabajo $--$ (color negro) y el modo de trabajo $-+$ (color rojo).

2.5. Modos de trabajo

Los modos de trabajo son diferentes configuraciones que puede adoptar el robot al posicionar el punto efector en un determinado punto. Estas configuraciones son consecuencia del ángulo que forman el brazo distal y proximal entre sí, tanto del lado izquierdo como del derecho. A continuación, en la figura (19) se pueden observar los modos de trabajo que puede adoptar el robot SCARA paralelo.

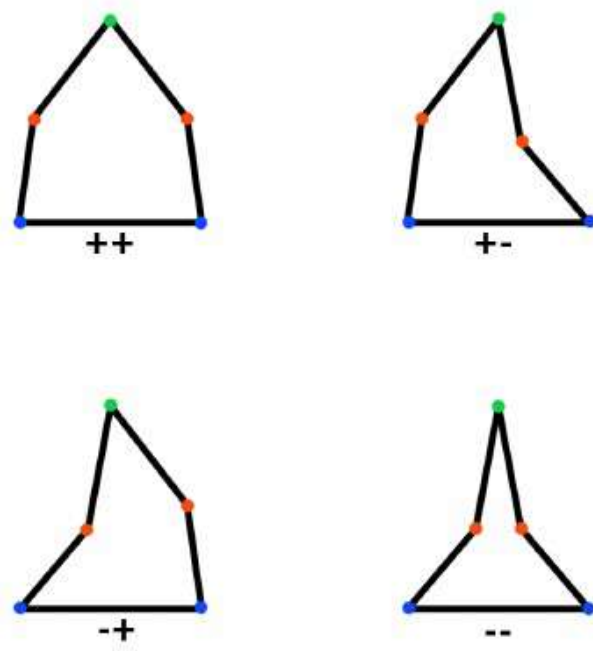


Figura 19: Modos de trabajo del robot SCARA paralelo [5].

2.6. Cinemática

La cinemática se corresponde con el estudio del movimiento de un cuerpo, sin considerar las causas que lo producen.

En esta sección analizaremos la cinemática del robot SCARA paralelo, en principal para calcular la posición del robot. Además, la cinemática consiste en determinar las velocidades y aceleraciones de los brazos del robot, pero en este proyecto nos centraremos específicamente en determinar la posición del robot.

2.6.1. Cinemática inversa

La cinemática indirecta nos permite determinar la posición del mecanismo en función de conocer/definir previamente la posición del punto efector, la distancia entre servomotores y las longitudes de los eslabones. En otras palabras, permite conocer la configuración del robot para una determinada posición del punto efector.

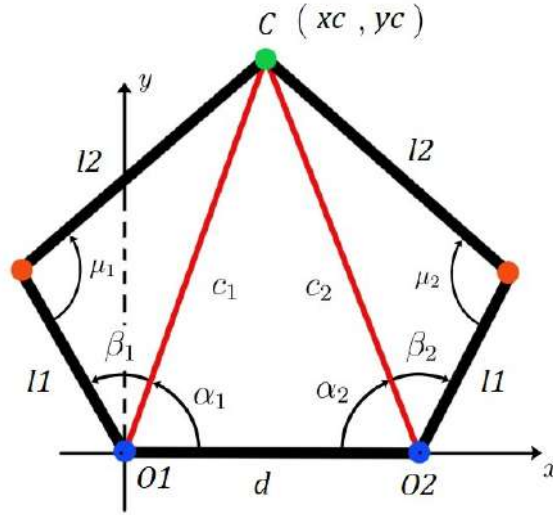


Figura 20: Esquema solución cinemática inversa [5].

El siguiente desarrollo que se presenta para conocer los ángulos de entrada se basa en lo realizado en [5] y el mismo fue utilizado para realizar el posicionamiento del robot en la automatización, como se verá más adelante. Las nomenclaturas utilizadas son basadas en las de la figura (21).

Para el cálculo de los ángulos $\beta_1, \beta_2, \alpha_1, \alpha_2$ en primera instancia se realiza el cálculo de la distancia de los segmentos c_1 y c_2 , luego, se procede a calcular los ángulos utilizando la ley del coseno.

$$c_i = \sqrt{(x_c - x_{O_i})^2 + (y_c - y_{O_i})^2} \quad (1)$$

Aplicando la ley del coseno y despejando se obtiene:

$$\alpha_i = \arccos \left(\frac{c_j^2 - c_i^2 - d^2}{-2c_i d} \right) \quad (2)$$

$$\beta_i = \arccos \left(\frac{l_2^2 - l_1^2 - c_i^2}{-2l_1 c_i} \right) \quad (3)$$

Ahora, desarrollando matemáticamente podemos simplificar las fórmulas

$$c_1^2 = (x_c - x_{O1})^2 + (y_c - y_{O1})^2 = x_c^2 + y_c^2 \quad (4)$$

$$\begin{aligned}
c_2^2 &= (x_c - x_{O2})^2 + (y_c - y_{O2})^2 \\
&= (x_c - d)^2 + y_c^2 \\
&= x_c^2 - 2x_c d + d^2 + y_c^2 \\
&= c_1^2 - 2x_c d + d^2
\end{aligned} \tag{5}$$

$$\begin{aligned}
\alpha_1 &= \arccos \left(\frac{c_2^2 - c_1^2 - d^2}{-2c_1 d} \right) \\
&= \arccos \left(\frac{c_1^2 - 2x_c d + d^2 - c_1^2 - d^2}{-2c_1 w} \right) \\
&= \arccos \left(\frac{x_c}{c_1} \right)
\end{aligned} \tag{6}$$

$$\begin{aligned}
\alpha_2 &= \arccos \left(\frac{c_1^2 - c_2^2 - d^2}{-2c_2 d} \right) \\
&= \arccos \left(\frac{c_1^2 - (c_1^2 - 2x_c d + d^2) - d^2}{-2c_2 d} \right) \\
&= \arccos \left(\frac{-x_c + d}{c_2} \right)
\end{aligned} \tag{7}$$

Entonces, los ángulos θ_1 y θ_2 resultan de sumar los ángulos β y α . Según el resultado de esta suma, podemos determinar el modo de trabajo del robot, a partir de las cuatro combinaciones siguientes:

Modo de trabajo	Brazo izquierdo	Brazo derecho
+ +	$\alpha_1 + \beta_1$	$\alpha_2 + \beta_2$
+ -	$\alpha_1 + \beta_1$	$\alpha_2 - \beta_2$
- +	$\alpha_1 - \beta_1$	$\alpha_2 + \beta_2$
- -	$\alpha_1 - \beta_1$	$\alpha_2 - \beta_2$

Tabla 1: Modos de trabajo en función de la combinación de β y α .

También en [[8]] se presenta un desarrollo vectorial, en donde es interesante ver cómo el modo de trabajo del robot resulta de realizar la intersección entre dos circunferencias, una de radio l_2 centrada en el punto efector y otra de radio l_1 centrada O_2 . Para conocer el valor de θ_1 se realiza la intersección también entre dos circunferencias, como se mencionó anteriormente, pero la circunferencia de radio l_1 se encuentra centrada en O_1 .

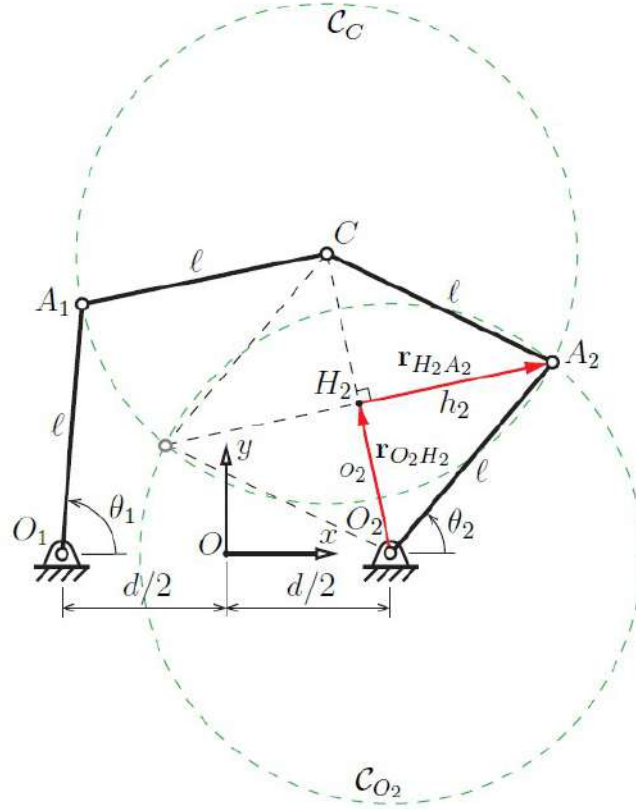


Figura 21: Esquema solución cinemática inversa [8].

2.6.2. Cinemática directa

La cinemática directa permite determinar la posición del mecanismo y el punto efector, conociendo los ángulos de entrada, la distancia entre los servomotores y las longitudes de los eslabones. En nuestro caso, los ángulos de entrada se corresponderán con los ángulos en los que se quieren posicionar a los servomotores.

A continuación, se presenta el desarrollo basado en [8] para conocer la posición del mecanismo teniendo como dato los ángulos que forman los brazos proximales con la horizontal. Se presenta este desarrollo ya que fue el que se utilizó para realizar el programa de simulación de movimiento del robot en Matlab, como se verá más adelante.

Como podemos ver en la figura (22) las dos posiciones posibles del punto efector al ingresar determinado par de ángulos se obtienen a partir de resolver la intersección entre dos circunferencias de radio l_2 centradas en la A_i como se desarrolla en [5]. A continuación, se desarrollará la cinemática de otra manera, la cual consiste en resolver vectorialmente la posición del punto efector.

$$\vec{r}_{OC} = \vec{r}_{OO_2} + \vec{r}_{O_2A_2} + \vec{r}_{A_2C} = \vec{r}_{OA_2} + \vec{r}_{A_2H} + \vec{r}_{HC} \quad (8)$$

Donde

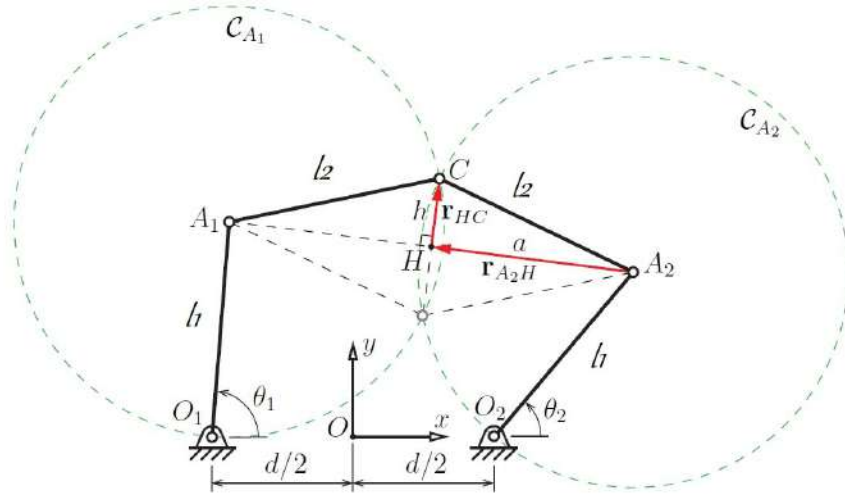


Figura 22: Esquema solución cinemática directa [8].

$$\vec{r}_{OA_2} = \vec{r}_{OO_2} + \vec{r}_{O_2A_2} \quad (9)$$

$$\vec{r}_{A_2H} = \frac{1}{2}(\vec{r}_{OA_1} + \vec{r}_{OA_2}) \quad (10)$$

$$\vec{r}_{OA_1} = \begin{pmatrix} -\frac{d}{2} + l_1 \cos(\theta_1) \\ l_1 \sin(\theta_1) \end{pmatrix} \quad (11)$$

$$\vec{r}_{OA_2} = \begin{pmatrix} \frac{d}{2} + l_1 \cos(\theta_2) \\ l_1 \sin(\theta_2) \end{pmatrix} \quad (12)$$

Así logramos expresar $\vec{r}_{A_2H}$ en función de: la distancia entre servos, longitudes de los brazos y ángulos de entrada.

$$\vec{r}_{A_2H} = \frac{1}{2} \left(\begin{pmatrix} -\frac{d}{2} + l_1 \cos(\theta_1) \\ l_1 \sin(\theta_1) \end{pmatrix} - \begin{pmatrix} \frac{d}{2} + l_1 \cos(\theta_2) \\ l_1 \sin(\theta_2) \end{pmatrix} \right) = \frac{1}{2} \begin{pmatrix} l_1 \cos(\theta_1) - l_1 \cos(\theta_2) \\ l_1 \sin(\theta_1) - l_1 \sin(\theta_2) \end{pmatrix} \quad (13)$$

$$\vec{r}_{A_2H} = \frac{1}{2} \begin{pmatrix} -d + l_1 \cos(\theta_1) - l_1 \cos(\theta_2) \\ l_1 \sin(\theta_1) - l_1 \sin(\theta_2) \end{pmatrix} \quad (14)$$

Ahora, se calcula el vector $\vec{r}_{HC}$ en función de $\vec{r}_{A_2H}$ para ello, al ser perpendiculares entre sí, se utiliza la siguiente matriz de rotación:

$$\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \quad (15)$$

Una vez rotado $\vec{r}_{HC}$, debemos normalizar $\vec{r}_{A_2H}$, esto se logra dividiendo al vector por su módulo, el cual es a , de la siguiente forma:

$$\widehat{r_{A_2H}} = \frac{\vec{r}_{A_2H}}{a} \quad (16)$$

Luego, para obtener $\vec{r}_{HC}$ debemos multiplicar al versor anteriormente calculado por el módulo de $\vec{r}_{HC}$, el cual es h y rotar el vector multiplicando por la matriz de rotación, obteniendo:

$$\vec{r}_{HC} = -\gamma \frac{h}{a} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \vec{r}_{A_2H} \quad (17)$$

Donde γ es un número 1 o -1 según la solución que se desea encontrar, si la 1 o 2 y $h = \sqrt{l^2 - a^2}$. Si $\gamma = 1$ se obtiene el modo de ensamblaje donde el punto efector se ubica en el punto C indicado en la figura (22), si $\gamma = -1$ se obtiene el otro modo de ensamblaje donde el punto efector se ubica a 180° del punto C.

Desarrollando, obtenemos:

$$\vec{r}_{HC} = -\gamma \frac{h}{2a} \begin{pmatrix} -l_1 \sin(\theta_1) + l_1 \sin(\theta_2) \\ -d + l_1 \cos(\theta_1) - l_1 \cos(\theta_2) \end{pmatrix} \quad (18)$$

$$\begin{aligned} \vec{r}_{OC} &= \begin{pmatrix} \frac{d}{2} + l_1 \cos(\theta_2) \\ l_1 \sin(\theta_2) \end{pmatrix} + \frac{1}{2} \begin{pmatrix} -d + l_1 \cos(\theta_1) - l_1 \cos(\theta_2) \\ l_1 \sin(\theta_1) - l_1 \sin(\theta_2) \end{pmatrix} - \gamma \frac{h}{2a} \begin{pmatrix} -l_1 \sin(\theta_1) + l_1 \sin(\theta_2) \\ l_1 \cos(\theta_1) - l_1 \cos(\theta_2) \end{pmatrix} \\ &= \begin{pmatrix} \frac{1}{2} l_1 \cos(\theta_1) - \frac{1}{2} l_1 \cos(\theta_2) + \gamma \frac{h}{2a} (l_1 \sin(\theta_1) - l_1 \sin(\theta_2)) \\ \frac{1}{2} l_1 \sin(\theta_1) + \frac{1}{2} l_1 \sin(\theta_2) + \gamma \frac{h}{2a} (d - l_1 \cos(\theta_1) + l_1 \cos(\theta_2)) \end{pmatrix} \end{aligned} \quad (19)$$

Resultando finalmente:

$$x_C = \frac{1}{2} (l_1 \cos(\theta_1) - l_1 \cos(\theta_2)) + \gamma \frac{h}{2a} (l_1 \sin(\theta_1) - l_1 \sin(\theta_2))$$

$$y_C = \frac{1}{2} (l_1 \sin(\theta_1) + l_1 \sin(\theta_2)) + \gamma \frac{h}{2a} (d - l_1 \cos(\theta_1) + l_1 \cos(\theta_2))$$

Donde a es igual a:

$$a = |\vec{r}_{A_2H}| = \sqrt{(-d + l_1 \cos(\theta_1) - l_1 \cos(\theta_2))^2 + (l_1 \sin(\theta_1) - l_1 \sin(\theta_2))^2} \quad (20)$$

Debemos tener en cuenta que en función de lo desarrollado en la cinemática inversa, el ángulo que a nosotros nos interesará será $\theta_{2b} = 180 - \theta_2$, ya que, como se verá más adelante, los ángulos ingresados como dato en el programa de IposPLUS son los ángulos internos que forman los brazos proximales con la horizontal.

3. Desarrollo

3.1. Optimización tablero eléctrico

Como se mencionó en la sección 1.2 el robot dispone de un tablero de control, en donde se encuentran los controladores de ambos servomotores e interruptores automáticos. El mismo se encuentra sobre la mesa de trabajo del robot y ocupa una gran cantidad de espacio. Además, los cables ingresan al mismo y egresan del mismo en zonas donde puede resultar incómoda su presencia. Por último, el mismo posee un ventilador para extracción de aire caliente y refrigerar el interior del gabinete.

Se propone disminuir la distancia existente entre los controladores e interruptores automáticos. Según el fabricante SEW Eurodrive en el manual de usuario, los mismos pueden ubicarse pegados uno al lado del otro sin inconvenientes. Disminuir la distancia entre componentes logrará aumentar el espacio disponible alrededor de los mismos y, además, brindará la opción de utilizar un tablero más angosto que el actual.

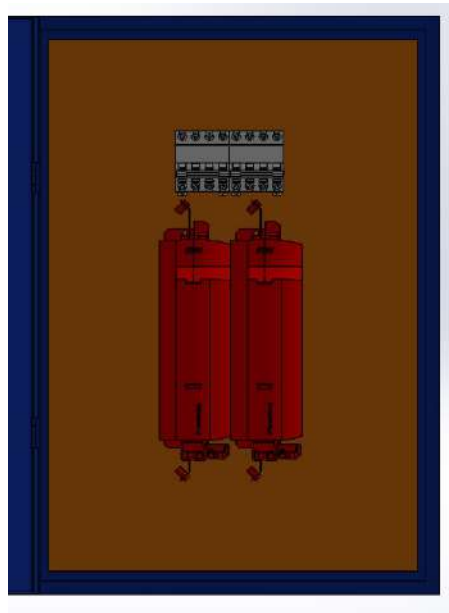


Figura 23: Componentes dentro del gabinete [Propia].

Esta liberación de espacio también permite utilizar cablecanales para emproljar el interior del tablero.

Con el fin de evitar la apertura de la tapa principal del tablero a la hora de operar sobre los interruptores automáticos para encender los controladores, se pensó en realizar un calado para colocar una ventana la cual permite el acceso al usuario hacia los interruptores, sin necesidad de abrir la totalidad del gabinete. Además, permitirá la visualización del estado de los controladores si fuese necesario.

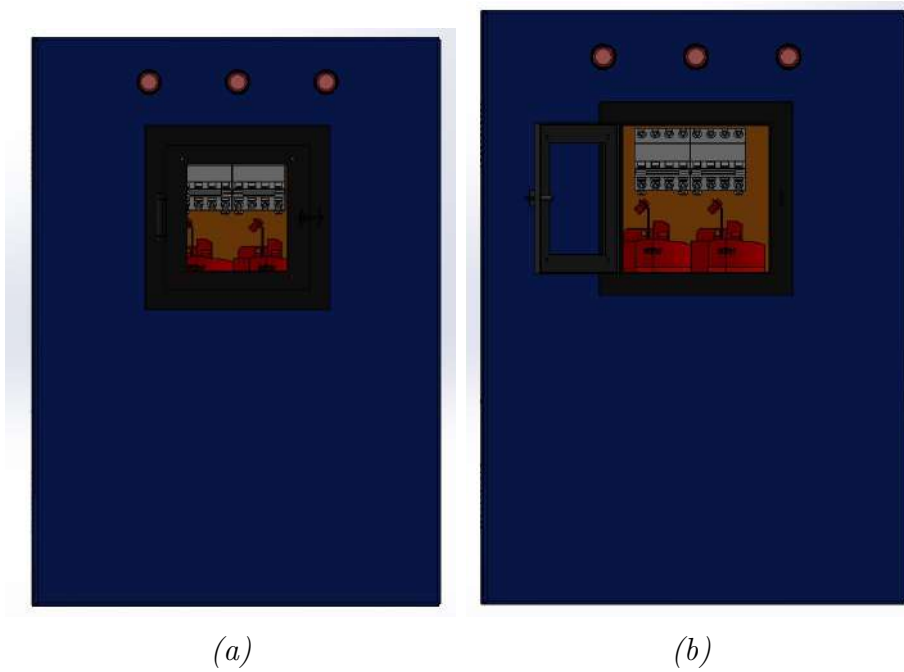


Figura 24: Ventana de acceso [Propia].

El ingreso y el egreso del cableado será todo por un mismo lateral, como ilustra la figura (25).

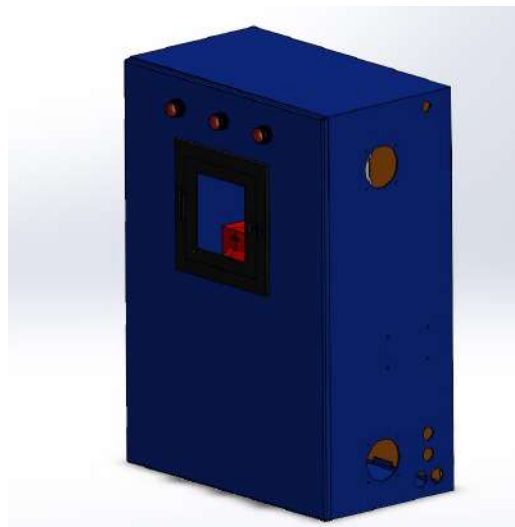


Figura 25: Entrada y salida de cableado [Propia].

Por último, se busca mejorar la refrigeración del gabinete agregando un ventilador de extracción de aire adicional y realizando "branqueas" para permitir el acceso de aire al interior del gabinete, como se observa en la figura (26). Lo que se busca es obtener un flujo de aire por la parte superior e interior a los controladores, que es donde los mismos liberan el calor durante su funcionamiento.

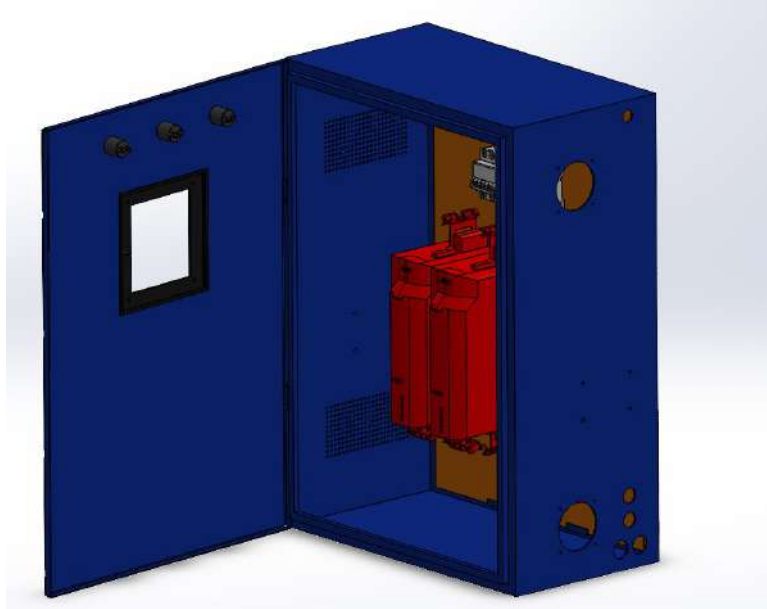


Figura 26: Refrigeración gabinete [Propia].

3.2. Soporte

Su función principal es sostener el conjunto conformado por los brazos y servomotores, logrando así la horizontalización del robot. Además, se consideró durante su diseño que el mismo no genere interferencias mecánicas, las cuales no permitan aprovechar ciertos sectores del espacio de trabajo del robot.

Teniendo en cuenta lo mencionado al principio del párrafo anterior, el soporte deberá tener la capacidad de soportar al robot, sin deformarse y sin tener grandes desplazamientos, los cuales afecten la precisión del robot en su punto efector, debido a la prolongación de los mismos. También se realizó un estudio de frecuencia para evitar que el funcionamiento del robot haga que este se encuentre cercano a algún modo de resonancia. Por último, se hizo un análisis de pandeo a la estructura para asegurarse de que las columnas del pórtico no se vean afectadas. Otra característica importante es que el mismo pueda ser transportable y no tener un peso excesivo, ya que el mismo se encontrará sobre una mesa de madera ubicada en el Fablab.

La estructura que mejor se adapta a las necesidades del proyecto es la de tipo pórtico, debido a su gran rigidez y estabilidad. Además, este tipo de estructura posee gran espacio entre columnas para que el robot pueda pasar de un lado a otro libremente y aprovechar la totalidad de su espacio de trabajo. Por ello, se decidió que el tipo de soporte a diseñar será de este tipo. En la figura (27) podemos ver una captura del mismo.

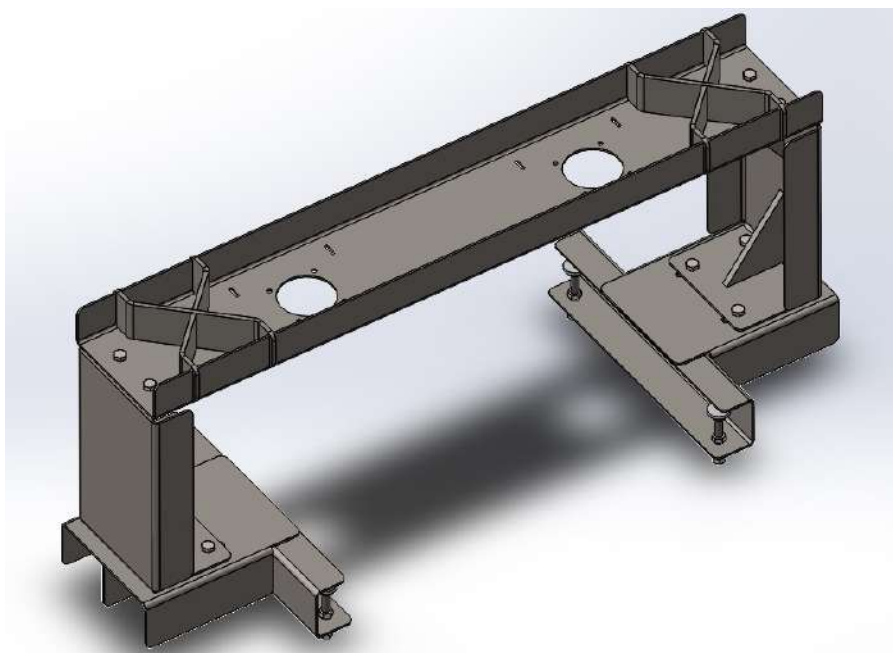


Figura 27: Soporte tipo pórtico [Propia].

Teniendo en cuenta que la estructura debe tener el menor peso posible, mediante simulaciones estáticas de elementos finitos, se determinó el mínimo espesor de chapa a utilizar ($1/8''$) que garantiza la resistencia mecánica del mismo y desplazamientos verticales bajos durante la deformación elástica, dichas simulaciones se encontrarán más adelante. Además, este espesor de chapa permite es sencillo de plegar y cortar mediante corte laser. Dado que no se tienen un ambiente hostil que requiera materiales específicos, se optó por utilizar una chapa de acero al carbono ASTM a36. La unión entre de las columnas tanto a la base como al travesaño, se hará utilizando bulones M10 de cabeza hexagonal grado 10.9 que se consiguen en el mercado. Este tipo de unión garantiza la posibilidad de desmontar el soporte cuando sea necesario para trasladarlo. Un problema que se presentó durante el desarrollo del mismo fue que la mesa posee un ancho de 683 mm, siendo menor al necesario para que el robot circule libremente de un lado a otro. Colocar el soporte longitudinalmente a la mesa no era viable, ya que imposibilitaría aprovechar la totalidad del espacio de trabajo del robot. En la figura (28) se observa una representación esquemática del robot con sus dimensiones y las dimensiones del espacio de trabajo. Esto último, llevó a diseñar unas bases (figura (29)) las cuales se vinculen a la mesa mediante un sistema similar al utilizado en las prensas sargento (figura (30)). El sistema de prisionero se pensó utilizando también bulones M10; esto permite hacer las bases adaptables a diferentes espesores de mesa, sujetando firmemente las bases a la mesa logrando una buena estabilidad de todo el conjunto sin necesidad de agujerear la mesa para atornillar el soporte a la misma.

Estas bases están conformadas por dos piezas plegadas en chapa como se observa en la figura (29), las cuales se vinculan entre sí mediante soldadura (en los anexos se pueden visualizar los planos de conjunto soldado). La primera pieza es la C que posee los agujeros para pasar los pernos prisioneros y la segunda pieza es la base donde se encuentran las ranuras para vincular las columnas y poder regular la ubicación de las mismas.

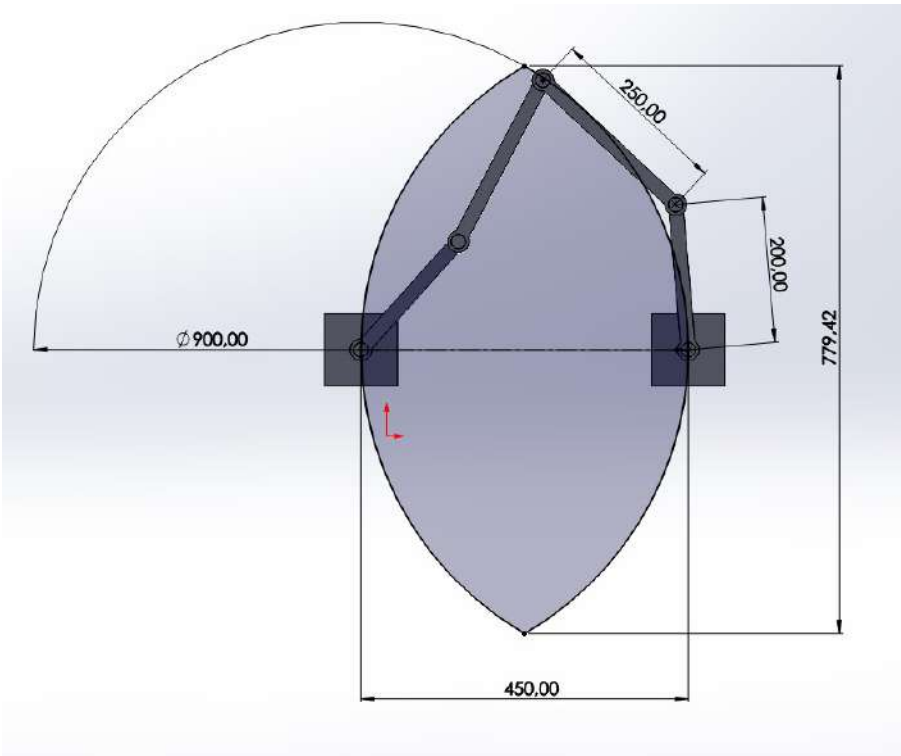
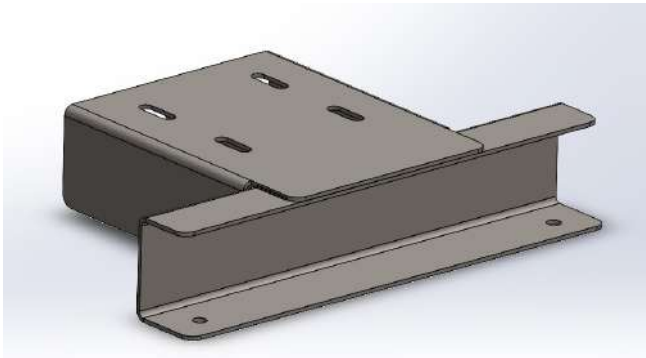
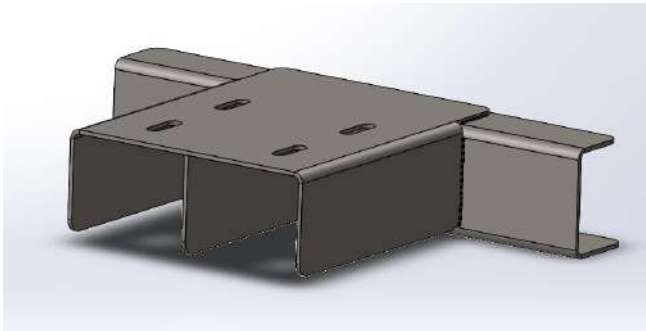


Figura 28: Dimensiones máximas espacio de trabajo robot [Propia].

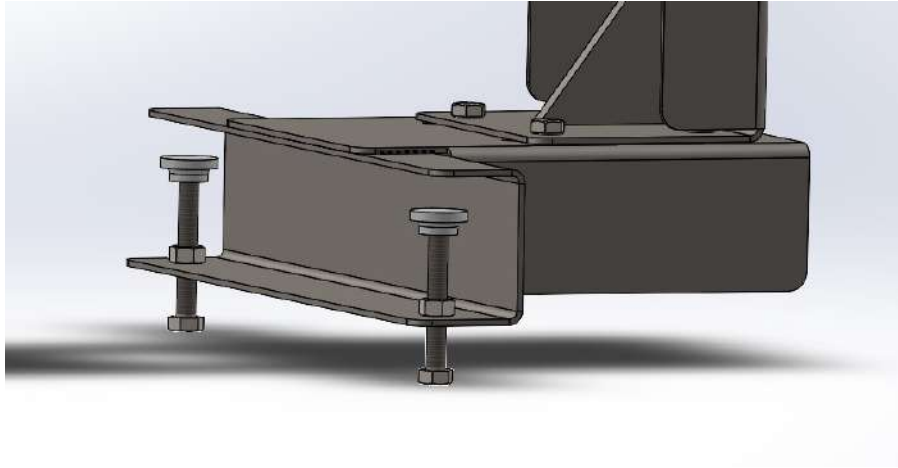


(a)

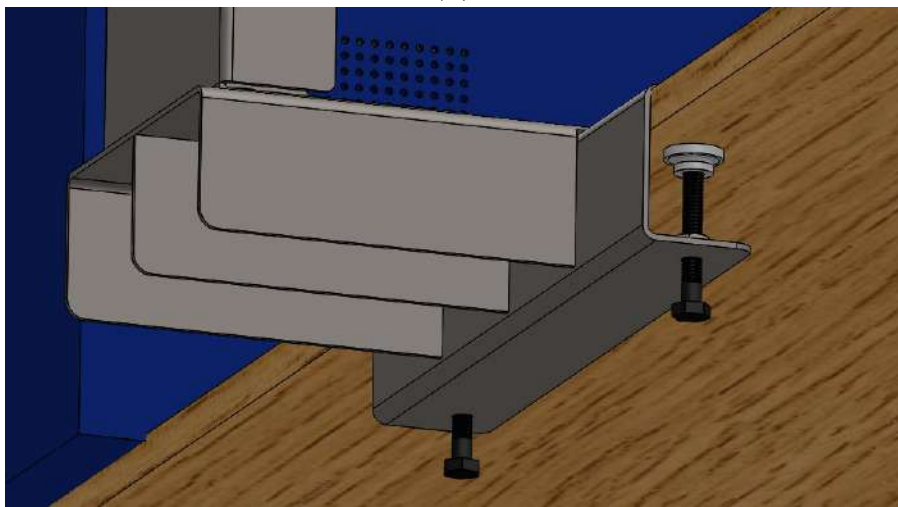


(b)

Figura 29: Bases de vinculación pórtico [Propia].



(a)



(b)

Figura 30: Vinculación bases a la mesa [Propia].

El travesaño posee un largo de 1050 mm. La longitud del mismo se definió en función de que el robot posea la posibilidad de pasar de un lado a otro sin interferencias mecánicas. Se determinó la distancia mínima necesaria teniendo en cuenta que la máxima amplitud que puede adquirir el robot al pasar hacia el otro lado equivale a la longitud del brazo proximal. Para encontrar la distancia se realizó un croquis en SolidWorks utilizando el ensamblaje esquemático del robot, el cual se observa en la figura (31). Por seguridad se decidieron agregar 100 mm de cada lado, resultando una longitud total de 1050 mm. Sin embargo, la distancia libre entre columna y columna es de 930 mm como se puede observar en la figura (32a), la cual sigue siendo segura para evitar colisiones. La distancia del travesaño hacia la mesa se definió de aproximadamente 300 mm (figura (32b)). Esto último permitirá que el día que se implemente un actuador sobre el punto efector haya espacio suficiente para evitar colisiones del mismo con el travesaño..

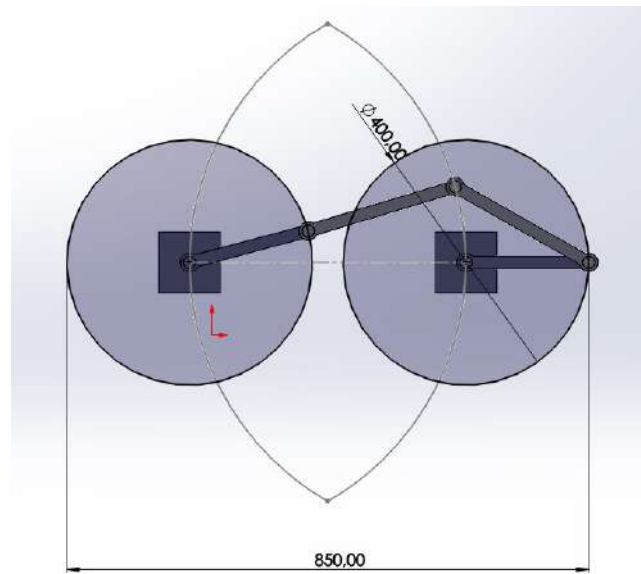
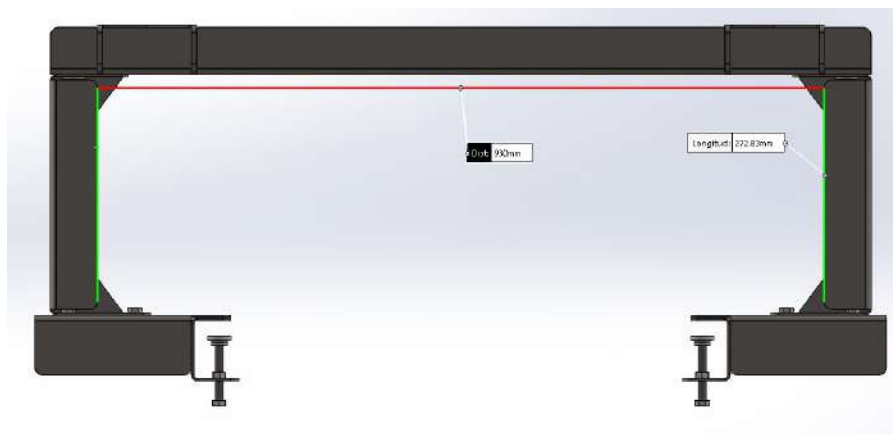
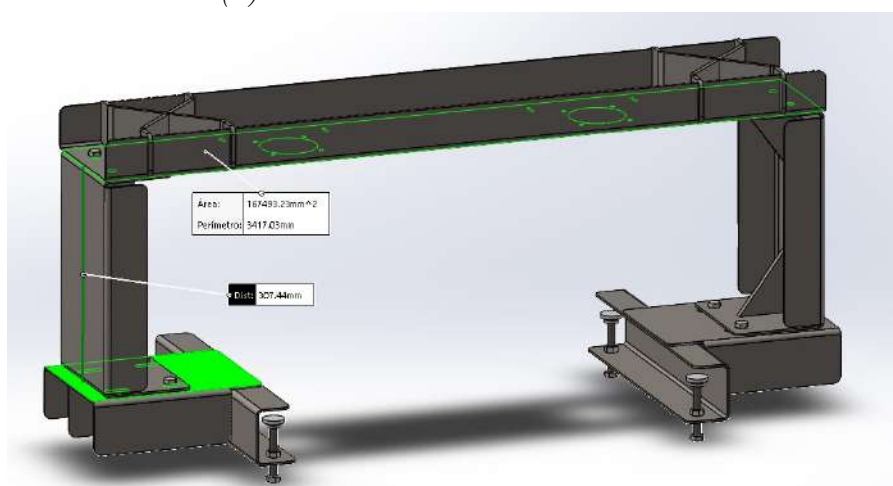


Figura 31: Definición distancia libre mínima [Propia].



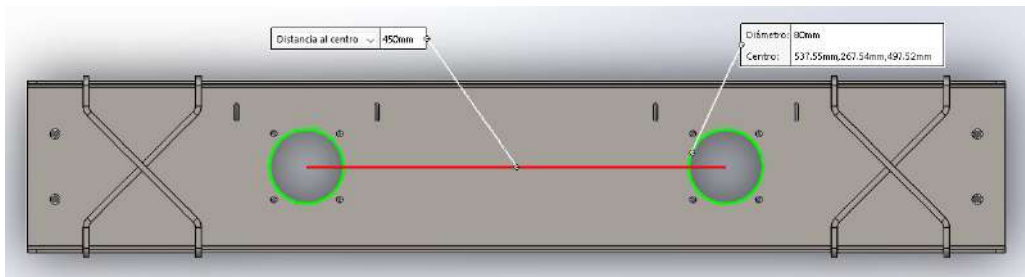
(a) Distancia libre entre columnas.



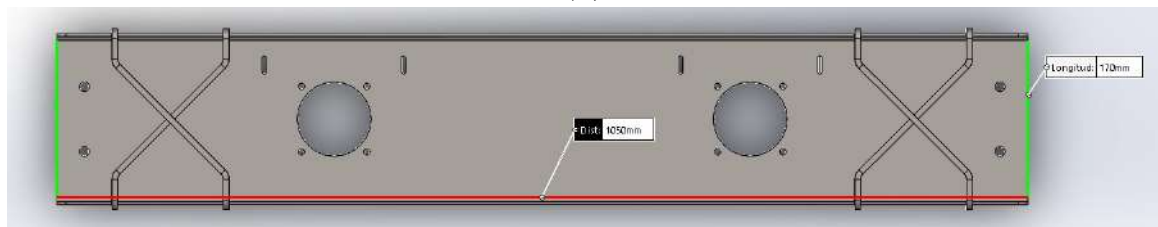
(b) Altura columnas.

Figura 32: Dimensiones generales soporte. [Propia].

La distancia entre servos se estableció como 450 mm, por lo cual el travesaño posee dos agujeros de un diámetro de 80 mm separados entre centros a 450 mm. Además, se tienen cuatro agujeros M6 para fijar los servomotores al mismo. En las ranuras se colocarán los soportes correspondientes para los límites de carrera del robot. El ancho del travesaño es de 170 mm para asegurar un correcto montaje de los servomotores. Posee una sección transversal tipo C con ramas hacia arriba ya que es un tipo de sección la cual garantiza una elevada resistencia a la flexión. La misma sección transversal se utiliza en las columnas pero en este caso se busca aumentar la resistencia al pandeo.



(a)



(b)

Figura 33: Dimensiones generales travesaño [Propia].

Los refuerzos para aumentar la rigidez a la torsión van soldados al travesaño. Para posicionar las chapas plegadas que conforman los refuerzos, se realizaron cortes en las pestañas del travesaño. Las mismas pueden observarse en la figura (34).

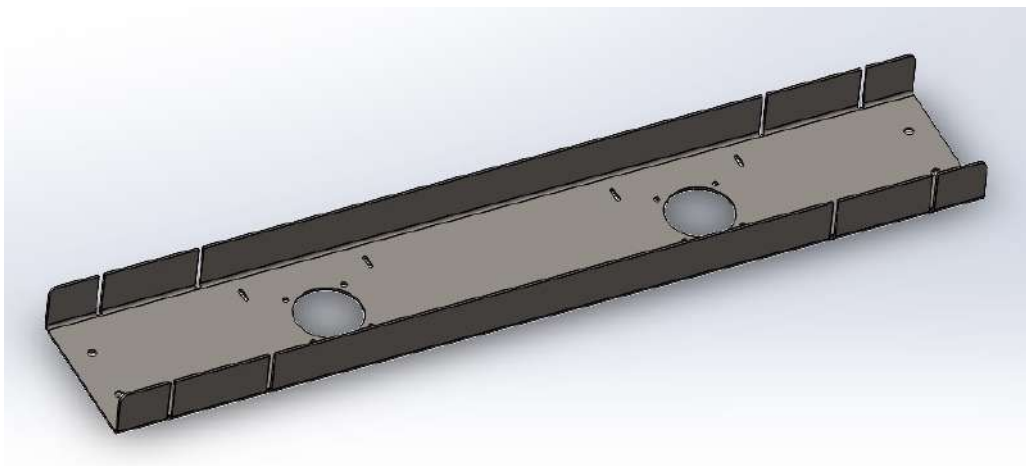


Figura 34: Ranuras para posicionamiento de los refuerzos [Propia].

A su vez, las chapas que conforman los refuerzos poseen ranuras para encastrarse entre sí y formar la X.

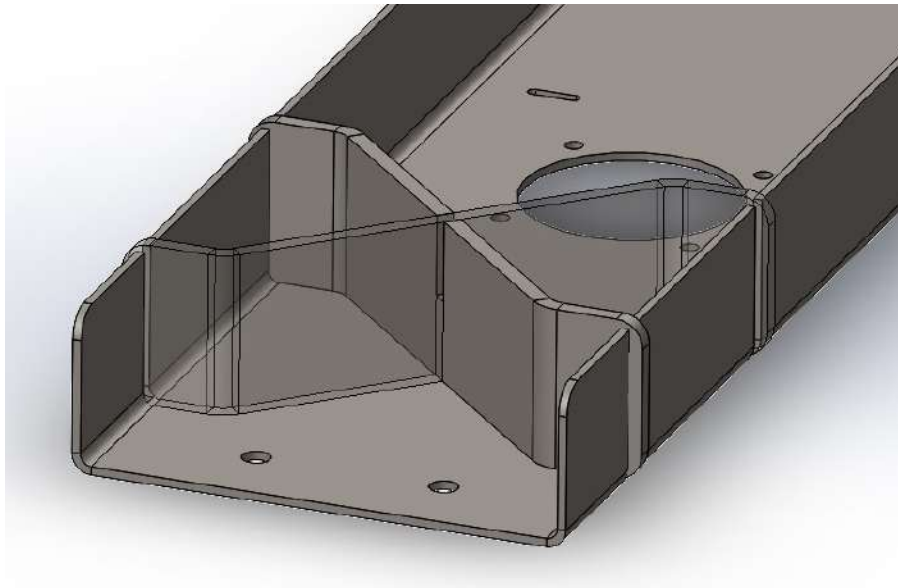


Figura 35: Encastre chapas refuerzos [Propia].

Finalmente, la figura (36) muestra de forma esquemática de como se verá el conjunto montado completamente en la mesa de trabajo.

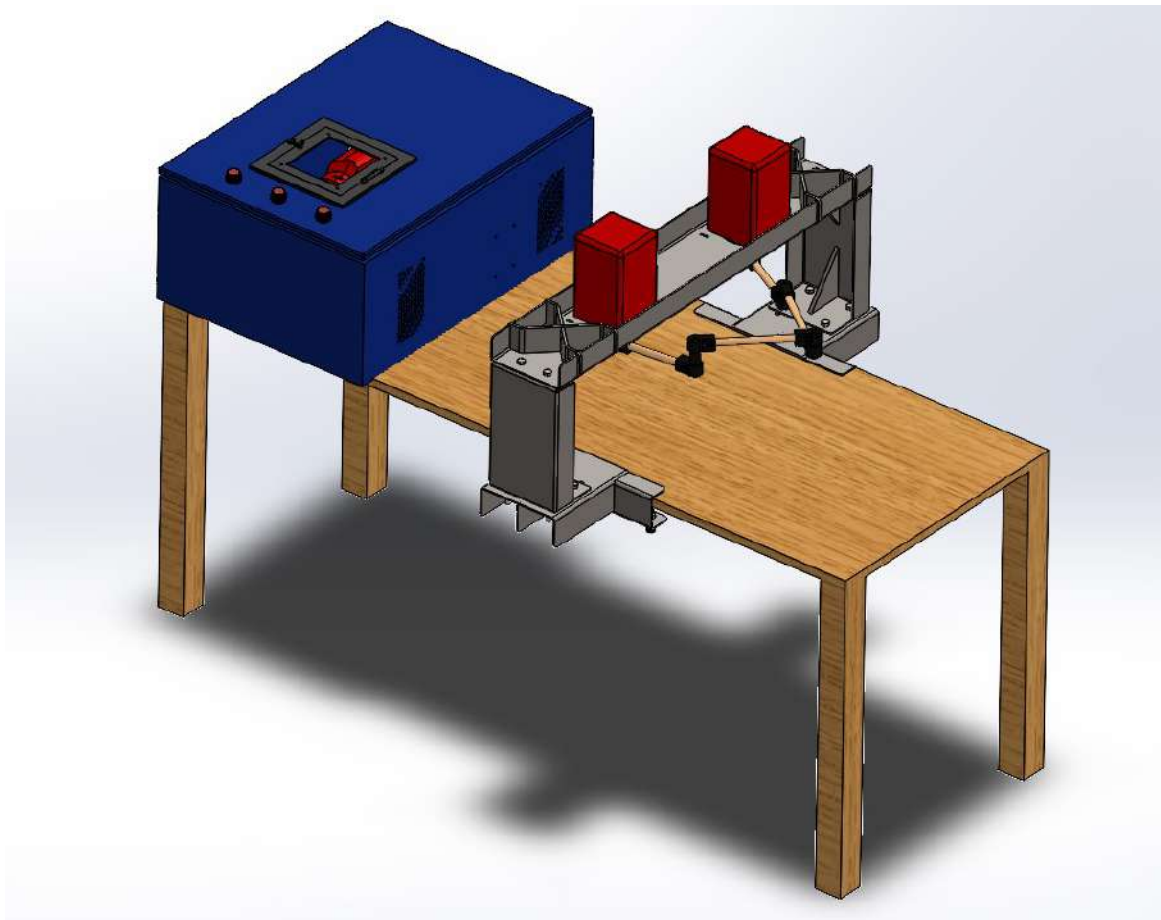


Figura 36: Encastre chapas refuerzos [Propia].

3.3. Simulaciones

Para fabricar la estructura y que esta tenga un correcto desempeño resistencial, se realizaron las siguientes simulaciones.

3.3.1. Simulación estática

Se realizaron simulaciones estáticas para determinar el mínimo espesor que garantice la resistencia estructural de todo el soporte.

En primera instancia, se plantearon las siguientes condiciones en la simulación:

- Espesor de chapa: 1/8"
- Carga por cada servomotor = 200 N = 20,41 kg.
- Momento torsor = 80 Nm.
- Material: SAE 1020 (simil propiedades ASTM a36)

Las simulaciones estáticas realizadas representan una situación de flexo-torsión. Se consideró simular a flexo-torsión debido a que el transporte de un objeto con un actuador, ubicado en el punto efector, tenderá a torsionar la estructura. Dicha situación se representa en la figura (37). El criterio de resistencia utilizado para evaluar la resistencia de la estructura es el criterio de las tensiones de Von Mises.

Adoptando una postura conservadora, se supone una carga exageradamente elevada de 200 N (el fabricante SEW indica que el conjunto servo-reductor pesa 7 kg aproximadamente). La máxima distancia de la carga hacia el eje medio del travesaño se produce cuando ambos brazos se encuentran totalmente estirados. En esta situación, la carga se encuentra aplicada a una distancia $d = 389,71$ mm como se puede observar en la figura (38). Tomando como distancia 400 mm y una carga de 200N aplicada sobre el punto efector, se obtiene un momento torsor de 80 Nm. La aplicación de dicha carga equivale al momento M_t que se indica en la figura (37).

Las cargas y momento torsor se aplicaron como se observa en la figura (39) (si bien en las imágenes de la configuración de la simulación se observa el soporte definitivo, las simulaciones preliminares se realizaron adoptando el mismo criterio). Para aplicar las cargas correspondientes a los pesos de los servomotores, se realizó una circunferencia de diámetro 120 mm y se generó una línea de partición, representando así la superficie de contacto entre el servomotor y el soporte. El momento torsor se aplicó respecto del eje medio del travesaño, ubicado sobre la superficie donde se apoyan los servomotores. Se suponen 80 Nm constantes a lo largo de toda la estructura.

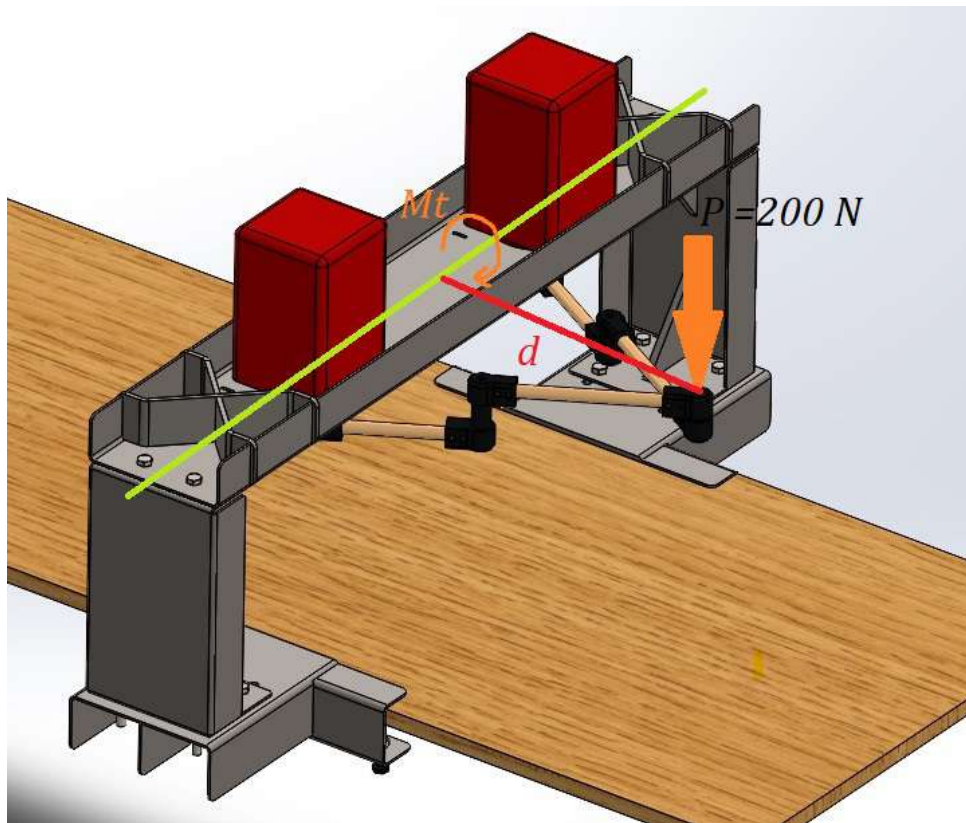


Figura 37: Momento torsor generado por una carga suspendida en el punto efector. [Propia].

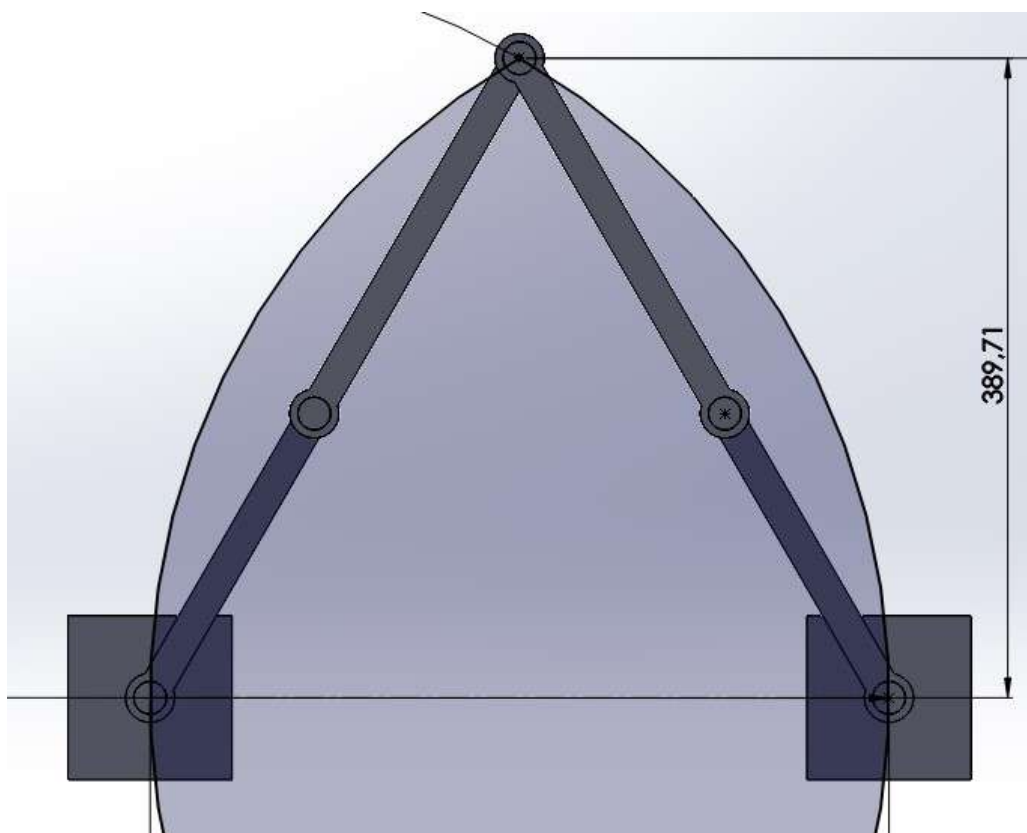
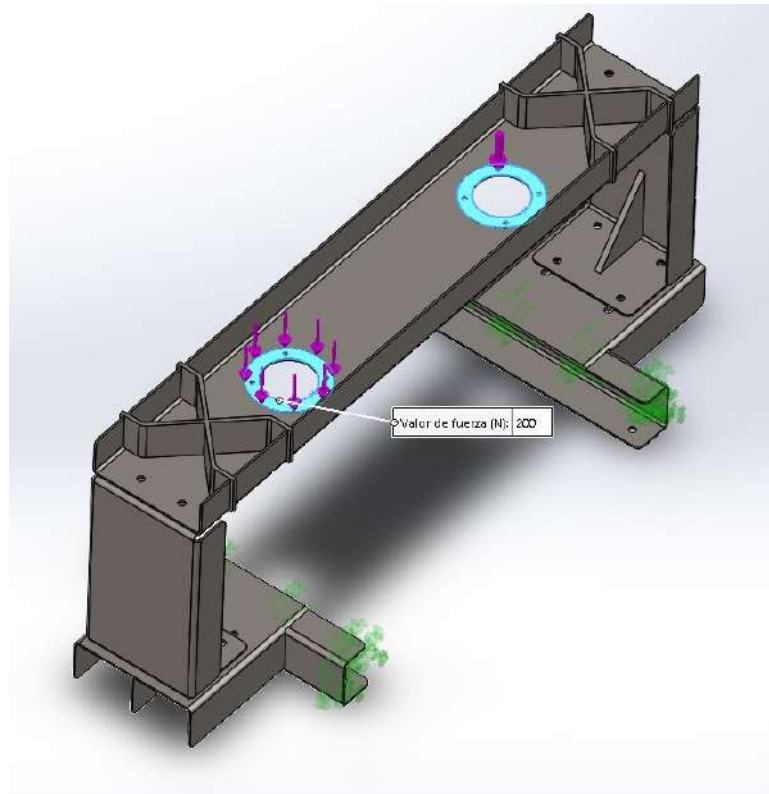
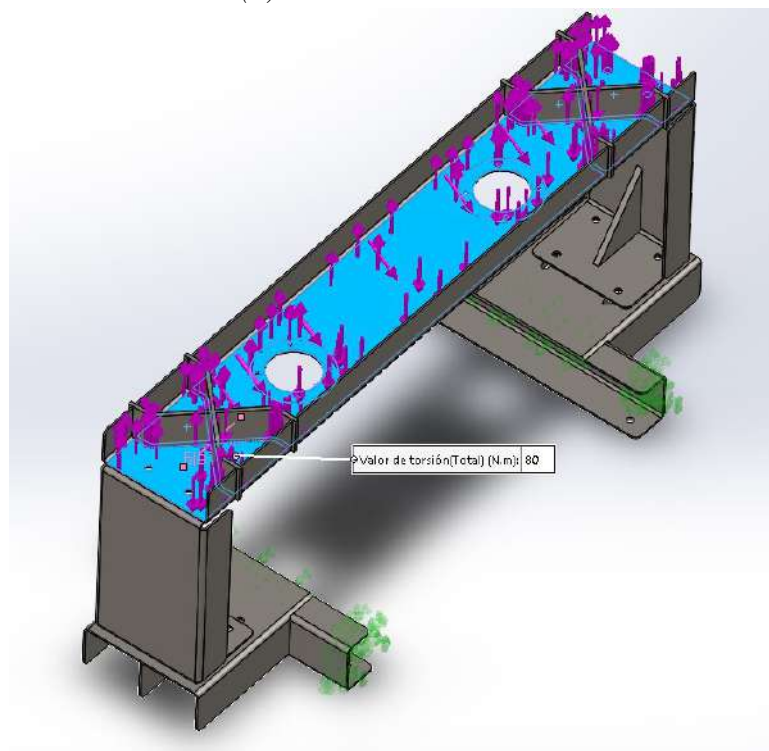


Figura 38: Máxima distancia del punto efector. [Propia].



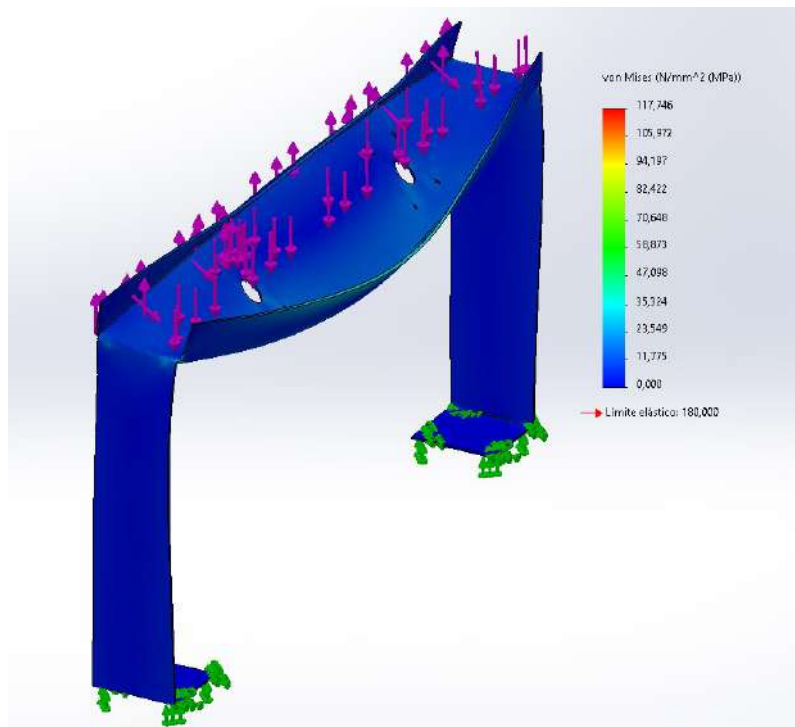
(a) *Peso servomotores.*



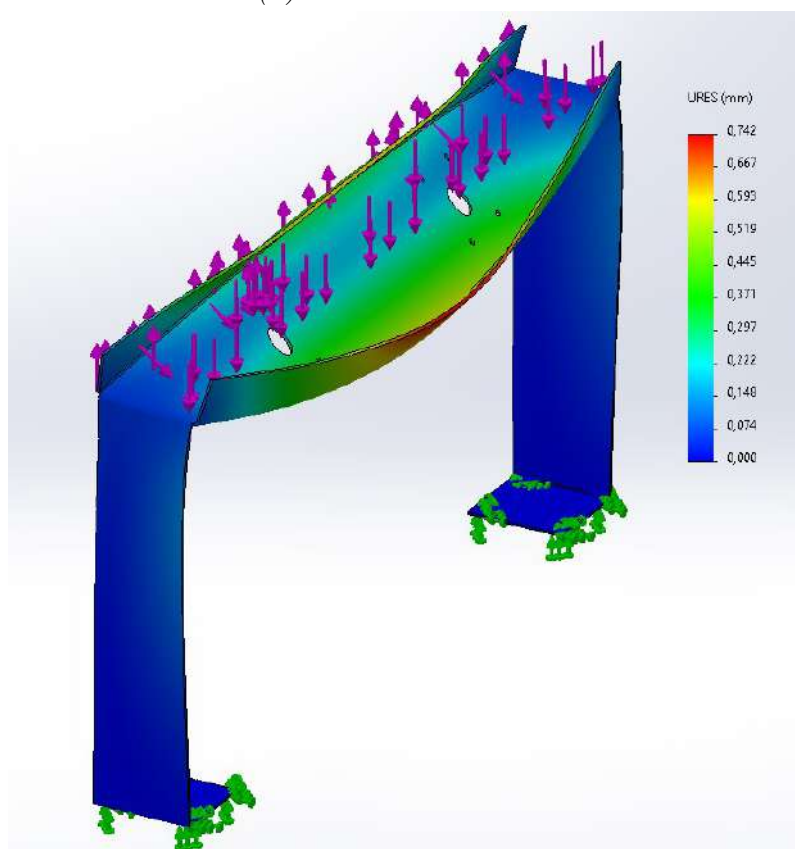
(b) *Momento torsor.*

Figura 39: Cargas aplicadas en la simulación [Propia].

Primeramente, se realizó la simulación para una estructura simple, sin refuerzos, y se obtuvieron los siguientes resultados:



(a) Tensiones de VM.



(b) Desplazamientos.

Figura 40: Simulación estática preliminar 1 [Propia].

Los resultados que se observan en la figura (40a) son óptimos. Vemos que ningún valor supera el límite elástico de la estructura. Los valores máximos

Se puede observar en la figura (40b) que el desplazamiento máximo en el eje Y es de 0,742 mm. Si bien no es un valor elevado y debemos tener en cuenta que estamos suponiendo una situación extrema conservadora, se optó por analizar diferentes formas de reforzar el travesaño para aumentar la rigidez a la torsión.

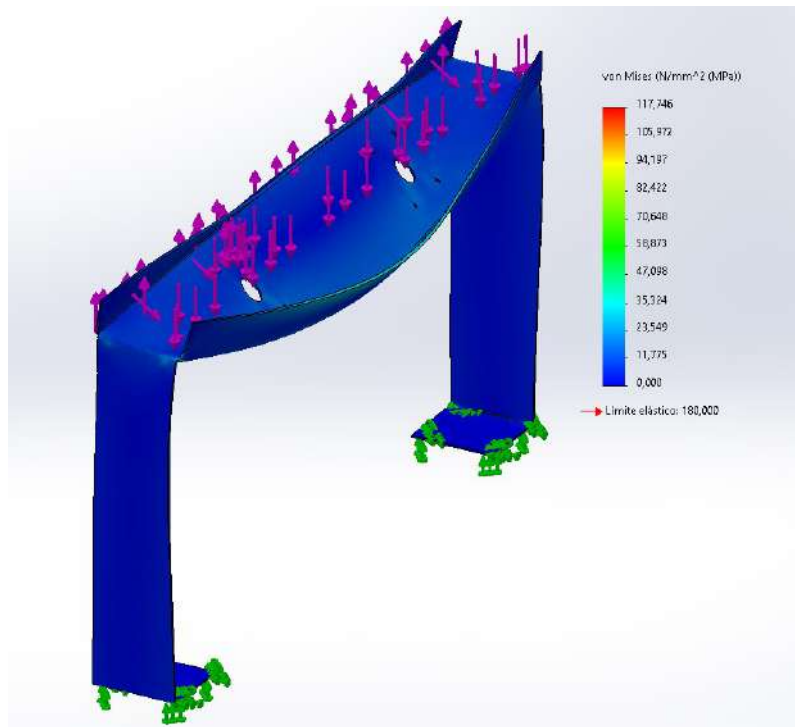
Para aumentar la rigidez, lo que se busca con los refuerzos es generar una mayor sección transversal en la estructura, provocando también un cierre de gran parte de ella. Primero, se optó por reforzar la estructura colocando placas transversales tanto cerca de los apoyos como en el centro de la estructura. En lo que respecta a las tensiones de Von Mises no se ven reflejados grandes cambios, como se puede observar en la figura (41a), pero si hubo una disminución del desplazamiento, como indica la figura (41b) a un valor de 0,456 mm. Segundo, se probó reforzar la estructura aplicando refuerzos en X, ya que los mismos generan un aumento de la sección transversal a lo largo de una longitud mayor a diferencia de los refuerzos de placas transversales, que únicamente aumentan la sección transversal en donde se ubican las placas. Los mismos se ubicaron únicamente cerca de los extremos, con el objetivo de reducir material ya que reforzar en el medio no generaría grandes cambios. Como podemos contemplar en la figura (42a) las tensiones de Von Mises son correctas. La figura (42b) muestra una disminución considerable de los desplazamientos con respecto a las simulaciones anteriores, un valor máximo de 0,219 mm es un valor aceptable teniendo en cuenta que se trata de una condición límite.

Teniendo en cuenta los resultados obtenidos, se optó por reforzar la estructura utilizando refuerzos en X.

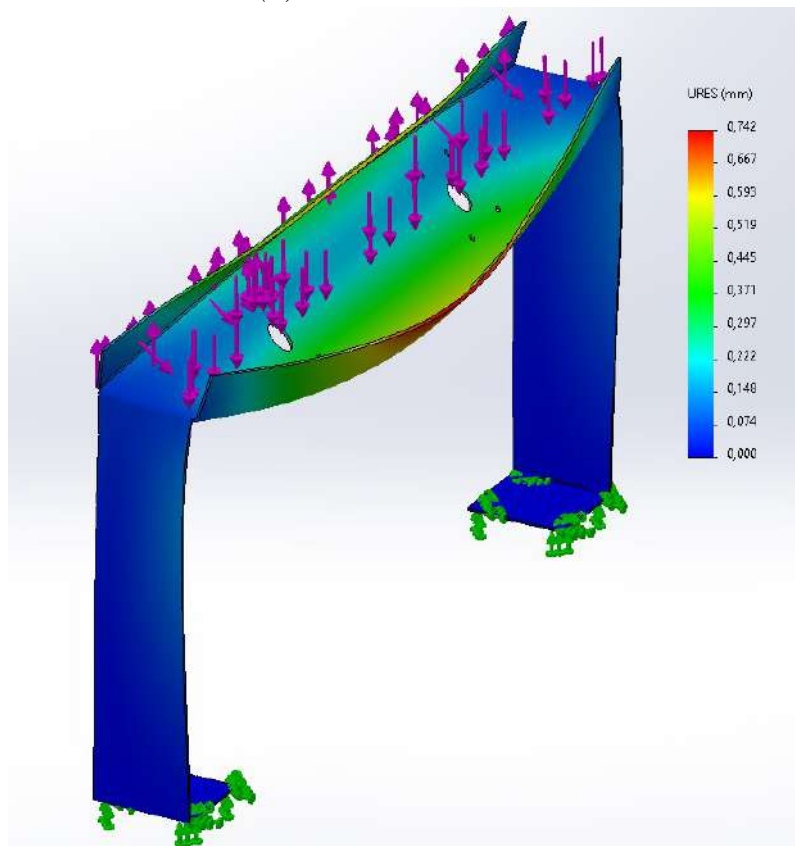
Finalmente, teniendo en cuenta las consideraciones explicadas al principio de la sección para el diseño del soporte, se realizaron las simulaciones a la estructura final. Se planteó una condición de contorno de geometría fija para las bases, en la parte interna que tendrá contacto la mesa por medio de los prisioneros, como se puede observar en la figura (43a). En la figura (43b) observamos el mallado utilizado para la simulación, se utilizó la configuración predefinida por el software, ya que la estructura no tiene complejidad geométrica que requiera de un mallado riguroso.

Como se observa en la figura (44a) los valores de tensiones de Von Mises son adecuados. Por otro lado, los desplazamientos que se observan en la figura (44b) son valores aceptables, el desplazamiento máximo es de 0,264 mm.

Nota: como se mencionó anteriormente, la situación para la cual se realizaron las simulaciones son casos extremos. Para esta aplicación didáctica el soporte no tendrá aplicada una carga elevada en el punto efector, por lo cual podemos despreciar la torsión y decir que solo trabaja a flexión, donde además, las cargas aplicadas sobre el mismo serán 70N las cuales corresponden con los pesos de los servomotores. En conclusión, los desplazamientos y valores de tensiones de Von Mises serán mucho menores a los visualizados en las simulaciones.

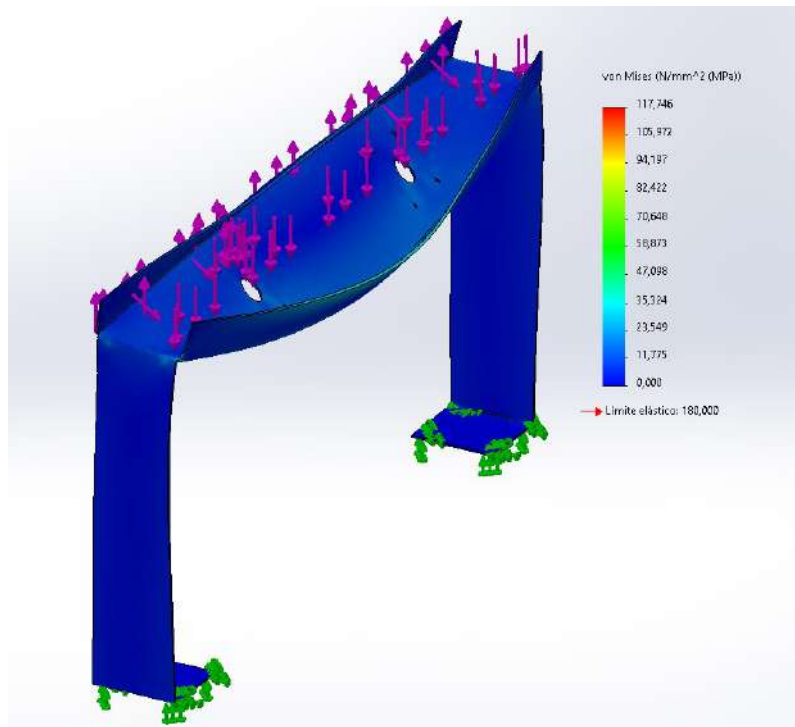


(a) Tensiones de VM.

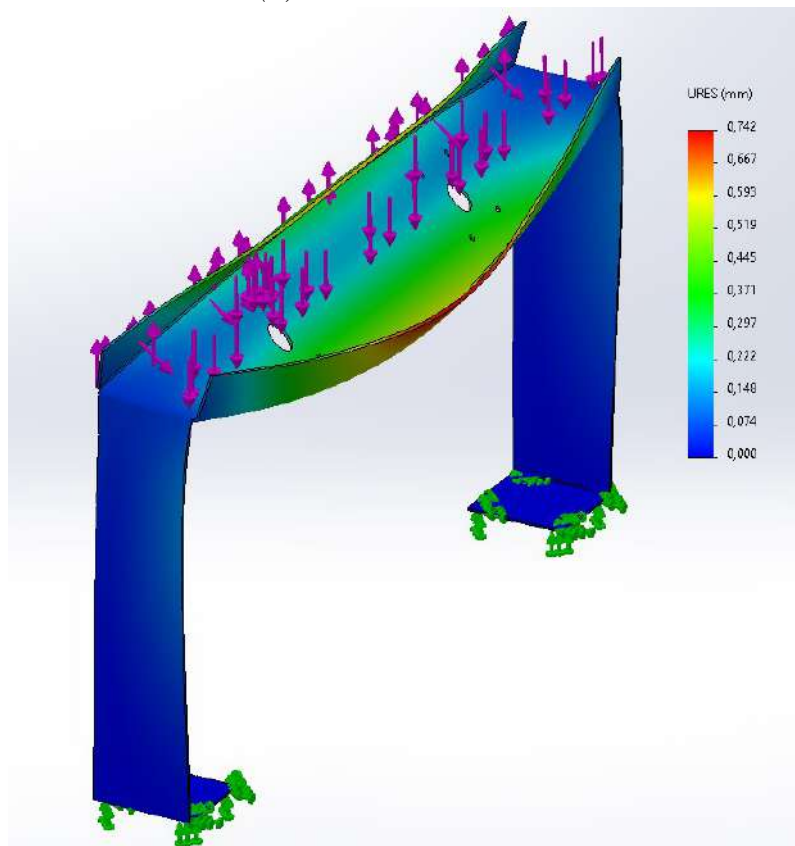


(b) Desplazamientos.

Figura 41: Simulación estática preliminar 2 [Propia].

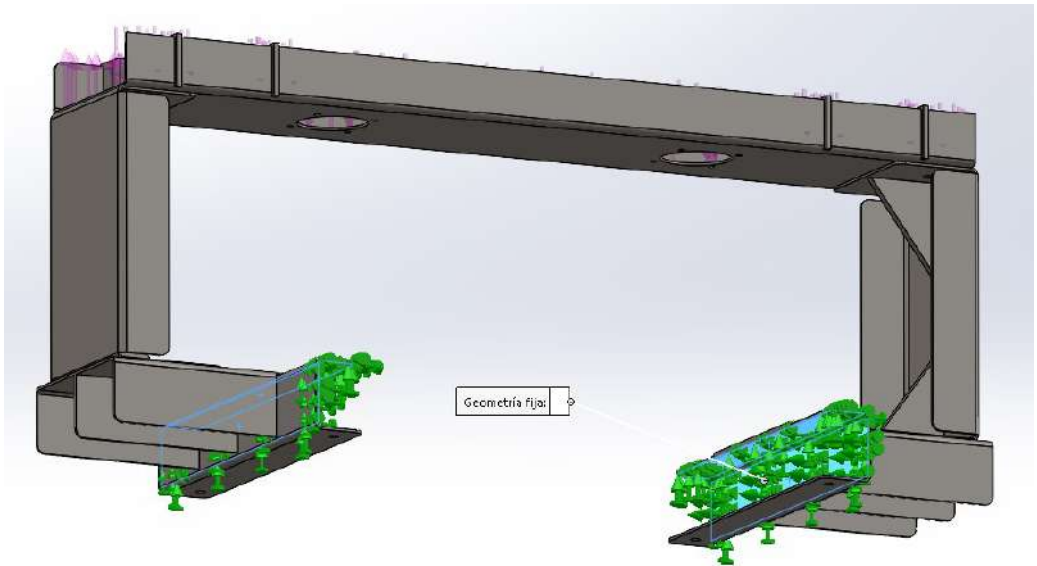


(a) Tensiones de VM.

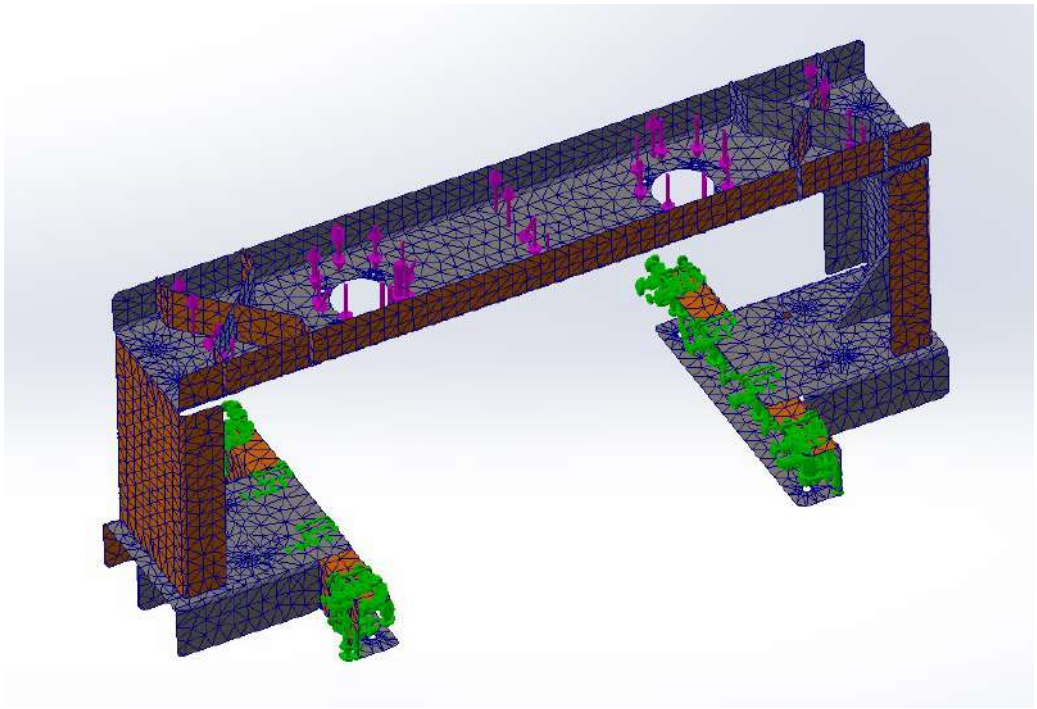


(b) Desplazamientos.

Figura 42: Simulación estática preliminar 3 [Propia].

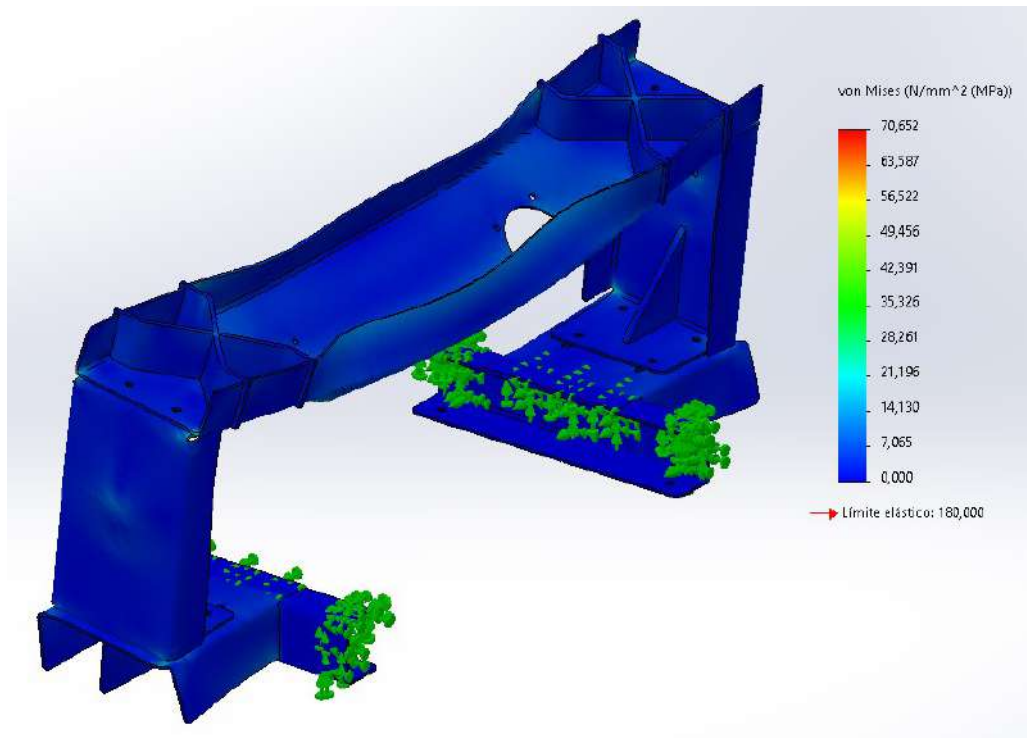


(a) Condiciones de contorno.

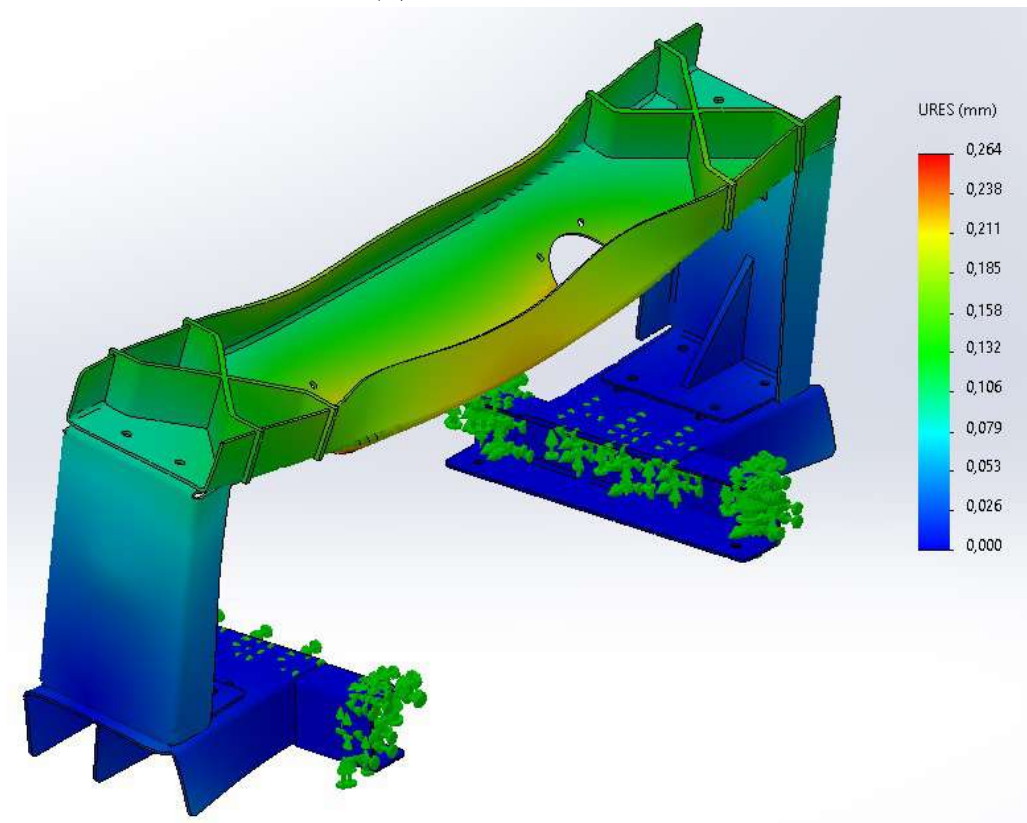


(b) Mallado.

Figura 43: Preparación simulación estática final [Propia].



(a) Tensiones de VM.



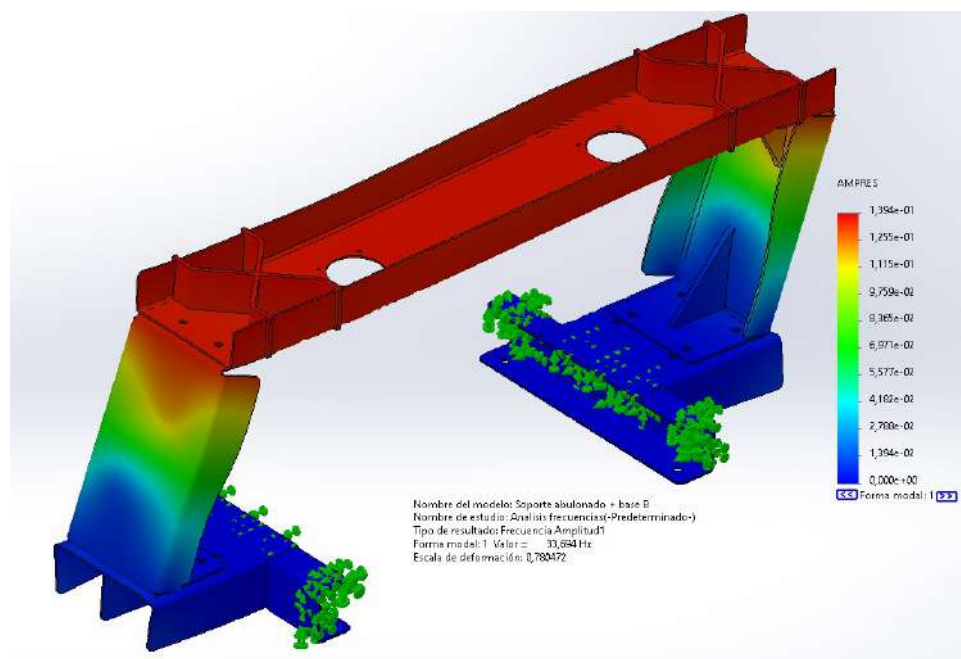
(b) Desplazamientos.

Figura 44: Simulación estática final [Propia].

3.3.2. Análisis de vibraciones

Todas las estructuras poseen frecuencias naturales y modos de vibración, los cuales dependen básicamente de la rigidez y masa de la estructura. Dentro de estos modos de vibración, la estructura tendrá diferentes modos de respuesta ante dicha excitación. Estas excitaciones pueden generar desplazamientos elevados y repetitivos, pudiendo generar una falla estructural en la estructura. Por ello, resulta muy importante analizar las vibraciones en el soporte.

Al tratarse de una estructura la cual soportará componentes en movimiento, sobre todo los servomotores, los cuales generarán fuerzas dinámicas. Debemos evitar que se produzcan excitaciones con frecuencias cercanas a las frecuencias naturales de la misma. En la simulación se supusieron dos masas de 20kg aplicadas sobre el soporte.



(a) Frecuencia natural 1.

Nº de modo	Frecuencia
1	33,694
2	35,478
3	36,764
4	52,299
5	68,951

(b) Frecuencias naturales de la estructura.

Figura 45: Análisis de vibraciones [Propia].

Como podemos observar en la figura (45a) el primer modo de frecuencia ocurre para una excitación de 33,69 Hz. Teniendo en cuenta que los servomotores pueden alcanzar una velocidad de 3000 RPM (50Hz) podemos decir que a partir de una velocidad de 1980 RPM la estructura comenzará a verse afectada por las vibraciones en sus modos de frecuencia. Sin embargo, los servomotores se moverán a una velocidad de 800 RPM la cual está por debajo de ese límite. Esto último nos permite concluir que

la estructura no tendrá problemas de resonancia.

3.3.3. Pandeo

Otro aspecto a considerar es el pandeo, se debe verificar que la estructura no sufra de pandeo sobre todo en sus columnas. Para ello, se realizó un análisis de pandeo. Las condiciones de la simulación fueron las mismas que las que se utilizaron para la simulación estática.

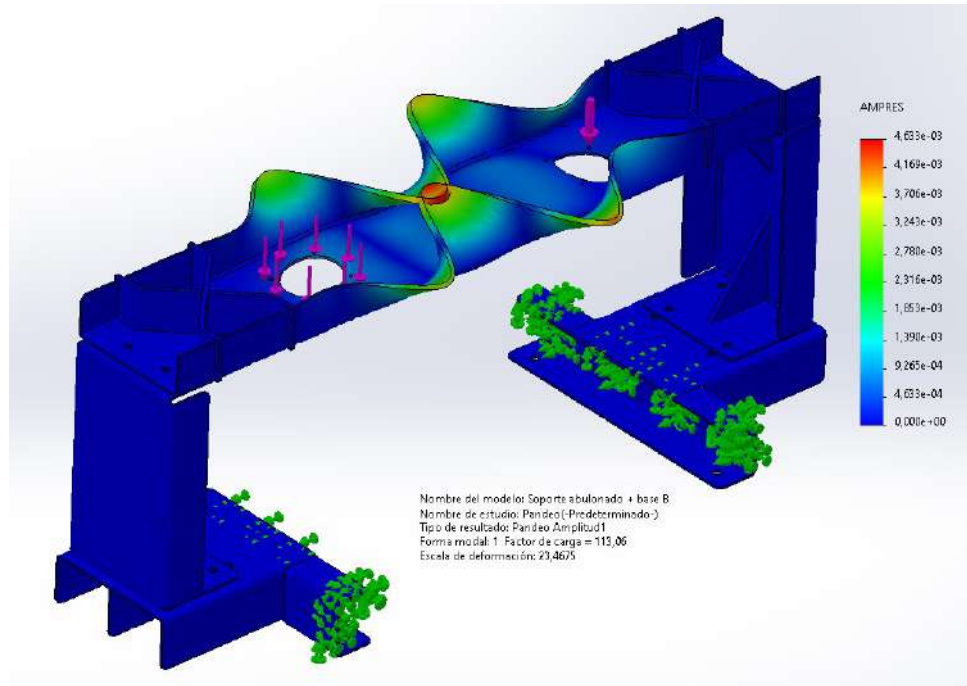


Figura 46: Simulación de pandeo. [Propia].

Se observa en la figura (50) que el factor de carga para que se produzca el primer modo de pandeo es de 113, lo cual equivale a cuántas veces más se debe aplicar la carga con la que fue realizada la simulación. La estructura verifica correctamente el pandeo y este último no será un riesgo para la misma.

3.4. Cálculos tensiones Von Mises soporte

Para validar mediante una segunda vía la resistencia a la flexión, se decidió realizar un cálculo analítico para las tensiones de Von Mises. Teniendo en cuenta que el robot será utilizado para aplicaciones didácticas, en un futuro no se suspenderán grandes cargas sobre el punto efector, como se mencionó anteriormente en el apartado de simulaciones. Esto nos permite decir que la flexión es más influyente sobre la estructura que la torsión, pudiendo despreciar esta última en los cálculos analíticos. También, el esfuerzo cortante puede despreciarse frente a los efectos de la flexión.

El pórtico se encuentra vinculado mediante dos apoyos fijos, esto quiere decir que nos encontramos en presencia de un pórtico hiperestático. Debido a la complejidad de resolución analítica de este tipo

de estructuras, se optó por utilizar un software de cálculo estructural para determinar los esfuerzos internos denominado Ftool.

Las condiciones para el cálculo son las mismas que se establecieron para la simulación estática. Se aplicarán dos cargas de 200 N, una por cada lugar donde va montado cada servomotor. A continuación, en la siguiente tabla se resumen los datos utilizados para los cálculos:

Largo travesaño	1050	[mm]
Altura columnas	300	[mm]
Distancia del punto de aplicación de la carga	300	[mm]
Carga aplicada por servomotor	200	[N]
Espesor de chapa adoptado	3,175	[mm]
σ_{el}	294,8	[MPa]
FOS	3	[—]

Tabla 2: Condiciones para cálculo estructural.

Aplicamos las cargas sobre el pórtico:



Figura 47: Cargas aplicadas en FTool [Propia].

Se realizaron los cálculos y los resultados arrojados por el programa fueron los siguientes:

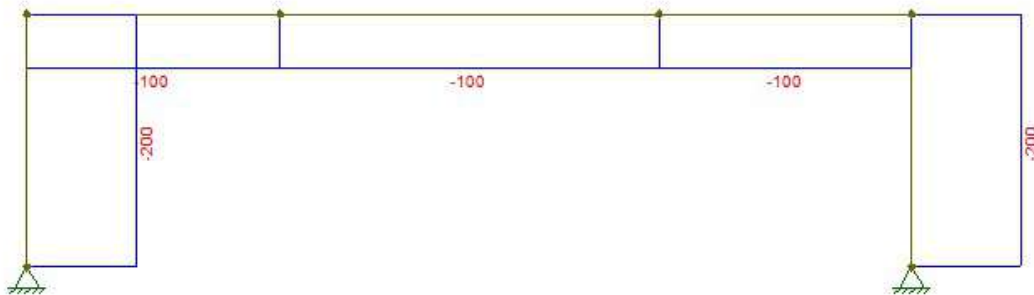


Figura 48: Esfuerzo normal en N [Propia].

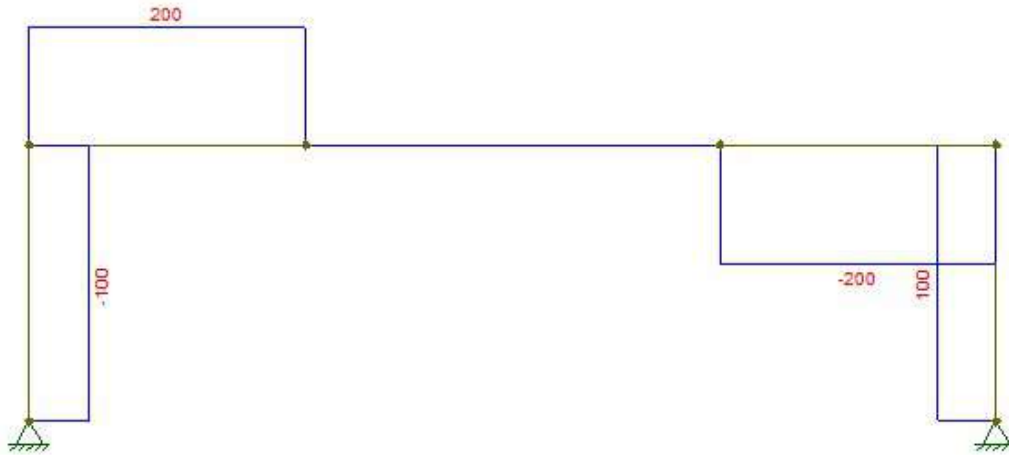


Figura 49: Esfuerzo cortante en N [Propia].

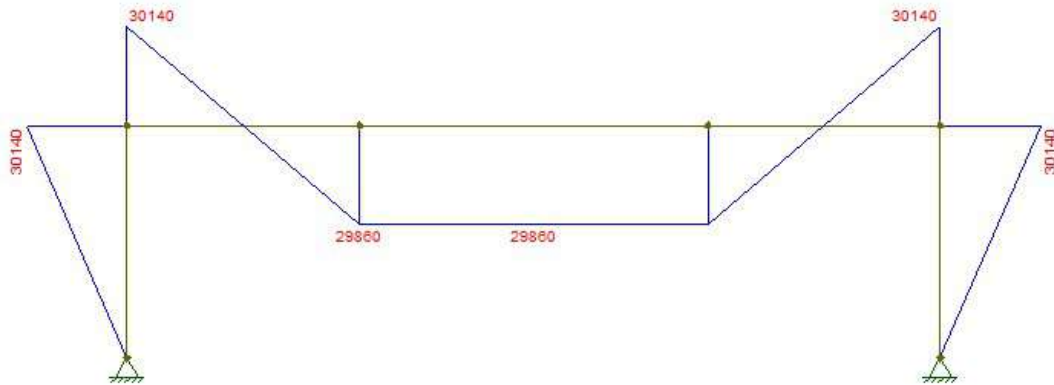


Figura 50: Momento flector en Nmm [Propia].

Una vez obtenidos los esfuerzos internos, se calculan las tensiones de Von Mises para los puntos críticos de las secciones transversales más comprometidas.

Utilizando el cálculo de propiedades para un área en Solidworks se determinó la posición del eje neutro y el valor de inercia de la sección, como se observa en la figura (52). Esto es válido tanto para el travesaño como para las columnas, ya que comparten las mismas dimensiones. Los puntos críticos a analizar serán los más alejados de la línea neutra y estos se indican en la figura (51).

La misma posee las siguientes propiedades, las cuales se utilizaron para los cálculos:

I_{xx}	296131,94	$[mm^4]$
Área	931,38	$[mm^2]$
Dist. máxima al eje neutro	46,77	$[mm]$

Tabla 3: Resumen propiedades sección transversal.

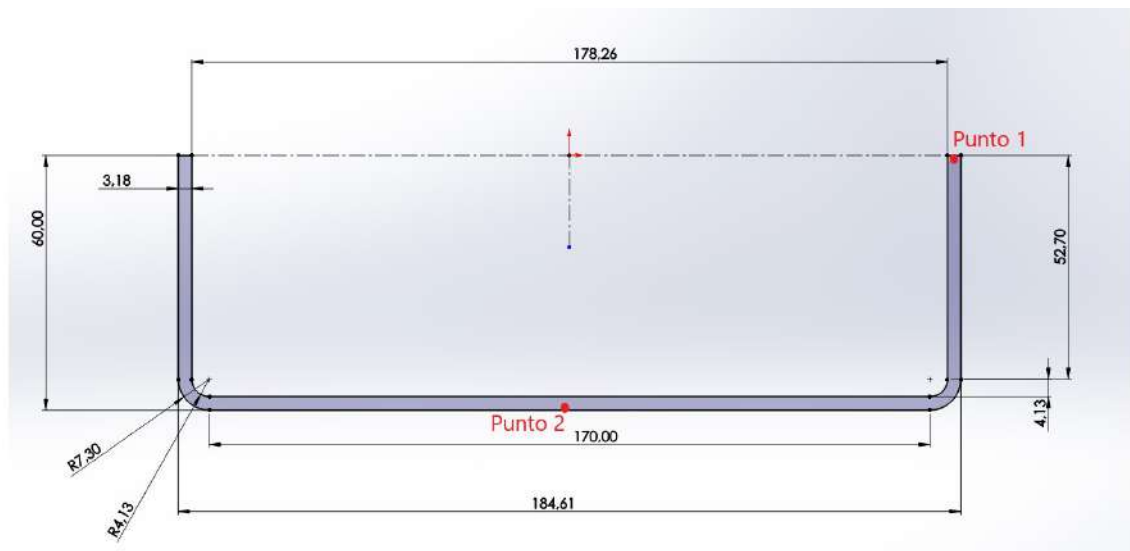


Figura 51: Puntos críticos sección transversal [Propia].

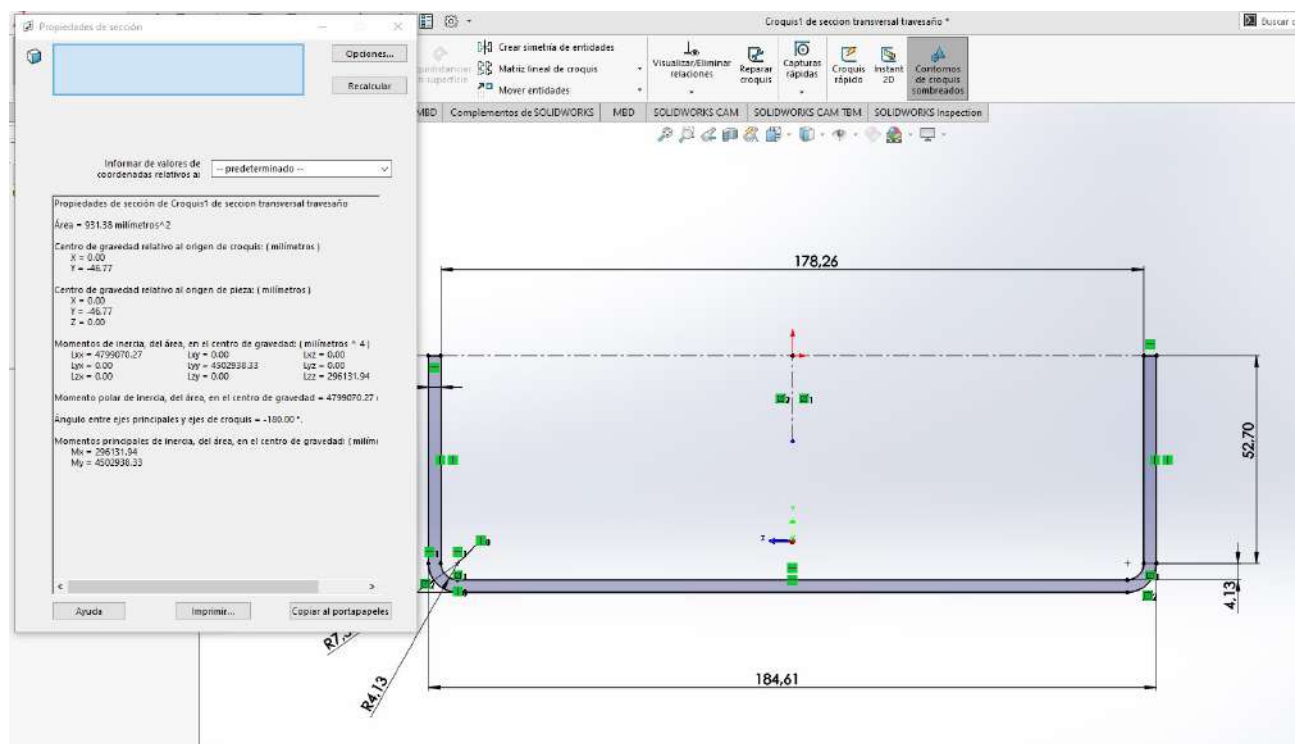


Figura 52: Propiedades sección transversal [Propia].

Para calcular la tensión de Von Mises se plantea el efecto de superposición entre el modelo de flexión pura y esfuerzo axial.

Teniendo en cuenta la convención de signos, el momento flector máximo es positivo y el esfuerzo normal es negativo. Esto hace que en el punto 1 las tensiones normales debido a ambos efectos sean de compresión y en el punto 2 las tensiones normales debidas a la flexión son de tracción.

A continuación se muestran las ecuaciones utilizadas para realizar los cálculos correspondientes tanto para el travesaño como para las columnas.

Punto 1:

$$\sigma_x = -\frac{N}{A} - \frac{cM}{I_{xx}} \quad (21)$$

Punto 2:

$$\sigma_x = -\frac{N}{A} + \frac{cM}{I_{xx}} \quad (22)$$

En cada punto se presenta un estado planar de tensiones, por lo que la tensión de Von Mises se calcula como:

$$\sigma_{vm} = \sqrt{\sigma_x^2 + 3\tau_{xy}^2} \quad (23)$$

Al no haber torsión $\tau_{xy} = 0$ resultando:

$$\sigma_{vm} = \sqrt{\sigma_x^2} = \sigma_x \quad (24)$$

La condición de resistencia a verificar es $\sigma_{vm} < \sigma_{adm}$.

En las capturas extraídas de la planilla de cálculo [7], podemos ver que la condición se verifica correctamente, estando los valores de tensión de Von Mises muy por debajo de la tensión admisible.

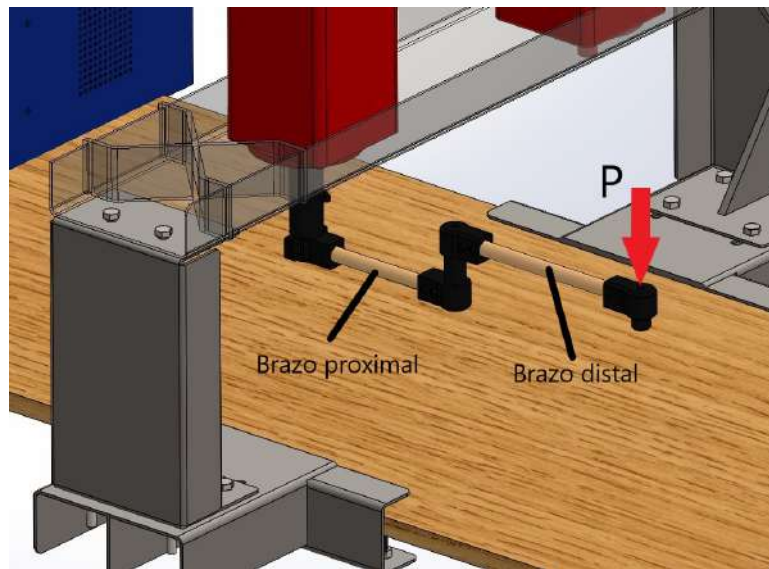
Travesaño			
Punto 1		Punto 2	
σ_N	0.19 [Mpa]	σ_N	0.11 [Mpa]
σ_{flex}	8.33 [Mpa]	σ_{flex}	2.36 [Mpa]
σ_{adm}	98.27 [Mpa]	σ_{adm}	98.27 [Mpa]
σ_{vm}	8.52 [Mpa]	σ_{vm}	2.46 [Mpa]
VERIFICA		VERIFICA	
Columna izq/der			
Punto 1		Punto 2	
σ_N	0.38 [Mpa]	σ_N	0.21 [Mpa]
σ_{flex}	8.33 [Mpa]	σ_{flex}	2.36 [Mpa]
σ_{adm}	98.27 [Mpa]	σ_{adm}	98.27 [Mpa]
σ_{vm}	8.71 [Mpa]	σ_{vm}	2.57 [Mpa]
VERIFICA		VERIFICA	

Figura 53: Cálculos tensiones de Von Mises [Propia].

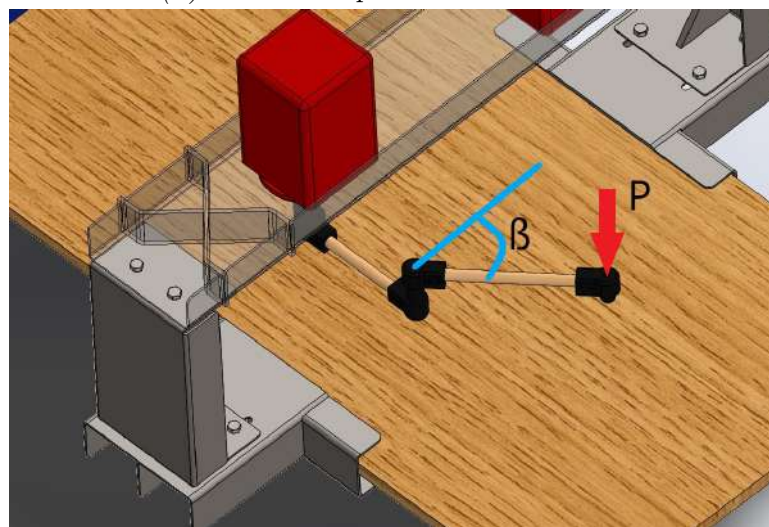
3.5. Cálculo carga máxima brazos

Debemos determinar si el robot es capaz de soportar carga suspendida sobre el punto efector. Para ello, teniendo en cuenta el material con el que se encuentra construido actualmente el robot, se calculó la fuerza máxima que son capaces de soportar los brazos.

Durante el posicionamiento del robot se pueden dar diferentes configuraciones en los brazos, donde en cada una se generarán esfuerzos de diferentes tipos. Se busca cubrir la mayor cantidad de posicionamientos posibles a partir de estudiar la cadena cinemática completamente estirada y cuando el brazo distal forma un ángulo de 90° con el proximal. Para realizar los cálculos, se estudiará un único brazo, realizando una apertura de la cadena cinemática. Esto nos pone en una situación más conservadora, ya que toda la carga se distribuye en una sola cadena cinemática en vez de en las dos. La figura (54) muestra lo explicado en este párrafo.



(a) Brazo completamente estirado.



(b) Brazo en situación intermedia.

Figura 54: Posiciones del brazo para analizar los esfuerzos [Propia].

Los esfuerzos internos se calcularon para la situación en donde el brazo se encuentra formando un ángulo entre 0° y 90° , de manera tal que se pueda generalizar el cálculo de las reacciones y esfuerzos internos en función de β . Cabe aclarar que se presentarán únicamente las ecuaciones finales, obviando el desarrollo realizado.

En primer lugar, se obtuvieron las reacciones en el empotramiento.

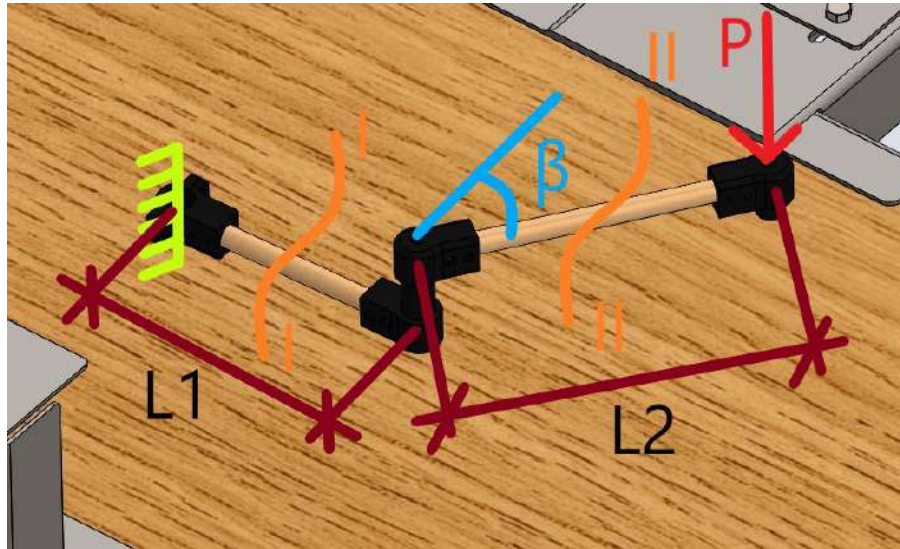


Figura 55: Esquema brazo en situación intermedia [Propia].

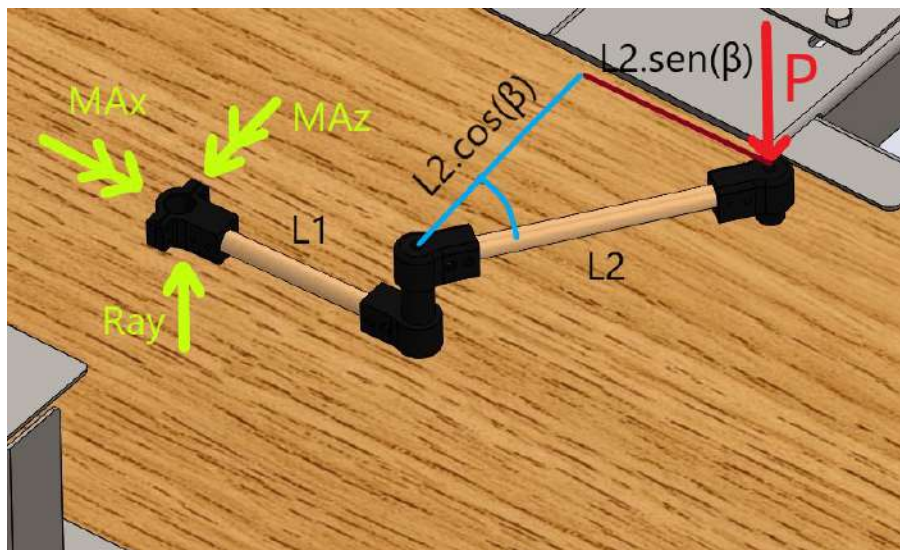


Figura 56: Reacciones en los apoyos [Propia].

$$\sum M_y = 0; \sum F_x = 0; \sum F_z = 0$$

$$\sum M_x^A = 0 \Rightarrow M_{Ax} = P \cdot L_2 \cdot \cos(\beta)$$

$$\sum M_z^A = 0 \Rightarrow M_{Az} = P.(L_1 + L_2.\text{sen}(\beta))$$

$$\sum F_y = 0 \Rightarrow R_{Ay} = P$$

Ahora, obtenemos los esfuerzos internos:

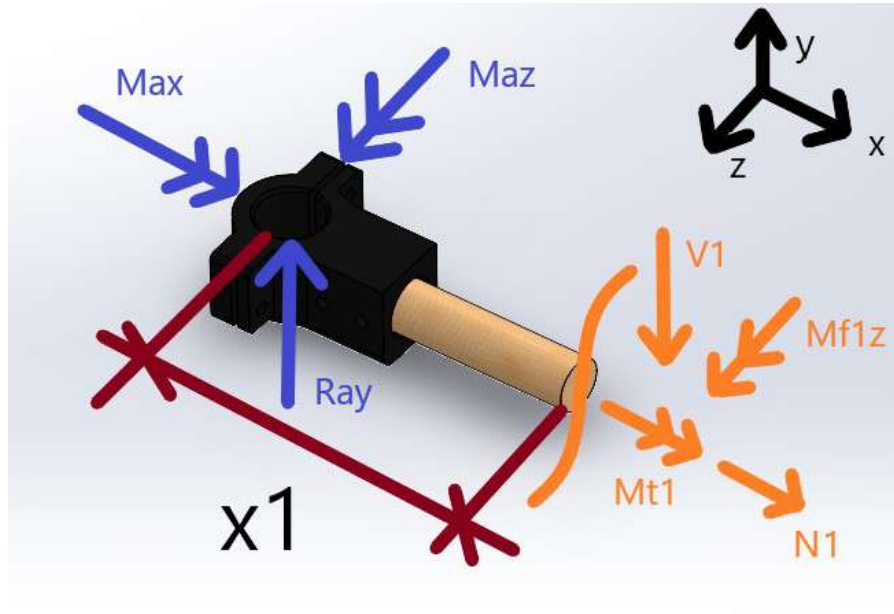


Figura 57: Sección I-I [Propia].

$$\sum F_x = 0 \Rightarrow N_1 = 0$$

$$\sum F_y = 0 \Rightarrow V_1 = P$$

$$\sum M_x^I = 0 \Rightarrow Mt_1 = -P.L_2.\cos(\beta)$$

$$\sum M_z^I = 0 \Rightarrow Mf_{1z} = P.x_1 - P.(L_1 + L_2.\text{sen}(\beta)) \quad (25)$$

Luego, se resuelven los esfuerzos internos para la sección II-II por derecha, ya que los cálculos resultarán más simples de desarrollar.

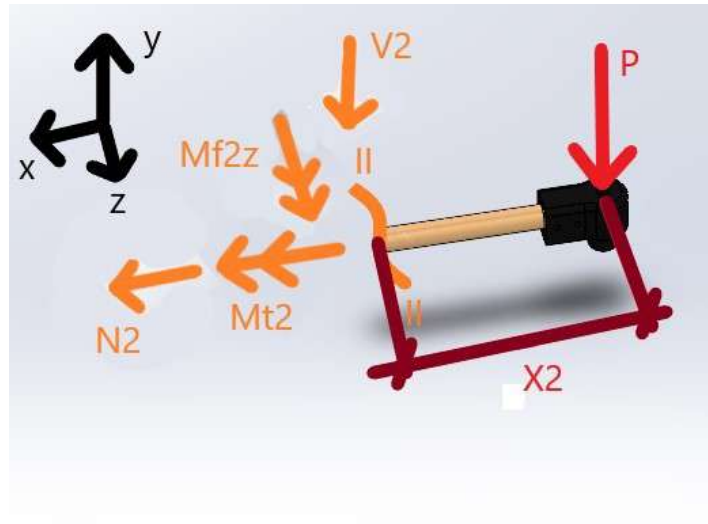


Figura 58: Sección I-I [Propia].

$$\sum F_x = 0 \Rightarrow N_2 = 0$$

$$\sum F_y = 0 \Rightarrow V_2 = P$$

$$\sum M_x^I = 0 \Rightarrow Mt_2 = 0$$

$$\sum M_z^I = 0 \Rightarrow Mf_{2z} = P \cdot x_2$$

Para determinar la carga P máxima que pueden soportar los brazos, debemos analizar los puntos críticos de la sección transversal más comprometida, la cual es donde la combinación entre el momento flector y momento torsor es máxima. Al tratarse de una sección circular, los puntos críticos se producirán en los extremos, como se visualiza en la figura (59). El esfuerzo cortante puede despreciarse en este análisis.



Figura 59: Puntos críticos sección transversal brazos [Propia].

Utilizando el criterio de resistencia de Von Mises se plantea la siguiente condición a cumplir:

$$\sigma_{vm} < \sigma_{adm}$$

Donde

$$\sqrt{\sigma_x^2 + 3\tau_{xy}^2} < \sigma_{adm}$$

Aplicando el modelo de flexión y de torsión la fórmula de la tensión de Von Mises resulta:

$$\sqrt{\left(\frac{32Mf_{max}}{\pi d^3}\right)^2 + \left(\frac{16Mt}{\pi d^3}\right)^2} < \sigma_{adm}$$

Para determinar en qué configuración los brazos se encuentran en la peor condición, se realizaron los cálculos de tensiones de Von Mises desde $\beta = 0^\circ$ hasta $\beta = 90^\circ$ con intervalos de 10° . Los cálculos se realizaron en la hoja de cálculos para ambos puntos críticos y se obtuvieron los siguientes resultados:

Largo brazo proximal	L1	200,00	[mm]
Largo brazo distal	L2	250,00	[mm]
Inclinación brazo distal	β	90,00	[°]
Diámetro brazo	$\emptyset$	16,00	[mm]
Límite de tensión elástica material brazos	σ_{el}	50,52	[Mpa]
Factor de seguridad operativo	FOS	1,00	[-]
Carga aplicada sobre el punto efector	P	5,00	[N]
		0,51	[kg]

Figura 60: Datos brazos [Propia].

I_{xx}	3216,99	[mm ⁴]
J	6433,98	[mm ³]
Área	201,06	[mm ²]
Dist. máxima al eje neutro	8	[mm]

Tabla 4: Resumen propiedades sección transversal brazos.

Betha [°]	Cálculo reacciones			Cálculo esfuerzos internos			
90				Sección I-I		Sección II-II	
	Ray	5.00	[N]	V1	5.00	V2	5.00 [N]
	Maz	2250.00	[Nmm]	Mt1	0.00	Mt2	0 [Nmm]
	Max	0.00	[Nmm]	Mf1	-2250	Mf2	-1250 [Nmm]
				x1	0	x2	250 [mm]

(a) Reacciones y esfuerzos internos.

Brazo proximal					
Punto 1			Punto 2		
τ	0,00	[Mpa]	τ	0,00	[Mpa]
σ_{flex}	5,60	[Mpa]	σ_{flex}	5,60	[Mpa]
σ_{adm}	50,52	[Mpa]	σ_{adm}	50,52	[Mpa]
σ_{vm}	5,60	[Mpa]	σ_{vm}	5,60	[Mpa]
VERIFICA			VERIFICA		

(b) Tensiones de Von Mises.

Figura 61: Cálculos $\beta = 90^\circ$ [Propia].

0				Sección I-I		Sección II-II	
	Ray	5.00	[N]	V1	5.00	V2	5.00 [N]
	Maz	1000.00	[Nmm]	Mt1	-1250.00	Mt2	0 [Nmm]
	Max	1250.00	[Nmm]	Mf1	-1000	Mf2	-1250 [Nmm]
				x1	0	x2	250 [mm]

(a) Reacciones y esfuerzos internos.

Punto 1			Punto 2		
τ	1,55	[Mpa]	τ	1,55	[Mpa]
σ_{flex}	2,49	[Mpa]	σ_{flex}	2,49	[Mpa]
σ_{adm}	50,52	[Mpa]	σ_{adm}	50,52	[Mpa]
σ_{vm}	2,93	[Mpa]	σ_{vm}	2,93	[Mpa]
VERIFICA			VERIFICA		

(b) Tensiones de Von Mises.

Figura 62: Cálculos $\beta = 0^\circ$ [Propia].

Podemos observar que las mayores tensiones de Von Mises en la sección crítica se producen cuando el brazo se encuentra totalmente estirado, por lo cual la máxima carga que son capaces de soportar los mismos se calculará para dicha condición.

La fuerza máxima a soportar se despeja de la fórmula de Von Mises. Cuando los brazos están completamente estirados se produce flexión pura, por lo cual la fórmula de Von Mises resulta:

$$\sigma_{vm} = \frac{32Mf_{max}}{\pi d^3}$$

Estableciendo la condición de resistencia, despejando el momento flector máximo y reemplazando por la expresión obtenida en la ecuación (25), obtenemos:

$$\frac{32Mf_{max}}{\pi d^3} < \sigma_{adm}$$

Como se considera FOS = 1, entonces:

$$\frac{32Mf_{max}}{\pi d^3} < \sigma_{el}$$

$$\frac{32(P.x_1 - P.(L_1 + L_2.\text{sen}(90)))}{\pi d^3} < \sigma_{el}$$

El momento flector máximo se produce para $x_1 = 0$, resultando:

$$\frac{32P(L_1 + L_2)}{\pi d^3} < \sigma_{el}$$

Despejando P:

$$P < \frac{\sigma_{el}\pi d^3}{32(L_1 + L_2)} = 15,05N = 1,53kg$$

El valor de tensión de límite elástico de la madera (ver figura (63)) se consiguió en el estudio que se presenta en [] a partir de conocer el módulo elástico y plantear la regla del 0,2 % utilizando la ley de Hooke, suponiendo que el material se deforma elásticamente. Se adoptó el valor mínimo existente para tomar la condición más conservadora. Verdaderamente se desconoce de qué tipo de madera están contruidos los brazos, pero este valor obtenido nos permite ver que para una próxima etapa en donde se implemente un actuador sobre el robot para trasladar objetos, se deben actualizar los materiales y juntas utilizados actualmente.

Módulos de rotura y elasticidad a la flexión estática

Especie		Flexión estática	
Nombre común	Nombre científico	MOR (kg/cm ²)	MOE (kg/cm ²)
Alecrín	<i>Holocalix balansae</i>	1117.9	203596
Anchico blanco	<i>Albizia hasleri</i>	1000	113750
Anchico colorado	<i>Piptadenia rigida</i>	1191	157801
Camboatá blanco	<i>Cupania vernalis</i>	797.6	180619
Camboatá colorado	<i>Matayba eleagnoides</i>	636.7	117415
Eucaliptus duni	<i>Eucaliptus dunnii</i>	536	45774
Grevilea	<i>Grevillea robusta</i>	521	45693
Guatambú amarillo	<i>Aspidosperma australe</i>	1420	136000
Guatambú blanco	<i>Balfourodendron riedelianum</i>	1250	115300
Guayaibí	<i>Patagonula americana</i>	1360	110000
Guayaibí amarillo	<i>Terminalia triflora</i>	1220	115700
Hovenia	<i>Hovenia dulces</i>	1154	108075
Kiri	<i>Pawlonia sps.</i>	257	25261
Lapacho negro	<i>Tabebuia ipe</i>	1300	157000
Laurel amarillo	<i>Nectandra lanceolata</i>	730	89740
Loro blanco	<i>Bastardiopsis densiflora</i>	612	59440
Mora amarilla	<i>Clorophora tinctoria</i>	1400	149000
Mora colorada	<i>Clorophora tinctoria</i> var. <i>polyneura</i>	1305	133000
Paraíso	<i>Melia azedarach</i>	853	72505
Persigüero bravo	<i>Prunus subcoriácea</i>	1700	142000
Peteribí	<i>Cordia tricótoma</i>	825	110000
Pino elioti	<i>Pinus elliotii</i>	920	52416
Pino Paraná de reforestaciones	<i>Araucaria angustifolia</i>	872	113103
Pino Paraná natural	<i>Araucaria angustifolia</i>	708	100400
Pino ponderosa	<i>Pinus ponderosa</i>	658	90000
Pino taeda	<i>Pinus taeda</i> L.	776	63784
Rabo itá	<i>Lonchocarpus leucanthus</i>	1158.6	205196
Rabo molle	<i>Lonchocarpus muehlbergianus</i>	551.6	111461
Urunday	<i>Astronium balansae</i>	1210	109500
Urundel	<i>Astronium urundeuva</i>	1626	159000

Figura 63: Datos módulo elasticidad y rotura maderas [Propia].

3.6. Simulación de movimientos

La resolución de la cinemática para conocer la posición del robot se realiza a través de Matlab. El entorno de programación utilizado para la automatización de los servomotores no tiene la capacidad de trabajar con números flotantes y realizar cálculos, por lo cual se utilizó el software mencionado anteriormente.

Se construyó un programa que combina la cinemática inversa y la cinemática directa. Utilizando y ampliando lo realizado en [2]. El mismo posee los siguientes objetivos:

- Calcular los puntos de la trayectoria a realizar.
- Mediante cinemática inversa obtener los ángulos que deben adoptar los brazos para posicionar el robot a partir de un modo de trabajo definido.
- Con los ángulos calculados, mediante cinemática directa obtener la posición de todos los eslabones que conforman el robot para los puntos definidos.
- Mostrar gráficamente la evolución de la posición del robot conforme los brazos adoptan los diferentes ángulos, para visualizar como se comporta al realizar la trayectoria y donde se posiciona.
- Transformar los ángulos de los brazos a pulsos para poder ingresarlos como dato en el programa realizado para la automatización de los movimientos.

3.6.1. Selección de trayectorias

Las trayectorias definidas que realizará el robot son:

- Cuatro líneas horizontales.
- Cuatro líneas verticales.
- Cuatro líneas diagonales.
- Cuatro cuadrados.
- Cuatro triángulos.
- Cuatro circunferencias.
- Una trayectoria de zig-zag la cual haría el robot en caso de trasladar un objeto.
- Un logo.

Los puntos en el plano cartesiano para las trayectorias correspondientes a líneas y circunferencias, se obtuvieron utilizando sus respectivas fórmulas. A modo de ejemplo, se observa en la figura (64) los puntos definidos para una circunferencia de radio r , las demás trayectorias podrán observarse en los programas que se encuentran en el anexo. Los puntos en el plano cartesiano de figuras como cuadrados, triángulos, trayectoria de zig-zag y logo se obtuvieron combinando líneas.

```

switch circ
case 1
    r = 10; %Radio de la circunferencia [mm]
case 2
    r = 20; %Radio de la circunferencia [mm]
case 3
    r = 30; %Radio de la circunferencia [mm]
case 4
    r = 50; %Radio de la circunferencia [mm]
end

alpha = linspace(2*pi,0 , 20); % Ángulos de 0 a 2*pi
xp = r * cos(alpha)+desfasaje_0;
yp = r * sin(alpha)+330-r;

```

Figura 64: Cálculo puntos (x,y) circunferencia [Propia].

Para cada trayectoria se definió una función de Matlab, las cuales se invocarán desde un programa principal. En el caso de las líneas, triángulos, cuadrados y circunferencias, cada función recibirá un parámetro determinado que se definirá desde el programa principal, el cual determinará el tamaño de la línea, triángulo, cuadrado o circunferencia a realizar.

En el programa principal se ingresan los parámetros fundamentales para realizar los cálculos mediante cinemática inversa y directa. Se debe ingresar la distancia entre servomotores y los largos de los brazos del robot. También, se define el modo de trabajo en el que se encontrará el robot al posicionarse en los respectivos puntos definidos para cada trayectoria. Para no complejizar la programación, todas las trayectorias que realizará el robot en el presente proyecto las hará posicionado en el modo de trabajo (++) definido en la tabla 1.

```

modo_tbj =1; %ind
trayectoria = 1;
arh = 1; % si arh
arv = 0; % si arv
% si arh = arv =

rh = 4; % rh = 1
rv = 4; % rv = 1
circ = 4; % circ
rd = 4; % rd = 1
cuad = 4; % cuad
triang = 4 ;% tri
graf = 1; % graf

```

Figura 65: Selección de trayectorias y modo de trabajo [Propia].

Variable "modo tbj": permite seleccionar entre el modo de trabajo que adoptará el robot al posicionarse en determinado punto, como se mencionó anteriormente.

- modo tbj = 1 → modo de trabajo ++
- modo tbj = 2 → modo de trabajo +-
- modo tbj = 3 → modo de trabajo -+
- modo tbj = 4 → modo de trabajo --

Variable "trayectoria": permite seleccionar el tipo de trayectoria a realizar. Al definir el valor de la variable, se utiliza una sentencia case (ver figura (66)), para invocar la función correspondiente a cada trayectoria.

- trayectoria = 1 → circunferencia
- trayectoria = 2 → recta
- trayectoria = 3 → triángulo
- trayectoria = 4 → cuadrado
- trayectoria = 5 → logo
- trayectoria = 6 → traslado zig-zag

De cada trayectoria a realizar, excepto el logo y el traslado en zig-zag, se pueden realizar trayectorias con diferentes medidas. Para seleccionar una medida específica se deben utilizar las siguientes variables:

- **rh:** permite seleccionar entre 4 longitudes diferentes para la recta horizontal.
- **rv:** permite seleccionar entre 4 longitudes diferentes para la recta vertical.
- **rv:** permite seleccionar entre 4 longitudes diferentes para la recta diagonal.
- **circ:** permite seleccionar entre 4 radios diferentes para la circunferencia.
- **cuad:** permite seleccionar entre 4 longitudes de lado diferentes para el cuadrado.
- **triang:** permite seleccionar entre 4 longitudes de base diferentes para el triángulo.

Específicamente para las rectas, a fin de utilizar una sola función para graficar las mismas y no asignarles tantos valores posibles de adoptar a la variable "trayectoria", también se utilizan las variables "arh" y "arv" las cuales a partir de las siguientes combinaciones de valores (previamente definido trayectoria = 2) permiten:

- $arv = 0$ y $arh = 1 \rightarrow$ realiza una recta vertical.
- $arv = 1$ y $arh = 0 \rightarrow$ realiza una recta horizontal.
- $arv = 0$ y $arh = 0 \rightarrow$ realiza una recta diagonal.

La variable "graf" permite activar ($= 1$) o desactivar ($= 0$) la simulación gráfica del movimiento del robot.

```
switch trayectoria
case 1 %realiza una circunferencia
[dita1_g, dita2_g, xc, yc] = cin_inv_SCARA_circ(L1,L2,modo_tbj,d,circ);
```

Figura 66: Sentencia case para selección de trayectoria [Propia].

Ya calculados los ángulos que deben adoptar los brazos del robot respecto de la horizontal, utilizando la cinemática inversa para que el mismo siga la trayectoria definida, se ingresan los ángulos obtenidos a la función que, mediante la cinemática directa descrita en este informe, obtendrá la posición de todos los eslabones que conforman la cadena cinemática del robot para luego graficar su posición para cada par de ángulos calculados y ver cómo realiza el movimiento.

```
for i = 1:length(theta1)
[rOC_sol1, rOC_sol2, O1, O2, rOA1, rOA2, rO1A1, rO2A2] = cin_dir_SCARA(L1,L2,theta1(i),theta2(i),d);
% Mostrar resultados
fprintf('Posición %.f\n',i);
fprintf('Solución 1: x1 = %.2f, y1 = %.2f\n', rOC_sol1(1), rOC_sol1(2));
fprintf('Solución 2: x2 = %.2f, y2 = %.2f\n', rOC_sol2(1), rOC_sol2(2));
fprintf('\n',i);
```

Figura 67: Invocación función cinemática directa [Propia].

La estructura for se utiliza para ingresar como dato en la función cada par de ángulos calculados y que la misma devuelva el punto correspondiente de ubicación del punto efector.

3.6.2. Pasaje de ángulos en grados a pulsos

Obtenidos los ángulos de entrada, se deben calcular los mismos en pulsos de servomotor para ingresar en la programación de los servomotores. Para ello, debemos tener en cuenta que una vuelta completa de cada servomotor equivale a 4096 pulsos y que el reductor de ejes paralelos acoplado al servomotor posee un índice de reducción de 19,71. El resultado obtenido mediante la cinemática inversa da como resultado los ángulos indicados en la figura (21) pero, debido a la forma en la cual el robot realiza el 0 máquina, se deben ingresar en pulsos a la programación en IposPLUS los ángulos indicados en la figura (22). Para ello, debemos restar 180° al ángulo obtenido, como se observa en la figura (68).

```
dita2_g_b = 180 - dita2_g;
dita1_p = -1*round((dita1_g/360)*ir*pulso); % paso el angulo de grados a pulsos para cargar en el servo
dita2_p = round(1*(dita2_g/360)*ir*pulso);
dita2_p_b = round(1*(dita2_g_b/360)*ir*pulso);
```

Figura 68: Cálculos pulsos servomotores [Propia].

3.6.3. Gráficas trayectorias y evolución del movimiento

Las gráficas de cada trayectoria, espacio de trabajo del robot y evolución del movimiento se realizaron en un mismo plot de Matlab.

Se representó el espacio de trabajo graficando las circunferencias que definen al mismo a partir de su intersección.

```
%% Cálculo espacio de trabajo robot scara paralelo

r = L1 + L2; % Radio de la circunferencia suma de la longitud de ambos brazos
alpha = linspace(0, 2*pi, 100); % Ángulos de 0 a 2*pi
x1 = r * cos(alpha)-d/2;
y1 = r * sin(alpha);

x2 = r * cos(alpha)+d/2;
y2 = r * sin(alpha);
```

Figura 69: Cálculos gráfica espacio de trabajo [Propia].

$(x1,y1)$ y $(x2,y2)$ corresponden a las coordenadas de las circunferencias que se intersecan para formar el espacio de trabajo respectivamente.

Las gráficas de las trayectorias se obtienen a partir de los arreglos x_c e y_c . Estos poseen las coordenadas de los puntos (x,y) correspondientes a la gráfica seleccionada.

El movimiento del robot se logra realizando una gráfica la cual tenga la capacidad de actualizarse a partir de actualizar los valores de posición del robot. Inicialmente se crean gráficos vacíos (ver figura (70a)) , los mismos, como se mencionó anteriormente, se irán actualizando a medida que la función `for` vaya iterando. Por último, para actualizar la gráfica se utilizan los comandos `set()` que se visualizan en la figura (70b).

Las gráficas correspondientes al espacio de trabajo y trayectorias permanecerán fijas en todo momento.

```

figure;
hold on; axis equal; grid on;
xlim([-420 420]); ylim([-400 400])
set(gcf, 'WindowState', 'maximized');
title('Disposición robot SCARA paralelo');
brazo_dist1 = plot([0,0],[0,0],'b-','LineWidth',2);
brazo_dist2 = plot([0,0],[0,0],'b-','LineWidth',2);
brazo_dist1_sol2 = plot([0,0],[0,0],'g--','LineWidth',2);
brazo_dist2_sol2 = plot([0,0],[0,0],'g--','LineWidth',2);
brazo_prox1 = plot([0,0],[0,0],'r-','LineWidth',2);
brazo_prox2 = plot([0,0],[0,0],'r-','LineWidth',2);
efector = plot(0, 0, 'ko', 'MarkerFaceColor', 'g', 'MarkerSize', 8); % efector
efector_sol2 = plot(0, 0, 'ko', 'MarkerFaceColor', 'm', 'MarkerSize', 8); % efector solucion 2
motor1 = plot(-d/2, 0, 'ko', 'MarkerFaceColor', 'k');
motor2 = plot(d/2, 0, 'ko', 'MarkerFaceColor', 'k');
plot(x1, y1, 'k');
plot(x2, y2, 'k');
plot(xc, yc, 'm');

```

(a) Inicialización gráfica.

```

% Actualizar líneas y puntos
set(brazo_prox1, 'XData', [O1(1), rOA1(1)], 'YData', [O1(2), rOA1(2)]);
set(brazo_prox2, 'XData', [O2(1), rOA2(1)], 'YData', [O2(2), rOA2(2)]);
set(brazo_dist1, 'XData', [rOA1(1), rOC_sol1(1)], 'YData', [rOA1(2), rOC_sol1(2)]);
set(brazo_dist2, 'XData', [rOA2(1), rOC_sol1(1)], 'YData', [rOA2(2), rOC_sol1(2)]);
set(brazo_dist1_sol2, 'XData', [rOA1(1), rOC_sol2(1)], 'YData', [rOA1(2), rOC_sol2(2)]);
set(brazo_dist2_sol2, 'XData', [rOA2(1), rOC_sol2(1)], 'YData', [rOA2(2), rOC_sol2(2)]);
set(efector, 'XData', rOC_sol1(1), 'YData', rOC_sol1(2)); %modo de ensamblaje 1.
set(efector_sol2, 'XData', rOC_sol2(1), 'YData', rOC_sol2(2)); %modo de ensamblaje 1.
drawnow;
pause(0.08); % control de velocidad

```

(b) Actualización gráfica.

Figura 70: Gráfica movimiento [Propia].

A continuación, se adjuntan capturas de los tipos de trayectorias a realizar por el robot a modo de ejemplo.

Recta horizontal:

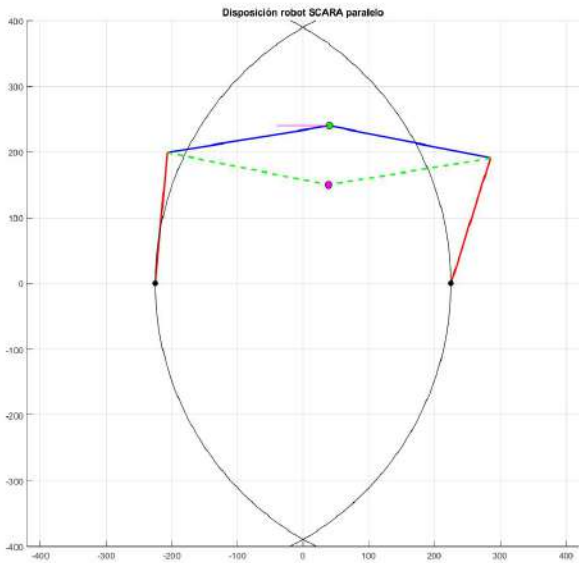


Figura 71: Simulación movimiento robot para recta horizontal [Propia].

Recta vertical:

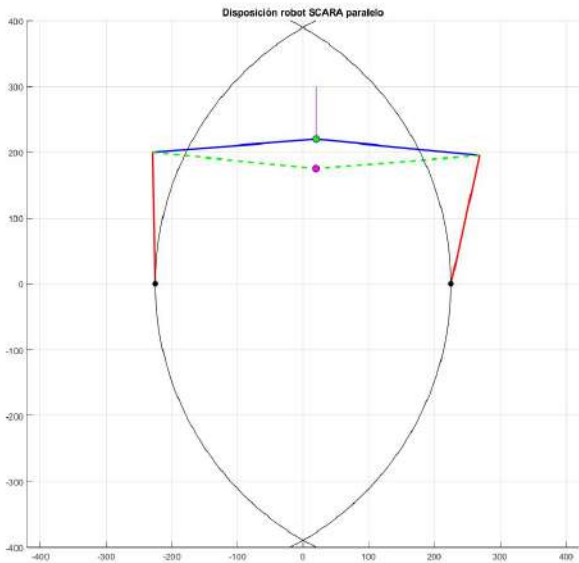


Figura 72: Simulación movimiento robot para recta vertical [Propia].

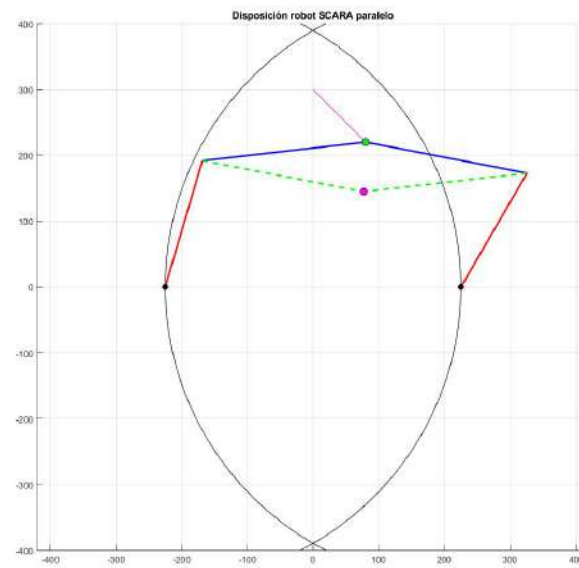
Recta diagonal:

Figura 73: Simulación movimiento robot para recta diagonal [Propia].

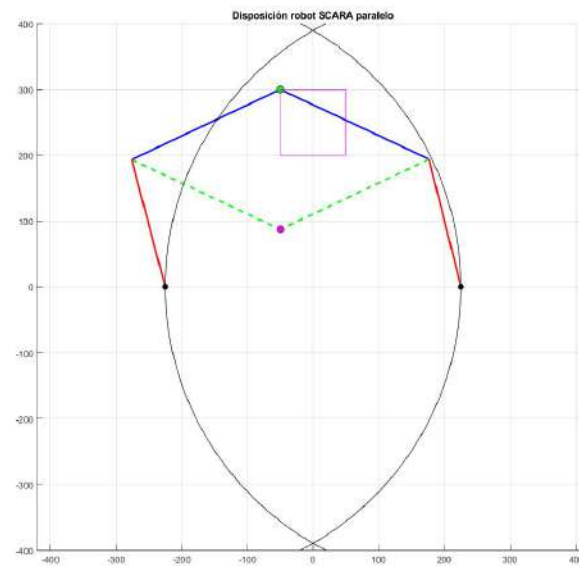
Cuadrado:

Figura 74: Simulación movimiento robot para cuadrado [Propia].

Triángulo:

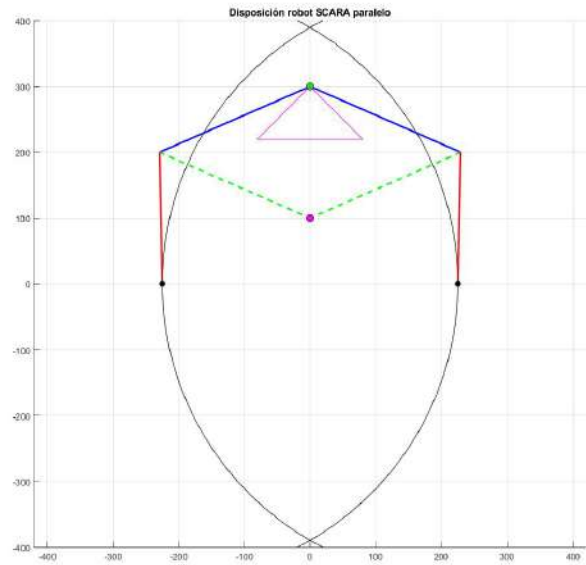


Figura 75: Simulación movimiento robot para triángulo [Propia].

Circunferencia:

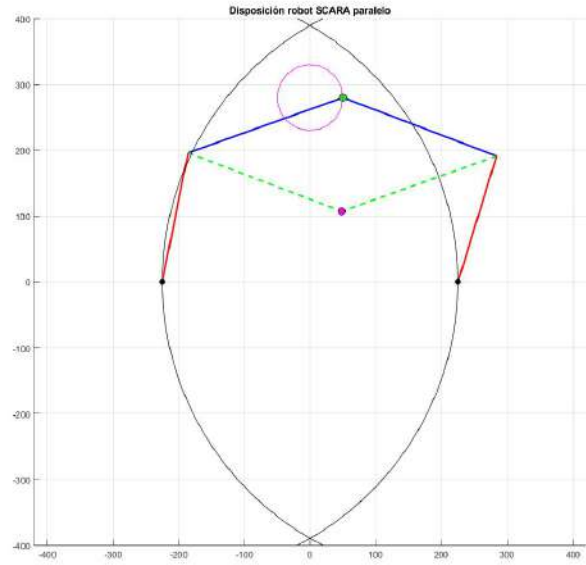


Figura 76: Simulación movimiento robot para circunferencia [Propia].

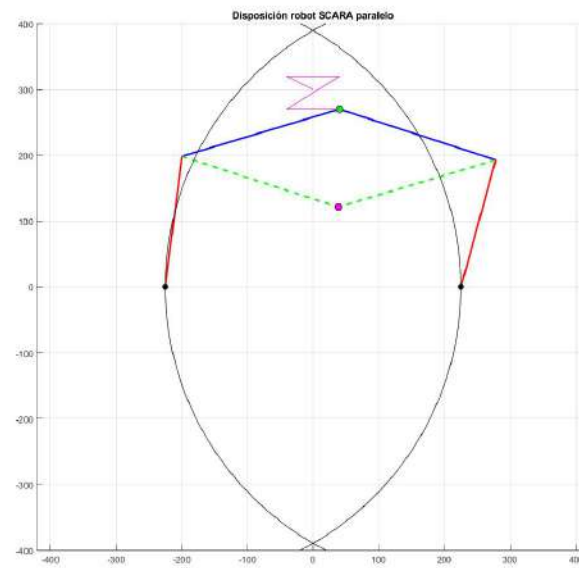
Traslado zig-zag:

Figura 77: Simulación movimiento robot para traslado zig-zag [Propia].

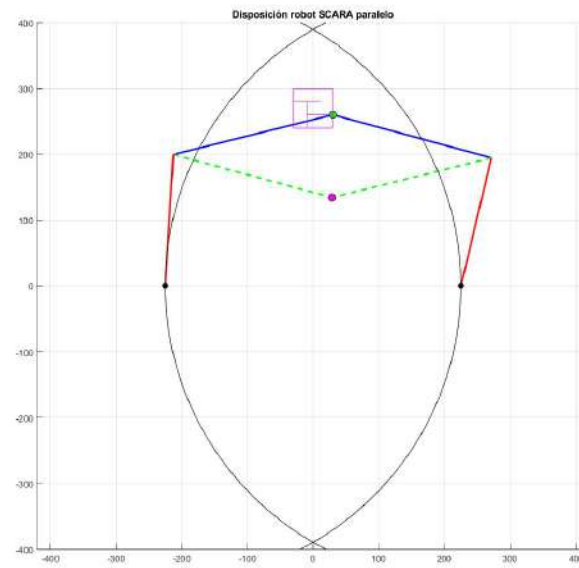
Logo:

Figura 78: Simulación movimiento robot para logo [Propia].

3.7. Automatización

La automatización del robot se logró a partir de realizar la programación en IposPLUS compiler. IposPLUS compiler es un entorno de programación el cual, mediante la escritura de un código en lenguaje C, permite realizar programas para controlar los servomotores a partir de una secuencia de tareas. Dicho entorno de programación se encuentra dentro del software libre de ingeniería ofrecido por el fabricante de los servomotores SEW Eurodrive que se denomina Movitools. Dentro de Movitools podemos parametrizar los variadores vectoriales, crear proyectos, poner en marcha los servomotores, visualizar su diagnóstico, etc. También se utiliza la extensión AppBuilder para lograr la interfaz humano-máquina.

En esta sección se presentarán los programas desarrollados para lograr las trayectorias que se definieron en la sección anterior. La explicación de cómo se configuran los servomotores, funcionamiento del entorno de programación y la utilización de AppBuilder se encuentran en [2].

Como se mencionó anteriormente, los servomotores contaban con una automatización inicial realizada en [2]. De la misma se mantendrán las funciones AngGiro1() y Homming(). Además, la comunicación para el funcionamiento en conjunto de los servomotores seguirá siendo de maestro-esclavo, por lo cual se utilizarán las estructuras MOVLINK y MLDATA para el pasaje de datos entre los mismos. Como bien se menciona en [2] el programa del controlador esclavo únicamente recibe y devuelve valores de variables, por lo que no se vio modificado en esta nueva etapa de programación, ya que cumple con su función correctamente. Las funciones y estructuras mencionadas pueden visualizarse en el anexo B.

Cada trayectoria a realizar se consideró dividida en veinte intervalos de ángulos, para lograr que el robot siga dicha trayectoria, ya que al moverse los servomotores con velocidad constante la trayectoria que realiza el punto efector entre ángulo y ángulo es aleatoria. Para ingresar estos valores de ángulo, se definieron veinte variables (figura (80)), donde cada par de variables adopta los pares de ángulos en pulsos para cada punto a atravesar por el punto efector para realizar la trayectoria. También se definieron nuevas variables necesarias para el funcionamiento del programa; las mismas pueden observarse en la figura (79).

```
//ANGULOS DITA, ESTADOS y MODOS
#define finTRAY H128 // indica fin de trayectoria para salir del while
#define home H129 // perdido( estado inicial) lost=1, after homming lost=0 (SERVO 1)
#define dita1 H130 // valor de ángulo theta1 en pulsos
#define dita2 H131 // valor de ángulo theta2 en pulsos
#define st1 H132 // estado servo 1 ( moving=0; ready=1; finish=2; lost=3)
#define st2 H133 // estado servo 2 ( moving=0; ready=1; finish=2; lost=3)
#define stop H134 //parada
#define cpr H135 //numero de posicion puntos intermedios
#define TRAY H136 // permite seleccionar trayectoria
#define INT H137 //se usa para pasar del punto inicial de la trayectoria a los demas y comenzar
#define band_home H138 // se utiliza para hacer home con los switches una sola vez
#define b_stop H139 // bandera para activar estado de espera previo al home luego de usar STOP
#define START H140 // habilita el posicionamiento al primer punto
#define seg H145 // indica que segmento se realizo en el logo
```

Figura 79: Definición variables generales [Propia].

```

#define p1_dita1 H17 //
#define p1_dita2 H18 //
#define p2_dita1 H19 //
#define p2_dita2 H20 //
#define p3_dita1 H21 //
#define p3_dita2 H22 //
#define p4_dita1 H23 //
#define p4_dita2 H24 //
#define p5_dita1 H25 //
#define p5_dita2 H26 //
#define p6_dita1 H27 //
#define p6_dita2 H28 //
#define p7_dita1 H29 //
#define p7_dita2 H30 //
#define p8_dita1 H31 //
#define p8_dita2 H32 //
#define p9_dita1 H33 //
#define p9_dita2 H34 //
#define p10_dita1 H35 //
#define p10_dita2 H36 //
#define p11_dita1 H37 //
#define p11_dita2 H38 //
#define p12_dita1 H39 //
#define p12_dita2 H40 //
#define p13_dita1 H41 //
#define p13_dita2 H42 //
#define p14_dita1 H43 //
#define p14_dita2 H44 //
#define p15_dita1 H45 //
#define p15_dita2 H46 //
#define p16_dita1 H47 //
#define p16_dita2 H48 //
#define p17_dita1 H49 //
#define p17_dita2 H50 //
#define p18_dita1 H51 //
#define p18_dita2 H52 //
#define p19_dita1 H53 //
#define p19_dita2 H54 //
#define p20_dita1 H55 //
#define p20_dita2 H56 //

```

Figura 80: Definición variables para θ_1 y θ_2 [Propia].

A continuación se mencionan las variables de mayor relevancia:

- **TRAY**: indica un número que permitirá seleccionar dentro de un case la trayectoria a realizar.
- **START**: se utiliza para posicionar el robot en el primer punto de la trayectoria a realizar.
- **INI**: da comienzo a la realización de la trayectoria.
- **cpr**: indica en que punto intermedio de la trayectoria se encuentra el robot.
- **finTRAY**: indica que finalizó la trayectoria para permitir nuevamente al usuario seleccionar una trayectoria nueva.

3.7.1. Función trayectoria

Ya definida la trayectoria a realizar y asignados los valores correspondientes de ángulos en pulsos, esta función se encarga de asignar los valores definidos a las variables `dita1` y `dita2`, para invocar a la función `AngGiro1()` y realizar el movimiento de los servomotores. Al presionar START, la función le asignará los valores a `dita1` y `dita2` para que, mediante la función `AngGiro1()` los servos se muevan la cantidad de pulsos correspondientes para posicionar al robot en el primer punto de la trayectoria. Luego, dentro de la misma función se espera la orden de iniciar la trayectoria pulsando INI desde el menú, como se verá más adelante. Presionado INI en función del valor de `cpr` se asignará a `dita1` y `dita2` el valor correspondiente para el punto intermedio al cual se debe mover el robot, una vez realizado el movimiento, `cpr` incrementa su valor. Este proceso se repetirá hasta que se alcance el valor de `cpr = 20` que indica que se llegó al punto final de la trayectoria. Luego de esto, se finaliza el repetir mientras y el programa queda listo nuevamente para seleccionar una nueva trayectoria.

```
trayectoria()
{
    //defino los valores de los angulos en PULSOS para llegar al punto deseado con un paso de 20
    // copiar y pegar desde MATLAB/OCTAVE
    finTRAY=0;
    cpr = 1; // contador de en que punto se encuentra el robot de todos los que fueron discretizados.

    // este if permite que el robot vaya desde el home al punto inicial de la recta
    if ( cpr == 1 && stop != 1 ) {

        dita1 = p1_dita1;
        dita2 = p1_dita2;
        AngGiro1();
        cpr = cpr + 1;

    } // end if

    // se establece un estado de espera en el punto inicial hasta que el usuario aprete un boton
    while ( INI != 1 && stop != 1 && cpr != 1 ) {

    } // end while de espera

    if ( INI == 1 && stop != 1 && cpr != 1 ) {

        while ( cpr <= 20 && cpr >= 2 && stop != 1 && finTRAY != 1 ) {

            switch (cpr) {

                case 2:

                    dita1 = p2_dita1;
                    dita2 = p2_dita2;
                    break;
                case 3:

                    dita1 = p3_dita1;
                    dita2 = p3_dita2;
                    break;
                case 4:

                    dita1 = p4_dita1;
                    dita2 = p4_dita2;
                    break;
            }
        }
    }
}
```

Figura 81: Función trayectoria parte inicial [Propia].

```

    case 20:
        dita1 = p20_dita1;
        dita2 = p20_dita2;
        finTRAY=1;
        break;

    } // end switch

    AngGiro1(); //invoco a la funcion angulo de giro para llevar los servos a posicion

    if (cpr < 20 && stop != 1) { // incrementa cpr hasta que llega a 20
        cpr = cpr + 1;
    } // end if

    } // end while

    INI =0; //se reinicia INI

    } // end if
} // end trayectoria

```

Figura 82: Función trayectoria parte final [Propia].

3.7.2. Función logopoli

Esta función, al igual que trayectoria, se encarga de asignar los valores que deben adoptar dita1 y dita2 para realizar el logo. El logo se encuentra compuesto por diez segmentos diferentes formados por rectas horizontales y verticales, el robot realizará cada segmento a partir de dividir al mismo en veinte puntos intermedios, como se realizó con las trayectorias. Esto permite realizar el logo sin necesidad de agregar al programa variables para asignar valores de ángulo en pulsos. Solamente se debió realizar una nueva función que, para cada segmento realice el mismo trabajo que la función trayectoria. En el primer segmento se lleva al robot al primer punto para luego mediante la variable INI se de inicio a la realización del logo.

```

logopoli()
{
    finTRAY =0;
    cpr=1;

    if (seg == 1) {
        if ( cpr == 1 && stop != 1 ) {

            dita1 = p1_dita1;
            dita2 = p1_dita2;
            AngGiro1();
            cpr = cpr + 1;

        } // end if

        // se establece un estado de espera en el punto inicial hasta que el usuario aprete un boton

        while ( INI != 1 && stop != 1 && cpr != 1 ) {

        } // end while de espera

        if ( INI == 1 && stop != 1 && cpr != 1 ) {

            while ( cpr <= 20 && cpr >= 2 && stop != 1 && finTRAY != 1 ) {

                switch (cpr) {

                    case 2:

                        dita1 = p2_dita1;
                        dita2 = p2_dita2;
                        break;

                    case 3:

                        dita1 = p3_dita1;
                        dita2 = p3_dita2;
                        break;

                    case 4:

                        dita1 = p4_dita1;
                        dita2 = p4_dita2;
                        break;

```

Figura 83: Función logopoli parte 1 [Propia].

```

    case 20:
        dital = p20_dital;
        dita2 = p20_dita2;
        finTRAY=1;
        break;

} // end switch

AngGiro1(); //invoco a la funcion angulo de giro para llevar los servos a posicion

```

Figura 84: Función logopoli parte 2 [Propia].

```

if (cpr < 20 && stop != 1) { // incrementa cpr hasta que llega a 20
    cpr = cpr + 1;
} // end if

} // end while

```

Figura 85: Función logopoli parte 3 [Propia].

```

INI =0;

} // end if
} //end if primer segmento logo

if (seg > 1 && seg <=10 ) {

if ( stop != 1 ) {

while ( cpr <= 20 && cpr >= 1 && stop != 1 && finTRAY != 1 ) {

    switch (cpr) {
        case 1:

            dital = p2_dital;
            dita2 = p2_dita2;
            break;
        case 2:

            dital = p2_dital;
            dita2 = p2_dita2;
            break;
        case 3:

            dital = p3_dital;
            dita2 = p3_dita2;
            break;
        case 4:

            dital = p4_dital;
            dita2 = p4_dita2;
            break;

```

Figura 86: Función logopoli parte 4 [Propia].

```

    case 20:
        dital = p20_dital;
        dita2 = p20_dita2;
        finTRAY=1;
        break;

    } // end switch

    AngGiro1(); //invoco a la funcion angulo de giro para llevar los servos a posicion

    if (cpr < 20 && stop != 1) { // incrementa cpr hasta que llega a 20
        cpr = cpr + 1;
    } // end if

    } // end while

    } // end if
} //end if

```

Figura 87: Función logopoli parte 5 [Propia].

3.7.3. Función homming

Esta función se mantiene igual que la programación base, pero se realizó una leve modificación en el programa principal para poder hacer home cuando se desee sin la necesidad de volver a establecer el cero máquina mediante los límites de carrera. En síntesis, cuando se inician los servomotores solo bastará con encontrar el cero máquina una única vez, luego el programa lleva el robot a homming estableciendo el valor 0 a dital y dita2.

```

if (st1==0 && (home==1)) home=0; // este if no deja que se haga homming si los servos se mueven

if ((st1!=0)&& (st2!=0) &&(home==1) && band_home == 0 ) {

    st1=3; //estado inicial perdido
    st2=3; //estado inicial perdido
    Homming ();
    cpr=0;
    seg=0;
    TRAY=0;
    dital=0;
    dita2=0;
    H142=0;
    band_home=1;

} //end if homming

```

Figura 88: Función homming parte 1 [Propia].

Una vez establecido el home la variable **band home** adopta el valor 1 para indicar que ya se realizó la búsqueda del 0 máquina y a partir de ahí utilizar el home que se ve en la figura (89)

```

if ((st1!=0)&& (st2!=0) &&(home==1) && band_home == 1 ) {
    H142=124;
    dital=0;
    dita2=0;
    H150=0;
    H151=0;
    H152=0;
    H153=0;
    H154=0;
    H155=0;
    H156=0;
    H157=0;
    H158=0;
    H159=0;
    AngGirol();
    home=0;
} //end homming 2

```

Figura 89: Función homming parte 2 [Propia].

3.7.4. Programa principal

El programa principal se encuentra compuesto por la parte de home indicada en el apartado anterior, un estado de espera en donde el programa queda a la espera de indicaciones, la parte de stop y la asignación de los valores de las variables correspondientes a los ángulos según las trayectorias a realizar.

Estado de espera:

```

while(TRAY == 0 && home!=1 && stop!=1 && START == 1) {
    H142=700; | // variable para determinar si el programa avanza correctamente
}; //espera
H142=222; // variable para determinar si el programa avanza correctamente

```

Figura 90: Estado de espera en programa principal [Propia].

Una vez apretado start, el programa espera que se seleccione una determinada trayectoria.

STOP:

Al apretar STOP desde el menú, el programa se detiene. Entonces, robot quedará detenido en el momento que se apriete dicho botón, por una cuestión de seguridad se estableció que se deba volver al home de forma obligatoria luego de apretar STOP. Además, todas las variables se reinician.

```
// -----STOP-----
if(stop==1){
    START=0;
    TRAY=0;
    cpr=0;
    seg=0;
    INI=0;
    finTRAY=0;
    _Wait (500);
    stop=0;
    b_stop=1;
    H42=22;
} //end if stop */
H142=8888;
```

Figura 91: Stop [Propia].

Luego de presionar stop, el programa ingresa en un while que hasta no apretar home no cambia el valor de la variable **b stop** evitando que el programa avance.

```
while (b_stop == 1) { // estado de espera previo a hacer home luego de apretar STOP
    while (home = 0) {
    }

    b_stop=0;
} //end while
```

Figura 92: Home post stop [Propia].

Selección de trayectoria:

Cuando se presiona START y se ingresa un número para elegir una trayectoria, mediante una sentencia case desde el programa principal, se asignan los valores de pulsos correspondientes para las variables de dita1 y dita2 en cada punto. Estos valores fueron cargados según lo calculado en matlab. Como se mencionó previamente, una vez asignados los valores, se invoca la función trayectoria(), finalizada la trayectoria se reinicia el valor de TRAY. Las figuras (93) y (94) ilustran lo mencionado para el caso donde TRAY es igual a 1, pero lo mismo se repite para las demás trayectorias, se podrá visualizar en el anexo correspondiente.

```
// ===== TRAYECTORIA =====
if (START == 1) { // boton START da la orden para iniciar la trayectoria.
    _Wait(500); // espera 0,5seg para evitar cualquier problema al poner START en 0.
    START=0;

    switch(TRAY) {

        case 1:

            // recta horizontal l = 20mm

            p1_dita1 = -21078;
            p1_dita2 = 19903;

            p2_dita1 = -21012;
            p2_dita2 = 19872;

            p3_dita1 = -20946;
            p3_dita2 = 19940;

            p4_dita1 = -20880;
            p4_dita2 = 20008;

            p5_dita1 = -20814;
            p5_dita2 = 20076;

            p6_dita1 = -20747;
            p6_dita2 = 20144;

            p7_dita1 = -20681;
            p7_dita2 = 20212;

            p8_dita1 = -20614;
            p8_dita2 = 20279;

            p9_dita1 = -20548;
            p9_dita2 = 20346;

            p10_dita1 = -20481;
            p10_dita2 = 20414;

            p11_dita1 = -20414;
            p11_dita2 = 20481;

            p12_dita1 = -20346;
            p12_dita2 = 20548;
```

Figura 93: Selección de trayectoria parte 1 [Propia].

```
p13_dita1 = -20279;
p13_dita2 = 20614;

p14_dita1 = -20212;
p14_dita2 = 20681;

p15_dita1 = -20144;
p15_dita2 = 20747;

p16_dita1 = -20076;
p16_dita2 = 20814;

p17_dita1 = -20008;
p17_dita2 = 20880;

p18_dita1 = -19940;
p18_dita2 = 20946;

p19_dita1 = -19872;
p19_dita2 = 21012;

p20_dita1 = -19803;
p20_dita2 = 21078;

trayectoria();
TRAY=0; //se reinicia TRAY para seleccionar nuevamente una trayectoria
break;
```

Figura 94: Selección de trayectoria parte 2 [Propia].

Al definir "TRAY = 26" se realizará el logo, por lo cual lo mencionando anteriormente para las trayectorias se repetirá diez veces, una vez para cada segmento.

```
case 26 :  
  
    seg = 1;  
    while (seg >= 1 && seg <=10 && stop != 1) {  
        if (seg == 1) {  
            p1_dita1 = -20302;  
            p1_dita2 = 21564;  
  
            p2_dita1 = -20437;  
            p2_dita2 = 21433;  
  
            p3_dita1 = -20571;  
            p3_dita2 = 21301;  
  
            p4_dita1 = -20704;  
            p4_dita2 = 21169;  
  
            p5_dita1 = -20838;  
            p5_dita2 = 21037;  
  
            p6_dita1 = -20971;  
            p6_dita2 = 20904;  
  
            p7_dita1 = -21103;  
            p7_dita2 = 20771;  
  
            p8_dita1 = -21235;  
            p8_dita2 = 20638;  
  
            p9_dita1 = -21367;  
            p9_dita2 = 20504;  
  
            p10_dita1 = -21499;  
            p10_dita2 = 20369;  
  
            p11_dita1 = -21630;  
            p11_dita2 = 20235;  
  
            p12_dita1 = -21761;  
            p12_dita2 = 20099;  
  
            p13_dita1 = -21891;  
            p13_dita2 = 19963;  
  
            p14_dita1 = -22022;  
            p14_dita2 = 19827;  
  
            p15_dita1 = -22152;  
            p15_dita2 = 19690;
```

Figura 95: Selección logo en programa principal parte 1 [Propia].

```
p15_dital = -22152;  
p15_dita2 = 19690;  
  
p16_dital = -22281;  
p16_dita2 = 19553;  
  
p17_dital = -22411;  
p17_dita2 = 19415;  
  
p18_dital = -22540;  
p18_dita2 = 19276;  
  
p19_dital = -22669;  
p19_dita2 = 19137;  
  
p20_dital = -22798;  
p20_dita2 = 18997;  
  
logopoli();  
H150 = 1;  
seg = seg +1;  
} // end if
```

Figura 96: Selección logo en programa principal parte 2 [Propia].

```
logopoli();  
H159 = 1;  
seg = seg +1;  
} // end if  
  
} // end while seg  
finTRAY=1;  
TRAY=0;  
  
break;  
  
} // end switch  
} // end IF  
  
} // end while  
} // end main
```

Figura 97: Selección logo en programa principal parte 3 [Propia].

3.7.5. Menú

Como se mencionó anteriormente, mediante AppBuilder se elaboró un menú el cual permite al usuario:

- Realizar la búsqueda del 0 máquina y volver a home el robot.
- Seleccionar la trayectoria a realizar mediante el recuadro de texto y luego presionando OK.
- Posicionar el robot en el primer punto de la trayectoria apretando el boton START.
- Dar inicio al movimiento presionando INI.
- Detener el robot en cualquier momento presionando STOP.

<u>Realizar linea horizontal</u>	<u>Realizar linea vertical</u>	<u>Realizar linea diagonal</u>	<u>Movimiento en zigzag</u>
TRAY = 1 -> recta largo = 20mm	TRAY = 5 -> recta largo = 20mm	TRAY = 9	TRAY = 25
TRAY = 2 -> recta largo = 40mm	TRAY = 6 -> recta largo = 40mm	TRAY = 10	<u>Logo</u>
TRAY = 3 -> recta largo = 60mm	TRAY = 7 -> recta largo = 60mm	TRAY = 11	TRAY = 26
TRAY = 4 -> recta largo = 80mm	TRAY = 8 -> recta largo = 80mm	TRAY = 12	
<u>Realizar cuadrado</u>	<u>Realizar triangulo</u>	<u>Realizar Circunferencia</u>	
TRAY = 13 -> lado = 20mm	TRAY = 17 -> base = 20mm	TRAY = 21 -> radio = 10mm	
TRAY = 14 -> lado = 40mm	TRAY = 18 -> base = 40mm	TRAY = 22 -> radio = 20mm	
TRAY = 15 -> lado = 80mm	TRAY = 19 -> base = 60mm	TRAY = 23 -> radio = 30mm	
TRAY = 16 -> lado = 100mm	TRAY = 20 -> base = 100mm	TRAY = 24 -> radio = 50mm	

Figura 98: Menú humano-maquina [Propia].

Como se observa en la figura (98) en el mismo figuran los de TRAY a ingresar para realizar cada trayectoria y un pequeño instructivo de uso.

3.8. Análisis velocidades angulares.

A fines de comprender que cómo se comportan las velocidades angulares de los brazos distales cuando se fuerza al punto efector a realizar una trayectoria determinada, se realizaron simulaciones de movimiento en SolidWorks.

Las trayectorias definidas son:

- Recta diagonal.
- Recta horizontal.
- Recta vertical.

Para los tres casos, se definió una recta con una longitud de 70,71 mm y el punto efector tendrá el siguiente perfil de velocidad:

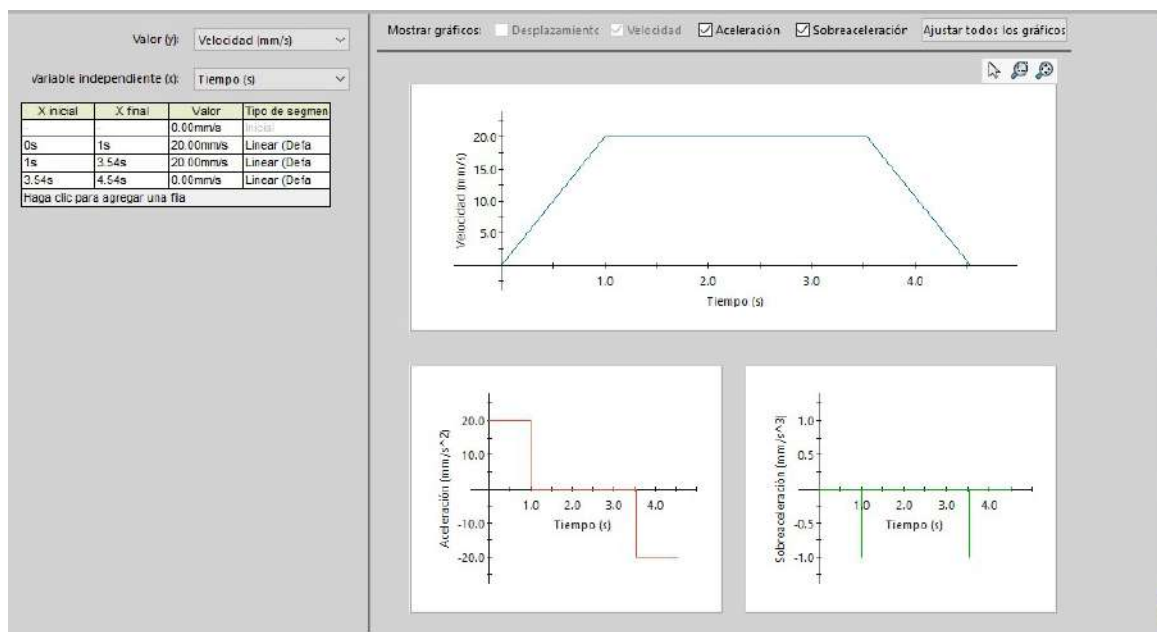


Figura 99: Perfil de velocidad punto efector en estudio de movimiento [Propia].

Los valores definidos para el perfil de velocidad se adoptaron al azar, ya que sólo interesa comprender cómo se comportan las velocidades y que tipo de función se visualiza.

Luego, se realizó un croquis con la trayectoria a seguir por el punto efector y se aplicó la relación de posición "función de trayectoria". En el estudio de movimiento se aplicó un motor lineal que cumpla con la ley de velocidad que se observa en la figura (99).

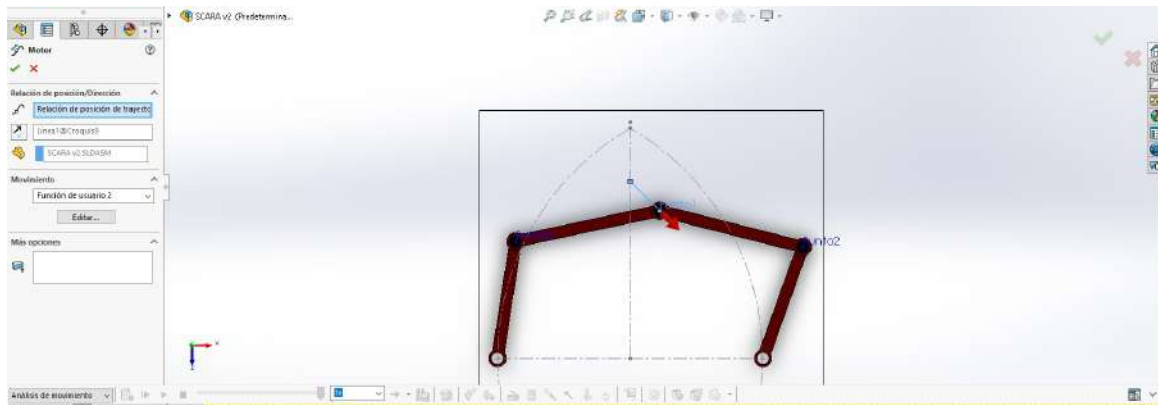


Figura 100: Estudio de movimiento [Propia].

Una vez realizado el estudio de movimiento para las tres trayectorias, se obtuvieron los siguientes resultados. En cada gráfica se comparan los valores de velocidad angular para ambos brazos.

Recta diagonal:

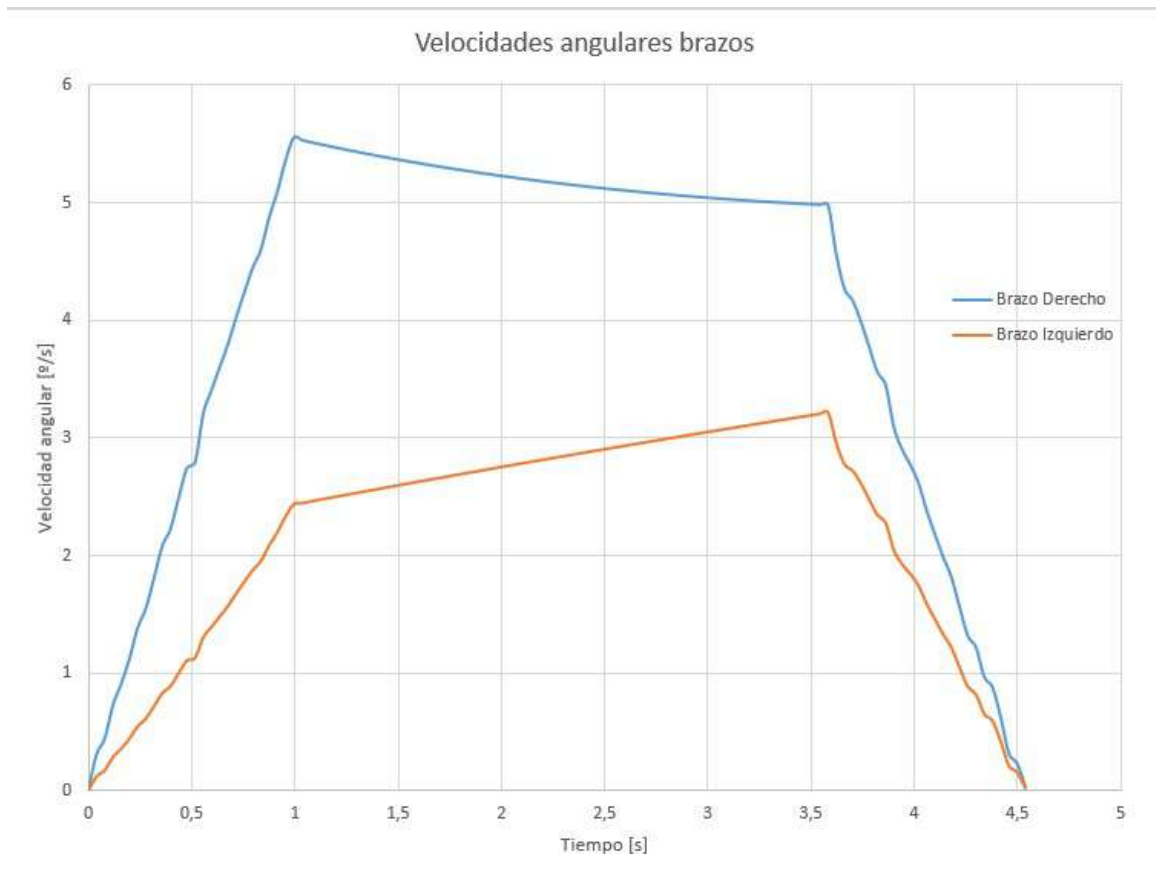


Figura 101: Resultado velocidades angulares recta diagonal [Propia].

Recta horizontal:

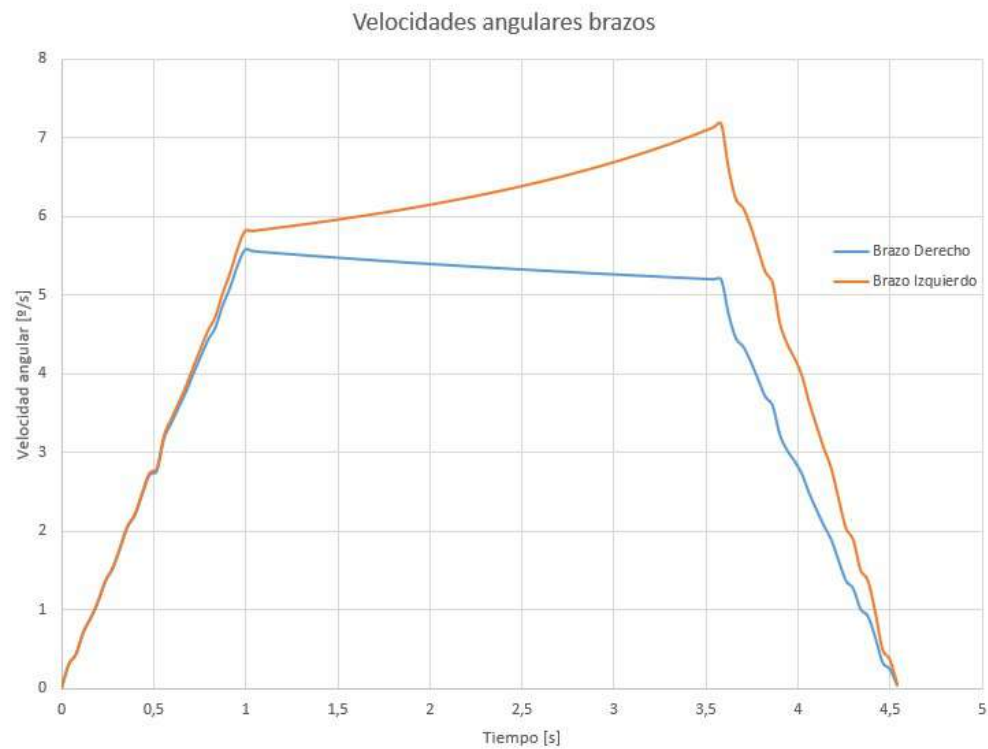


Figura 102: Resultado velocidades angulares recta horizontal [Propia].

Recta vertical:

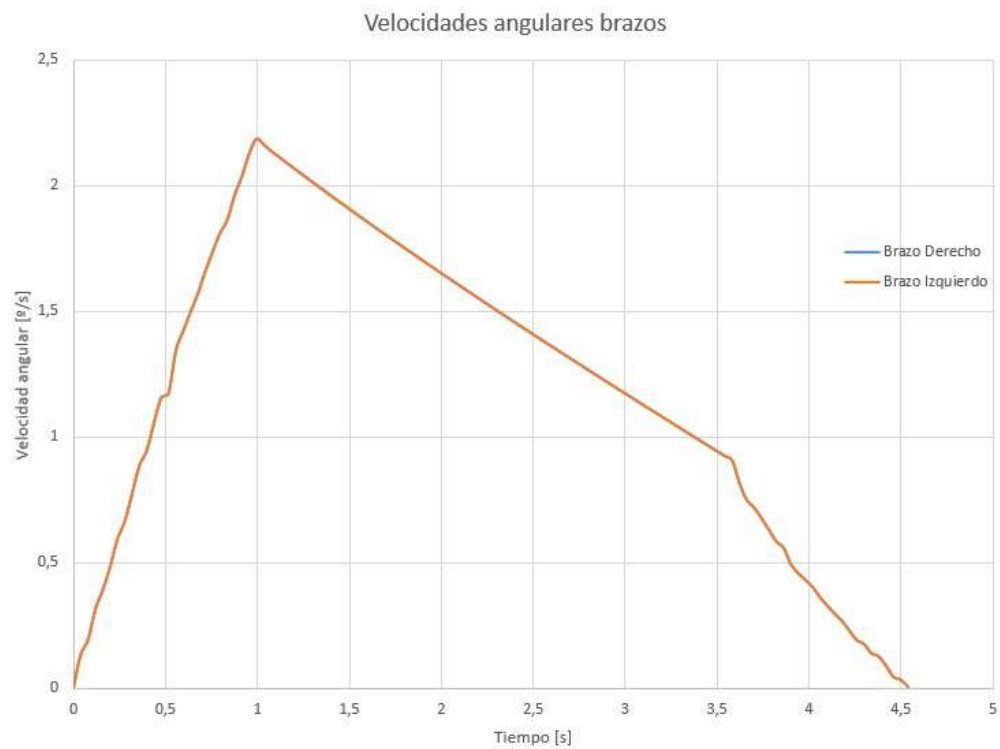


Figura 103: Resultado velocidades angulares recta vertical [Propia].

Analizando los resultados obtenidos, no se observa con claridad que tipo de curva describen las velocidades angulares de los brazos, parecieran ser del tipo cuadráticas sobre todo en los segmentos en donde la velocidad lineal del punto efector es constante. Si podemos observar que para lograr una trayectoria recta entre dos puntos, las velocidades angulares de los brazos deben ser diferentes, como podemos observar para los casos de las figuras (101) y (102). Las curvas poseen diferentes pendientes, por lo cual las aceleraciones angulares de cada brazo en particular son distintas. Para el caso de la línea vertical (figura (103)), se observa que los brazos poseen el mismo valor de velocidad angular, teniendo en cuenta que para realizar dicha recta ambos se deben mover en sentidos opuestos.

3.9. Construcción y montaje.

Definidos los espesores de chapa, el material y el diseño del soporte en la correspondiente sección, se realizaron los trabajos de corte y plegado de las diferentes partes del soporte. A continuación podemos visualizar dicho trabajo en la figura (104).



Figura 104: Chapas cortadas y plegadas[Propia].

Se realizaron las soldaduras correspondientes, los trabajos de pintura en color gris texturado y se procedió a montar el soporte, obteniendo el siguiente resultado:

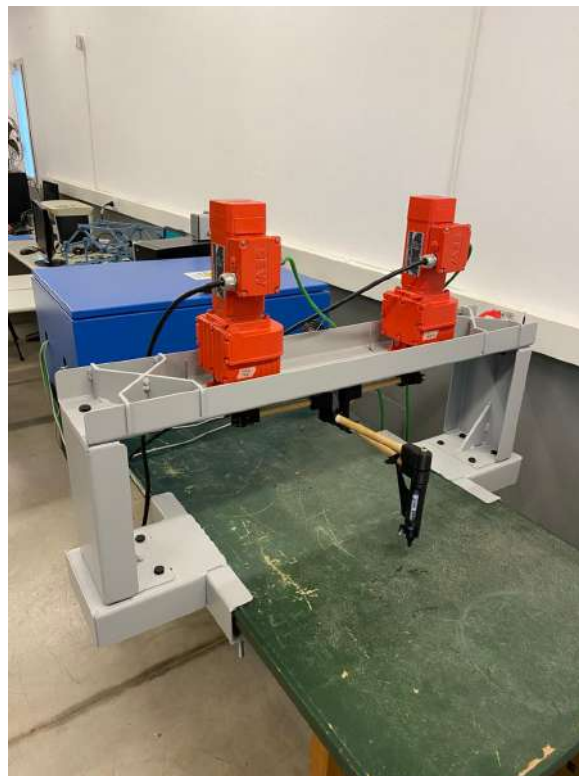


Figura 105: Montaje final [Propia].

4. Pruebas experimentales

Lograda la automatización de los movimientos y verificado su funcionamiento, se realizaron pruebas experimentales para visualizar las trayectorias definidas. Primero, se llevaron a cabo pruebas con el robot en posición vertical (ver figura (106a)) como se encontraba en un principio. Segundo, se llevaron a cabo las mismas pruebas, pero con el robot ya montado en la estructura diseñada (ver figura (106a)). En ambos casos, se utilizó el mismo soporte para el fibrón a fin de obtener igualdad de condiciones.

Se debe tener en cuenta que lo resuelto en el presente proyecto para la cadena cinemática es la posición, es decir, podemos conocer dónde se posicionará el robot pero no de qué manera realiza el movimiento para llegar a determinada posición. Esto genera que, al indicarle al robot que se desplace de un punto a otro, el punto efector posea un movimiento aleatorio. Por ello, para que el punto efector copie las trayectorias, se discretizaron las mismas en 20 puntos, buscando disminuir el movimiento aleatorio. Este efecto se notará en las siguientes pruebas. Para cada trayectoria, se midieron las dimensiones correspondientes a fin de evaluar con qué precisión el robot dibuja las mismas.



(a) Robot en vertical.



(b) Robot en horizontal.

Figura 106: Disposiciones del robot para las pruebas [Propia].

4.1. Pruebas en vertical

Lineas horizontales:

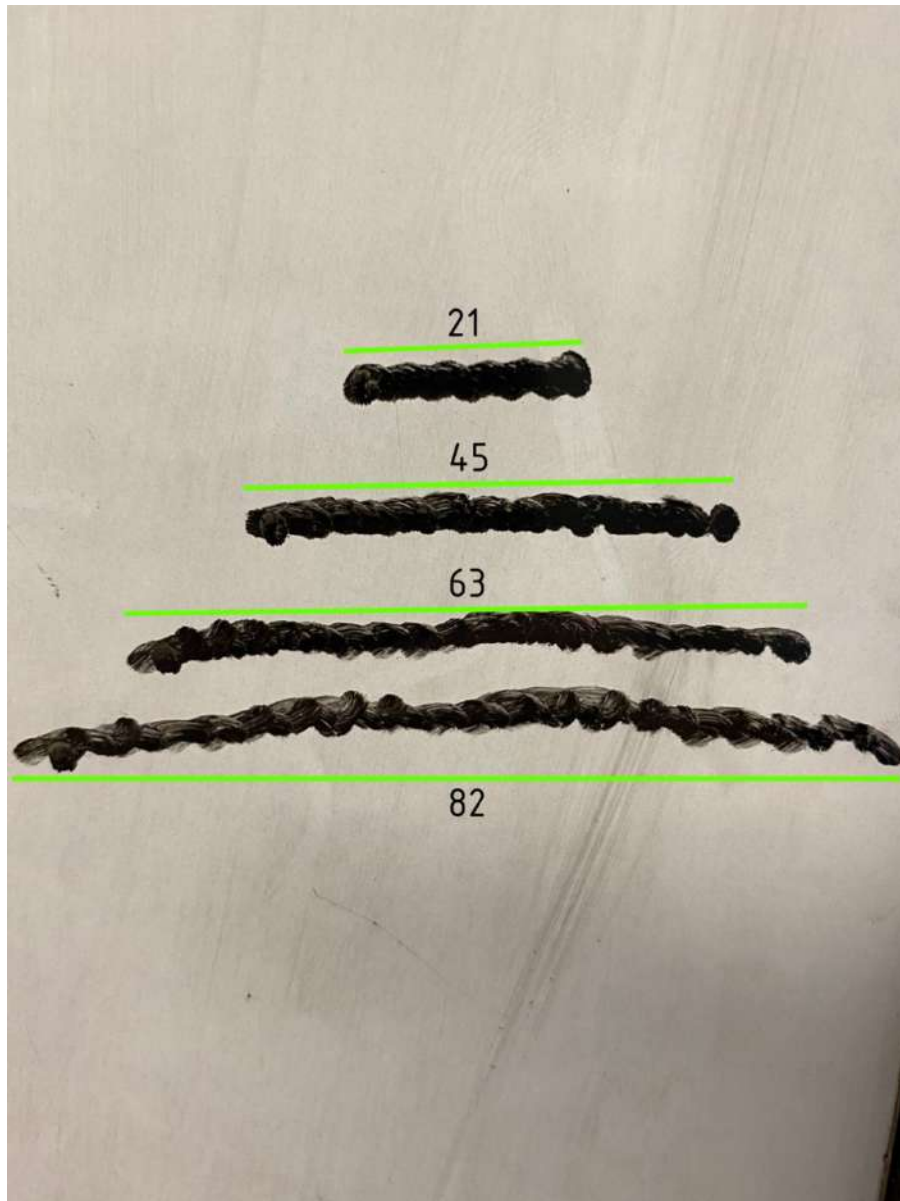


Figura 107: Comparación líneas horizontales - prueba vertical [Propia].

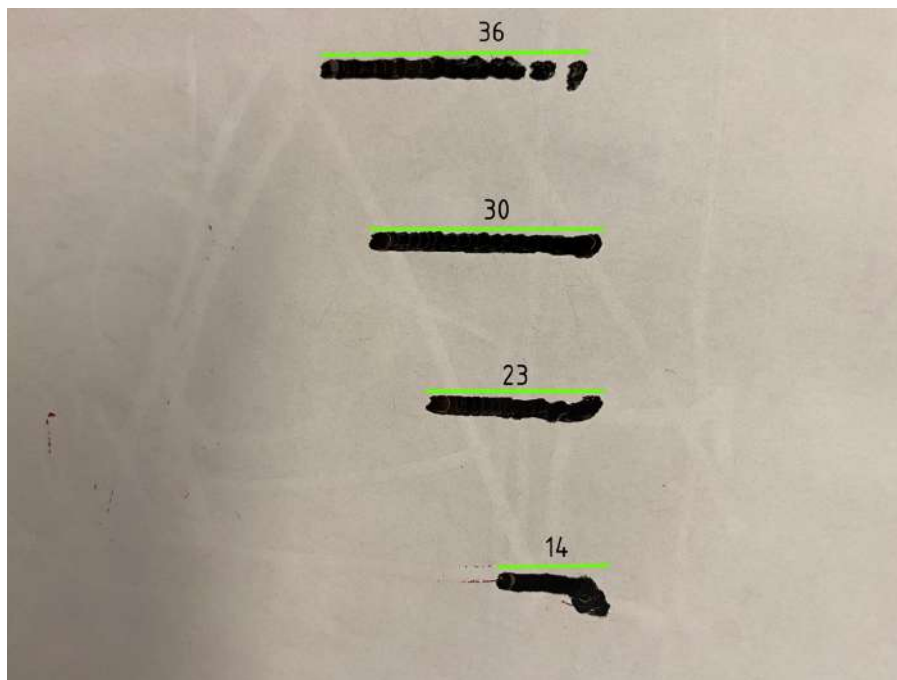
Lineas verticales:

Figura 108: Comparación líneas verticales - prueba vertical [Propia].

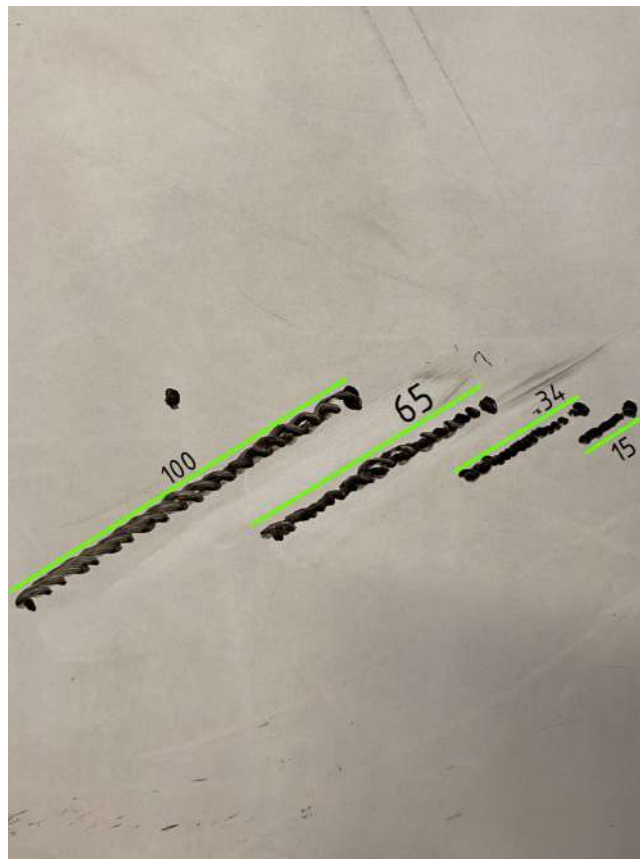
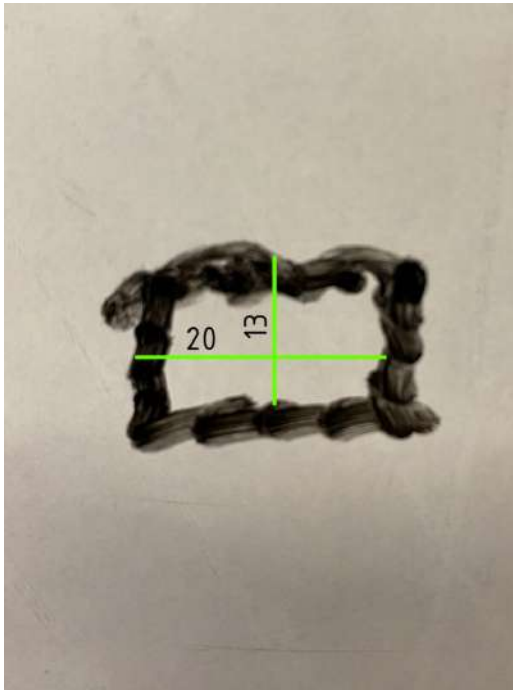
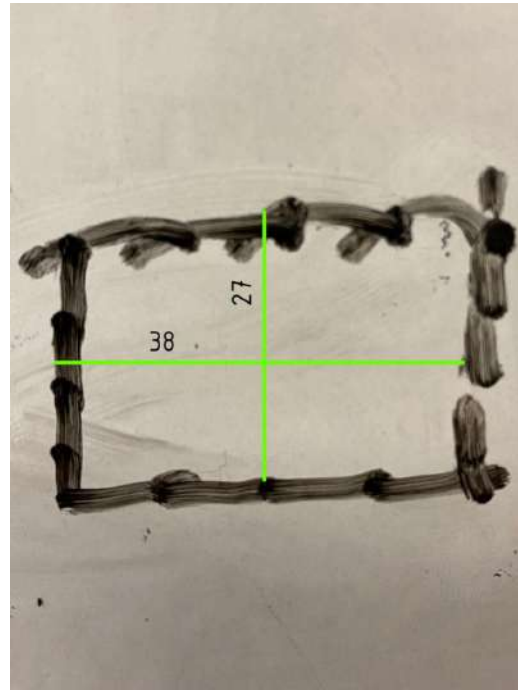
Lineas diagonales:

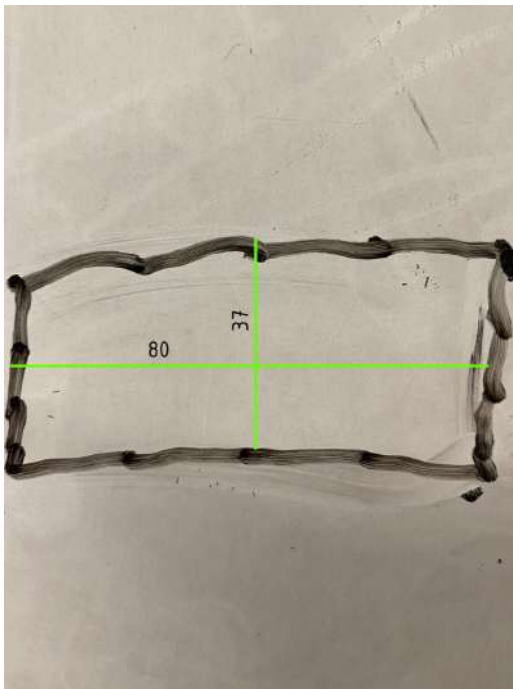
Figura 109: Comparación líneas diagonales - prueba vertical [Propia].

Cuadrados:

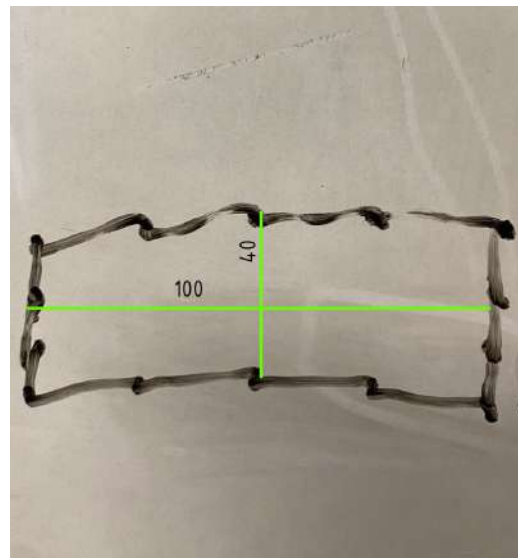
(a) Lados 20mm.



(b) Lados 40mm.

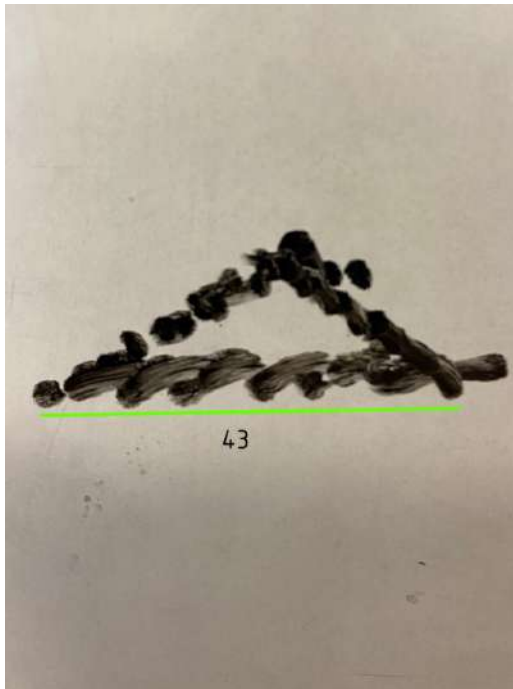
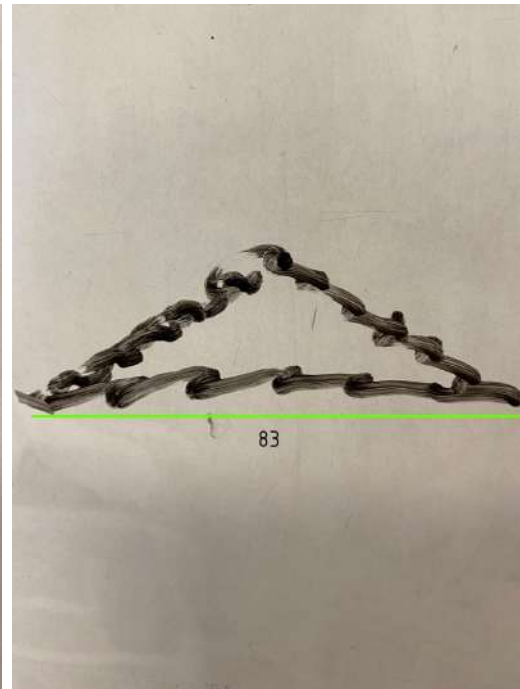
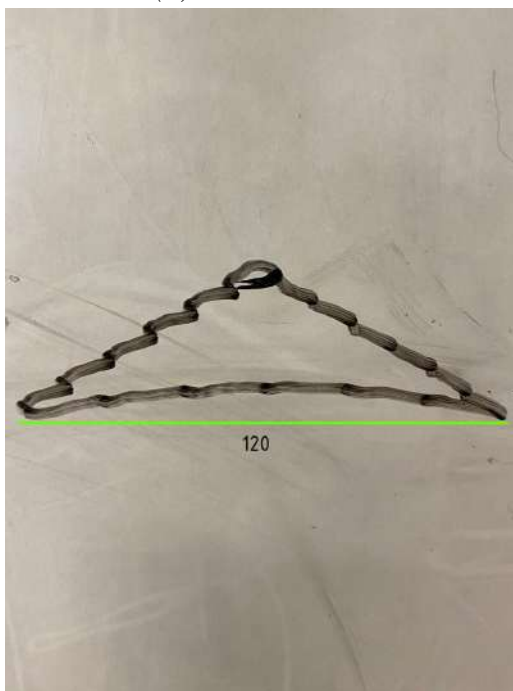
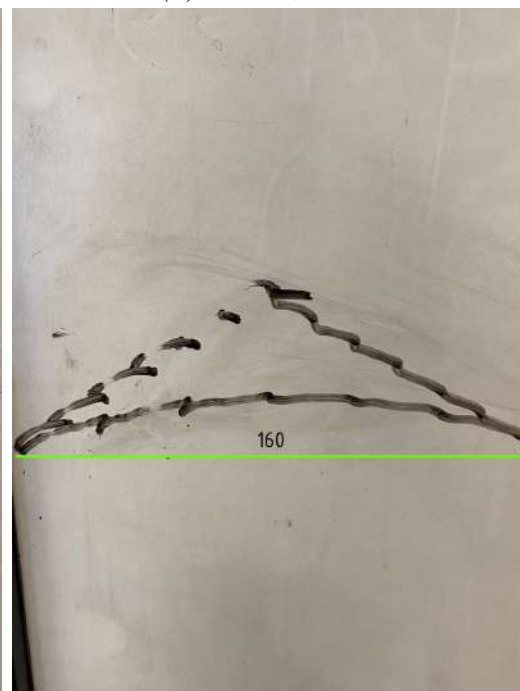


(c) Lados 80mm.

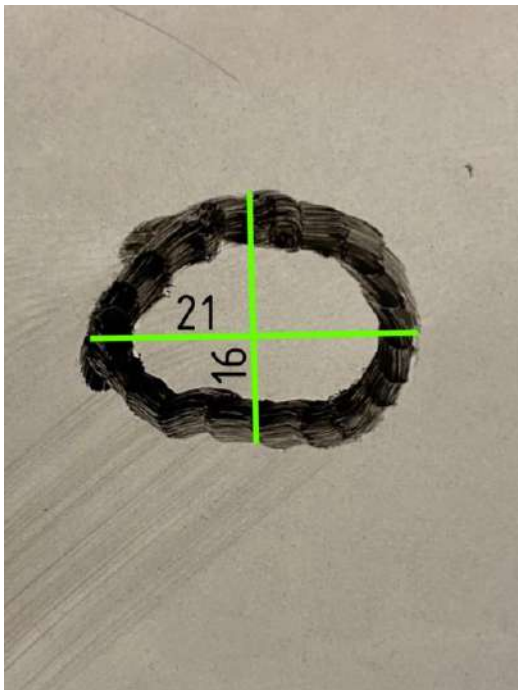


(d) Lados 100mm.

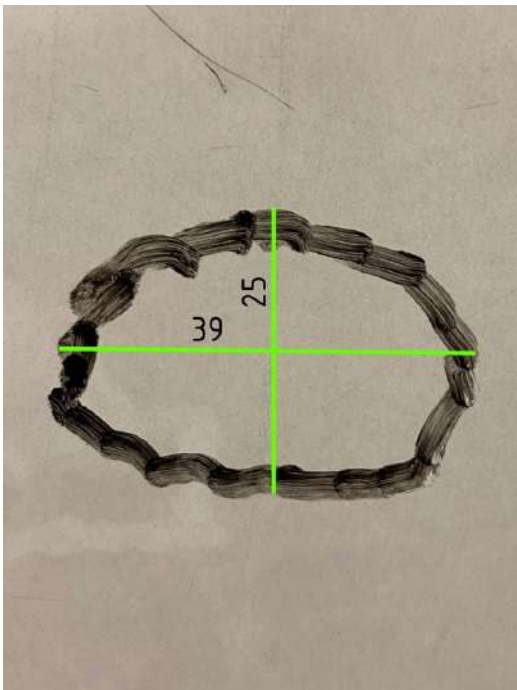
Figura 110: Cuadrados - prueba vertical [Propia].

Triángulos:*(a) Base 20mm.**(b) Base 40mm.**(c) Base 60mm.**(d) Base 100mm.**Figura 111: Triángulos - prueba vertical [Propia].*

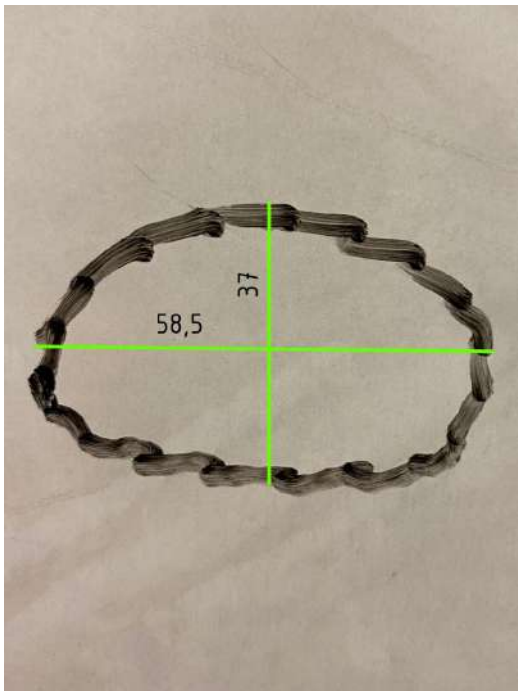
Circunferencias:



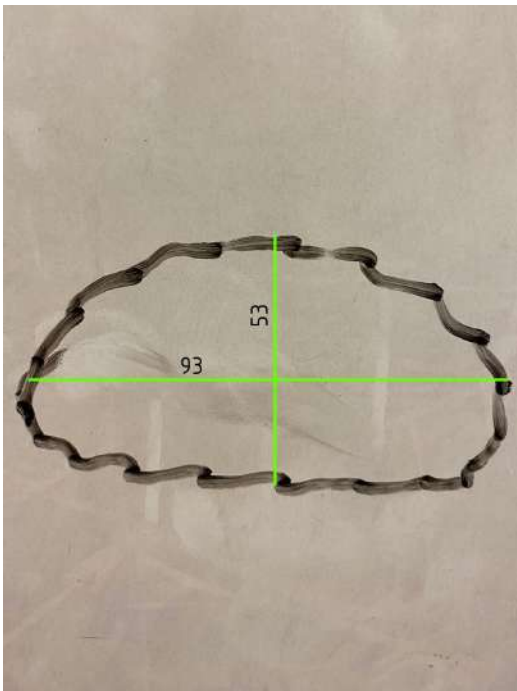
(a) Radio 10mm.



(b) Radio 20mm.



(c) Radio 30mm.



(d) Radio 50mm.

Figura 112: Circunferencias - prueba vertical [Propia].

Logo:

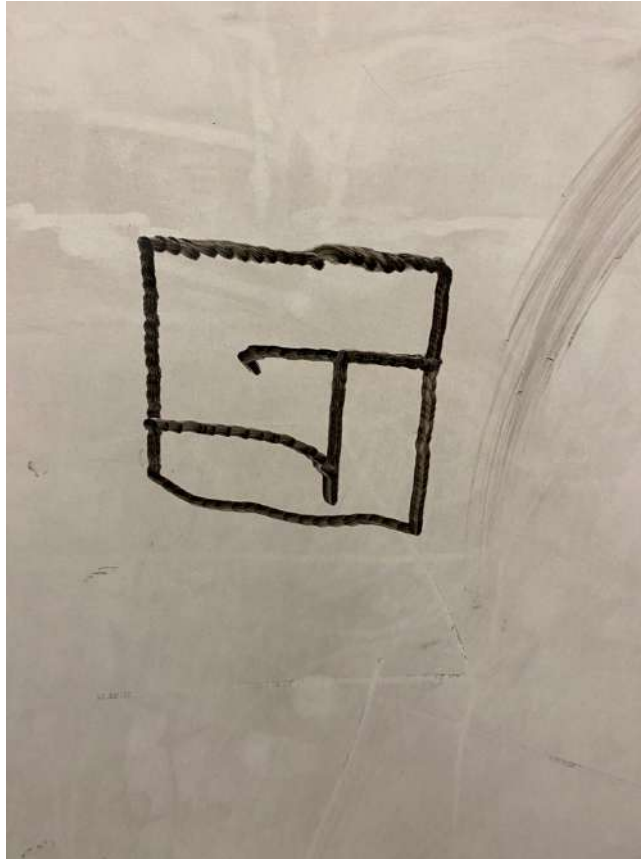


Figura 113: Logo - prueba vertical [Propia].

4.2. Pruebas en horizontal

Lineas horizontales:

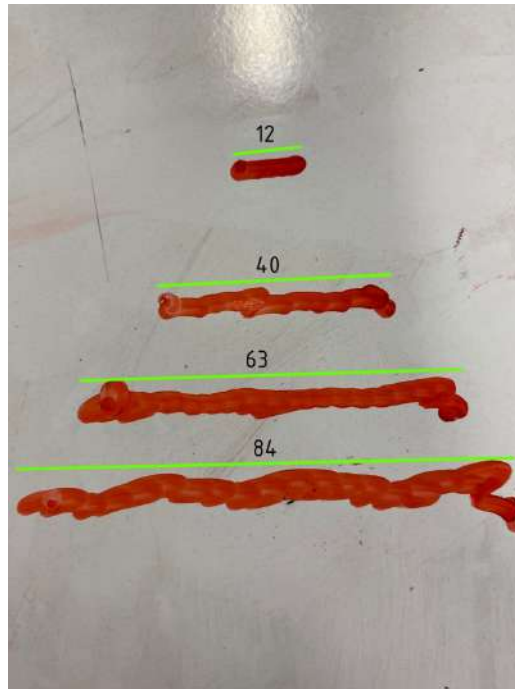


Figura 114: Comparación líneas horizontales - prueba horizontal [Propia].

Lineas verticales:

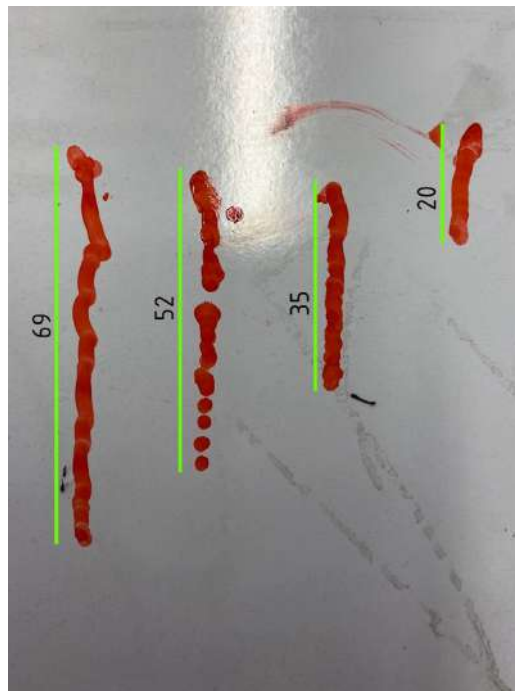


Figura 115: Comparación líneas verticales - prueba horizontal [Propia].

Lineas diagonales:

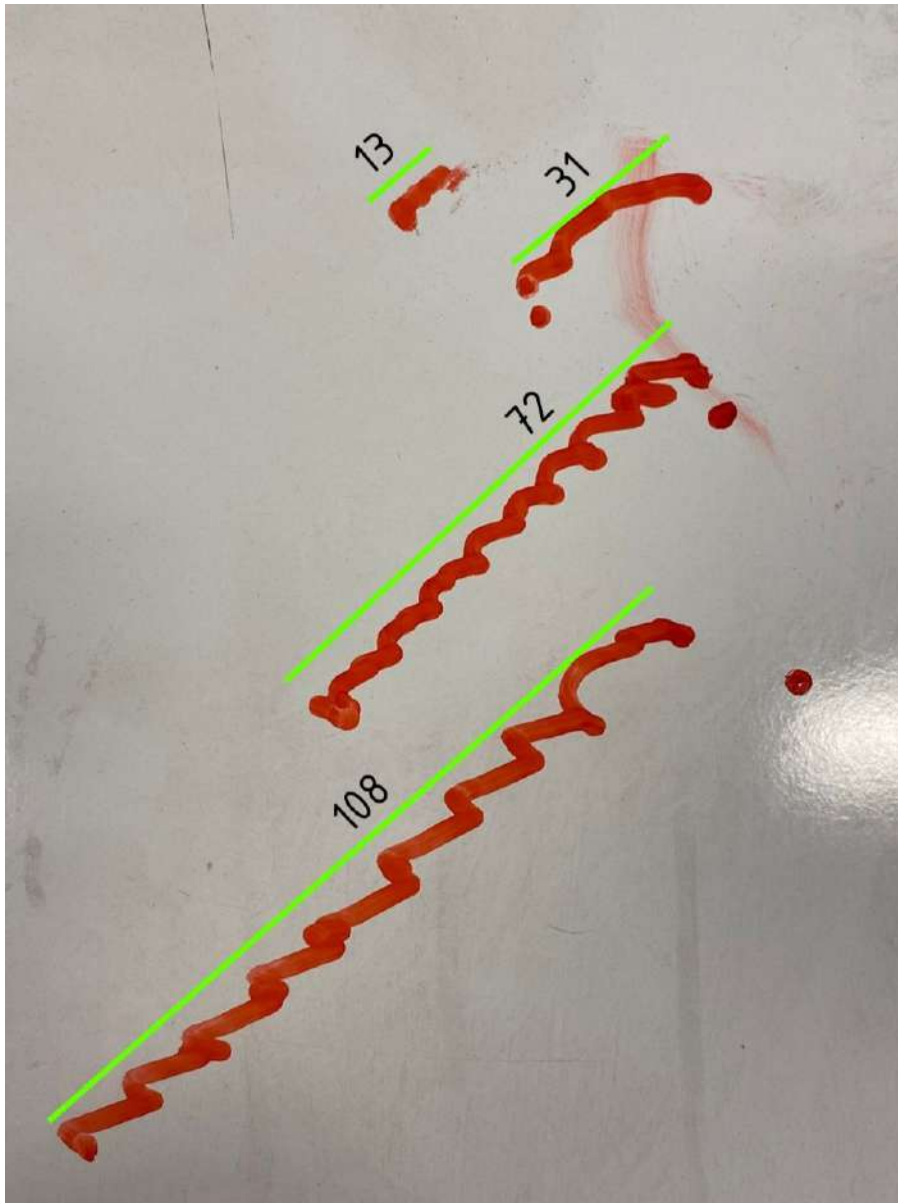
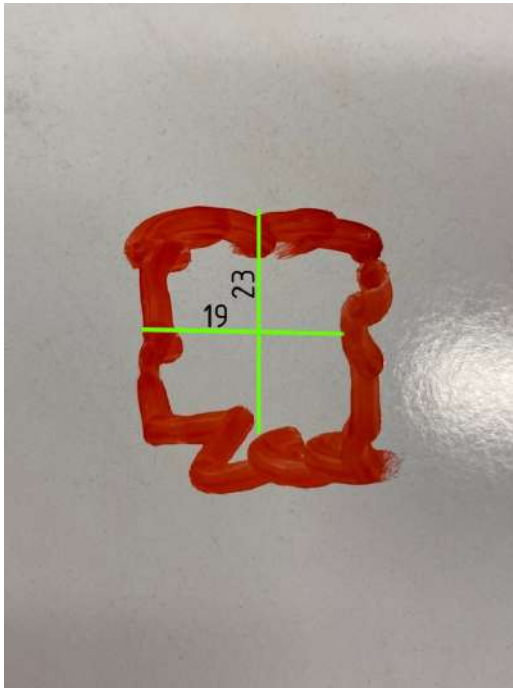
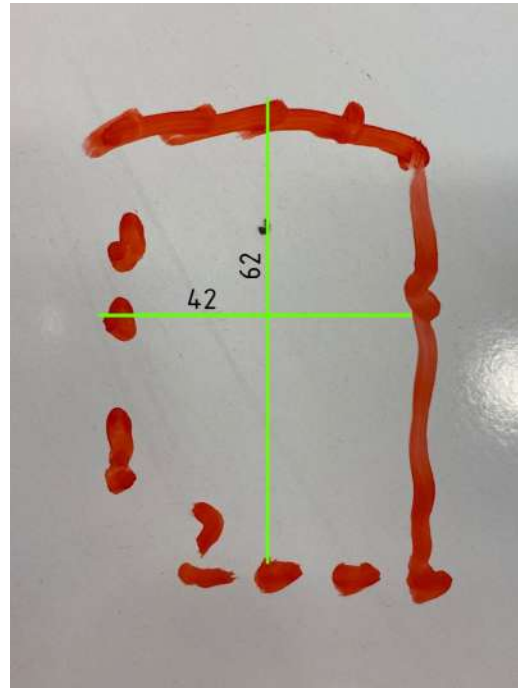


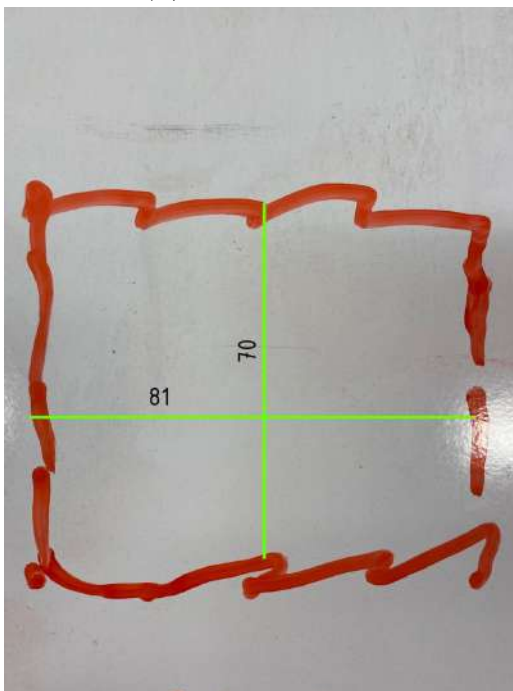
Figura 116: Comparación líneas diagonales - prueba horizontal [Propia].

Cuadrados:

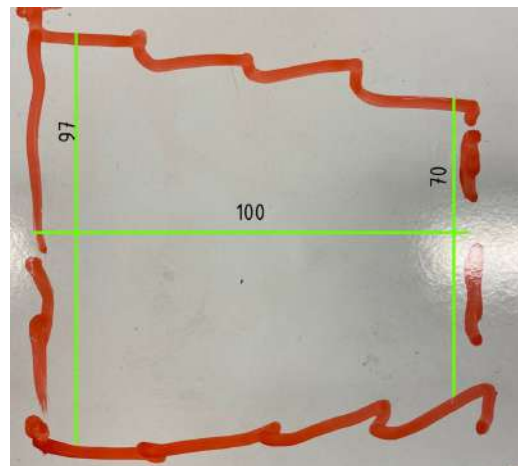
(a) Lados 20mm.



(b) Lados 40mm.



(c) Lados 80mm.



(d) Lados 100mm.

Figura 117: Cuadrados - prueba horizontal [Propia].

Triángulos:

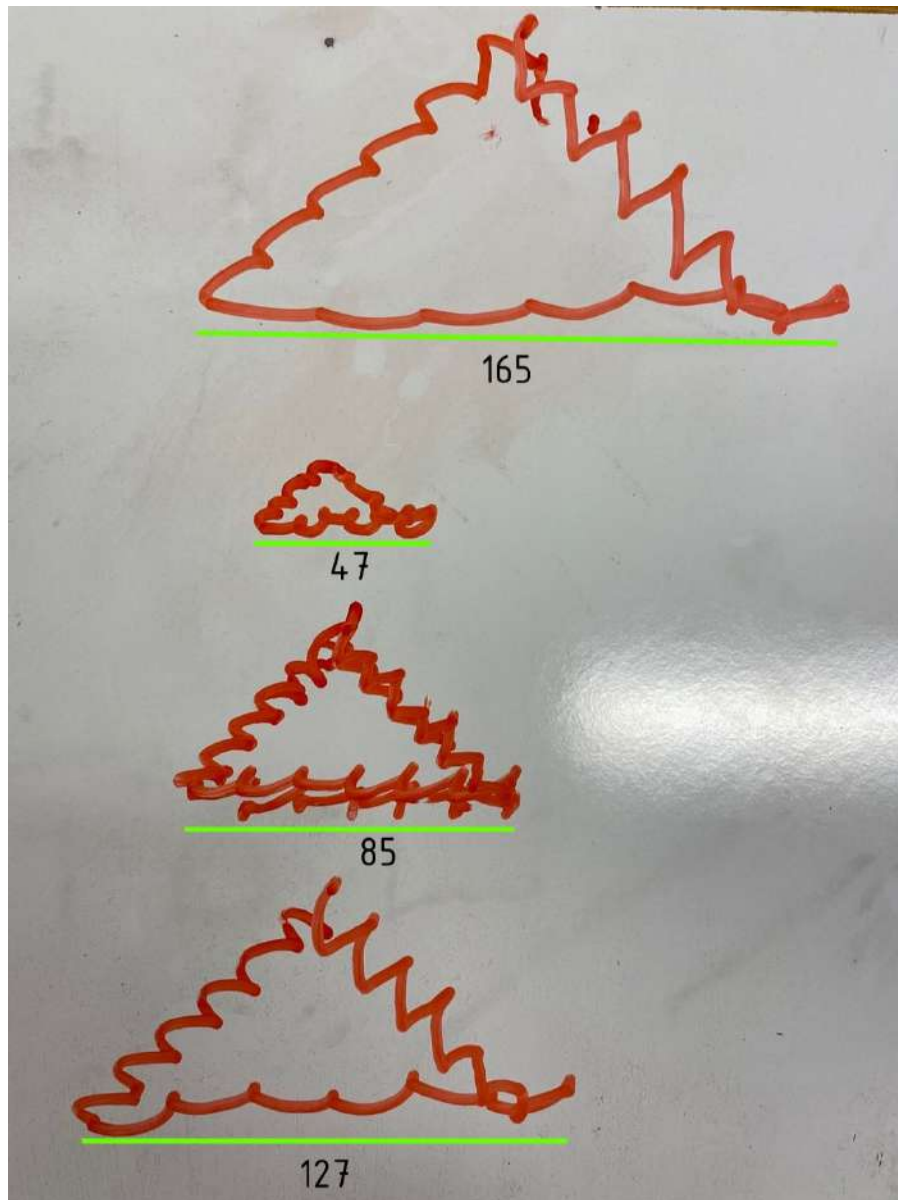
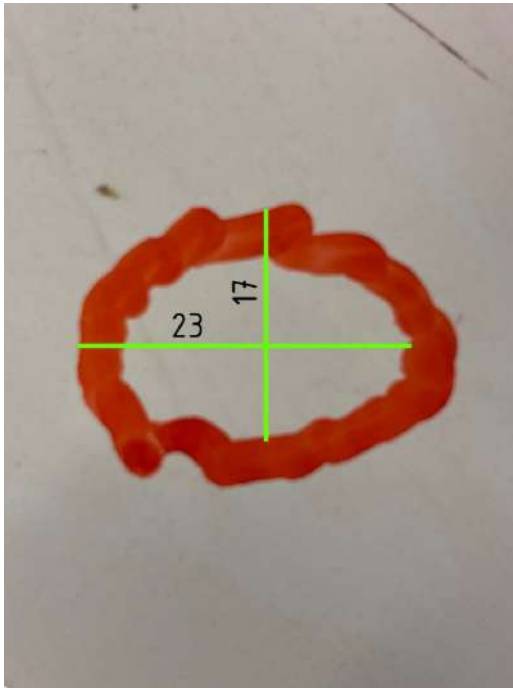
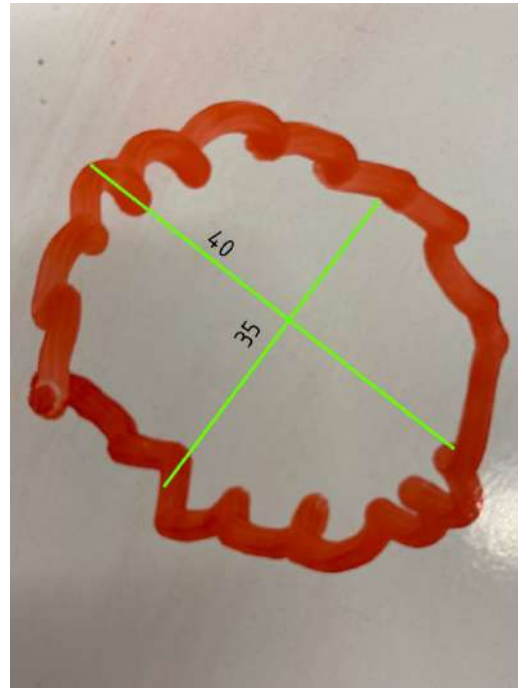


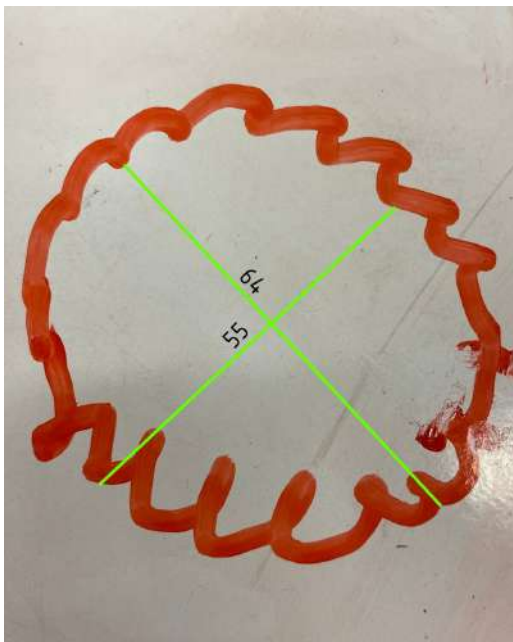
Figura 118: Triángulos - prueba horizontal [Propia].

Circunferencias:

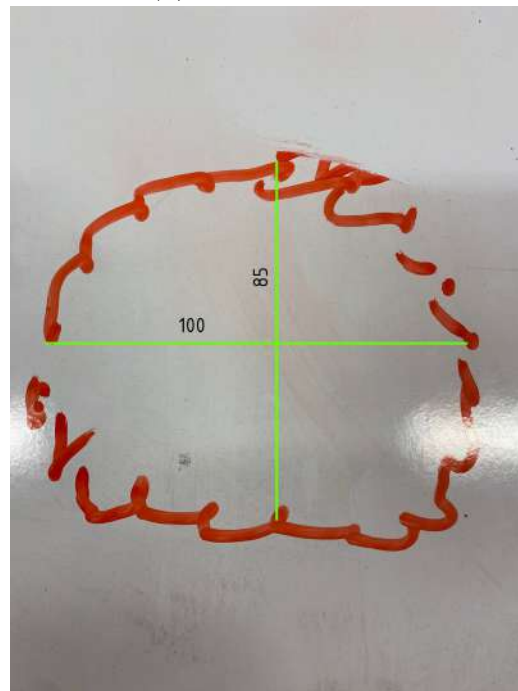
(a) Radio 10mm.



(b) Radio 20mm.



(c) Radio 30mm.



(d) Radio 50mm.

Figura 119: Circunferencias - prueba horizontal [Propia].

Logo:

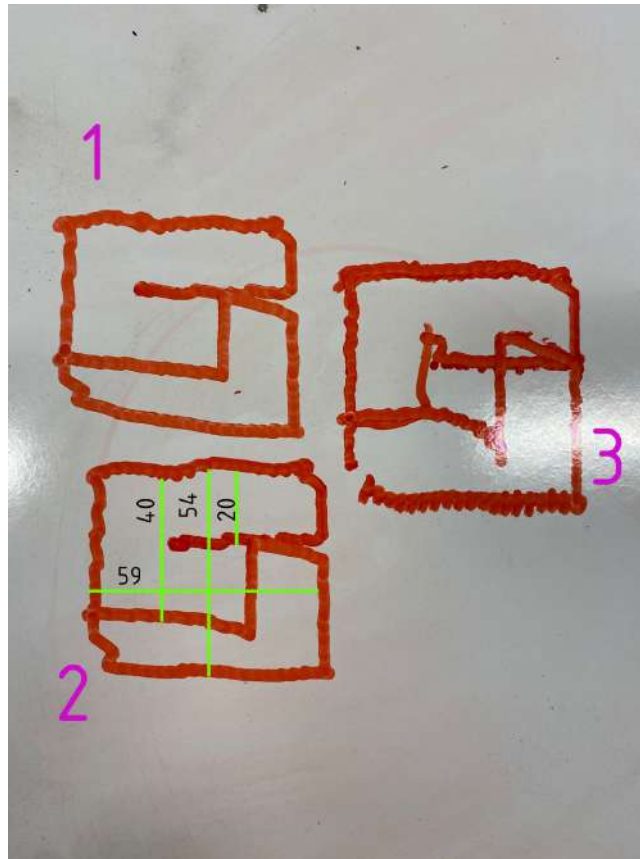


Figura 120: Logo - prueba horizontal [Propia].

zig-zag:



Figura 121: Movimiento zigzag - prueba horizontal [Propia].

4.3. Análisis resultados experimentales

Habiendo realizado las pruebas, podemos notar los siguientes patrones: a medida que aumentamos la dimensión de las trayectorias, se percibe un aumento de la visibilidad de la trayectoria aleatoria que realiza el punto efector entre punto y punto. Esto se ve mucho más marcado en los cuadrados, triángulos y circunferencias. Las medidas obtenidas son razonables, teniendo en cuenta las características constructivas del robot, las cuales afectan su precisión. Las líneas horizontales y verticales demuestran resultados prolijos, en las líneas verticales no se visualizan con claridad las trayectorias aleatorias entre los puntos. Se puede observar en las figuras (110) y (117) que los segmentos realizados en vertical siempre resultan más cortos que la dimensión establecida, lo mismo para el caso de las líneas verticales, si comparamos las figuras (107) y (108), (114) y (115) respectivamente. Las rectas diagonales poseen una mayor notoriedad de las trayectorias aleatorias, si las comparamos con las verticales u horizontales. Los triángulos suelen tener una especie de "panza" en su base en el caso de la figura (111), la cual no se nota con claridad en la figura (118). Con respecto a las circunferencias, se notan al menos dos radios diferentes en todas las trayectorias realizadas, uno mayor y uno menor. Por último, en los logos no se percibe la trayectoria aleatoria, ya que está compuesto por diferentes segmentos discretizados en 20 a comparación de las otras trayectorias, cuya totalidad de las mismas se encuentra discretizada. También, podemos notar que en las pruebas horizontales las trayectorias resultaron ser más desprolijas a comparación de las realizadas en las pruebas verticales.

5. Conclusiones finales.

Como conclusiones finales de este proyecto, en primera instancia puedo decir qué al brindar un mayor enfoque al desarrollo del soporte y automatización, no se pudieron llevar a cabo las modificaciones propuestas para el gabinete, sin embargo, se dejan planteadas las mismas para un desarrollo futuro.

En lo que respecta al soporte la estructura cumplió ampliamente los objetivos planteados logrando la disposición horizontal del robot, el soporte garantiza la rigidez estructural necesaria para no afectar la precisión del robot y no posee interferencias mecánicas, pudiendo aprovechar la totalidad del espacio de trabajo sin problemas.

El programa de simulación de movimiento en Matlab fue de gran utilidad a la hora de definir las trayectorias y para calcular los ángulos en pulsos necesarios para generar los movimientos. En lo que respecta a la automatización, la programación lograda en IPOSPlus cumple con los objetivos planteados, brindando la posibilidad de que los servomotores adopten los valores de ángulos de pulsos necesarios para otorgar el movimiento correspondiente al robot, además, se logró implementar un menú amigable con el usuario donde el mismo pueda seleccionar las trayectorias, iniciar o detener el movimiento en cualquier momento.

Hubo un cambio respecto a la idea inicial, que consistía en montar un actuador sobre el punto efector para transportar objetos. Se decidió implementar un fibrón para que el robot dibuje sobre una pizarra las diferentes trayectorias presentadas en el desarrollo del proyecto. Esta readaptación, permitió visualizar las capacidades y limitaciones tanto mecánicas como de automatización para este tipo de aplicación u otras que involucren el movimiento del punto efector describiendo de una trayectoria definida.

En lo que respecta a los resultados de las pruebas experimentales podemos concluir que los movimientos aleatorios del punto efector del robot se deben a que los servomotores se mueven a la misma velocidad, debido al tipo de configuración utilizada, haciendo que las trayectorias no describan líneas perfectas, lo que llevó a dividir en puntos cada trayectoria realizada a fines de disminuir este fenómeno. Por ello, a medida que aumentamos la longitud de una trayectoria o disminuimos la cantidad de puntos en la que se divide la misma, esta trayectoria aleatoria se hace más notoria. Además, debido cómo están configurados, los servomotores al realizar un movimiento, aceleran desde la posición inicial y se frenan en la posición final establecida, por lo que no realizan un movimiento continuo al describir la trayectoria, generando vibraciones sobre la cadena cinemática las cuales complementadas con los juegos de las juntas del robot, influyen considerablemente en la posición del fibrón sobre la pizarra, afectando la prolijidad de las trayectorias y las dimensiones obtenidas de las mismas, sobre todo en la disposición horizontal. También, al utilizar un soporte de gran longitud y por su forma de fijación, se observó que primero se desplaza el punto efector y luego la punta del fibrón, lo cual también afectó el dibujo de las trayectorias. Se pudo observar que la presión del fibrón contra la pizarra también afecta el desplazamiento del mismo por la fricción generada, efecto que se destacó en las pruebas horizontales debido a la acción de la gravedad sobre el conjunto. El efecto mencionado anteriormente y el juego de las juntas, influyeron en el aumento de la desprolijidad de las trayectorias en las pruebas horizontales

respecto de las verticales.

Respecto al análisis de velocidades angulares realizado se vio que, para lograr una trayectoria recta del punto efector, los servomotores deben moverse a velocidades diferentes y variables a lo largo del movimiento, lo cual es complejo de lograr con el tipo de control utilizado en este proyecto, excepto para el caso de la línea vertical, en donde ambos servomotores se mueven a la misma velocidad. Lo mencionado anteriormente explica que en las rectas verticales se perciban menos las trayectorias aleatorias en comparación con las otras curvas dibujadas, ya que los servomotores se mueven a la misma velocidad en sentidos opuestos. Teniendo en cuenta lo mencionado respecto a los juegos y basándonos en los cálculos de capacidad portante de los brazos, se recomienda en la próxima etapa actualizar las juntas y materiales constructivos del robot, para lograr una mejor precisión en los movimientos y aumentar su capacidad de carga. Como conclusión final del análisis comentado, el robot realiza trayectorias aceptables si estas son pequeñas o se aumenta la cantidad de puntos que dividen a la misma. Se pueden mejorar las trayectorias realizadas implementando las siguientes mejoras a lo que se dispone actualmente: disminuir la velocidad de los servomotores a fines de reducir las aceleraciones de arranque y parada en cada punto, mejorar la rigidez del soporte del fibrón.

Se deja una gran base para optar por dos caminos en una futura etapa de desarrollo: realizar una investigación para mejorar la automatización, contemplando la posibilidad de utilizar un software que pueda configurar velocidades diferentes en los servomotores y sacar el máximo potencial del robot para describir todo tipo de trayectorias o implementar un actuador sobre el punto efector para lograr transportar objetos, ya que el robot puede usarse sin inconvenientes para este tipo de aplicación didáctica debido a que no influye la trayectoria que realiza el robot entre punto y punto. También, se propone estudiar los esfuerzos dinámicos generados durante el movimiento del robot.

Para finalizar, considero que este proyecto me desafió día a día, permitiéndome aplicar un poco de todos los conceptos adquiridos durante el cursado de la carrera y aprender conceptos nuevos, sobre todo relacionados con la automatización, además, demuestra que la Ingeniería Mecánica es muy importante en el diseño de sistemas mecatrónicos.

Referencias Bibliográficas

- [1] Jairo Orlando Grande Murcia Brian Alexis Bernal Romero. «DISEÑO Y CONSTRUCCIÓN DE UN ROBOT PARALELO PLANAR TIPO SCARA CON 2 GDL RRRRR PARA FINES DIDÁCTICOS». Tesis doct. Universidad Distrital Francisco José de Caldas, Facultad Tecnológica Bogotá DC, 2019.
- [2] Paloma Barat Carnino. «IMPLEMENTACIÓN OPERATIVA DE PROTOTIPO ROBÓTICO». Tesis doct. Universidad Nacal de Rosario, Facultad de Ciencias Exactas, Ingeniería y Agrimensura, Escuela de Ingeniería Mecánica, 2023.
- [3] M. Díaz-Rodríguez y et al. «Aplicación de los robots paralelos». En: *En Manipuladores paralelos: Síntesis, análisis y aplicaciones (Pereira: Universidad Tecnológica de Pereira)*. (2018).
- [4] Mitsubishi Electric. *Mitsubishi Electric*. URL: <https://es.mitsubishielectric.com/fa/products/rbt/robot/horizontal>.
- [5] Brandon J. Gamble. «5-Bar Linkage Kinematic Solver and Simulator». Tesis doct. University of Vermont, College of Engineering y Mathematical Sciences, Department of Mechanical Engineering, 2020.
- [6] Russell C. Hibbeler. *Mecánica de materiales. 8va Ed.* PEARSON EDUCACIÓN, México., 2011.
- [7] Iván. de Lajonquiere. *Hoja de cálculos*. URL: <https://docs.google.com/spreadsheets/d/1MXW-UAFwJNZMRq3L8BxoN366vV1cEY1/edit?usp=sharing&ouid=106306679842349764148&rtpof=true&sd=true>.
- [8] Inc. Mecademic. *Mecademic DexTAR Manual de Usuario*. URL: https://www.researchgate.net/publication/277299749_User's_manual_for_the_DexTAR_five-bar_parallel_robot.
- [9] Christian Gustavo Morales Muñoz. «DISEÑO Y SIMULACIÓN DE UN CONTROLADOR DE POSICIÓN BASADO EN MODOS DESLIZANTES PARA UN ROBOT TIPO SCARA PARALELO». Tesis doct. Escuela Politécnica Nacional de Ecuador, Facultad de Ingeniería Eléctrica y Electrónica, 2017.
- [10] Robert L. Norton. *Diseño de Maquinaria. 4ta Ed.* McGraw-Hill Interamericana de España S.L., 2009.
- [11] Sew Eurodrive del Perú SAC. *Programación en IPOS - Control de movimiento de motores y servos - Nivel 1*. 2017. URL: <https://pdfcoffee.com/programacion-en-ipos-pdf-free.html>.
- [12] Kawasaki Robotics. *Robot Kawasaki*. URL: <https://kawasakirobotics.com/es/products-robots/yf002n/>.

Índice de figuras

1.	Robot paralelo para pintado [3]	1
2.	Máquina de pruebas ideada por Eric Gough [3]	2
3.	Simulador de vuelo Lufthansa basado en plataforma de Stewart [[3]]	2
4.	Robot paralelo centro de mecanizado Variax [3]	3
5.	Robot paralelo centro de mecanizado Okuma PM600 [3].	3
6.	Robot paralelo para mecanizado de blocks de motor [1].	4
7.	Robot delta desarrollado por Clavel [3].	5
8.	Robot delta YF002N desarrollado por Kawasaki Robotics [12].	5
9.	Robot SCARA serie [4].	6
10.	Robots SCARA paralelos.	6
11.		7
12.		10
13.	Determinación espacio de trabajo [9].	12
14.	Espacio de trabajo [9].	12
15.	Singularidad serie [5].	13
16.	Singularidades serie al alcanzar la frontera del espacio de trabajo [9].	13
17.	Singularidad paralela [5].	14
18.	Evitar singularidades paralelas variando el modo de trabajo [5].	14
19.	Modos de trabajo del robot SCARA paralelo [5].	15
20.	Esquema solución cinemática inversa [5].	16
21.	Esquema solución cinemática inversa [8].	18
22.	Esquema solución cinemática directa [8].	19
23.	Componentes dentro del gabinete [Propia].	21
24.	Ventana de acceso [Propia].	22
25.	Entrada y salida de cableado [Propia].	22

26.	Refrigeración gabinete [Propia].	23
27.	Soporte tipo pórtico [Propia].	24
28.	Dimensiones máximas espacio de trabajo robot [Propia].	25
29.	Bases de vinculación pórtico [Propia].	25
30.	Vinculación bases a la mesa [Propia].	26
31.	Definición distancia libre mínima [Propia].	27
32.	Dimensiones generales soporte. [Propia].	27
33.	Dimensiones generales travesaño [Propia].	28
34.	Ranuras para posicionamiento de los refuerzos [Propia].	28
35.	Encastre chapas refuerzos [Propia].	29
36.	Encastre chapas refuerzos [Propia].	29
37.	Momento torsor generado por una carga suspendida en el punto efector. [Propia]. . .	31
38.	Máxima distancia del punto efector. [Propia].	31
39.	Cargas aplicadas en la simulación [Propia].	32
40.	Simulación estática preliminar 1 [Propia].	33
41.	Simulación estática preliminar 2 [Propia].	35
42.	Simulación estática preliminar 3 [Propia].	36
43.	Preparación simulación estática final [Propia].	37
44.	Simulación estática final [Propia].	38
45.	Análisis de vibraciones [Propia].	39
46.	Simulación de pandeo. [Propia].	40
47.	Cargas aplicadas en FTool [Propia].	41
48.	Esfuerzo normal en N [Propia].	41
49.	Esfuerzo cortante en N [Propia].	42
50.	Momento flector en Nmm [Propia].	42
51.	Puntos críticos sección transversal [Propia].	43
52.	Propiedades sección transversal [Propia].	43

53.	Cálculos tensiones de Von Mises [Propia].	44
54.	Posiciones del brazo para analizar los esfuerzos [Propia].	45
55.	Esquema brazo en situación intermedia [Propia].	46
56.	Reacciones en los apoyos [Propia].	46
57.	Sección I-I [Propia].	47
58.	Sección I-I [Propia].	48
59.	Puntos críticos sección transversal brazos [Propia].	48
60.	Datos brazos [Propia].	49
61.	Cálculos $\beta = 90^\circ$ [Propia].	50
62.	Cálculos $\beta = 0^\circ$ [Propia].	50
63.	Datos módulo elasticidad y rotura maderas [Propia].	52
64.	Cálculo puntos (x,y) circunferencia [Propia].	54
65.	Selección de trayectorias y modo de trabajo [Propia].	54
66.	Sentencia case para selección de trayectoria [Propia].	56
67.	Invocación función cinemática directa [Propia].	56
68.	Cálculos pulsos servomotores [Propia].	57
69.	Cálculos gráfica espacio de trabajo [Propia].	57
70.	Gráfica movimiento [Propia].	58
71.	Simulación movimiento robot para recta horizontal [Propia].	59
72.	Simulación movimiento robot para recta vertical [Propia].	59
73.	Simulación movimiento robot para recta diagonal [Propia].	60
74.	Simulación movimiento robot para cuadrado [Propia].	60
75.	Simulación movimiento robot para triángulo [Propia].	61
76.	Simulación movimiento robot para circunferencia [Propia].	61
77.	Simulación movimiento robot para traslado zig-zag [Propia].	62
78.	Simulación movimiento robot para logo [Propia].	62
79.	Definición variables generales [Propia].	63

80.	Definición variables para θ_1 y θ_2 [Propia].	64
81.	Función trayectoria parte inicial [Propia].	65
82.	Función trayectoria parte final [Propia].	66
83.	Función logopoli parte 1 [Propia].	66
84.	Función logopoli parte 2 [Propia].	67
85.	Función logopoli parte 3 [Propia].	67
86.	Función logopoli parte 4 [Propia].	67
87.	Función logopoli parte 5 [Propia].	68
88.	Función homming parte 1 [Propia].	68
89.	Función homming parte 2 [Propia].	69
90.	Estado de espera en programa principal [Propia].	69
91.	Stop [Propia].	70
92.	Home post stop [Propia].	70
93.	Selección de trayectoria parte 1 [Propia].	71
94.	Selección de trayectoria parte 2 [Propia].	71
95.	Selección logo en programa principal parte 1 [Propia].	72
96.	Selección logo en programa principal parte 2 [Propia].	73
97.	Selección logo en programa principal parte 3 [Propia].	73
98.	Menú humano-maquina [Propia].	74
99.	Perfil de velocidad punto efector en estudio de movimiento [Propia].	75
100.	Estudio de movimiento [Propia].	76
101.	Resultado velocidades angulares recta diagonal [Propia].	76
102.	Resultado velocidades angulares recta horizontal [Propia].	77
103.	Resultado velocidades angulares recta vertical [Propia].	77
104.	Chapas cortadas y plegadas[Propia].	79
105.	Montaje final [Propia].	79
106.	Disposiciones del robot para las pruebas [Propia].	80

107. Comparación líneas horizontales - prueba vertical [Propia].	81
108. Comparación líneas verticales - prueba vertical [Propia].	82
109. Comparación líneas diagonales - prueba vertical [Propia].	82
110. Cuadrados - prueba vertical [Propia].	83
111. Triángulos - prueba vertical [Propia].	84
112. Circunferencias - prueba vertical [Propia].	85
113. Logo - prueba vertical [Propia].	86
114. Comparación líneas horizontales - prueba horizontal [Propia].	87
115. Comparación líneas verticales - prueba horizontal [Propia].	87
116. Comparación líneas diagonales - prueba horizontal [Propia].	88
117. Cuadrados - prueba horizontal [Propia].	89
118. Triángulos - prueba horizontal [Propia].	90
119. Circunferencias - prueba horizontal [Propia].	91
120. Logo - prueba horizontal [Propia].	92
121. Movimiento zigzag - prueba horizontal [Propia].	92

Índice de Tablas

1.	Modos de trabajo en función de la combinación de β y α	17
2.	Condiciones para cálculo estructural.	41
3.	Resumen propiedades sección transversal.	42
4.	Resumen propiedades sección transversal brazos.	49

6. Anexos

6.1. Anexo A: Programas Matlab

6.2. Anexo B: Programas IPOSPPlus

6.3. Anexo C: Planimetría

```

1 clc, clear all, close all
2
3 %% datos robot paralelo SCARA y selección de trayectorias a realizar
4 % Longitud de las Barras:
5 d = 450; % [mm]
6 L1 = 200; % [mm]
7 L2 = 250; % [mm]
8 modo_tbj = 1; %indica modo de trabajo
9 trayectoria = 6; % 1 = circunferencia, 2 = recta, 3 = triangulo, 4 = cuadrado, 5 logo, 6 zigzag
10 arh = 0; % si arh = 1 la función recta hace una recta horizontal
11 arv = 1; % si arv = 1 la función recta hace una recta vertical
12 % si arh = arv = 0 funcion recta hace la recta diagonal
13
14 rh = 4; % rh = 1 -> l = 20mm; rh = 2 -> l = 40mm; rh = 3 -> l = 60mm; rh = 4 -> l = 80mm
15 rv = 1; % rv = 1 -> l = 20mm; rv = 2 -> l = 40mm; rv = 3 -> l = 60mm; rv = 4 -> l = 80mm
16 circ = 1; % circ = 1 -> r = 10mm; circ = 2 -> r = 20mm; circ = 3 -> r = 30mm; circ = 4 -> r = 50mm;
17 rd = 4; % rd = 1 -> l = (2)^1/2*10mm; rd = 2 -> l = (2)^1/2*25mm; rd = 3 -> l = (2)^1/2*30mm; rd = 4 -> l = (2)^1/2*40mm
18 cuad = 4; % cuad = 1 -> l = 20mm; cuad = 2 -> l = 40mm; cuad = 3 -> l = 80mm; cuad = 4 -> l = 100mm
19 triang = 2; % triang = 1 -> b = 20mm; triang = 2 -> b = 40mm; triang = 3 -> b = 60mm; triang = 4 -> b = 80mm
20 graf = 1; % graf = 1 hace gráfica de la evolución del robot, si graf = 0 no grafica nada (en Octave pone graf 0 así no jode la
grafica en blanco)
21 modo_home = 1; %indica el tipo de home que hace el robot, 1 es para los brazos paralelos a la horizontal y 2 es para los brazos
con desfase de 90°
22 pulso = 4096; %4096 pulsos equivalen a una vuelta del servomotor
23 ir = 19.71; %índice de reducción reductor de ejes paralelos
24
25 %% Cálculo espacio de trabajo robot scara paralelo
26
27 r = L1 + L2; % Radio de la circunferencia suma de la longitud de ambos brazos
28 alpha = linspace(0, 2*pi, 100); % Ángulos de 0 a 2*pi
29 x1 = r * cos(alpha)-d/2;
30 y1 = r * sin(alpha);
31
32 x2 = r * cos(alpha)+d/2;
33 y2 = r * sin(alpha);
34
35 %% cálculo angulos servos para realizar trayectoria
36
37 switch trayectoria
38
39     case 1 %realiza una circunferencia centrada en (0,310)
40         [dita1_g, dita2_g,xc,yc] = cin_inv_SCARA_circ(L1,L2,modo_tbj,d,circ);
41
42         theta1 = dita1_g;
43         theta2 = dita2_g;
44     case 2 %realiza una recta desde el punto (0,300)
45
46         [dita1_g, dita2_g,xc,yc] = cin_inv_SCARA_recta(L1,L2,modo_tbj,d,rd,rv,rh,arh,arv);
47
48         theta1 = dita1_g;
49         theta2 = dita2_g;
50
51     case 3 %realiza un triángulo desde el punto (0,300)
52         [dita1_g, dita2_g,xc,yc] = cin_inv_SCARA_triang(L1,L2,modo_tbj,d,triang);
53
54         theta1 = dita1_g;
55         theta2 = dita2_g;
56
57     case 4 %realiza un cuadrado
58         [dita1_g, dita2_g,xc,yc] = cin_inv_SCARA_cuad(L1,L2,modo_tbj,d,cuad);

```

```

59
60     theta1 = dita1_g;
61     theta2 = dita2_g;
62
63     case 5 %hace el logo
64         [dita1_g, dita2_g,xc,yc] = logopoli(L1,L2,modo_tbj,d);
65
66         theta1 = dita1_g;
67         theta2 = dita2_g;
68     case 6 %calcula el zig zag para traslado
69         [dita1_g, dita2_g,xc,yc] = cin_inv_SCARA_trasladozigzag(L1,L2,modo_tbj,d);
70
71         theta1 = dita1_g;
72         theta2 = dita2_g;
73 end
74
75 %% Pasaje de angulos en ° a pulsos
76
77 % calcula los pulsos a ingresar a IPOS si el home se hace con los
78 % brazos paralelos a la horizontal
79 if modo_home == 1
80     dita2_g_b = 180 - dita2_g;
81     dita1_p = -1*round((dita1_g/360)*ir*pulso); % paso el angulo de grados a pulsos para cargar en el servo
82     dita2_p = round(1*(dita2_g/360)*ir*pulso);
83     dita2_p_b = round(1*(dita2_g_b/360)*ir*pulso);
84 end
85
86 % calcula los pulsos a ingresa a IPOS si el home se hace con el robot
87 % formando una casita (desfasaje 90°).
88 if modo_home == 2
89     dita2_g_b = 180 - dita2_g;
90     dita1_p = -1*round((dita1_g/360)*ir*pulso)-round((90/360)*ir*pulso); % paso el angulo de grados a pulsos para cargar en el
91     dita2_p = round((dita2_g/360)*ir*pulso)-round((90/360)*ir*pulso);
92     dita2_p_b = round((dita2_g_b/360)*ir*pulso)-round((90/360)*ir*pulso);
93 end
94
95 %% inicialización para la gráfica de la disposición del robot
96
97 if graf == 1
98     figure;
99     hold on; axis equal; grid on;
100     xlim([-420 420]); ylim([-400 400])
101     set(gcf, 'WindowState', 'maximized');
102     title('Disposición robot SCARA paralelo');
103     brazo_dist1 = plot([0,0],[0,0],'b-','LineWidth',2);
104     brazo_dist2 = plot([0,0],[0,0],'b-','LineWidth',2);
105     brazo_dist1_sol2 = plot([0,0],[0,0],'g--','LineWidth',2);
106     brazo_dist2_sol2 = plot([0,0],[0,0],'g--','LineWidth',2);
107     brazo_prox1 = plot([0,0],[0,0],'r-','LineWidth',2);
108     brazo_prox2 = plot([0,0],[0,0],'r-','LineWidth',2);
109     efector = plot(0, 0, 'ko', 'MarkerFaceColor', 'g', 'MarkerSize', 8); % efector
110     efector_sol2 = plot(0, 0, 'ko', 'MarkerFaceColor', 'm', 'MarkerSize', 8); % efector solucion 2
111     motor1 = plot(-d/2, 0, 'ko', 'MarkerFaceColor', 'k');
112     motor2 = plot(d/2, 0, 'ko', 'MarkerFaceColor', 'k');
113     plot(x1, y1,'k');
114     plot(x2, y2,'k');
115     plot(xc, yc,'m');
116 end
117

```

```

118 %% cálculo posición del punto efector
119
120 fprintf('-----Posición punto efector en coordenadas cartesianas (x,y).-----\n');
121
122 for i = 1:length(theta1)
123
124     [rOC_sol1, rOC_sol2, O1, O2, rOA1, rOA2, rO1A1, rO2A2] = cin_dir_SCARA(L1, L2, theta1(i), theta2(i), d); %esta función resuelve la cinemática directa, obteniendo los puntos a través de los ángulos de entrada.
125
126     % Mostrar resultados
127     fprintf('Posición %.f\n', i);
128     fprintf('Solución 1: x1 = %.2f, y1 = %.2f\n', rOC_sol1(1), rOC_sol1(2));
129     fprintf('Solución 2: x2 = %.2f, y2 = %.2f\n', rOC_sol2(1), rOC_sol2(2));
130     fprintf('\n', i);
131
132
133     %gráfica espacio de trabajo
134
135     if graf == 1
136         % Actualizar líneas y puntos
137         set(brazo_prox1, 'XData', [O1(1), rOA1(1)], 'YData', [O1(2), rOA1(2)]);
138         set(brazo_prox2, 'XData', [O2(1), rOA2(1)], 'YData', [O2(2), rOA2(2)]);
139         set(brazo_dist1, 'XData', [rOA1(1), rOC_sol1(1)], 'YData', [rOA1(2), rOC_sol1(2)]);
140         set(brazo_dist2, 'XData', [rOA2(1), rOC_sol1(1)], 'YData', [rOA2(2), rOC_sol1(2)]);
141         set(brazo_dist1_sol2, 'XData', [rOA1(1), rOC_sol2(1)], 'YData', [rOA1(2), rOC_sol2(2)]);
142         set(brazo_dist2_sol2, 'XData', [rOA2(1), rOC_sol2(1)], 'YData', [rOA2(2), rOC_sol2(2)]);
143         set(efector, 'XData', rOC_sol1(1), 'YData', rOC_sol1(2)); %modo de ensamblaje 1.
144         set(efector_sol2, 'XData', rOC_sol2(1), 'YData', rOC_sol2(2)); %modo de ensamblaje 1.
145         drawnow;
146         pause(0.08); % control de velocidad
147     end
148
149 end
150 fprintf('-----\n');
151 %% Muestra resultados
152
153
154 % ----- Mostrar resultados de angulos en pulsos para ingresar a IPOS -----
155
156 fprintf('-----Valores de pulsos a ingresar en IPOS.-----\n');
157
158 if trayectoria == 5
159     fprintf('====1er segmento.====\n');
160
161     for i = 1:20
162
163         fprintf('p%.0f_dita1 = %0.f\n', i, dita1_p(i));
164         fprintf('p%.0f_dita2 = %0.f\n', i, dita2_p_b(i));
165         fprintf('\n', i);
166     end
167
168     fprintf('====2do segmento.====\n');
169
170     for i = 21:40
171
172         fprintf('p%.0f_dita1 = %0.f\n', i-20, dita1_p(i));
173         fprintf('p%.0f_dita2 = %0.f\n', i-20, dita2_p_b(i));
174         fprintf('\n', i);
175     end
176

```

```

177 fprintf('====3er segmento.=====\n');
178
179     for i = 41:60
180
181         fprintf('p%.0f_dita1 = %0.f\n',i-40, dita1_p(i));
182         fprintf('p%.0f_dita2 = %0.f\n ',i-40, dita2_p_b(i));
183         fprintf('\n',i);
184     end
185
186 fprintf('====4to segmento.=====\n');
187
188     for i = 61:80
189
190         fprintf('p%.0f_dita1 = %0.f\n',i-60, dita1_p(i));
191         fprintf('p%.0f_dita2 = %0.f\n ',i-60, dita2_p_b(i));
192         fprintf('\n',i);
193     end
194
195 fprintf('====5to segmento.=====\n');
196
197     for i = 81:100
198
199         fprintf('p%.0f_dita1 = %0.f\n',i-80, dita1_p(i));
200         fprintf('p%.0f_dita2 = %0.f\n ',i-80, dita2_p_b(i));
201         fprintf('\n',i);
202     end
203
204 fprintf('====6to segmento.=====\n');
205
206     for i = 100:120
207
208         fprintf('p%.0f_dita1 = %0.f\n',i-100, dita1_p(i));
209         fprintf('p%.0f_dita2 = %0.f\n ',i-100, dita2_p_b(i));
210         fprintf('\n',i);
211     end
212
213 fprintf('====7mo segmento.=====\n');
214
215     for i = 121:140
216
217         fprintf('p%.0f_dita1 = %0.f\n',i-120, dita1_p(i));
218         fprintf('p%.0f_dita2 = %0.f\n ',i-120, dita2_p_b(i));
219         fprintf('\n',i);
220     end
221
222 fprintf('====8vo segmento.=====\n');
223
224     for i = 141:160
225
226         fprintf('p%.0f_dita1 = %0.f\n',i-140, dita1_p(i));
227         fprintf('p%.0f_dita2 = %0.f\n ',i-140, dita2_p_b(i));
228         fprintf('\n',i);
229     end
230
231
232 fprintf('====9no segmento.=====\n');
233
234     for i = 161:180
235
236         fprintf('p%.0f_dita1 = %0.f\n',i-160, dita1_p(i));

```

```

237         fprintf('p%.0f_dita2 = %.0f\n ',i-160, dita2_p_b(i));
238         fprintf('\n',i);
239     end
240     fprintf('=====10mo segmento.=====\\n');
241
242     for i = 181:200
243
244         fprintf('p%.0f_dita1 = %0.f\\n',i-180, dita1_p(i));
245         fprintf('p%.0f_dita2 = %.0f\\n ',i-180, dita2_p_b(i));
246         fprintf('\n',i);
247     end
248 else
249
250     for i = 1:length(theta1)
251
252         fprintf('p%.0f_dita1 = %0.f\\n',i, dita1_p(i));
253         fprintf('p%.0f_dita2 = %.0f\\n ',i, dita2_p_b(i));
254         fprintf('\n',i);
255     end
256 end
257 fprintf('-----\\n');
258
259 % ----- Mostrar resultados de angulos en grados para corroborar -----
260
261 fprintf('-----Angulos que adoptan los brazos.-----\\n');
262     for i = 1:length(theta1)
263
264         fprintf('Posición %.f\\n',i);
265         fprintf('theta 1 = %.2f°, theta 2 = %.2f°\\n ', theta1(i), dita2_g_b(i));
266         fprintf('\n',i);
267
268     end
269 fprintf('-----\\n');
270
271
272
273
274

```

```

1
2 function [dita1_g, dita2_g] = cin_inv_SCARA(xef,yef,L1,L2,modo_tbj,d)
3
4 % Longitud de las Barras:
5 w = d; % [mm]
6 a1 = L1; % [mm]
7 b1 = L2; % [mm]
8 b2 = L2; % [mm]
9 a2 = L1; % [mm]
10
11 desfasaje_O = 225;
12
13 xp = xef + desfasaje_O;
14 yp = yef;
15
16 %1 = beta1 y beta2 positivos (++)
17 %2 =beta1 positivo y beta2 negativo (+-)
18 %3 == beta1 negativo y beta1 posutivo (-+)
19 %4 ==beta1 y beta2 negativos(--)
20
21
22 %=====CALCULO DE POSICIONES=====
23 c1= sqrt((xp).^2+ yp.^2);
24 c2= sqrt((xp-d).^2+yp.^2);
25 alfa1= acos((xp)./(c1));
26 alfa2= acos((-xp+w)./(c2));
27 beta1= acos((b1.^2-a1.^2-c1.^2)./(-2*c1.*a1));
28 beta2= acos((b2.^2-a2.^2-c2.^2)./(-2*c2.*a2));
29
30 for i=1:length(c1)
31     if c1(i) > a1+b1
32         error('Punto fuera del espacio de trabajo. Brazos servo 1');
33     end
34 end
35
36 for i=1:length(c2)
37     if c2(i) > a2+b2
38         error('Punto fuera del espacio de trabajo. Brazos servo 2');
39     end
40 end
41
42 switch modo_tbj
43
44     case 1 %beta1 y beta2 positivos (++)
45         dita1=alfa1+beta1;
46         dita2= alfa2+beta2;
47
48     case 2 %beta1 positivo y beta2 negativo (+-)
49         dita1=alfa1+beta1;
50         dita2= alfa2-beta2;
51
52     case 3 %beta1 negativo y beta1 posutivo (-+)
53         dita1=alfa1-beta1;
54         dita2= alfa2+beta2;
55
56     case 4 %4 beta1 y beta2 negativos(--)
57         dita1=alfa1-beta1;
58         dita2= alfa2-beta2;
59 end
60

```

```
61  dita1_g = rad2deg(dita1);  
62  dita2_g = 180 - rad2deg(dita2);  
63  end  
64
```

```

1
2 function [rOC_sol1, rOC_sol2, O1,O2, rOA1, rOA2, rO1A1, rO2A2] = cin_dir_SCARA(L1,L2,theta1,theta2,d)
3
4 gamma= 1; % indica el modo de ensamblaje para la combinación de ángulos
5 theta1_rad = deg2rad(theta1); % [rad]
6 theta2_rad = deg2rad(theta2); % [rad]
7 E = [ 0 -1; 1 0]; % matriz de rotacion para calcular rHC a partir de rA2H
8 %% cálculo posición del punto efector
9
10 O1 = [-d/2;0]; %ubicación motor 1
11 O2 = [d/2;0]; %ubicación motor 2
12
13 rOA1 = [-d/2+L1*cos(theta1_rad), L1*sin(theta1_rad)]; %posición codo A1
14 rOA2 = [d/2+L1*cos(theta2_rad), L1*sin(theta2_rad)]; %posición codo A2
15
16 rO1A1 = [-d/2+L1*cos(theta1_rad)+d/2, L1*sin(theta1_rad)]; %calculo vector para luego graficar el mecanismo
17 rO2A2 = [d/2+L1*cos(theta2_rad)-d/2, L1*sin(theta2_rad)]; %calculo vector para luego graficar el mecanismo
18
19 A1A2 = rOA2-rOA1;
20
21 a = norm(A1A2)*0.5; %mitad segmento A-B
22 h = sqrt(L2^2-a^2); %altura triángulo
23
24 if 2*a > 2*L2
25     error('Los brazos no se cruzan: no hay solución real (punto fuera de alcance)');
26 end
27
28 rA2H = 0.5*(rOA1-rOA2);
29 rHC = -gamma*(h/a)*E*rA2H'; %vector perpendicular a A-B
30
31 rOC_sol1 = rOA2 + rA2H + rHC'; %solución modo de ensamblaje 1
32 rOC_sol2 = rOA2 + rA2H - rHC'; %solución modo de ensamblaje 2
33 end

```

```

1 function [dita1_g, dita2_g,xc,yc] = cin_inv_SCARA_trasladozigzag(L1,L2,modo_tbj,d)
2
3 % Longitud de las Barras:
4 w = d; % [mm]
5 a1 = L1; % [mm]
6 b1 = L2; % [mm]
7 b2 = L2; % [mm]
8 a2 = L1; % [mm]
9
10 desfasaje_O = 225;
11
12 %1 = beta1 y beta2 positivos (++)
13 %2 =beta1 positivo y beta2 negativo (+-)
14 %3 == beta1 negativo y beta1 posutivo (-+)
15 %4 ==beta1 y beta2 negativos(--)
16
17
18 %=====CALCULO DE POSICIONES=====
19
20 for i=1:6
21
22     switch i
23     case 1 %calcula la trayectoria para posicionarse el primer punto 1.1
24         xp1 = [0:-5:-40] + desfasaje_O;
25         yp1 = -0.5*(xp1-desfasaje_O) + 300;
26
27         xc1 = xp1 - desfasaje_O;
28         yc1 = yp1;
29
30         c1_1= sqrt((xp1).^2+ yp1.^2);
31         c2_1= sqrt((xp1-d).^2+yp1.^2);
32         alfa1_1= acos((c2_1.^2-c1_1.^2-w^2)./(-2*c1_1*w));
33         alfa2_1= acos((c1_1.^2-c2_1.^2-w^2)./(-2*c2_1*w));
34         beta1_1= acos((b1.^2-a1.^2-c1_1.^2)./(-2*c1_1.*a1));
35         beta2_1= acos((b2.^2-a2.^2-c2_1.^2)./(-2*c2_1.*a2));
36
37         switch modo_tbj
38
39             case 1
40                 dita1_1=alfa1_1+beta1_1;
41                 dita2_1= alfa2_1+beta2_1;
42             case 2
43                 dita1_1=alfa1_1+beta1_1;
44                 dita2_1= alfa2_1-beta2_1;
45             case 3
46                 dita1_1=alfa1_1-beta1_1;
47                 dita2_1= alfa2_1+beta2_1;
48             case 4
49                 dita1_1=alfa1_1-beta1_1;
50                 dita2_1= alfa2_1-beta2_1;
51         end
52
53         dita1_g_1 = rad2deg(dita1_1);
54         dita2_g_1 = 180 - rad2deg(dita2_1);
55
56     case 2 %calcula el movimiento hacia la posición 1.2
57         xp2 = [-40:10:40] + desfasaje_O;
58         yp2 = ones(1,9)*320;
59
60         xc2 = xp2 - desfasaje_O;

```

```

61     yc2 = yp2;
62
63     c1_2= sqrt((xp2).^2+ yp2.^2);
64     c2_2= sqrt((xp2-d).^2+yp2.^2);
65     alfa1_2= acos((c2_2.^2-c1_2.^2-w^2)./(-2*c1_2*w));
66     alfa2_2= acos((c1_2.^2-c2_2.^2-w^2)./(-2*c2_2*w));
67     beta1_2= acos((b1.^2-a1.^2-c1_2.^2)./(-2*c1_2.*a1));
68     beta2_2= acos((b2.^2-a2.^2-c2_2.^2)./(-2*c2_2.*a2));
69
70     switch modo_tbj
71
72         case 1
73             dita1_2=alfa1_2+beta1_2;
74             dita2_2= alfa2_2+beta2_2;
75         case 2
76             dita1_2=alfa1_2+beta1_2;
77             dita2_2= alfa2_2-beta2_2;
78         case 3
79             dita1_1=alfa1_1-beta1_1;
80             dita2_1= alfa2_1+beta2_1;
81         case 4
82             dita1_2=alfa1_2-beta1_2;
83             dita2_2= alfa2_2-beta2_2;
84     end
85
86     dita1_g_2 = rad2deg(dita1_2);
87     dita2_g_2 = 180 - rad2deg(dita2_2);
88
89     case 3 %calcula el movimiento hacia la posición 2.1
90         xp3 = [40:-10:-40] + desfasaje_O;
91         yp3 = (5/8)*(xp3-desfasaje_O) + 295;
92
93         xc3 = xp3 - desfasaje_O;
94         yc3 = yp3;
95
96         c1_3= sqrt((xp3).^2+ yp3.^2);
97         c2_3= sqrt((xp3-d).^2+yp3.^2);
98         alfa1_3= acos((c2_3.^2-c1_3.^2-w^2)./(-2*c1_3*w));
99         alfa2_3= acos((c1_3.^2-c2_3.^2-w^2)./(-2*c2_3*w));
100        beta1_3= acos((b1.^2-a1.^2-c1_3.^2)./(-2*c1_3.*a1));
101        beta2_3= acos((b2.^2-a2.^2-c2_3.^2)./(-2*c2_3.*a2));
102
103        switch modo_tbj
104
105            case 1
106                dita1_3=alfa1_3+beta1_3;
107                dita2_3= alfa2_3+beta2_3;
108            case 2
109                dita1_3=alfa1_3+beta1_3;
110                dita2_3= alfa2_3-beta2_3;
111            case 3
112                dita1_3=alfa1_3-beta1_3;
113                dita2_3= alfa2_3+beta2_3;
114            case 4
115                dita1_3=alfa1_3-beta1_3;
116                dita2_3= alfa2_3-beta2_3;
117        end
118
119        dita1_g_3 = rad2deg(dita1_3);
120        dita2_g_3 = 180 - rad2deg(dita2_3);

```

```

121
122 case 4 %calcula el movimiento hacia la posición 2.2
123 xp4 = [-40:10:40] + desfasaje_O;
124 yp4 = ones(1,9)*((5/8)*(185-desfasaje_O) + 295); %tener en cuenta q es -40+225 (desfasaje origen)
125
126 xc4 = xp4 - desfasaje_O;
127 yc4 = yp4;
128
129 c1_4= sqrt((xp4).^2+ yp4.^2);
130 c2_4= sqrt((xp4-d).^2+yp4.^2);
131 alfa1_4= acos((c2_4.^2-c1_4.^2-w^2)./(-2*c1_4*w));
132 alfa2_4= acos((c1_4.^2-c2_4.^2-w^2)./(-2*c2_4*w));
133 beta1_4= acos((b1.^2-a1.^2-c1_4.^2)./(-2*c1_4.*a1));
134 beta2_4= acos((b2.^2-a2.^2-c2_4.^2)./(-2*c2_4.*a2));
135
136 switch modo_tbj
137
138     case 1
139         dita1_4=alfa1_4+beta1_4;
140         dita2_4= alfa2_4+beta2_4;
141     case 2
142         dita1_4=alfa1_4+beta1_4;
143         dita2_4= alfa2_4-beta2_4;
144     case 3
145         dita1_4=alfa1_4-beta1_4;
146         dita2_4= alfa2_4+beta2_4;
147     case 4
148         dita1_4=alfa1_4-beta1_4;
149         dita2_4= alfa2_4-beta2_4;
150     end
151
152     dita1_g_4 = rad2deg(dita1_4);
153     dita2_g_4 = 180 - rad2deg(dita2_4);
154
155 end
156 end
157
158
159
160 dita1_g = [dita1_g_1, dita1_g_2, dita1_g_3, dita1_g_4];
161 dita2_g = [dita2_g_1, dita2_g_2, dita2_g_3, dita2_g_4];
162
163
164
165 xc = [xc1, xc2, xc3, xc4]; %concatenación para obtener la figura triangular graficando xc e yc.
166
167 yc = [yc1, yc2, yc3, yc4]; %concatenación para obtener la figura triangular graficando xc e yc.
168
169 theta1 = dita1_g;
170 theta2 = dita2_g;
171
172 end

```

```

1
2 function [dita1_g, dita2_g,xc,yc] = cin_inv_SCARA_circ(L1,L2,modo_tbj,d,circ)
3
4 % Longitud de las Barras:
5 w = d; % [mm]
6 a1 = L1; % [mm]
7 b1 = L2; % [mm]
8 b2 = L2; % [mm]
9 a2 = L1; % [mm]
10
11 desfasaje_O = 225;
12
13
14 %POSICION EFECTOR FINAL CIRCUNFERENCIA
15
16
17 switch circ
18     case 1
19         r = 10; %Radio de la circunferencia [mm]
20     case 2
21         r = 20; %Radio de la circunferencia [mm]
22     case 3
23         r = 30; %Radio de la circunferencia [mm]
24     case 4
25         r = 50; %Radio de la circunferencia [mm]
26 end
27
28 alpha = linspace(2*pi,0 , 20); % Ángulos de 0 a 2*pi
29 xp = r * cos(alpha)+desfasaje_O;
30 yp = r * sin(alpha)+330-r;
31
32 % guardo los puntos en estas variables para luego graficar la trayectoria
33 % el programa principal.
34
35 xc = xp - desfasaje_O;
36 yc = yp;
37
38 %1 = beta1 y beta2 positivos (++)
39 %2 =beta1 positivo y beta2 negativo (+-)
40 %3 == beta1 negativo y beta1 posutivo (-+)
41 %4 ==beta1 y beta2 negativos(--)
42
43
44 %=====CALCULO DE POSICIONES=====
45 c1= sqrt((xp).^2+ yp.^2);
46 c2= sqrt((xp-d).^2+yp.^2);
47 alfa1= acos((c2.^2-c1.^2-w^2)./(-2*c1*w));
48 alfa2= acos((c1.^2-c2.^2-w^2)./(-2*c2*w));
49 beta1= acos((b1.^2-a1.^2-c1.^2)./(-2*c1.*a1));
50 beta2= acos((b2.^2-a2.^2-c2.^2)./(-2*c2.*a2));
51
52 switch modo_tbj
53
54     case 1
55         dita1=alfa1+beta1;
56         dita2= alfa2+beta2;
57     case 2
58         dita1=alfa1+beta1;
59         dita2= alfa2-beta2;
60     case 3

```

```
61      dita1=alfa1-beta1;
62      dita2= alfa2+beta2;
63  case 4
64      dita1=alfa1-beta1;
65      dita2= alfa2-beta2;
66  end
67
68  dita1_g = rad2deg(dita1);
69  dita2_g = 180 - rad2deg(dita2);
70
71  end
72
```

```

1 function [dita1_g, dita2_g,xc,yc] = cin_inv_SCARA_cuad(L1,L2,modo_tbj,d,cuad)
2
3 % Longitud de las Barras:
4 w = d; % [mm]
5 a1 = L1; % [mm]
6 b1 = L2; % [mm]
7 b2 = L2; % [mm]
8 a2 = L1; % [mm]
9
10
11 switch cuad
12     case 1
13         l = 20; %[mm] longitud lado del cuadrado
14     case 2
15         l = 40; %[mm] longitud lado del cuadrado
16     case 3
17         l = 80; %[mm] longitud lado del cuadrado
18     case 4
19         l = 100; %[mm] longitud lado del cuadrado
20 end
21
22 desfasaje_O = 225;
23
24 %1 = beta1 y beta2 positivos (++)
25 %2 =beta1 positivo y beta2 negativo (+-)
26 %3 == beta1 negativo y beta1 posutivo (-+)
27 %4 ==beta1 y beta2 negativos(--)
28
29
30 %=====CALCULO DE POSICIONES=====
31
32
33 %cálcula lado superior
34 xp1 = linspace(-l*0.5,l*0.5,5) + desfasaje_O;
35 yp1 = ones(1,5)*300;
36
37 % guardo los puntos en estas variables para luego graficar la trayectoria
38 % en el programa principal.
39 xc1 = xp1 - desfasaje_O;
40 yc1 = yp1;
41
42 c1_1= sqrt((xp1).^2+ yp1.^2);
43 c2_1= sqrt((xp1-d).^2+yp1.^2);
44 alfa1_1= acos((c2_1.^2-c1_1.^2-w^2)./(-2*c1_1*w));
45 alfa2_1= acos((c1_1.^2-c2_1.^2-w^2)./(-2*c2_1*w));
46 beta1_1= acos((b1.^2-a1.^2-c1_1.^2)./(-2*c1_1.*a1));
47 beta2_1= acos((b2.^2-a2.^2-c2_1.^2)./(-2*c2_1.*a2));
48
49 switch modo_tbj
50
51     case 1
52         dita1_1=alfa1_1+beta1_1;
53         dita2_1= alfa2_1+beta2_1;
54     case 2
55         dita1_1=alfa1_1+beta1_1;
56         dita2_1= alfa2_1-beta2_1;
57     case 3
58         dita1_1=alfa1_1-beta1_1;
59         dita2_1= alfa2_1+beta2_1;
60     case 4

```

```

61         dita1_1=alfa1_1-beta1_1;
62         dita2_1= alfa2_1-beta2_1;
63     end
64
65     dita1_g_1 = rad2deg(dita1_1);
66     dita2_g_1 = 180 - rad2deg(dita2_1);
67
68     %cálcula lado derecho hacia abajo
69     xp2 = ones(1,5)*(l*0.5 + desfasaje_O);
70     yp2 = linspace(300,-l + 300,5);
71
72     xc2 = xp2 - desfasaje_O;
73     yc2 = yp2;
74
75     c1_2= sqrt((xp2).^2+ yp2.^2);
76     c2_2= sqrt((xp2-d).^2+yp2.^2);
77     alfa1_2= acos((c2_2.^2-c1_2.^2-w^2)./(-2*c1_2*w));
78     alfa2_2= acos((c1_2.^2-c2_2.^2-w^2)./(-2*c2_2*w));
79     beta1_2= acos((b1.^2-a1.^2-c1_2.^2)./(-2*c1_2.*a1));
80     beta2_2= acos((b2.^2-a2.^2-c2_2.^2)./(-2*c2_2.*a2));
81
82     switch modo_tbj
83
84         case 1
85             dita1_2=alfa1_2+beta1_2;
86             dita2_2= alfa2_2+beta2_2;
87         case 2
88             dita1_2=alfa1_2+beta1_2;
89             dita2_2= alfa2_2-beta2_2;
90         case 3
91             dita1_1=alfa1_1-beta1_1;
92             dita2_1= alfa2_1+beta2_1;
93         case 4
94             dita1_2=alfa1_2-beta1_2;
95             dita2_2= alfa2_2-beta2_2;
96     end
97
98     dita1_g_2 = rad2deg(dita1_2);
99     dita2_g_2 = 180 - rad2deg(dita2_2);
100
101     %cálcula lado inferior hacia la izquierda
102     xp3 = linspace(l*0.5,-l*0.5,5) + desfasaje_O;
103     yp3 =ones(1,5)*(300-l);
104
105     xc3 = xp3 - desfasaje_O;
106     yc3 = yp3;
107
108     c1_3= sqrt((xp3).^2+ yp3.^2);
109     c2_3= sqrt((xp3-d).^2+yp3.^2);
110     alfa1_3= acos((c2_3.^2-c1_3.^2-w^2)./(-2*c1_3*w));
111     alfa2_3= acos((c1_3.^2-c2_3.^2-w^2)./(-2*c2_3*w));
112     beta1_3= acos((b1.^2-a1.^2-c1_3.^2)./(-2*c1_3.*a1));
113     beta2_3= acos((b2.^2-a2.^2-c2_3.^2)./(-2*c2_3.*a2));
114
115     switch modo_tbj
116
117         case 1
118             dita1_3=alfa1_3+beta1_3;
119             dita2_3= alfa2_3+beta2_3;
120         case 2

```

```

121         dita1_3=alfa1_3+beta1_3;
122         dita2_3= alfa2_3-beta2_3;
123     case 3
124         dita1_3=alfa1_3-beta1_3;
125         dita2_3= alfa2_3+beta2_3;
126     case 4
127         dita1_3=alfa1_3-beta1_3;
128         dita2_3= alfa2_3-beta2_3;
129     end
130
131     dita1_g_3 = rad2deg(dita1_3);
132     dita2_g_3 = 180 - rad2deg(dita2_3);
133
134
135     %cálcula lado derecho hacia arriba
136     xp4 = ones(1,5)*(-l*0.5 + desfasaje_O);
137     yp4 = linspace(-l + 300,300,5);
138
139     xc4 = xp4 - desfasaje_O;
140     yc4 = yp4;
141
142     c1_4= sqrt((xp4).^2+ yp4.^2);
143     c2_4= sqrt((xp4-d).^2+yp4.^2);
144     alfa1_4= acos((c2_4.^2-c1_4.^2-w.^2)./(-2*c1_4*w));
145     alfa2_4= acos((c1_4.^2-c2_4.^2-w.^2)./(-2*c2_4*w));
146     beta1_4= acos((b1.^2-a1.^2-c1_4.^2)./(-2*c1_4.*a1));
147     beta2_4= acos((b2.^2-a2.^2-c2_4.^2)./(-2*c2_4.*a2));
148
149     switch modo_tbj
150
151     case 1
152         dita1_4=alfa1_4+beta1_4;
153         dita2_4= alfa2_4+beta2_4;
154     case 2
155         dita1_4=alfa1_4+beta1_4;
156         dita2_4= alfa2_4-beta2_4;
157     case 3
158         dita1_4=alfa1_4-beta1_4;
159         dita2_4= alfa2_4+beta2_4;
160     case 4
161         dita1_4=alfa1_4-beta1_4;
162         dita2_4= alfa2_4-beta2_4;
163     end
164
165     dita1_g_4 = rad2deg(dita1_4);
166     dita2_g_4 = 180 - rad2deg(dita2_4);
167
168
169
170
171
172     dita1_g = [dita1_g_1, dita1_g_2, dita1_g_3, dita1_g_4];
173     dita2_g = [dita2_g_1, dita2_g_2, dita2_g_3, dita2_g_4];
174
175
176     xc = [xc1, xc2, xc3, xc4]; %concatenación para obtener la figura triangular graficando xc e yc.
177
178     yc = [yc1, yc2, yc3, yc4]; %concatenación para obtener la figura triangular graficando xc e yc.
179
180     theta1 = dita1_g;

```

```
181  theta2 = dita2_g;  
182  
183  end  
184  
185  
186  
187  
188
```

```

1 function [dita1_g, dita2_g,xc,yc] = logopoli(L1,L2,modo_tbj,d)
2
3 % Longitud de las Barras:
4 w = d; % [mm]
5 a1 = L1; % [mm]
6 b1 = L2; % [mm]
7 b2 = L2; % [mm]
8 a2 = L1; % [mm]
9
10 arr_xp = ones(10,20); %la fila representa el xc de la trayectoria a hacer
11 arr_yp = ones(10,20); %la fila representa el yc de la trayectoria a hacer
12 arr_xc = ones(10,20); %la fila representa el xc de la trayectoria a hacer
13 arr_yc = ones(10,20); %la fila representa el yc de la trayectoria a hacer
14 arr_alfa1 = ones(10,20);
15 arr_alfa2 = ones(10,20);
16 arr_beta1 = ones(10,20);
17 arr_beta2 = ones(10,20);
18 arr_dita1 = ones(10,20);
19 arr_dita2 = ones(10,20);
20 arr_dita1_g = ones(10,20);
21 arr_dita2_g = ones(10,20);
22
23 desfasaje_O = 225;
24
25 % guardo los puntos en estas variables para luego graficar la trayectoria
26 % el programa principal.
27
28 % xc = xp - desfasaje_O;
29 % yc = yp;
30
31 %1 = beta1 y beta2 positivos (++)
32 %2 =beta1 positivo y beta2 negativo (+-)
33 %3 == beta1 negativo y beta1 posutivo (-+)
34 %4 ==beta1 y beta2 negativos(--)
35
36
37 %=====CALCULO DE POSICIONES=====
38
39
40 %calcula primer linea
41 arr_xp(1,:) = linspace(10,-30,20) + desfasaje_O;
42 arr_yp(1,:) = ones(1,20)*280;
43
44 arr_xc(1,:) = arr_xp(1,:) - desfasaje_O;
45 arr_yc(1,:) = arr_yp(1,:);
46
47 arr_xp(2,:) = ones(1,20)*(-30 + desfasaje_O);
48 arr_yp(2,:) = linspace(280,300,20);
49
50 arr_xc(2,:) = arr_xp(2,)- desfasaje_O;
51 arr_yc(2,:) = arr_yp(2,:);
52
53 arr_xp(3,:) = linspace(-30,30,20) + desfasaje_O;
54 arr_yp(3,:) = ones(1,20)*300;
55
56 arr_xc(3,:) = arr_xp(3,:) - desfasaje_O;
57 arr_yc(3,:) = arr_yp(3,:);
58
59 arr_xp(4,:) = ones(1,20)*(30 + desfasaje_O);
60 arr_yp(4,:) = linspace(300,-60 + 300,20);

```

```

61
62     arr_xc(4,:) = arr_xp(4,:) - desfasaje_O;
63     arr_yc(4,:) = arr_yp(4,:);
64
65     arr_xp(5,:) = linspace(30,-30,20) + desfasaje_O;
66     arr_yp(5,:) = ones(1,20)*(300-60);
67
68     arr_xc(5,:) = arr_xp(5,:) - desfasaje_O;
69     arr_yc(5,:) = arr_yp(5,:);
70
71     arr_xp(6,:) = ones(1,20)*(-30 + desfasaje_O);
72     arr_yp(6,:) = linspace(240,280,20);
73
74     arr_xc(6,:) = arr_xp(6,:) - desfasaje_O;
75     arr_yc(6,:) = arr_yp(6,:);
76
77     arr_xp(7,:) = linspace(-30,-10,20) + desfasaje_O;
78     arr_yp(7,:) = ones(1,20)*(280);
79
80     arr_xc(7,:) = arr_xp(7,:) - desfasaje_O;
81     arr_yc(7,:) = arr_yp(7,:);
82
83     arr_xp(8,:) = ones(1,20)*(-10 + desfasaje_O);
84     arr_yp(8,:) = linspace(280,240,20);
85
86     arr_xc(8,:) = arr_xp(8,:) - desfasaje_O;
87     arr_yc(8,:) = arr_yp(8,:);
88
89     arr_xp(9,:) = ones(1,20)*(-10 + desfasaje_O);
90     arr_yp(9,:) = linspace(240,260,20);
91
92     arr_xc(9,:) = arr_xp(9,:) - desfasaje_O;
93     arr_yc(9,:) = arr_yp(9,:);
94
95     arr_xp(10,:) = linspace(-10,30,20) + desfasaje_O;
96     arr_yp(10,:) = ones(1,20)*(260);
97
98     arr_xc(10,:) = arr_xp(10,:) - desfasaje_O;
99     arr_yc(10,:) = arr_yp(10,:);
100
101
102
103 for i=1:10
104     arr_c1(i,:)= sqrt((arr_xp(i,:).^2+ arr_yp(i,:).^2);
105     arr_c2(i,:)= sqrt((arr_xp(i,:)-d).^2+arr_yp(i,:).^2);
106     arr_alfa1(i,:)= acos((arr_c2(i,:).^2-arr_c1(i,:).^2-w^2)./(-2*arr_c1(i,:)*w));
107     arr_alfa2(i,:)= acos((arr_c1(i,:).^2-arr_c2(i,:).^2-w^2)./(-2*arr_c2(i,:)*w));
108     arr_beta1(i,:)= acos((b1.^2-a1.^2-arr_c1(i,:).^2)./(-2*arr_c1(i,:)*a1));
109     arr_beta2(i,:)= acos((b2.^2-a2.^2-arr_c2(i,:).^2)./(-2*arr_c2(i,:)*a2));
110
111     switch modo_tbj
112
113         case 1
114             arr_dita1(i,:)= arr_alfa1(i,:)+arr_beta1(i,:);
115             arr_dita2(i,:)= arr_alfa2(i,:)+arr_beta2(i,:);
116         case 2
117             arr_dita1(i,:)= arr_alfa1(i,:)+arr_beta1(i,:);
118             arr_dita2(i,:)= arr_alfa2(i,:)-arr_beta2(i,:);
119         case 3
120             arr_dita1(i,:)= arr_alfa1(i,:)-arr_beta1(i,:);

```

```

121         arr_dita2(i,:)= arr_alfa2(i,:)+arr_beta2(i,:);
122     case 4
123         arr_dita1(i,:)= arr_alfa1(i,:)-arr_beta1(i,:);
124         arr_dita2(i,:)= arr_alfa2(i,:)-arr_beta2(i,:);
125     end
126
127     arr_dita1_g(i,:) = rad2deg(arr_dita1(i,:));
128     arr_dita2_g(i,:) = 180 - rad2deg(arr_dita2(i,:));
129 end
130
131
132
133
134 dita1_g = [arr_dita1_g(1,:), arr_dita1_g(2,:), arr_dita1_g(3,:), arr_dita1_g(4,:), arr_dita1_g(5,:), arr_dita1_g(6,:), arr_dita1_g(7,:), ✓
arr_dita1_g(8,:), arr_dita1_g(9,:), arr_dita1_g(10,:)];
135 dita2_g = [arr_dita2_g(1,:), arr_dita2_g(2,:), arr_dita2_g(3,:), arr_dita2_g(4,:), arr_dita2_g(5,:), arr_dita2_g(6,:), arr_dita2_g(7,:), ✓
arr_dita2_g(8,:), arr_dita2_g(9,:), arr_dita2_g(10,:)];
136
137
138 xc = [arr_xc(1,:), arr_xc(2,:), arr_xc(3,:), arr_xc(4,:), arr_xc(5,:), arr_xc(6,:), arr_xc(7,:), arr_xc(8,:), arr_xc(9,:), arr_xc(10,:)]; % ✓
concatenación para obtener la figura triangular graficando xc e yc.
139
140 yc = [arr_yc(1,:), arr_yc(2,:), arr_yc(3,:), arr_yc(4,:), arr_yc(5,:), arr_yc(6,:), arr_yc(7,:), arr_yc(8,:), arr_yc(9,:), arr_yc(10,:)]; % ✓
concatenación para obtener la figura triangular graficando xc e yc.
141
142 theta1 = dita1_g;
143 theta2 = dita2_g;
144
145 end
146

```

```

1 function [dita1_g, dita2_g,xc,yc] = cin_inv_SCARA_recta(L1,L2,modo_tbj,d,rd,rv,rh,arh,arv)
2
3 % Longitud de las Barras:
4 w = d; % [mm]
5 a1 = L1; % [mm]
6 b1 = L2; % [mm]
7 b2 = L2; % [mm]
8 a2 = L1; % [mm]
9
10 desfasaje_O = 225;
11
12 if (arv == 1 && arh == 1) % hace recta en diagonal
13     error('Elija una opción de las siguientes: arv = 0 && arh = 0 -> recta diagonal / arv = 1 && arh = 0 -> recta vertical / arv =
0 && arh = 1 -> recta horizontal');
14 end
15
16 if (arv == 0 && arh == 0) % hace recta en diagonal
17     switch rd
18         case 1
19             l = 10; %[mm] longitud cateto recta
20         case 2
21             l = 25; %[mm] longitud cateto recta
22         case 3
23             l = 50; %[mm] longitud cateto recta
24         case 4
25             l = 80; %[mm] longitud cateto recta
26     end
27
28     %POSICION EFECTOR FINAL
29
30     xp = linspace(0,l,20) + desfasaje_O;
31     yp = -(xp-desfasaje_O) + 300;
32
33     % guardo los puntos en estas variables para luego graficar la trayectoria
34     % el programa principal.
35
36     xc = xp - desfasaje_O;
37     yc = yp;
38
39     %1 = beta1 y beta2 positivos (++)
40     %2 =beta1 positivo y beta2 negativo (+-)
41     %3 == beta1 negativo y beta1 posutivo (-+)
42     %4 ==beta1 y beta2 negativos(--)
43 end
44
45 if (arv == 1 && arh == 0) % hace la recta vertical
46     switch rv
47         case 1
48             l = 20; %[mm] longitud recta vertical
49             %POSICION EFECTOR FINAL
50             xp = ones(1,20)*(-40 + desfasaje_O);
51             yp = linspace(300,-l + 300,20);
52
53             xc = xp - desfasaje_O;
54             yc = yp;
55
56         case 2
57             l = 40; %[mm] longitud recta vertical
58             %POSICION EFECTOR FINAL
59             xp = ones(1,20)*(-20 + desfasaje_O);

```

```

60     yp = linspace(300,-l + 300,20);
61
62     xc = xp - desfasaje_O;
63     yc = yp;
64
65     case 3
66         l = 60; %[mm] longitud recta vertical
67         %POSICION EFECTOR FINAL
68         xp = ones(1,20)*(desfasaje_O);
69         yp = linspace(300,-l + 300,20);
70
71         xc = xp - desfasaje_O;
72         yc = yp;
73
74     case 4
75         l = 80; %[mm] longitud recta vertical
76         %POSICION EFECTOR FINAL
77         xp = ones(1,20)*(20 + desfasaje_O);
78         yp = linspace(300,-l + 300,20);
79
80         xc = xp - desfasaje_O;
81         yc = yp;
82     end
83 end
84
85 if (arv == 0 && arh == 1) % hace la recta horizontal
86     switch rh
87         case 1
88             l = 20; %[mm] longitud recta horizontal
89             %POSICION EFECTOR FINAL
90             xp = linspace(-l*0.5,l*0.5,20) + desfasaje_O;
91             yp = ones(1,20)*300;
92
93             xc = xp - desfasaje_O;
94             yc = yp;
95
96         case 2
97             l = 40; %[mm] longitud recta horizontal
98             %POSICION EFECTOR FINAL
99             xp = linspace(-l*0.5,l*0.5,20) + desfasaje_O;
100             yp = ones(1,20)*300-20;
101
102             xc = xp - desfasaje_O;
103             yc = yp;
104
105         case 3
106             l = 60; %[mm] longitud recta horizontal
107             %POSICION EFECTOR FINAL
108             xp = linspace(-l*0.5,l*0.5,20) + desfasaje_O;
109             yp = ones(1,20)*300-40;
110
111             xc = xp - desfasaje_O;
112             yc = yp;
113
114         case 4
115             l = 80; %[mm] longitud recta vertical
116             %POSICION EFECTOR FINAL
117             xp = linspace(-l*0.5,l*0.5,20) + desfasaje_O;
118             yp = ones(1,20)*300-60;
119

```

```

120         xc = xp - desfasaje_O;
121         yc = yp;
122     end
123 end
124
125 %=====CALCULO DE POSICIONES=====
126 c1= sqrt((xp).^2+ yp.^2);
127 c2= sqrt((xp-d).^2+yp.^2);
128 alfa1= acos((c2.^2-c1.^2-w.^2)/(-2*c1*w));
129 alfa2= acos((c1.^2-c2.^2-w.^2)/(-2*c2*w));
130 beta1= acos((b1.^2-a1.^2-c1.^2)/(-2*c1.*a1));
131 beta2= acos((b2.^2-a2.^2-c2.^2)/(-2*c2.*a2));
132
133 switch modo_tbj
134
135     case 1
136         dita1=alfa1+beta1;
137         dita2= alfa2+beta2;
138     case 2
139         dita1=alfa1+beta1;
140         dita2= alfa2-beta2;
141     case 3
142         dita1=alfa1-beta1;
143         dita2= alfa2+beta2;
144     case 4
145         dita1=alfa1-beta1;
146         dita2= alfa2-beta2;
147 end
148
149 dita1_g = rad2deg(dita1);
150 dita2_g = 180 - rad2deg(dita2);
151
152 end

```

```

1 function [dita1_g, dita2_g,xc,yc] = cin_inv_SCARA_triang(L1,L2,modo_tbj,d,triang)
2
3 % Longitud de las Barras:
4 w = d; % [mm]
5 a1 = L1; % [mm]
6 b1 = L2; % [mm]
7 b2 = L2; % [mm]
8 a2 = L1; % [mm]
9
10 desfasaje_O = 225;
11
12 switch triang
13     case 1
14         b = 20; %[mm] longitud lado del cuadrado
15     case 2
16         b = 40; %[mm] longitud lado del cuadrado
17     case 3
18         b = 60; %[mm] longitud lado del cuadrado
19     case 4
20         b = 80; %[mm] longitud lado del cuadrado
21 end
22
23
24
25 %POSICION EFECTOR FINAL CIRCUNFERENCIA
26
27 % xp = [0:5:100] + desfasaje_O;
28 % yp = -(xp-desfasaje_O) + 300;
29
30 % guardo los puntos en estas variables para luego graficar la trayectoria
31 % el programa principal.
32
33 % xc = xp - desfasaje_O;
34 % yc = yp;
35
36 %1 = beta1 y beta2 positivos (++)
37 %2 =beta1 positivo y beta2 negativo (+-)
38 %3 == beta1 negativo y beta1 posutivo (-+)
39 %4 ==beta1 y beta2 negativos(--)
40
41
42 %=====CALCULO DE POSICIONES=====
43
44 for i=1:3
45
46     switch i
47         case 1 %calcula la primera recta decreciente
48             xp1 = linspace(0,b,7) + desfasaje_O;
49             yp1 = -(xp1-desfasaje_O) + 300;
50
51             xc1 = xp1 - desfasaje_O;
52             yc1 = yp1;
53
54             c1_1= sqrt((xp1).^2+ yp1.^2);
55             c2_1= sqrt((xp1-d).^2+yp1.^2);
56             alfa1_1= acos((c2_1.^2-c1_1.^2-w.^2)./(-2*c1_1*w));
57             alfa2_1= acos((c1_1.^2-c2_1.^2-w.^2)./(-2*c2_1*w));
58             beta1_1= acos((b1.^2-a1.^2-c1_1.^2)./(-2*c1_1.*a1));
59             beta2_1= acos((b2.^2-a2.^2-c2_1.^2)./(-2*c2_1.*a2));
60

```

```

61     switch modo_tbj
62
63         case 1
64             dita1_1=alfa1_1+beta1_1;
65             dita2_1= alfa2_1+beta2_1;
66         case 2
67             dita1_1=alfa1_1+beta1_1;
68             dita2_1= alfa2_1-beta2_1;
69         case 3
70             dita1_1=alfa1_1-beta1_1;
71             dita2_1= alfa2_1+beta2_1;
72         case 4
73             dita1_1=alfa1_1-beta1_1;
74             dita2_1= alfa2_1-beta2_1;
75     end
76
77     dita1_g_1 = rad2deg(dita1_1);
78     dita2_g_1 = 180 - rad2deg(dita2_1);
79
80     case 2 %calcula la trayectoria horizontal para formar la base del triángulo
81         xp2 = linspace(b,-b,7) + desfasaje_O;
82         yp1_21 = -b + 300; %yp1(21)
83         yp2 = ones(1,7)*yp1_21;
84
85         xc2 = xp2 - desfasaje_O;
86         yc2 = yp2;
87
88         c1_2= sqrt((xp2).^2+ yp2.^2);
89         c2_2= sqrt((xp2-d).^2+yp2.^2);
90         alfa1_2= acos((c2_2.^2-c1_2.^2-w^2)./(-2*c1_2*w));
91         alfa2_2= acos((c1_2.^2-c2_2.^2-w^2)./(-2*c2_2*w));
92         beta1_2= acos((b1.^2-a1.^2-c1_2.^2)./(-2*c1_2.*a1));
93         beta2_2= acos((b2.^2-a2.^2-c2_2.^2)./(-2*c2_2.*a2));
94
95     switch modo_tbj
96
97         case 1
98             dita1_2=alfa1_2+beta1_2;
99             dita2_2= alfa2_2+beta2_2;
100        case 2
101            dita1_2=alfa1_2+beta1_2;
102            dita2_2= alfa2_2-beta2_2;
103        case 3
104            dita1_1=alfa1_1-beta1_1;
105            dita2_1= alfa2_1+beta2_1;
106        case 4
107            dita1_2=alfa1_2-beta1_2;
108            dita2_2= alfa2_2-beta2_2;
109    end
110
111    dita1_g_2 = rad2deg(dita1_2);
112    dita2_g_2 = 180 - rad2deg(dita2_2);
113
114    case 3 %calcula la trayectoria de la recta creciente para cerrar el triángulo
115        xp3a = linspace(-b,0,7) + desfasaje_O;
116        xp3 = xp3a(2:end);
117        yp3 =(xp3-desfasaje_O) + 300;
118
119        xc3 = xp3 - desfasaje_O;
120        yc3 = yp3;

```

```

121
122     c1_3= sqrt((xp3).^2+ yp3.^2);
123     c2_3= sqrt((xp3-d).^2+yp3.^2);
124     alfa1_3= acos((c2_3.^2-c1_3.^2-w^2)./(-2*c1_3*w));
125     alfa2_3= acos((c1_3.^2-c2_3.^2-w^2)./(-2*c2_3*w));
126     beta1_3= acos((b1.^2-a1.^2-c1_3.^2)./(-2*c1_3.*a1));
127     beta2_3= acos((b2.^2-a2.^2-c2_3.^2)./(-2*c2_3.*a2));
128
129     switch modo_tbj
130
131         case 1
132             dita1_3=alfa1_3+beta1_3;
133             dita2_3= alfa2_3+beta2_3;
134         case 2
135             dita1_3=alfa1_3+beta1_3;
136             dita2_3= alfa2_3-beta2_3;
137         case 3
138             dita1_3=alfa1_3-beta1_3;
139             dita2_3= alfa2_3+beta2_3;
140         case 4
141             dita1_3=alfa1_3-beta1_3;
142             dita2_3= alfa2_3-beta2_3;
143         end
144
145         dita1_g_3 = rad2deg(dita1_3);
146         dita2_g_3 = 180 - rad2deg(dita2_3);
147     end
148 end
149
150 % dita1_g = dita1_g_1;
151 % dita2_g = dita2_g_1;
152 %
153 % dita1_g = dita1_g_2;
154 % dita2_g = dita2_g_2;
155 %
156 % dita1_g = dita1_g_3;
157 % dita2_g = dita2_g_3;
158
159
160
161 dita1_g = [dita1_g_1, dita1_g_2, dita1_g_3];
162 dita2_g = [dita2_g_1, dita2_g_2, dita2_g_3];
163
164 % xc = xc1;
165 % yc = yc1;
166 %
167 % xc = xc2;
168 % yc = yc2;
169 %
170 % xc = xc3;
171 % yc = yc3;
172
173 xc = [xc1, xc2, xc3]; %concatenación para obtener la figura triangular graficando xc e yc.
174
175 yc = [yc1, yc2, yc3]; %concatenación para obtener la figura triangular graficando xc e yc.
176
177 theta1 = dita1_g;
178 theta2 = dita2_g;
179
180 end

```

```

181
182
183
184 %
185 % for i=1:3
186 %
187 %     switch i
188 %         case 1 %cálcula la primera recta decreciente
189 %             xp1 = [0:5:80] + desfasaje_O;
190 %             yp1 = -(xp1-desfasaje_O) + 300;
191 %
192 %             xc1 = xp1 - desfasaje_O;
193 %             yc1 = yp1;
194 %
195 %             c1_1= sqrt((xp1).^2+ yp1.^2);
196 %             c2_1= sqrt((xp1-d).^2+yp1.^2);
197 %             alfa1_1= acos((c2_1.^2-c1_1.^2-w^2)./(-2*c1_1*w));
198 %             alfa2_1= acos((c1_1.^2-c2_1.^2-w^2)./(-2*c2_1*w));
199 %             beta1_1= acos((b1.^2-a1.^2-c1_1.^2)./(-2*c1_1.*a1));
200 %             beta2_1= acos((b2.^2-a2.^2-c2_1.^2)./(-2*c2_1.*a2));
201 %
202 %             switch modo_tbj
203 %
204 %                 case 1
205 %                     dita1_1=alfa1_1+beta1_1;
206 %                     dita2_1= alfa2_1+beta2_1;
207 %                 case 2
208 %                     dita1_1=alfa1_1+beta1_1;
209 %                     dita2_1= alfa2_1-beta2_1;
210 %                 case 3
211 %                     dita1_1=alfa1_1-beta1_1;
212 %                     dita2_1= alfa2_1+beta2_1;
213 %                 case 4
214 %                     dita1_1=alfa1_1-beta1_1;
215 %                     dita2_1= alfa2_1-beta2_1;
216 %             end
217 %
218 %             dita1_g_1 = rad2deg(dita1_1);
219 %             dita2_g_1 = 180 - rad2deg(dita2_1);
220 %
221 %         case 2 %cálcula la trayectoria horizontal para formar la base del triángulo
222 %             xp2 = [80:-10:-80] + desfasaje_O;
223 %             yp1_21 = -(305-desfasaje_O) + 300; %yp1(21)
224 %             yp2 = ones(1,17)*yp1_21;
225 %
226 %             xc2 = xp2 - desfasaje_O;
227 %             yc2 = yp2;
228 %
229 %             c1_2= sqrt((xp2).^2+ yp2.^2);
230 %             c2_2= sqrt((xp2-d).^2+yp2.^2);
231 %             alfa1_2= acos((c2_2.^2-c1_2.^2-w^2)./(-2*c1_2*w));
232 %             alfa2_2= acos((c1_2.^2-c2_2.^2-w^2)./(-2*c2_2*w));
233 %             beta1_2= acos((b1.^2-a1.^2-c1_2.^2)./(-2*c1_2.*a1));
234 %             beta2_2= acos((b2.^2-a2.^2-c2_2.^2)./(-2*c2_2.*a2));
235 %
236 %             switch modo_tbj
237 %
238 %                 case 1
239 %                     dita1_2=alfa1_2+beta1_2;
240 %                     dita2_2= alfa2_2+beta2_2;

```

```

241 %           case 2
242 %               dita1_2=alfa1_2+beta1_2;
243 %               dita2_2= alfa2_2-beta2_2;
244 %           case 3
245 %               dita1_1=alfa1_1-beta1_1;
246 %               dita2_1= alfa2_1+beta2_1;
247 %           case 4
248 %               dita1_2=alfa1_2-beta1_2;
249 %               dita2_2= alfa2_2-beta2_2;
250 %       end
251 %
252 %       dita1_g_2 = rad2deg(dita1_2);
253 %       dita2_g_2 = 180 - rad2deg(dita2_2);
254 %
255 %       case 3 %calcula la trayectoria de la recta creciente para cerrar el triángulo
256 %       xp3 = [-80:5:0] + desfasaje_O;
257 %       yp3 =(xp3-desfasaje_O) + 300;
258 %
259 %       xc3 = xp3 - desfasaje_O;
260 %       yc3 = yp3;
261 %
262 %       c1_3= sqrt((xp3).^2+ yp3.^2);
263 %       c2_3= sqrt((xp3-d).^2+yp3.^2);
264 %       alfa1_3= acos((c2_3.^2-c1_3.^2-w^2)./(-2*c1_3*w));
265 %       alfa2_3= acos((c1_3.^2-c2_3.^2-w^2)./(-2*c2_3*w));
266 %       beta1_3= acos((b1.^2-a1.^2-c1_3.^2)./(-2*c1_3.*a1));
267 %       beta2_3= acos((b2.^2-a2.^2-c2_3.^2)./(-2*c2_3.*a2));
268 %
269 %       switch modo_tbj
270 %
271 %           case 1
272 %               dita1_3=alfa1_3+beta1_3;
273 %               dita2_3= alfa2_3+beta2_3;
274 %           case 2
275 %               dita1_3=alfa1_3+beta1_3;
276 %               dita2_3= alfa2_3-beta2_3;
277 %           case 3
278 %               dita1_3=alfa1_3-beta1_3;
279 %               dita2_3= alfa2_3+beta2_3;
280 %           case 4
281 %               dita1_3=alfa1_3-beta1_3;
282 %               dita2_3= alfa2_3-beta2_3;
283 %       end
284 %
285 %       dita1_g_3 = rad2deg(dita1_3);
286 %       dita2_g_3 = 180 - rad2deg(dita2_3);
287 %   end
288 % end
289 %
290 % %   dita1_g = dita1_g_1;
291 % %   dita2_g = dita2_g_1;
292 % %
293 % %   dita1_g = dita1_g_2;
294 % %   dita2_g = dita2_g_2;
295 % %
296 % %   dita1_g = dita1_g_3;
297 % %   dita2_g = dita2_g_3;
298 %
299 %
300 %

```

```
301 %   dita1_g = [dita1_g_1, dita1_g_2, dita1_g_3];
302 %   dita2_g = [dita2_g_1, dita2_g_2, dita2_g_3];
303 %
304 % %   xc = xc1;
305 % %   yc = yc1;
306 % %
307 % %   xc = xc2;
308 % %   yc = yc2;
309 % %
310 % %   xc = xc3;
311 % %   yc = yc3;
312 %
313 %   xc = [xc1, xc2, xc3]; %concatenación para obtener la figura triangular graficando xc e yc.
314 %
315 %   yc = [yc1, yc2, yc3]; %concatenación para obtener la figura triangular graficando xc e yc.
316 %
317 %   theta1 = dita1_g;
318 %   theta2 = dita2_g;
319
```

```
//=====CONTROLADOR IZQUIERDO=====

/*=====
      IPOS Source File
=====*/

#include <constb.h>
#include <iob.h>
//#pragma initials 0 127
#pragma globals 700 800
//#pragma var 700 800

//esto se mantiene igual respecto al programa inicial ya que unicamente se pasan y leen dita1,dita2
// y st2

MOVLNK WR_ST2;
MLDATA WR_ST2D;
MOVLNK WR_DITA2;
MLDATA WR_DITA2D;
MOVLNK RD_ST2;
MLDATA RD_ST2D;
MOVLNK ARRANQUE_2;
MLDATA ARRANQUE_2D;

//Note:
//Movidrive inverters with activated technology function "ISYNCH" or "Cam"
//do not support variables H360 an higher.

//DEFINICION DE VARIABLES
#define tdelay H11 //delay entre un punto y el siguiente [ms]
#define tcom H12 //delay de comunicaciones, para evitar que se pierdan señales [ms]
#define vhome H13 //vel al hacer homing [RPM]
#define vel H14 //vel al moverse [RPM]
#define lost2 H16 //estado inicial perdido (SERVO 2)
#define p1_dita1 H17 //
#define p1_dita2 H18 //
#define p2_dita1 H19 //
#define p2_dita2 H20 //
#define p3_dita1 H21 //
#define p3_dita2 H22 //
#define p4_dita1 H23 //
#define p4_dita2 H24 //
#define p5_dita1 H25 //
#define p5_dita2 H26 //
#define p6_dita1 H27 //
#define p6_dita2 H28 //
#define p7_dita1 H29 //
#define p7_dita2 H30 //
#define p8_dita1 H31 //
#define p8_dita2 H32 //
#define p9_dita1 H33 //
#define p9_dita2 H34 //
#define p10_dita1 H35 //
#define p10_dita2 H36 //
#define p11_dita1 H37 //
#define p11_dita2 H38 //
#define p12_dita1 H39 //
#define p12_dita2 H40 //
#define p13_dita1 H41 //
#define p13_dita2 H42 //
#define p14_dita1 H43 //
#define p14_dita2 H44 //
#define p15_dita1 H45 //
#define p15_dita2 H46 //
#define p16_dita1 H47 //
#define p16_dita2 H48 //
#define p17_dita1 H49 //
#define p17_dita2 H50 //
#define p18_dita1 H51 //
#define p18_dita2 H52 //
#define p19_dita1 H53 //
#define p19_dita2 H54 //
#define p20_dita1 H55 //
#define p20_dita2 H56 //

//ANGULOS DITA, ESTADOS y MODOS
#define finTRAY H128 // indica fin de trayectoria para salir del while
#define home H129// perdido( estaod inicial) lost=1, after homming lost=0 (SERVO 1)
#define dita1 H130 // valor de ángulo theta1 en pulsos
#define dita2 H131 // valor de ángulo theta2 en pulsos
#define st1 H132 // estado servo 1 ( moving=0; ready=1; finish=2; lost=3)
#define st2 H133 // estado servo 2 ( moving=0; ready=1; finish=2; lost=3)
#define stop H134 //parada
#define cpr H135 //numero de posicion puntos intermedios
#define TRAY H136 // permite seleccionar trayectoria
#define INI H137 //se usa para pasar del punto inicial de la trayectoria a los demas y comenzar
#define band_home H138 // se utiliza para hacer home con los switchs una sola vez
#define b_stop H139 // bandera para activar estado de espera previo al home luego de usar STOP
#define START H140 // habilita el posicionamiento al primer punto
#define seg H145 // indica que segmento se realizo en el logo

/*===== logo =====*/

logopoli()
{

    finTRAY =0;
    cpr=1;

    if (seg == 1) {
        if ( cpr == 1 && stop != 1 ) {
```

```

        dital = p1_dital;
        dita2 = p1_dita2;
        AngGiro1();
        cpr = cpr + 1;

    } // end if

// se establece un estado de espera en el punto inicial hasta que el usuario aprete un boton

while ( INI != 1 && stop != 1 && cpr != 1 ) {

} // end while de espera

if ( INI == 1 && stop != 1 && cpr != 1 ) {

    while ( cpr <= 20 && cpr >= 2 && stop != 1 && finTRAY != 1 ) {

        switch (cpr) {

            case 2:

                dital = p2_dital;
                dita2 = p2_dita2;
                break;
            case 3:

                dital = p3_dital;
                dita2 = p3_dita2;
                break;
            case 4:

                dital = p4_dital;
                dita2 = p4_dita2;
                break;
            case 5:

                dital = p5_dital;
                dita2 = p5_dita2;
                break;
            case 6:

                dital = p6_dital;
                dita2 = p6_dita2;
                break;
            case 7:

                dital = p7_dital;
                dita2 = p7_dita2;
                break;
            case 8:

                dital = p8_dital;
                dita2 = p8_dita2;
                break;
            case 9:

                dital = p9_dital;
                dita2 = p9_dita2;
                break;
            case 10:

                dital = p10_dital;
                dita2 = p10_dita2;
                break;
            case 11:

                dital = p11_dital;
                dita2 = p11_dita2;
                break;
            case 12:

                dital = p12_dital;
                dita2 = p12_dita2;
                break;
            case 13:

                dital = p13_dital;
                dita2 = p13_dita2;
                break;
            case 14:

                dital = p14_dital;
                dita2 = p14_dita2;
                break;
            case 15:

                dital = p15_dital;
                dita2 = p15_dita2;
                break;
            case 16:

                dital = p16_dital;
                dita2 = p16_dita2;
                break;
            case 17:

                dital = p17_dital;
                dita2 = p17_dita2;
                break;

```

```

        case 18:

            dital = p18_dital;
            dita2 = p18_dita2;
            break;
        case 19:

            dital = p19_dital;
            dita2 = p19_dita2;
            break;
        case 20:

            dital = p20_dital;
            dita2 = p20_dita2;
            finTRAY=1;
            break;

    } // end switch

    AngGiro1(); //invoco a la funcion angulo de giro para llevar los servos a posicion

    if (cpr < 20 && stop != 1) { // incrementa cpr hasta que llega a 20
        cpr = cpr + 1;
    } // end if

} // end while

INI =0;

} // end if
} //end if primer segmento logo

if (seg > 1 && seg <=10 ) {

if ( stop != 1 ) {

    while ( cpr <= 20 && cpr >= 1 && stop != 1 && finTRAY != 1 ) {

        switch (cpr) {
            case 1:

                dital = p2_dital;
                dita2 = p2_dita2;
                break;
            case 2:

                dital = p2_dital;
                dita2 = p2_dita2;
                break;
            case 3:

                dital = p3_dital;
                dita2 = p3_dita2;
                break;
            case 4:

                dital = p4_dital;
                dita2 = p4_dita2;
                break;
            case 5:

                dital = p5_dital;
                dita2 = p5_dita2;
                break;
            case 6:

                dital = p6_dital;
                dita2 = p6_dita2;
                break;
            case 7:

                dital = p7_dital;
                dita2 = p7_dita2;
                break;
            case 8:

                dital = p8_dital;
                dita2 = p8_dita2;
                break;
            case 9:

                dital = p9_dital;
                dita2 = p9_dita2;
                break;
            case 10:

                dital = p10_dital;
                dita2 = p10_dita2;
                break;
            case 11:

                dital = p11_dital;
                dita2 = p11_dita2;
                break;
            case 12:

                dital = p12_dital;
                dita2 = p12_dita2;

```

```

        break;
    case 13:

        dital = p13_dital;
        dita2 = p13_dita2;
        break;
    case 14:

        dital = p14_dital;
        dita2 = p14_dita2;
        break;
    case 15:

        dital = p15_dital;
        dita2 = p15_dita2;
        break;
    case 16:

        dital = p16_dital;
        dita2 = p16_dita2;
        break;
    case 17:

        dital = p17_dital;
        dita2 = p17_dita2;
        break;
    case 18:

        dital = p18_dital;
        dita2 = p18_dita2;
        break;
    case 19:

        dital = p19_dital;
        dita2 = p19_dita2;
        break;
    case 20:

        dital = p20_dital;
        dita2 = p20_dita2;
        finTRAY=1;
        break;

    } // end switch

    AngGiro1(); //invoco a la funcion angulo de giro para llevar los servos a posicion

    if (cpr < 20 && stop != 1) { // incrementa cpr hasta que llega a 20
        cpr = cpr + 1;
    } // end if

    } // end while

} // end if
} //end if


} // end funcion logopoli

/*===== TRAYECTORIA =====*/

trayectoria()
{

    //defino los valores de los angulos en PULSOS para llegar al punto deseado con un paso de 20

    // copiar y pegar desde MATLAB/OCTAVE
    finTRAY=0;
    cpr = 1; // contador de en que punto se encuentra el robot de todos los que fueron discretizados.

    // este if permite que el robot vaya desde el home al punto inicial de la recta
    if ( cpr == 1 && stop != 1 ) {

        dital = p1_dital;
        dita2 = p1_dita2;
        AngGiro1();
        cpr = cpr + 1;

    } // end if

    // se establece un estado de espera en el punto inicial hasta que el usuario aprete un boton

    while ( INI != 1 && stop != 1 && cpr != 1 ) {

    } // end while de espera

    if ( INI == 1 && stop != 1 && cpr != 1 ) {

        while ( cpr <= 20 && cpr >= 2 && stop != 1 && finTRAY != 1 ) {

            switch (cpr) {

                case 2:

                    dital = p2_dital;
                    dita2 = p2_dita2;
                    break;

```

```

        case 3:

            dital = p3_dital;
            dita2 = p3_dita2;
            break;
        case 4:

            dital = p4_dital;
            dita2 = p4_dita2;
            break;
        case 5:

            dital = p5_dital;
            dita2 = p5_dita2;
            break;
        case 6:

            dital = p6_dital;
            dita2 = p6_dita2;
            break;
        case 7:

            dital = p7_dital;
            dita2 = p7_dita2;
            break;
        case 8:

            dital = p8_dital;
            dita2 = p8_dita2;
            break;
        case 9:

            dital = p9_dital;
            dita2 = p9_dita2;
            break;
        case 10:

            dital = p10_dital;
            dita2 = p10_dita2;
            break;
        case 11:

            dital = p11_dital;
            dita2 = p11_dita2;
            break;
        case 12:

            dital = p12_dital;
            dita2 = p12_dita2;
            break;
        case 13:

            dital = p13_dital;
            dita2 = p13_dita2;
            break;
        case 14:

            dital = p14_dital;
            dita2 = p14_dita2;
            break;
        case 15:

            dital = p15_dital;
            dita2 = p15_dita2;
            break;
        case 16:

            dital = p16_dital;
            dita2 = p16_dita2;
            break;
        case 17:

            dital = p17_dital;
            dita2 = p17_dita2;
            break;
        case 18:

            dital = p18_dital;
            dita2 = p18_dita2;
            break;
        case 19:

            dital = p19_dital;
            dita2 = p19_dita2;
            break;
        case 20:

            dital = p20_dital;
            dita2 = p20_dita2;
            finTRAY=1;
            break;

    } // end switch

    AngGiro1(); //invoco a la funcion angulo de giro para llevar los servos a posicion

    if (cpr < 20 && stop != 1) { // incrementa cpr hasta que llega a 20
        cpr = cpr + 1;
    } // end if

```

```

    } // end while

    INI =0; //se reinicia INI

    } // end if
} // end trayectoria

/*===== ANGULO DE GIRO =====*/

AngGiro1()
{

// ESCRIBE SERVO 1 A SERVO 2 QUE SE MUEVA (st2=0)==> mooving
WR_DITA2D.WritePar=dita2;
_MoviLink(WR_DITA2);
st2=0;
WR_ST2D.WritePar=st2;
_MoviLink(WR_ST2);

//MOVIMIENTO SERVO 1
st1=0; // ==> moving
_GoAbs( GO_WAIT,dital );
st1=2;
//ESPERA A ST2 READY (ST2=2)
while( st2 !=2 ) {
    _MoviLink( RD_ST2 ); // actual command call
st2=RD_ST2D.ReadPar;
} //end while
st1=1;
st2=1;
WR_ST2D.WritePar=st2;
_MoviLink(WR_ST2);
} //end AngGiro1

/*===== HOMMING =====*/

Homming()
{
WR_ST2D.WritePar=st2; // ENTREGA A SERVO2 ESTADO INICIAL PERDID
_MoviLink(WR_ST2);
//MOVIMIENTO SERVO 1 A HOME
st1=0; // ==> moving
_GoAbs( GO_WAIT,dital );
_Go0( GO0_U_W_ZP );
st1=2;
//ESPERA A ST READY (ST2=2)
while( st2 !=2 ) {
    _MoviLink( RD_ST2 ); // actual command call
st2=RD_ST2D.ReadPar;
} //end while
st1=1;
st2=1;
WR_ST2D.WritePar=st2;
_MoviLink(WR_ST2);
_Wait(500);
home=0;
} //end homming

/*=====
Main Function (IPOS Entry Function)
=====*/
main()
{
//CONCIONES INICIALES
st1=3; //estado inicial perdido
st2=3; //estado inicial perdido
home=0;
band_home=0;
cpr=0;
seg=0;
stop=0;
START=0;
TRAY=0;
INI=0;
b_stop=0;
finTRAY=0;
_BitClear( ControlWord, 0 ); //inhibicion desactivada
_BitClear( ControlWord, 1 ); //inhibicion con freno desactivada
_BitClear( ControlWord, 3 ); // sentido CCW
_BitSet(ControlWord,2); // sentido CW habilitado

tdelay=500;
tcom=50;

//INICIALIZACION

WR_ST2.BusType = ML_BT_SBUS1; // RS-485 #2
WR_ST2.Address = 1;
RD_ST2.Format = ML_FT_PAR; // 2 PD with parameter
WR_ST2.Service = ML_S_WRV; // WRITE
WR_ST2.Index = 11133; // Index of var H18)
WR_ST2.DPointer =numof(WR_ST2D); // Target structure

WR_DITA2.BusType = ML_BT_SBUS1; // RS-485 #2
WR_DITA2.Address = 1;
WR_DITA2.Format = ML_FT_PAR; // 2 PD with parameter
WR_DITA2.Service = ML_S_WRV; // WRITE
WR_DITA2.Index = 11131; // Index of var H18)
WR_DITA2.DPointer =numof(WR_DITA2D); // Target structure

```

```

RD_ST2.BusType = ML_BT_SBUS1; // RS-485 #2
RD_ST2.Address = 1;
RD_ST2.Format = ML_FT_PAR; // 2 PD with parameter
RD_ST2.Service = ML_S_RD; // Read
RD_ST2.Index = 11133; // Index of VER H18
RD_ST2.DPointer = numof(RD_ST2D); // Target structure

ARRANQUE_2.BusType = ML_BT_SBUS1; // RS-485 #2
ARRANQUE_2.Address = 1;
ARRANQUE_2.Format = ML_FT_PAR; // 2 PD with parameter
ARRANQUE_2.Service = ML_S_WRV; // WRITE
ARRANQUE_2.Index = 8844; // Index of var H18)
ARRANQUE_2.DPointer = numof(ARRANQUE_2D); // Target structure

//PROGRAMA PRINCIPAL CICLICO
while(1)
{
//===== HOMMING =====

/*****
*****CUIDADO RPIMERO ENERGIZAR CONTROLADOR DERECHO*****
*****/

if (st1==0 && (home==1)) home=0; // este if no deja que se haga homming si los servos se mueven

if ((st1!=0)&& (st2!=0) &&(home==1) && band_home == 0 ) {

    st1=3; //estado inicial perdido
    st2=3; //estado inicial perdido
    Homming ();
    cpr=0;
    seg=0;
    TRAY=0;
    dital=0;
    dita2=0;
    H142=0;
    band_home=1;

} //end if    homming

while (b_stop == 1) { // estado de espera previo a hacer home luego de apretar STOP

    while (home = 0) {

    }

    b_stop=0;

} //end while

if ((st1!=0)&& (st2!=0) &&(home==1) && band_home == 1 ) {
    H142=124;
    dital=0;
    dita2=0;
    H150=0;
    H151=0;
    H152=0;
    H153=0;
    H154=0;
    H155=0;
    H156=0;
    H157=0;
    H158=0;
    H159=0;
    AngGiro1();
    home=0;
} //end homming 2

//===== MOVIMIENTO=====

while(TRAY == 0 && home!=1 && stop!=1 && START == 1) {
    H142=700; // variable para determinar si el programa avanza correctamente
}; //espera
H142=222; // variable para determinar si el programa avanza correctamente

// -----STOP-----
if(stop==1){
    START=0;
    TRAY=0;
    cpr=0;
    seg=0;
    INI=0;
    finTRAY=0;
    _Wait (500);
    stop=0;
    b_stop=1;
    H42=22;
} //end if stop    */
H142=8888;

```

```
// ===== TRAYECTORIA =====

if (START == 1) { // boton START da la orden para iniciar la trayectoria.

    _Wait(500); // espera 0,5seg para evitar cualquier problema al poner START en 0.
    START=0;

    switch (TRAY) {

        case 1:

            // recta horizontal l = 20mm

            p1_dita1 = -21078;
            p1_dita2 = 19803;

            p2_dita1 = -21012;
            p2_dita2 = 19872;

            p3_dita1 = -20946;
            p3_dita2 = 19940;

            p4_dita1 = -20880;
            p4_dita2 = 20008;

            p5_dita1 = -20814;
            p5_dita2 = 20076;

            p6_dita1 = -20747;
            p6_dita2 = 20144;

            p7_dita1 = -20681;
            p7_dita2 = 20212;

            p8_dita1 = -20614;
            p8_dita2 = 20279;

            p9_dita1 = -20548;
            p9_dita2 = 20346;

            p10_dita1 = -20481;
            p10_dita2 = 20414;

            p11_dita1 = -20414;
            p11_dita2 = 20481;

            p12_dita1 = -20346;
            p12_dita2 = 20548;

            p13_dita1 = -20279;
            p13_dita2 = 20614;

            p14_dita1 = -20212;
            p14_dita2 = 20681;

            p15_dita1 = -20144;
            p15_dita2 = 20747;

            p16_dita1 = -20076;
            p16_dita2 = 20814;

            p17_dita1 = -20008;
            p17_dita2 = 20880;

            p18_dita1 = -19940;
            p18_dita2 = 20946;

            p19_dita1 = -19872;
            p19_dita2 = 21012;

            p20_dita1 = -19803;
            p20_dita2 = 21078;

            trayectoria();
            TRAY=0; //se reinicia TRAY para seleccionar nuevamente una trayectoria

        break;

        case 2: // realiza recta horizontal l = 40mm

            p1_dita1 = -22184;
            p1_dita2 = 19656;

            p2_dita1 = -22054;
            p2_dita2 = 19793;

            p3_dita1 = -21924;
            p3_dita2 = 19929;

            p4_dita1 = -21794;
            p4_dita2 = 20065;

            p5_dita1 = -21663;
            p5_dita2 = 20201;

            p6_dita1 = -21532;
            p6_dita2 = 20336;
```

```

p7_dita1 = -21400;
p7_dita2 = 20470;

p8_dita1 = -21268;
p8_dita2 = 20604;

p9_dita1 = -21136;
p9_dita2 = 20738;

p10_dita1 = -21004;
p10_dita2 = 20871;

p11_dita1 = -20871;
p11_dita2 = 21004;

p12_dita1 = -20738;
p12_dita2 = 21136;

p13_dita1 = -20604;
p13_dita2 = 21268;

p14_dita1 = -20470;
p14_dita2 = 21400;

p15_dita1 = -20336;
p15_dita2 = 21532;

p16_dita1 = -20201;
p16_dita2 = 21663;

p17_dita1 = -20065;
p17_dita2 = 21794;

p18_dita1 = -19929;
p18_dita2 = 21924;

p19_dita1 = -19793;
p19_dita2 = 22054;

p20_dita1 = -19656;
p20_dita2 = 22184;

    trayectoria();
    TRAY=0;

    break;

    case 3: // realiza recta horizontal l = 60 mm

p1_dita1 = -23184;
p1_dita2 = 19386;

p2_dita1 = -22988;
p2_dita2 = 19592;

p3_dita1 = -22792;
p3_dita2 = 19797;

p4_dita1 = -22595;
p4_dita2 = 20001;

p5_dita1 = -22398;
p5_dita2 = 20204;

p6_dita1 = -22201;
p6_dita2 = 20406;

p7_dita1 = -22003;
p7_dita2 = 20608;

p8_dita1 = -21806;
p8_dita2 = 20809;

p9_dita1 = -21607;
p9_dita2 = 21009;

p10_dita1 = -21408;
p10_dita2 = 21209;

p11_dita1 = -21209;
p11_dita2 = 21408;

p12_dita1 = -21009;
p12_dita2 = 21607;

p13_dita1 = -20809;
p13_dita2 = 21806;

p14_dita1 = -20608;
p14_dita2 = 22003;

p15_dita1 = -20406;
p15_dita2 = 22201;

p16_dita1 = -20204;
p16_dita2 = 22398;

p17_dita1 = -20001;
p17_dita2 = 22595;

```

```

p18_dita1 = -19797;
p18_dita2 = 22792;

p19_dita1 = -19592;
p19_dita2 = 22988;

p20_dita1 = -19386;
p20_dita2 = 23184;

    trayectoria();
    TRAY=0;

    break;

    case 4: // realiza recta horizontal l = 80 mm

p1_dita1 = -24105;
p1_dita2 = 19002;

p2_dita1 = -23838;
p2_dita2 = 19277;

p3_dita1 = -23572;
p3_dita2 = 19551;

p4_dita1 = -23305;
p4_dita2 = 19823;

p5_dita1 = -23039;
p5_dita2 = 20094;

p6_dita1 = -22773;
p6_dita2 = 20365;

p7_dita1 = -22506;
p7_dita2 = 20634;

p8_dita1 = -22240;
p8_dita2 = 20903;

p9_dita1 = -21973;
p9_dita2 = 21171;

p10_dita1 = -21706;
p10_dita2 = 21439;

p11_dita1 = -21439;
p11_dita2 = 21706;

p12_dita1 = -21171;
p12_dita2 = 21973;

p13_dita1 = -20903;
p13_dita2 = 22240;

p14_dita1 = -20634;
p14_dita2 = 22506;

p15_dita1 = -20365;
p15_dita2 = 22773;

p16_dita1 = -20094;
p16_dita2 = 23039;

p17_dita1 = -19823;
p17_dita2 = 23305;

p18_dita1 = -19551;
p18_dita2 = 23572;

p19_dita1 = -19277;
p19_dita2 = 23838;

p20_dita1 = -19002;
p20_dita2 = 24105;

    trayectoria();
    TRAY=0;

    break;

    case 5: // realiza recta vertical l = 20 mm

p1_dita1 = -22909;
p1_dita2 = 17763;

p2_dita1 = -22938;
p2_dita2 = 17797;

p3_dita1 = -22967;
p3_dita2 = 17830;

p4_dita1 = -22995;
p4_dita2 = 17863;

```

```

p5_dita1 = -23023;
p5_dita2 = 17896;

p6_dita1 = -23050;
p6_dita2 = 17928;

p7_dita1 = -23078;
p7_dita2 = 17959;

p8_dita1 = -23105;
p8_dita2 = 17990;

p9_dita1 = -23131;
p9_dita2 = 18020;

p10_dita1 = -23158;
p10_dita2 = 18050;

p11_dita1 = -23184;
p11_dita2 = 18079;

p12_dita1 = -23210;
p12_dita2 = 18108;

p13_dita1 = -23235;
p13_dita2 = 18136;

p14_dita1 = -23261;
p14_dita2 = 18164;

p15_dita1 = -23286;
p15_dita2 = 18192;

p16_dita1 = -23310;
p16_dita2 = 18219;

p17_dita1 = -23335;
p17_dita2 = 18245;

p18_dita1 = -23359;
p18_dita2 = 18271;

p19_dita1 = -23383;
p19_dita2 = 18297;

p20_dita1 = -23406;
p20_dita2 = 18322;

```

```

    trayectoria();
    TRAY=0;

```

```

    break;

```

```

    case 6: // realiza recta vertical l = 40 mm

```

```

p1_dita1 = -21697;
p1_dita2 = 19144;

p2_dita1 = -21754;
p2_dita2 = 19205;

p3_dita1 = -21810;
p3_dita2 = 19264;

p4_dita1 = -21864;
p4_dita2 = 19322;

p5_dita1 = -21916;
p5_dita2 = 19377;

p6_dita1 = -21968;
p6_dita2 = 19432;

p7_dita1 = -22018;
p7_dita2 = 19484;

p8_dita1 = -22067;
p8_dita2 = 19535;

p9_dita1 = -22115;
p9_dita2 = 19585;

p10_dita1 = -22161;
p10_dita2 = 19633;

p11_dita1 = -22207;
p11_dita2 = 19679;

p12_dita1 = -22251;
p12_dita2 = 19724;

p13_dita1 = -22294;
p13_dita2 = 19768;

p14_dita1 = -22336;
p14_dita2 = 19810;

p15_dita1 = -22376;

```

```

p15_dita2 = 19851;

p16_dita1 = -22416;
p16_dita2 = 19890;

p17_dita1 = -22454;
p17_dita2 = 19928;

p18_dita1 = -22491;
p18_dita2 = 19965;

p19_dita1 = -22527;
p19_dita2 = 20000;

p20_dita1 = -22562;
p20_dita2 = 20035;

    trayectoria();
    TRAY=0;

    break;

    case 7: // realiza recta vertical l = 60 mm

p1_dita1 = -20447;
p1_dita2 = 20447;

p2_dita1 = -20533;
p2_dita2 = 20533;

p3_dita1 = -20616;
p3_dita2 = 20616;

p4_dita1 = -20695;
p4_dita2 = 20695;

p5_dita1 = -20772;
p5_dita2 = 20772;

p6_dita1 = -20845;
p6_dita2 = 20845;

p7_dita1 = -20915;
p7_dita2 = 20915;

p8_dita1 = -20982;
p8_dita2 = 20982;

p9_dita1 = -21046;
p9_dita2 = 21046;

p10_dita1 = -21108;
p10_dita2 = 21108;

p11_dita1 = -21166;
p11_dita2 = 21166;

p12_dita1 = -21222;
p12_dita2 = 21222;

p13_dita1 = -21275;
p13_dita2 = 21275;

p14_dita1 = -21325;
p14_dita2 = 21325;

p15_dita1 = -21373;
p15_dita2 = 21373;

p16_dita1 = -21418;
p16_dita2 = 21418;

p17_dita1 = -21461;
p17_dita2 = 21461;

p18_dita1 = -21501;
p18_dita2 = 21501;

p19_dita1 = -21538;
p19_dita2 = 21538;

p20_dita1 = -21573;
p20_dita2 = 21573;

    trayectoria();
    TRAY=0;

    break;

    case 8: // realiza recta vertical l = 80 mm

p1_dita1 = -19144;
p1_dita2 = 21697;

p2_dita1 = -19264;
p2_dita2 = 21810;

```

```

p3_dita1 = -19377;
p3_dita2 = 21916;

p4_dita1 = -19484;
p4_dita2 = 22018;

p5_dita1 = -19585;
p5_dita2 = 22115;

p6_dita1 = -19679;
p6_dita2 = 22207;

p7_dita1 = -19768;
p7_dita2 = 22294;

p8_dita1 = -19851;
p8_dita2 = 22376;

p9_dita1 = -19928;
p9_dita2 = 22454;

p10_dita1 = -20000;
p10_dita2 = 22527;

p11_dita1 = -20068;
p11_dita2 = 22596;

p12_dita1 = -20130;
p12_dita2 = 22660;

p13_dita1 = -20187;
p13_dita2 = 22720;

p14_dita1 = -20239;
p14_dita2 = 22776;

p15_dita1 = -20286;
p15_dita2 = 22827;

p16_dita1 = -20329;
p16_dita2 = 22874;

p17_dita1 = -20367;
p17_dita2 = 22916;

p18_dita1 = -20400;
p18_dita2 = 22954;

p19_dita1 = -20429;
p19_dita2 = 22988;

p20_dita1 = -20453;
p20_dita2 = 23017;

```

```

    trayectoria();
    TRAY=0;

```

```

    break;

```

```

    case 9: // realiza recta diagonal 1

```

```

p1_dita1 = -20447;
p1_dita2 = 20447;

p2_dita1 = -20428;
p2_dita2 = 20495;

p3_dita1 = -20409;
p3_dita2 = 20543;

p4_dita1 = -20390;
p4_dita2 = 20591;

p5_dita1 = -20371;
p5_dita2 = 20638;

p6_dita1 = -20351;
p6_dita2 = 20686;

p7_dita1 = -20332;
p7_dita2 = 20733;

p8_dita1 = -20312;
p8_dita2 = 20780;

p9_dita1 = -20293;
p9_dita2 = 20827;

p10_dita1 = -20273;
p10_dita2 = 20874;

p11_dita1 = -20253;
p11_dita2 = 20921;

p12_dita1 = -20233;
p12_dita2 = 20968;

```

```

p13_dita1 = -20213;
p13_dita2 = 21014;

p14_dita1 = -20193;
p14_dita2 = 21061;

p15_dita1 = -20173;
p15_dita2 = 21107;

p16_dita1 = -20152;
p16_dita2 = 21153;

p17_dita1 = -20132;
p17_dita2 = 21199;

p18_dita1 = -20111;
p18_dita2 = 21245;

p19_dita1 = -20090;
p19_dita2 = 21291;

p20_dita1 = -20070;
p20_dita2 = 21336;

    trayectoria();
    TRAY=0;

    break;

    case 10: // realiza recta diagonal 2

p1_dita1 = -20447;
p1_dita2 = 20447;

p2_dita1 = -20400;
p2_dita2 = 20567;

p3_dita1 = -20351;
p3_dita2 = 20686;

p4_dita1 = -20303;
p4_dita2 = 20804;

p5_dita1 = -20253;
p5_dita2 = 20921;

p6_dita1 = -20203;
p6_dita2 = 21037;

p7_dita1 = -20152;
p7_dita2 = 21153;

p8_dita1 = -20101;
p8_dita2 = 21268;

p9_dita1 = -20049;
p9_dita2 = 21382;

p10_dita1 = -19996;
p10_dita2 = 21495;

p11_dita1 = -19943;
p11_dita2 = 21608;

p12_dita1 = -19889;
p12_dita2 = 21720;

p13_dita1 = -19834;
p13_dita2 = 21831;

p14_dita1 = -19779;
p14_dita2 = 21942;

p15_dita1 = -19724;
p15_dita2 = 22052;

p16_dita1 = -19667;
p16_dita2 = 22162;

p17_dita1 = -19610;
p17_dita2 = 22272;

p18_dita1 = -19553;
p18_dita2 = 22380;

p19_dita1 = -19495;
p19_dita2 = 22489;

p20_dita1 = -19436;
p20_dita2 = 22597;

```

```

    trayectoria();
    TRAY=0;

    break;

```

```
case 11: // realiza recta diagonal 3
```

```
p1_dita1 = -20447;
p1_dita2 = 20447;

p2_dita1 = -20351;
p2_dita2 = 20686;

p3_dita1 = -20253;
p3_dita2 = 20921;

p4_dita1 = -20152;
p4_dita2 = 21153;

p5_dita1 = -20049;
p5_dita2 = 21382;

p6_dita1 = -19943;
p6_dita2 = 21608;

p7_dita1 = -19834;
p7_dita2 = 21831;

p8_dita1 = -19724;
p8_dita2 = 22052;

p9_dita1 = -19610;
p9_dita2 = 22272;

p10_dita1 = -19495;
p10_dita2 = 22489;

p11_dita1 = -19377;
p11_dita2 = 22704;

p12_dita1 = -19256;
p12_dita2 = 22918;

p13_dita1 = -19134;
p13_dita2 = 23131;

p14_dita1 = -19009;
p14_dita2 = 23342;

p15_dita1 = -18882;
p15_dita2 = 23552;

p16_dita1 = -18752;
p16_dita2 = 23761;

p17_dita1 = -18621;
p17_dita2 = 23970;

p18_dita1 = -18487;
p18_dita2 = 24178;

p19_dita1 = -18351;
p19_dita2 = 24385;

p20_dita1 = -18213;
p20_dita2 = 24591;
```

```
trayectoria();
TRAY=0;
```

```
break;
```

```
case 12: // realiza recta diagonal 4
```

```
p1_dita1 = -20447;
p1_dita2 = 20447;

p2_dita1 = -20293;
p2_dita2 = 20827;

p3_dita1 = -20132;
p3_dita2 = 21199;

p4_dita1 = -19964;
p4_dita2 = 21563;

p5_dita1 = -19790;
p5_dita2 = 21920;

p6_dita1 = -19610;
p6_dita2 = 22272;

p7_dita1 = -19424;
p7_dita2 = 22618;

p8_dita1 = -19232;
p8_dita2 = 22961;

p9_dita1 = -19034;
p9_dita2 = 23300;

p10_dita1 = -18830;
p10_dita2 = 23636;
```

```

p11_dita1 = -18621;
p11_dita2 = 23970;

p12_dita1 = -18406;
p12_dita2 = 24302;

p13_dita1 = -18185;
p13_dita2 = 24633;

p14_dita1 = -17960;
p14_dita2 = 24962;

p15_dita1 = -17728;
p15_dita2 = 25292;

p16_dita1 = -17492;
p16_dita2 = 25621;

p17_dita1 = -17251;
p17_dita2 = 25952;

p18_dita1 = -17005;
p18_dita2 = 26283;

p19_dita1 = -16753;
p19_dita2 = 26615;

p20_dita1 = -16497;
p20_dita2 = 26950;

```

```

    trayectoria();
    TRAY=0;

```

```

    break;

```

```

    case 13: // realiza cuadrado l = 20mm

```

```

p1_dita1 = -21078;
p1_dita2 = 19803;

p2_dita1 = -20764;
p2_dita2 = 20127;

p3_dita1 = -20447;
p3_dita2 = 20447;

p4_dita1 = -20127;
p4_dita2 = 20764;

p5_dita1 = -19803;
p5_dita2 = 21078;

p6_dita1 = -19803;
p6_dita2 = 21078;

p7_dita1 = -19941;
p7_dita2 = 21211;

p8_dita1 = -20070;
p8_dita2 = 21336;

p9_dita1 = -20190;
p9_dita2 = 21454;

p10_dita1 = -20302;
p10_dita2 = 21564;

p11_dita1 = -20302;
p11_dita2 = 21564;

p12_dita1 = -20621;
p12_dita2 = 21252;

p13_dita1 = -20938;
p13_dita2 = 20938;

p14_dita1 = -21252;
p14_dita2 = 20621;

p15_dita1 = -21564;
p15_dita2 = 20302;

p16_dita1 = -21564;
p16_dita2 = 20302;

p17_dita1 = -21454;
p17_dita2 = 20190;

p18_dita1 = -21336;
p18_dita2 = 20070;

p19_dita1 = -21211;
p19_dita2 = 19941;

p20_dita1 = -21078;
p20_dita2 = 19803;

```

```

        trayectoria();
        TRAY=0;

        break;

        case 14: // realiza cuadrado l = 40mm

//p1_dita1 = -21697;
//p1_dita2 = 19144;

//p2_dita1 = -21078;
//p2_dita2 = 19803;

//p3_dita1 = -20447;
//p3_dita2 = 20447;

//p4_dita1 = -19803;
//p4_dita2 = 21078;

//p5_dita1 = -19144;
//p5_dita2 = 21697;

//p6_dita1 = -19144;
//p6_dita2 = 21697;
//
//p7_dita1 = -19418;
//p7_dita2 = 21955;

//p8_dita1 = -19656;
//p8_dita2 = 22184;

//p9_dita1 = -19861;
//p9_dita2 = 22386;

//p10_dita1 = -20035;
//p10_dita2 = 22562;

//p11_dita1 = -20035;
//p11_dita2 = 22562;

//p12_dita1 = -20675;
//p12_dita2 = 21938;

//p13_dita1 = -21309;
//p13_dita2 = 21309;

//p14_dita1 = -21938;
//p14_dita2 = 20675;

//p15_dita1 = -22562;
//p15_dita2 = 20035;

//p16_dita1 = -22562;
//p16_dita2 = 20035;

//p17_dita1 = -22386;
//p17_dita2 = 19861;

//p18_dita1 = -22184;
//p18_dita2 = 19656;

//p19_dita1 = -21955;
//p19_dita2 = 19418;

//p20_dita1 = -21697;
//p20_dita2 = 19144;
//
p1_dita1 = -22231;
p1_dita2 = 19719;

p2_dita1 = -21619;
p2_dita2 = 20364;

p3_dita1 = -20997;
p3_dita2 = 20997;

p4_dita1 = -20364;
p4_dita2 = 21619;

p5_dita1 = -19719;
p5_dita2 = 22231;

p6_dita1 = -19719;
p6_dita2 = 22231;

p7_dita1 = -19976;
p7_dita2 = 22481;

p8_dita1 = -20201;
p8_dita2 = 22705;

p9_dita1 = -20396;
p9_dita2 = 22903;

p10_dita1 = -20562;
p10_dita2 = 23077;

p11_dita1 = -20562;

```

```

p11_dita2 = 23077;

p12_dita1 = -21198;
p12_dita2 = 22454;

p13_dita1 = -21828;
p13_dita2 = 21828;

p14_dita1 = -22454;
p14_dita2 = 21198;

p15_dita1 = -23077;
p15_dita2 = 20562;

p16_dita1 = -23077;
p16_dita2 = 20562;

p17_dita1 = -22903;
p17_dita2 = 20396;

p18_dita1 = -22705;
p18_dita2 = 20201;

p19_dita1 = -22481;
p19_dita2 = 19976;

p20_dita1 = -22231;
p20_dita2 = 19719;

    trayectoria();
    TRAY=0;

    break;

    case 15: // realiza cuadrado 1 = 80mm

p1_dita1 = -22909;
p1_dita2 = 17763;

p2_dita1 = -21697;
p2_dita2 = 19144;

p3_dita1 = -20447;
p3_dita2 = 20447;

p4_dita1 = -19144;
p4_dita2 = 21697;

p5_dita1 = -17763;
p5_dita2 = 22909;

p6_dita1 = -17763;
p6_dita2 = 22909;

p7_dita1 = -18322;
p7_dita2 = 23406;

p8_dita1 = -18726;
p8_dita2 = 23803;

p9_dita1 = -19002;
p9_dita2 = 24105;

p10_dita1 = -19164;
p10_dita2 = 24312;

p11_dita1 = -19164;
p11_dita2 = 24312;

p12_dita1 = -20453;
p12_dita2 = 23017;

p13_dita1 = -21734;
p13_dita2 = 21734;

p14_dita1 = -23017;
p14_dita2 = 20453;

p15_dita1 = -24312;
p15_dita2 = 19164;

p16_dita1 = -24312;
p16_dita2 = 19164;

p17_dita1 = -24105;
p17_dita2 = 19002;

p18_dita1 = -23803;
p18_dita2 = 18726;

p19_dita1 = -23406;
p19_dita2 = 18322;

p20_dita1 = -22909;
p20_dita2 = 17763;

```

```

    trayectoria();

```

```

        TRAY=0;

        break;

        case 16: // realiza cuadrado l = 100mm

p1_dita1 = -23504;
p1_dita2 = 17031;

p2_dita1 = -22004;
p2_dita2 = 18807;

p3_dita1 = -20447;
p3_dita2 = 20447;

p4_dita1 = -18807;
p4_dita2 = 22004;

p5_dita1 = -17031;
p5_dita2 = 23504;

p6_dita1 = -17031;
p6_dita2 = 23504;

p7_dita1 = -17748;
p7_dita2 = 24122;

p8_dita1 = -18213;
p8_dita2 = 24591;

p9_dita1 = -18480;
p9_dita2 = 24917;

p10_dita1 = -18572;
p10_dita2 = 25093;

p11_dita1 = -18572;
p11_dita2 = 25093;

p12_dita1 = -20183;
p12_dita2 = 23424;

p13_dita1 = -21793;
p13_dita2 = 21793;

p14_dita1 = -23424;
p14_dita2 = 20183;

p15_dita1 = -25093;
p15_dita2 = 18572;

p16_dita1 = -25093;
p16_dita2 = 18572;

p17_dita1 = -24917;
p17_dita2 = 18480;

p18_dita1 = -24591;
p18_dita2 = 18213;

p19_dita1 = -24122;
p19_dita2 = 17748;

p20_dita1 = -23504;
p20_dita2 = 17031;

        trayectoria();
        TRAY=0;

        break;

        case 17: // realiza triangulo b = 20mm

p1_dita1 = -20447;
p1_dita2 = 20447;

p2_dita1 = -20325;
p2_dita2 = 20749;

p3_dita1 = -20200;
p3_dita2 = 21045;

p4_dita1 = -20070;
p4_dita2 = 21336;

p5_dita1 = -19936;
p5_dita2 = 21623;

p6_dita1 = -19798;
p6_dita2 = 21905;

p7_dita1 = -19656;
p7_dita2 = 22184;

p8_dita1 = -19656;
p8_dita2 = 22184;

p9_dita1 = -20088;

```

```

p9_dita2 = 21772;

p10_dita1 = -20515;
p10_dita2 = 21356;

p11_dita1 = -20938;
p11_dita2 = 20938;

p12_dita1 = -21356;
p12_dita2 = 20515;

p13_dita1 = -21772;
p13_dita2 = 20088;

p14_dita1 = -22184;
p14_dita2 = 19656;

p15_dita1 = -21905;
p15_dita2 = 19798;

p16_dita1 = -21623;
p16_dita2 = 19936;

p17_dita1 = -21336;
p17_dita2 = 20070;

p18_dita1 = -21045;
p18_dita2 = 20200;

p19_dita1 = -20749;
p19_dita2 = 20325;

p20_dita1 = -20447;
p20_dita2 = 20447;

    trayectoria();
    TRAY=0;

    break;

    case 18: // realiza triangulo b = 40mm

p1_dita1 = -20447;
p1_dita2 = 20447;

p2_dita1 = -20200;
p2_dita2 = 21045;

p3_dita1 = -19936;
p3_dita2 = 21623;

p4_dita1 = -19656;
p4_dita2 = 22184;

p5_dita1 = -19361;
p5_dita2 = 22733;

p6_dita1 = -19051;
p6_dita2 = 23272;

p7_dita1 = -18726;
p7_dita2 = 23803;

p8_dita1 = -18726;
p8_dita2 = 23803;

p9_dita1 = -19603;
p9_dita2 = 22977;

p10_dita1 = -20462;
p10_dita2 = 22146;

p11_dita1 = -21309;
p11_dita2 = 21309;

p12_dita1 = -22146;
p12_dita2 = 20462;

p13_dita1 = -22977;
p13_dita2 = 19603;

p14_dita1 = -23803;
p14_dita2 = 18726;

p15_dita1 = -23272;
p15_dita2 = 19051;

p16_dita1 = -22733;
p16_dita2 = 19361;

p17_dita1 = -22184;
p17_dita2 = 19656;

p18_dita1 = -21623;
p18_dita2 = 19936;

p19_dita1 = -21045;
p19_dita2 = 20200;

```

```

p20_dita1 = -20447;
p20_dita2 = 20447;

    trayectoria();
    TRAY=0;

    break;

    case 19: // realiza triangulo b = 60mm

```

```

p1_dita1 = -20447;
p1_dita2 = 20447;

p2_dita1 = -20070;
p2_dita2 = 21336;

p3_dita1 = -19656;
p3_dita2 = 22184;

p4_dita1 = -19208;
p4_dita2 = 23003;

p5_dita1 = -18726;
p5_dita2 = 23803;

p6_dita1 = -18213;
p6_dita2 = 24591;

p7_dita1 = -17670;
p7_dita2 = 25374;

p8_dita1 = -17670;
p8_dita2 = 25374;

p9_dita1 = -19002;
p9_dita2 = 24105;

p10_dita1 = -20297;
p10_dita2 = 22839;

p11_dita1 = -21573;
p11_dita2 = 21573;

p12_dita1 = -22839;
p12_dita2 = 20297;

p13_dita1 = -24105;
p13_dita2 = 19002;

p14_dita1 = -25374;
p14_dita2 = 17670;

p15_dita1 = -24591;
p15_dita2 = 18213;

p16_dita1 = -23803;
p16_dita2 = 18726;

p17_dita1 = -23003;
p17_dita2 = 19208;

p18_dita1 = -22184;
p18_dita2 = 19656;

p19_dita1 = -21336;
p19_dita2 = 20070;

p20_dita1 = -20447;
p20_dita2 = 20447;

```

```

    trayectoria();
    TRAY=0;

    break;

```

```

    case 20: // realiza triangulo b = 80mm
p1_dita1 = -20447;
p1_dita2 = 20447;

p2_dita1 = -19936;
p2_dita2 = 21623;

p3_dita1 = -19361;
p3_dita2 = 22733;

p4_dita1 = -18726;
p4_dita2 = 23803;

p5_dita1 = -18035;
p5_dita2 = 24853;

p6_dita1 = -17291;
p6_dita2 = 25897;

p7_dita1 = -16497;

```

```

p7_dita2 = 26950;

p8_dita1 = -16497;
p8_dita2 = 26950;

p9_dita1 = -18293;
p9_dita2 = 25183;

p10_dita1 = -20024;
p10_dita2 = 23447;

p11_dita1 = -21734;
p11_dita2 = 21734;

p12_dita1 = -23447;
p12_dita2 = 20024;

p13_dita1 = -25183;
p13_dita2 = 18293;

p14_dita1 = -26950;
p14_dita2 = 16497;

p15_dita1 = -25897;
p15_dita2 = 17291;

p16_dita1 = -24853;
p16_dita2 = 18035;

p17_dita1 = -23803;
p17_dita2 = 18726;

p18_dita1 = -22733;
p18_dita2 = 19361;

p19_dita1 = -21623;
p19_dita2 = 19936;

p20_dita1 = -20447;
p20_dita2 = 20447;

    trayectoria();
    TRAY=0;

    break;

    case 21: // realiza circ r = 10mm
p1_dita1 = -19152;
p1_dita2 = 20459;

p2_dita1 = -19306;
p2_dita2 = 20535;

p3_dita1 = -19510;
p3_dita2 = 20531;

p4_dita1 = -19744;
p4_dita2 = 20449;

p5_dita1 = -19983;
p5_dita2 = 20299;

p6_dita1 = -20203;
p6_dita2 = 20097;

p7_dita1 = -20382;
p7_dita2 = 19864;

p8_dita1 = -20500;
p8_dita2 = 19625;

p9_dita1 = -20543;
p9_dita2 = 19403;

p10_dita1 = -20507;
p10_dita2 = 19221;

p11_dita1 = -20392;
p11_dita2 = 19099;

p12_dita1 = -20211;
p12_dita2 = 19051;

p13_dita1 = -19983;
p13_dita2 = 19084;

p14_dita1 = -19733;
p14_dita2 = 19198;

p15_dita1 = -19490;
p15_dita2 = 19379;

p16_dita1 = -19281;
p16_dita2 = 19609;

p17_dita1 = -19132;
p17_dita2 = 19859;

```

```

p18_dita1 = -19057;
p18_dita2 = 20102;

p19_dita1 = -19065;
p19_dita2 = 20309;

p20_dita1 = -19152;
p20_dita2 = 20459;

    trayectoria();
    TRAY=0;

    break;

    case 22: // realiza circ r = 20mm
p1_dita1 = -18828;
p1_dita2 = 21409;

p2_dita1 = -19111;
p2_dita2 = 21533;

p3_dita1 = -19489;
p3_dita2 = 21499;

p4_dita1 = -19926;
p4_dita2 = 21314;

p5_dita1 = -20381;
p5_dita2 = 21003;

p6_dita1 = -20811;
p6_dita2 = 20602;

p7_dita1 = -21172;
p7_dita2 = 20154;

p8_dita1 = -21424;
p8_dita2 = 19703;

p9_dita1 = -21536;
p9_dita2 = 19290;

p10_dita1 = -21491;
p10_dita2 = 18956;

p11_dita1 = -21290;
p11_dita2 = 18733;

p12_dita1 = -20953;
p12_dita2 = 18651;

p13_dita1 = -20516;
p13_dita2 = 18727;

p14_dita1 = -20028;
p14_dita2 = 18959;

p15_dita1 = -19546;
p15_dita2 = 19325;

p16_dita1 = -19128;
p16_dita2 = 19782;

p17_dita1 = -18824;
p17_dita2 = 20275;

p18_dita1 = -18669;
p18_dita2 = 20744;

p19_dita1 = -18673;
p19_dita2 = 21136;

p20_dita1 = -18828;
p20_dita2 = 21409;

    trayectoria();
    TRAY=0;

    break;

    case 23: // realiza circ r = 30mm
p1_dita1 = -18465;
p1_dita2 = 22308;

p2_dita1 = -18854;
p2_dita2 = 22461;

p3_dita1 = -19380;
p3_dita2 = 22376;

p4_dita1 = -19997;
p4_dita2 = 22069;

p5_dita1 = -20654;
p5_dita2 = 21583;

```

```

p6_dita1 = -21291;
p6_dita2 = 20979;

p7_dita1 = -21844;
p7_dita2 = 20324;

p8_dita1 = -22248;
p8_dita2 = 19680;

p9_dita1 = -22448;
p9_dita2 = 19103;

p10_dita1 = -22414;
p10_dita2 = 18640;

p11_dita1 = -22144;
p11_dita2 = 18336;

p12_dita1 = -21667;
p12_dita2 = 18234;

p13_dita1 = -21034;
p13_dita2 = 18360;

p14_dita1 = -20318;
p14_dita2 = 18717;

p15_dita1 = -19602;
p15_dita2 = 19271;

p16_dita1 = -18973;
p16_dita2 = 19955;

p17_dita1 = -18511;
p17_dita2 = 20682;

p18_dita1 = -18267;
p18_dita2 = 21366;

p19_dita1 = -18258;
p19_dita2 = 21929;

p20_dita1 = -18465;
p20_dita2 = 22308;

    trayectoria();
    TRAY=0;

    break;

    case 24: // realiza circ r = 50mm
p1_dita1 = -17628;
p1_dita2 = 24010;

p2_dita1 = -18169;
p2_dita2 = 24184;

p3_dita1 = -18924;
p3_dita2 = 23942;

p4_dita1 = -19843;
p4_dita2 = 23326;

p5_dita1 = -20870;
p5_dita2 = 22435;

p6_dita1 = -21928;
p6_dita2 = 21401;

p7_dita1 = -22906;
p7_dita2 = 20348;

p8_dita1 = -23676;
p8_dita2 = 19367;

p9_dita1 = -24114;
p9_dita2 = 18523;

p10_dita1 = -24149;
p10_dita2 = 17868;

p11_dita1 = -23773;
p11_dita2 = 17455;

p12_dita1 = -23039;
p12_dita2 = 17345;

p13_dita1 = -22039;
p13_dita2 = 17596;

p14_dita1 = -20887;
p14_dita2 = 18222;

p15_dita1 = -19714;
p15_dita2 = 19163;

p16_dita1 = -18660;
p16_dita2 = 20295;

```

```

p17_dita1 = -17864;
p17_dita2 = 21475;

p18_dita1 = -17423;
p18_dita2 = 22566;

p19_dita1 = -17358;
p19_dita2 = 23446;

p20_dita1 = -17628;
p20_dita2 = 24010;

    trayectoria();
    TRAY=0;

    break;

    case 25: // realiza zig zag traslado
p1_dita1 = -20447;
p1_dita2 = 20447;

p2_dita1 = -20937;
p2_dita2 = 19656;

p3_dita1 = -21409;
p3_dita2 = 18828;

p4_dita1 = -21864;
p4_dita2 = 17948;

p5_dita1 = -22301;
p5_dita2 = 16993;

p6_dita1 = -22301;
p6_dita2 = 16993;

p7_dita1 = -21086;
p7_dita2 = 18465;

p8_dita1 = -19815;
p8_dita2 = 19815;

p9_dita1 = -18465;
p9_dita2 = 21086;

p10_dita1 = -16993;
p10_dita2 = 22301;

p11_dita1 = -16993;
p11_dita2 = 22301;

p12_dita1 = -18911;
p12_dita2 = 21484;

p13_dita1 = -20582;
p13_dita2 = 20582;

p14_dita1 = -22130;
p14_dita2 = 19600;

p15_dita1 = -23617;
p15_dita2 = 18542;

p16_dita1 = -23617;
p16_dita2 = 18542;

p17_dita1 = -22386;
p17_dita2 = 19861;

p18_dita1 = -21137;
p18_dita2 = 21137;

p19_dita1 = -19861;
p19_dita2 = 22386;

p20_dita1 = -18542;
p20_dita2 = 23617;

    trayectoria();
    TRAY=0;

    break;

case 26 :

    seg = 1;
    while (seg >= 1 && seg <=10 && stop != 1) {
        if (seg == 1) {
            p1_dita1 = -20302;
            p1_dita2 = 21564;

            p2_dita1 = -20437;
            p2_dita2 = 21433;

```

```

        p3_dita1 = -20571;
        p3_dita2 = 21301;

        p4_dita1 = -20704;
        p4_dita2 = 21169;

        p5_dita1 = -20838;
        p5_dita2 = 21037;

        p6_dita1 = -20971;
        p6_dita2 = 20904;

        p7_dita1 = -21103;
        p7_dita2 = 20771;

        p8_dita1 = -21235;
        p8_dita2 = 20638;

        p9_dita1 = -21367;
        p9_dita2 = 20504;

        p10_dita1 = -21499;
        p10_dita2 = 20369;

        p11_dita1 = -21630;
        p11_dita2 = 20235;

        p12_dita1 = -21761;
        p12_dita2 = 20099;

        p13_dita1 = -21891;
        p13_dita2 = 19963;

        p14_dita1 = -22022;
        p14_dita2 = 19827;

        p15_dita1 = -22152;
        p15_dita2 = 19690;

        p16_dita1 = -22281;
        p16_dita2 = 19553;

        p17_dita1 = -22411;
        p17_dita2 = 19415;

        p18_dita1 = -22540;
        p18_dita2 = 19276;

        p19_dita1 = -22669;
        p19_dita2 = 19137;

        p20_dita1 = -22798;
        p20_dita2 = 18997;

        logopoli();
        H150 = 1;
        seg = seg +1;
    } // end if

    if (seg == 2) {
p1_dita1 = -22798;
p1_dita2 = 18997;

p2_dita1 = -22775;
p2_dita2 = 18973;

p3_dita1 = -22751;
p3_dita2 = 18948;

p4_dita1 = -22728;
p4_dita2 = 18923;

p5_dita1 = -22704;
p5_dita2 = 18898;

p6_dita1 = -22679;
p6_dita2 = 18872;

p7_dita1 = -22655;
p7_dita2 = 18846;

p8_dita1 = -22630;
p8_dita2 = 18819;

p9_dita1 = -22605;
p9_dita2 = 18792;

p10_dita1 = -22579;
p10_dita2 = 18765;

p11_dita1 = -22553;
p11_dita2 = 18737;

p12_dita1 = -22527;
p12_dita2 = 18709;

p13_dita1 = -22501;
p13_dita2 = 18680;

p14_dita1 = -22474;

```

```

p14_dita2 = 18651;

p15_dita1 = -22447;
p15_dita2 = 18621;

p16_dita1 = -22420;
p16_dita2 = 18591;

p17_dita1 = -22392;
p17_dita2 = 18560;

p18_dita1 = -22364;
p18_dita2 = 18529;

p19_dita1 = -22336;
p19_dita2 = 18497;

p20_dita1 = -22308;
p20_dita2 = 18465;

        logopoli();
        H151 = 1;
        seg = seg +1;
    } // end if
    if (seg == 3) {
        p1_dita1 = -22308;
p1_dita2 = 18465;

p2_dita1 = -22116;
p2_dita2 = 18682;

p3_dita1 = -21923;
p3_dita2 = 18896;

p4_dita1 = -21730;
p4_dita2 = 19109;

p5_dita1 = -21535;
p5_dita2 = 19319;

p6_dita1 = -21340;
p6_dita2 = 19528;

p7_dita1 = -21144;
p7_dita2 = 19735;

p8_dita1 = -20946;
p8_dita2 = 19940;

p9_dita1 = -20747;
p9_dita2 = 20144;

p10_dita1 = -20548;
p10_dita2 = 20346;

p11_dita1 = -20346;
p11_dita2 = 20548;

p12_dita1 = -20144;
p12_dita2 = 20747;

p13_dita1 = -19940;
p13_dita2 = 20946;

p14_dita1 = -19735;
p14_dita2 = 21144;

p15_dita1 = -19528;
p15_dita2 = 21340;

p16_dita1 = -19319;
p16_dita2 = 21535;

p17_dita1 = -19109;
p17_dita2 = 21730;

p18_dita1 = -18896;
p18_dita2 = 21923;

p19_dita1 = -18682;
p19_dita2 = 22116;

p20_dita1 = -18465;
p20_dita2 = 22308;
        logopoli();
        H152 = 1;
        seg = seg +1;
    } // end if
    if (seg == 4) {
        p1_dita1 = -18465;
p1_dita2 = 22308;

p2_dita1 = -18560;
p2_dita2 = 22392;

p3_dita1 = -18651;
p3_dita2 = 22474;

p4_dita1 = -18737;
p4_dita2 = 22553;

```

```

p5_dita1 = -18819;
p5_dita2 = 22630;

p6_dita1 = -18898;
p6_dita2 = 22704;

p7_dita1 = -18973;
p7_dita2 = 22775;

p8_dita1 = -19044;
p8_dita2 = 22843;

p9_dita1 = -19112;
p9_dita2 = 22909;

p10_dita1 = -19177;
p10_dita2 = 22973;

p11_dita1 = -19238;
p11_dita2 = 23034;

p12_dita1 = -19296;
p12_dita2 = 23092;

p13_dita1 = -19351;
p13_dita2 = 23148;

p14_dita1 = -19403;
p14_dita2 = 23201;

p15_dita1 = -19452;
p15_dita2 = 23253;

p16_dita1 = -19498;
p16_dita2 = 23301;

p17_dita1 = -19541;
p17_dita2 = 23347;

p18_dita1 = -19581;
p18_dita2 = 23391;

p19_dita1 = -19618;
p19_dita2 = 23433;

p20_dita1 = -19653;
p20_dita2 = 23472;
    logopoli();
    H153 = 1;
    seg = seg +1;
} // end if
if (seg == 5) {
    p1_dita1 = -19653;
p1_dita2 = 23472;

p2_dita1 = -19857;
p2_dita2 = 23272;

p3_dita1 = -20061;
p3_dita2 = 23072;

p4_dita1 = -20263;
p4_dita2 = 22873;

p5_dita1 = -20466;
p5_dita2 = 22673;

p6_dita1 = -20668;
p6_dita2 = 22473;

p7_dita1 = -20870;
p7_dita2 = 22273;

p8_dita1 = -21071;
p8_dita2 = 22073;

p9_dita1 = -21272;
p9_dita2 = 21873;

p10_dita1 = -21472;
p10_dita2 = 21673;

p11_dita1 = -21673;
p11_dita2 = 21472;

p12_dita1 = -21873;
p12_dita2 = 21272;

p13_dita1 = -22073;
p13_dita2 = 21071;

p14_dita1 = -22273;
p14_dita2 = 20870;

p15_dita1 = -22473;
p15_dita2 = 20668;

p16_dita1 = -22673;
p16_dita2 = 20466;

```

```

p17_dita1 = -22873;
p17_dita2 = 20263;

p18_dita1 = -23072;
p18_dita2 = 20061;

p19_dita1 = -23272;
p19_dita2 = 19857;

p20_dita1 = -23472;
p20_dita2 = 19653;

        logopoli();
        H154 = 1;
        seg = seg +1;
    } // end if
    if (seg == 6) {
        p1_dita1 = -23472;
p1_dita2 = 19653;

p2_dita1 = -23446;
p2_dita2 = 19630;

p3_dita1 = -23419;
p3_dita2 = 19606;

p4_dita1 = -23391;
p4_dita2 = 19581;

p5_dita1 = -23362;
p5_dita2 = 19554;

p6_dita1 = -23332;
p6_dita2 = 19527;

p7_dita1 = -23301;
p7_dita2 = 19498;

p8_dita1 = -23269;
p8_dita2 = 19467;

p9_dita1 = -23236;
p9_dita2 = 19436;

p10_dita1 = -23201;
p10_dita2 = 19403;

p11_dita1 = -23166;
p11_dita2 = 19368;

p12_dita1 = -23130;
p12_dita2 = 19333;

p13_dita1 = -23092;
p13_dita2 = 19296;

p14_dita1 = -23053;
p14_dita2 = 19258;

p15_dita1 = -23013;
p15_dita2 = 19218;

p16_dita1 = -22973;
p16_dita2 = 19177;

p17_dita1 = -22931;
p17_dita2 = 19134;

p18_dita1 = -22887;
p18_dita2 = 19090;

p19_dita1 = -22843;
p19_dita2 = 19044;

p20_dita1 = -22798;
p20_dita2 = 18997;
        logopoli();
        H155 = 1;
        seg = seg +1;
    } // end if
    if (seg == 7) {
p1_dita1 = -22798;
p1_dita2 = 18997;

p2_dita1 = -22733;
p2_dita2 = 19067;

p3_dita1 = -22669;
p3_dita2 = 19137;

p4_dita1 = -22605;
p4_dita2 = 19207;

p5_dita1 = -22540;
p5_dita2 = 19276;

p6_dita1 = -22476;
p6_dita2 = 19346;

```

```

p7_dita1 = -22411;
p7_dita2 = 19415;

p8_dita1 = -22346;
p8_dita2 = 19484;

p9_dita1 = -22281;
p9_dita2 = 19553;

p10_dita1 = -22217;
p10_dita2 = 19622;

p11_dita1 = -22152;
p11_dita2 = 19690;

p12_dita1 = -22087;
p12_dita2 = 19759;

p13_dita1 = -22022;
p13_dita2 = 19827;

p14_dita1 = -21957;
p14_dita2 = 19895;

p15_dita1 = -21891;
p15_dita2 = 19963;

p16_dita1 = -21826;
p16_dita2 = 20031;

p17_dita1 = -21761;
p17_dita2 = 20099;

p18_dita1 = -21695;
p18_dita2 = 20167;

p19_dita1 = -21630;
p19_dita2 = 20235;

p20_dita1 = -21564;
p20_dita2 = 20302;
    logopoli();
    H156 = 1;
    seg = seg +1;
} // end if
if (seg == 8) {
    p1_dita1 = -21564;
p1_dita2 = 20302;

p2_dita1 = -21609;
p2_dita2 = 20347;

p3_dita1 = -21652;
p3_dita2 = 20390;

p4_dita1 = -21694;
p4_dita2 = 20433;

p5_dita1 = -21735;
p5_dita2 = 20473;

p6_dita1 = -21774;
p6_dita2 = 20513;

p7_dita1 = -21813;
p7_dita2 = 20551;

p8_dita1 = -21850;
p8_dita2 = 20588;

p9_dita1 = -21886;
p9_dita2 = 20624;

p10_dita1 = -21921;
p10_dita2 = 20658;

p11_dita1 = -21954;
p11_dita2 = 20692;

p12_dita1 = -21987;
p12_dita2 = 20724;

p13_dita1 = -22018;
p13_dita2 = 20754;

p14_dita1 = -22048;
p14_dita2 = 20784;

p15_dita1 = -22078;
p15_dita2 = 20812;

p16_dita1 = -22106;
p16_dita2 = 20840;

p17_dita1 = -22132;
p17_dita2 = 20866;

p18_dita1 = -22158;
p18_dita2 = 20890;

```

```

p19_dita1 = -22183;
p19_dita2 = 20914;

p20_dita1 = -22206;
p20_dita2 = 20937;
    logopoli();
    H157 = 1;
    seg = seg +1;
} // end if
if (seg == 9) {
p1_dita1 = -22206;
p1_dita2 = 20937;

p2_dita1 = -22195;
p2_dita2 = 20926;

p3_dita1 = -22183;
p3_dita2 = 20914;

p4_dita1 = -22171;
p4_dita2 = 20902;

p5_dita1 = -22158;
p5_dita2 = 20890;

p6_dita1 = -22146;
p6_dita2 = 20878;

p7_dita1 = -22132;
p7_dita2 = 20866;

p8_dita1 = -22119;
p8_dita2 = 20853;

p9_dita1 = -22106;
p9_dita2 = 20840;

p10_dita1 = -22092;
p10_dita2 = 20826;

p11_dita1 = -22078;
p11_dita2 = 20812;

p12_dita1 = -22063;
p12_dita2 = 20798;

p13_dita1 = -22048;
p13_dita2 = 20784;

p14_dita1 = -22033;
p14_dita2 = 20769;

p15_dita1 = -22018;
p15_dita2 = 20754;

p16_dita1 = -22003;
p16_dita2 = 20739;

p17_dita1 = -21987;
p17_dita2 = 20724;

p18_dita1 = -21971;
p18_dita2 = 20708;

p19_dita1 = -21954;
p19_dita2 = 20692;

p20_dita1 = -21938;
p20_dita2 = 20675;
    logopoli();
    H158 = 1;
    seg = seg +1;
} // end if
if (seg == 10) {
p1_dita1 = -21938;
p1_dita2 = 20675;

p2_dita1 = -21806;
p2_dita2 = 20809;

p3_dita1 = -21673;
p3_dita2 = 20943;

p4_dita1 = -21541;
p4_dita2 = 21076;

p5_dita1 = -21408;
p5_dita2 = 21209;

p6_dita1 = -21276;
p6_dita2 = 21342;

p7_dita1 = -21143;
p7_dita2 = 21475;

p8_dita1 = -21009;
p8_dita2 = 21607;

p9_dita1 = -20876;
p9_dita2 = 21740;

```

```
p10_dita1 = -20742;
p10_dita2 = 21872;

p11_dita1 = -20608;
p11_dita2 = 22003;

p12_dita1 = -20474;
p12_dita2 = 22135;

p13_dita1 = -20339;
p13_dita2 = 22267;

p14_dita1 = -20204;
p14_dita2 = 22398;

p15_dita1 = -20069;
p15_dita2 = 22529;

p16_dita1 = -19933;
p16_dita2 = 22661;

p17_dita1 = -19797;
p17_dita2 = 22792;

p18_dita1 = -19660;
p18_dita2 = 22922;

p19_dita1 = -19523;
p19_dita2 = 23053;

p20_dita1 = -19386;
p20_dita2 = 23184;

        logopoli();
        H159 = 1;
        seg = seg +1;
    } // end if

} // end while seg
finTRAY=1;
TRAY=0;

break;

    } // end switch
} // end IF

} // end while

} // end main
```

```
//=====CONTROLADOR DERECHO=====

/*=====
      IPOS Source File
=====*/
#include <constb.h>
#include <iob.h>
//#pragma initials 0 127
//#pragma globals 700 800
//#pragma var 201 251

//Note:
//Movidrive inverters with activated technology function "ISYNCH" or "Cam"
//do not support variables H360 an higher.

//DEFINICION DE VARIABLES
#define tdelay H11 //delay entre un punto y el siguiente [ms]
#define tcom H12 //delay de comunicaciones, para evitar que se pierdan señales [ms]
#define vhome H13 //vel al hacer homing [RPM]
#define vel H14 //vel al moverse [RPM]
#define lost1 H15// perdido( estaad inicial) lost=1, after homming lost=0 (SERVO 1)
#define lost2 H16 //estado inicial perdido (SERVO 2)

//ANGULOS DITA Y ESTADOS
#define dita1 H130
#define dita2 H131
#define prueba H132 // estado servo 1 ( moving=0; ready=1; finish=2)
#define st2 H133 // estado servo 2 ( moving=0; ready=1; finish=2)

/*=====*/
AngGiro2() {
// Wait(1000);
//H136=3;
_GoAbs( GO_WAIT,dita2 );
st2=2;
while( st2!=1 ){
//_Wait(1000);
//H136=4;
} //end AngGiro2

/*=====*/
Homming() {
//while( st2!=3 ){
st2=0;
_Go0( GO0_U_W_ZP );
st2=2;
while( st2!=1 ){

} // end homming

/*=====
      Main Function (IPOS Entry Function)
=====*/
main() {

//INICIALIZACION

tdelay=500;
tcom=50;
st2=2; //estado 2 es un estado de espera no va a hacer el giro ni tampoco va hacer home
_BitClear( ControlWord, 0 );
_BitClear( ControlWord, 1 );
_BitClear( ControlWord, 3 );
_BitSet( ControlWord,2 );

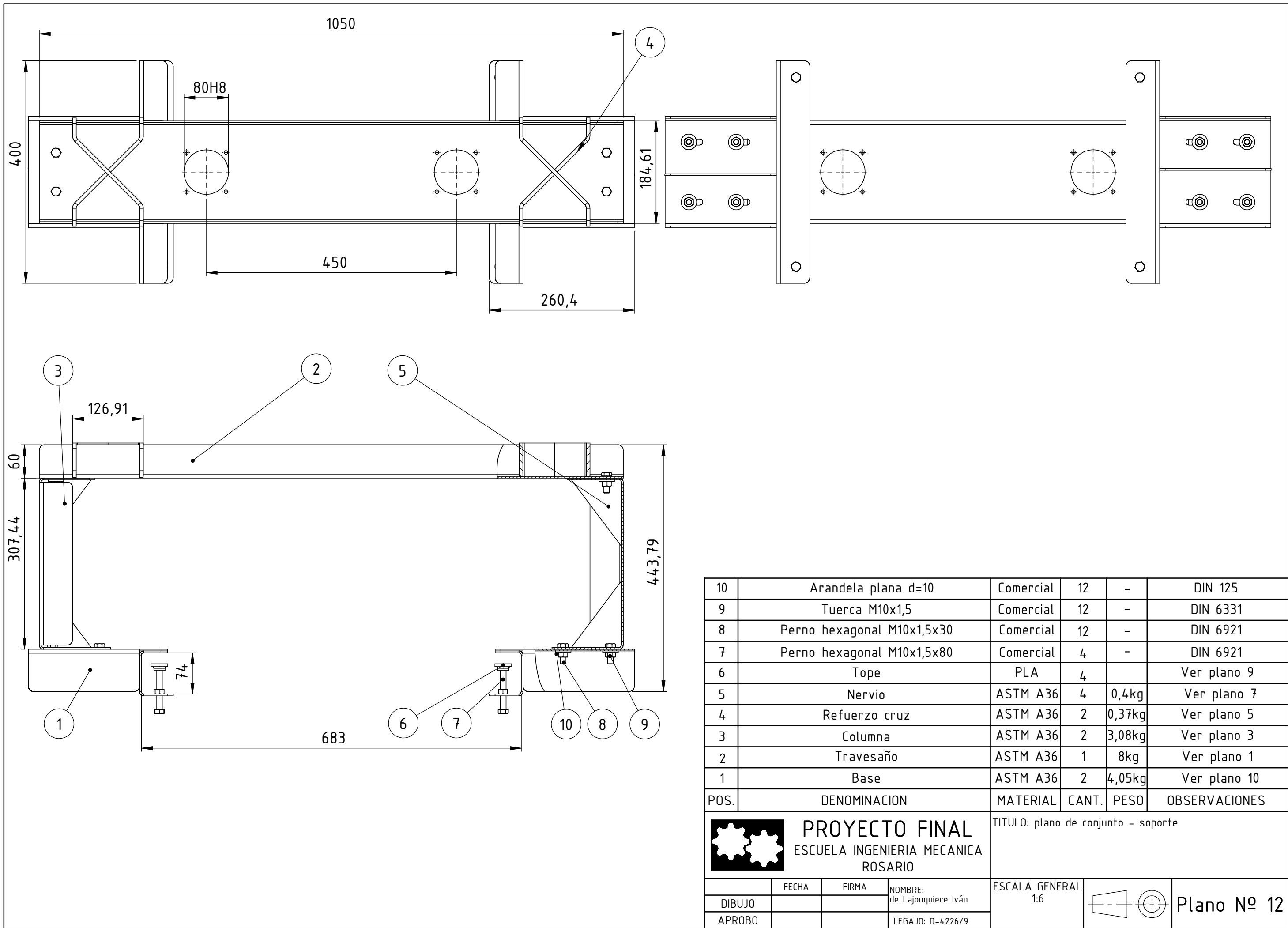
/*=====
      PROGRAMA DE POSICIONES
=====*/

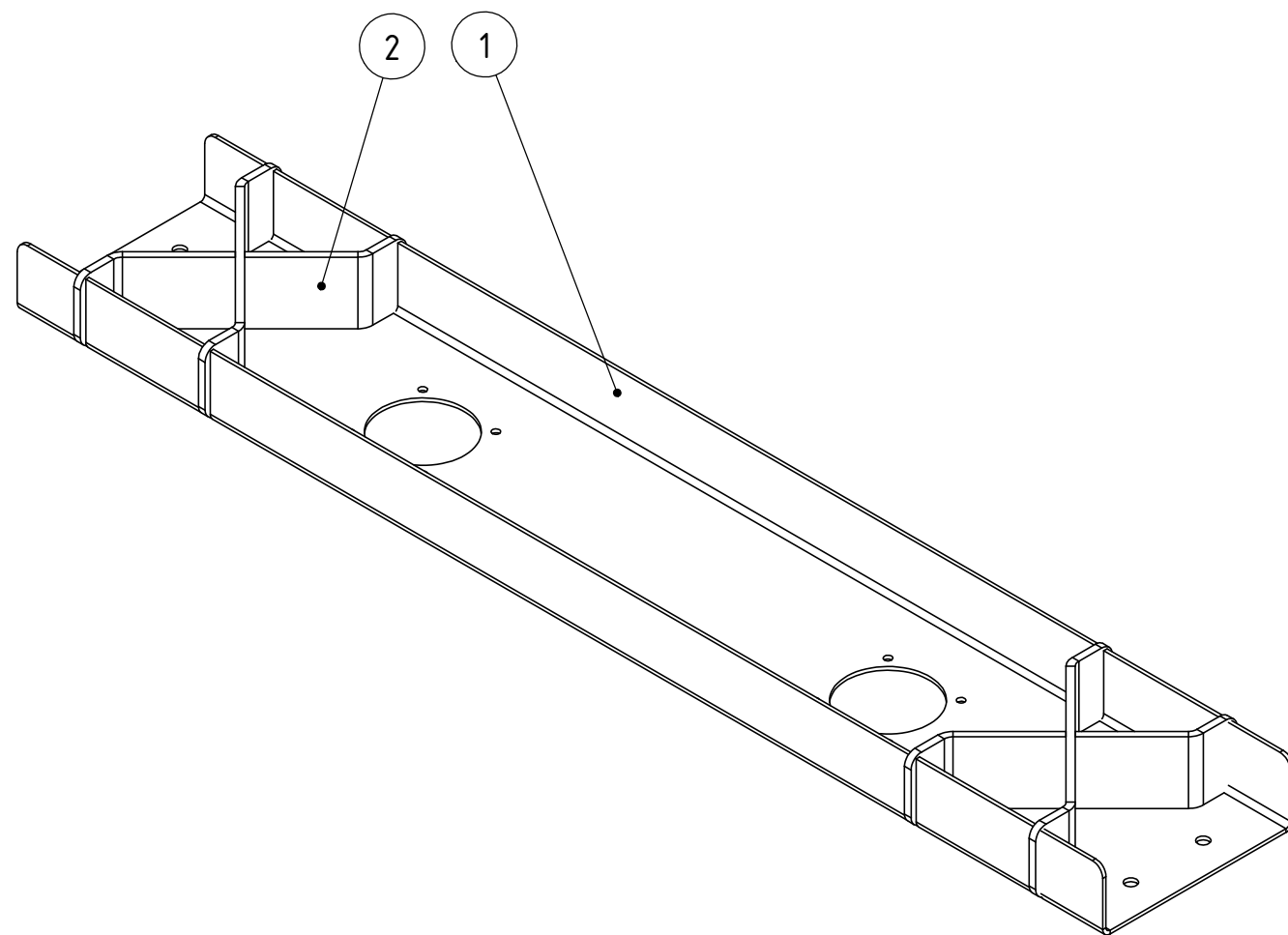
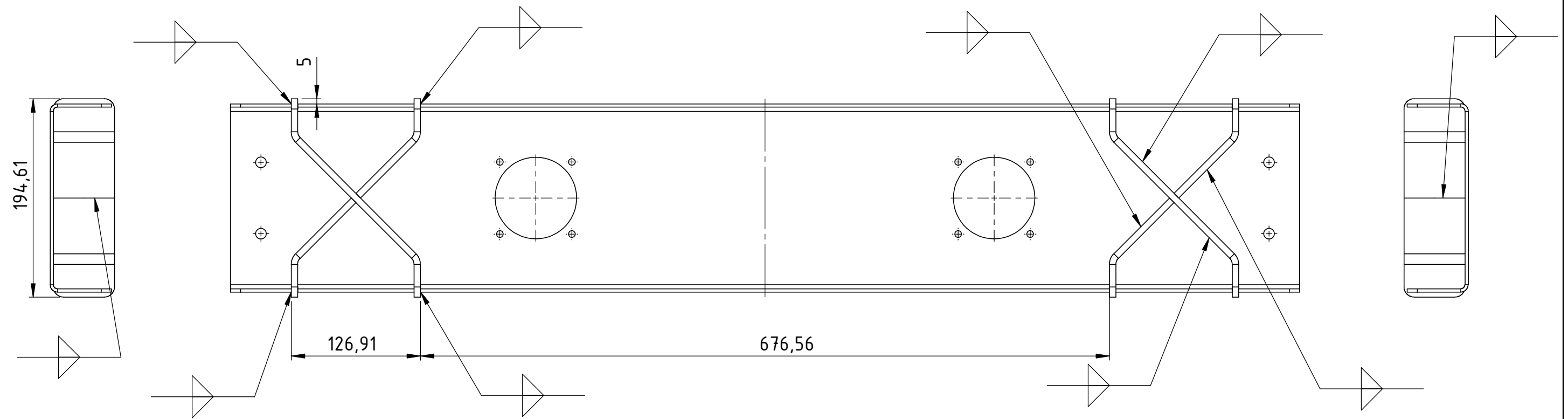
while (1) {
//H136=2;

if(st2==3) Homming();

if ((st2==0)) {
//H136=10;
//_Wait(1000);
AngGiro2();

} //end if *///while
} //end while
} //end main
```

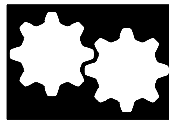
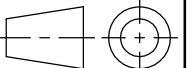


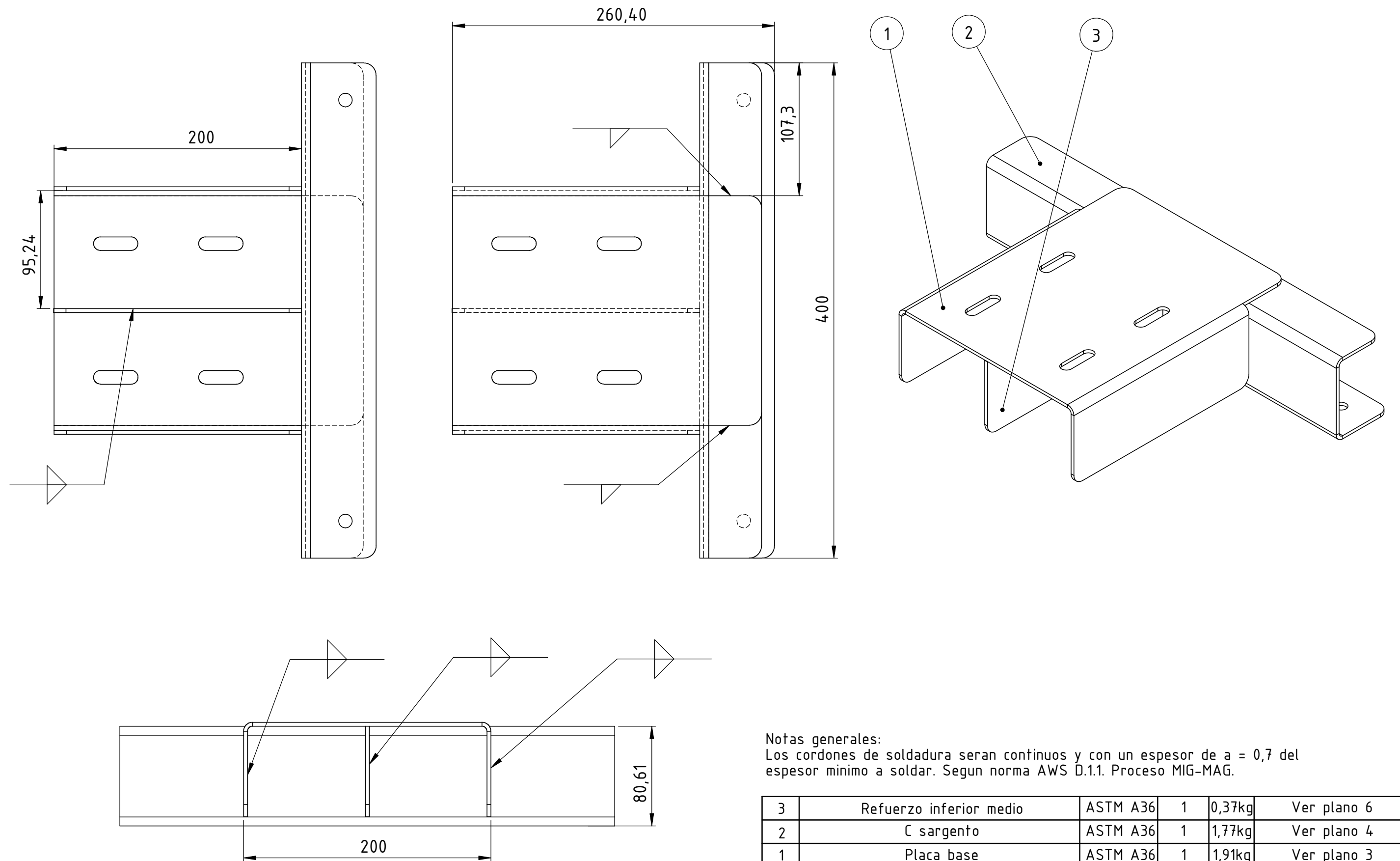


ISOMETRIA ESC 1:5

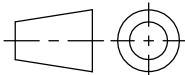
Notas generales:

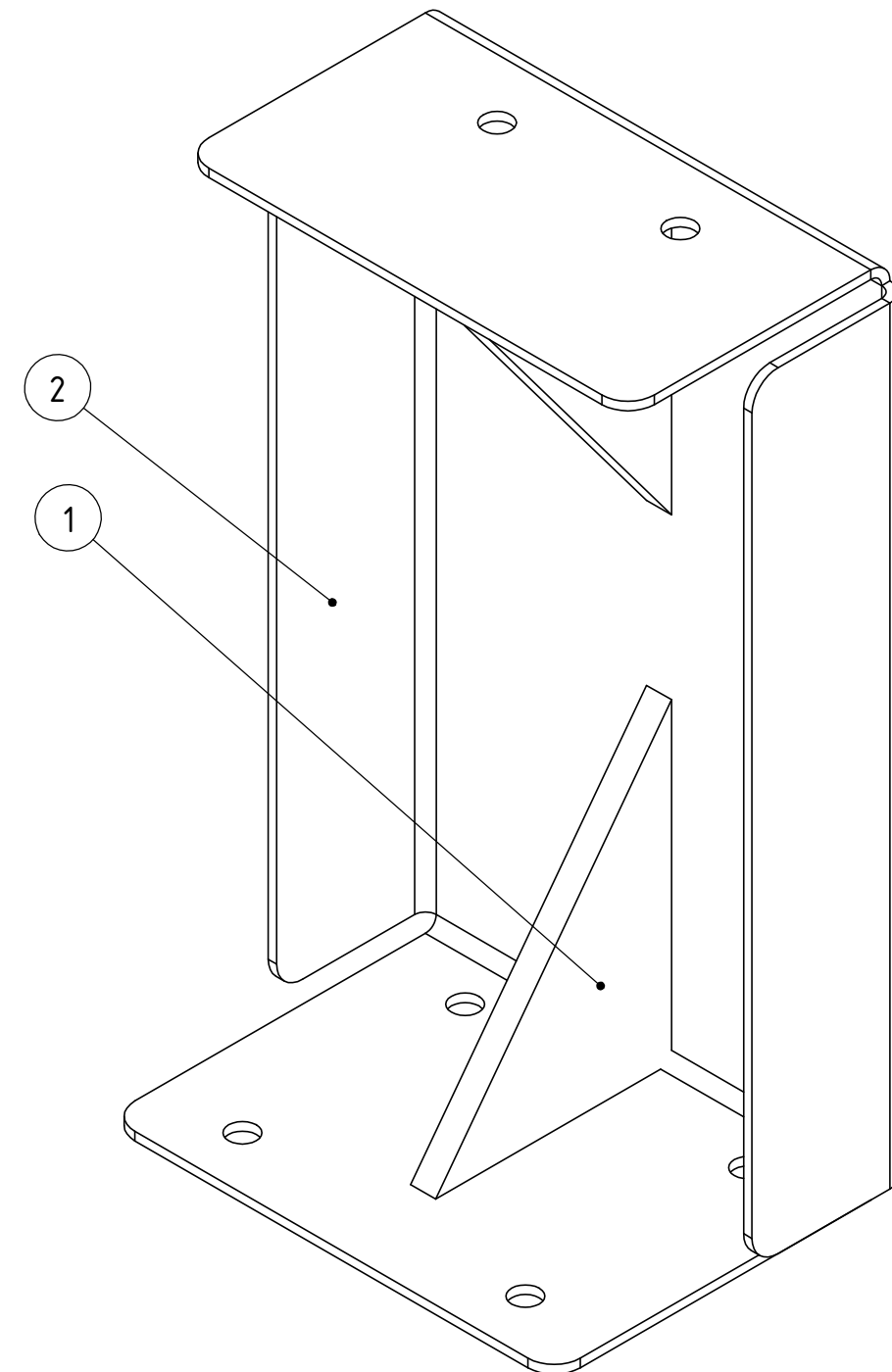
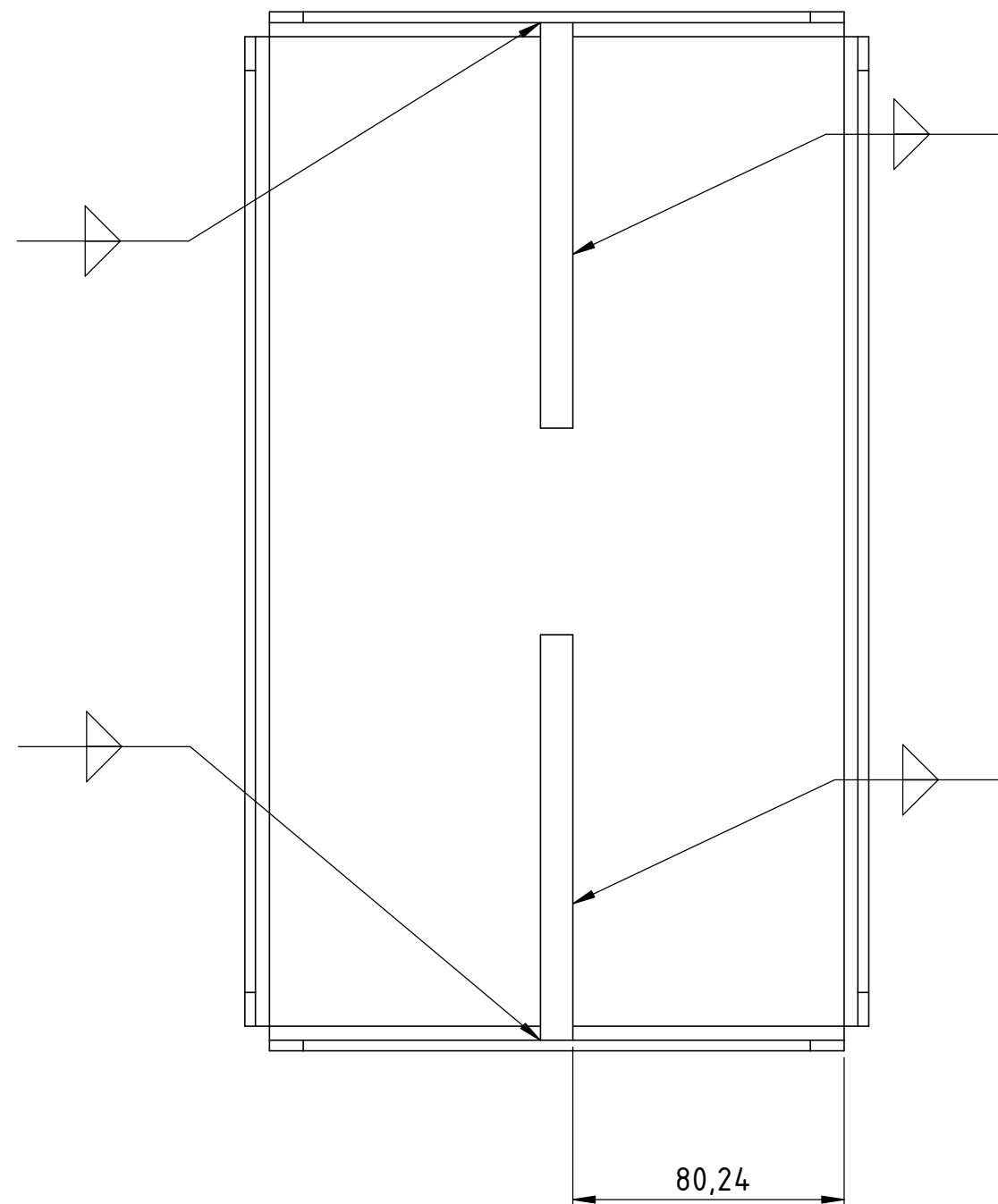
Los cordones de soldadura seran continuos y con un espesor de $a = 0,7$ del espesor minimo a soldar. Segun norma AWS D.1.1. Proceso MIG-MAG.

2	Chapa refuerzo	ASTM A36	4	0,71kg	Ver plano 5
1	Travesaño	ASTM A36	1	7,29kg	Ver plano 1
POS.	DENOMINACION	MATERIAL	CANT.	PESO	OBSERVACIONES
 PROYECTO FINAL ESCUOLA INGENIERIA MECANICA ROSARIO		TITULO: plano de conjunto soldado - travesaño			
	FECHA	FIRMA	NOMBRE:	ESCALA GENERAL 1:4	
DIBUJO			de Lajonquiere Iván		
APROBO			LEGAJO: D-4226/9	 Plano Nº 9	

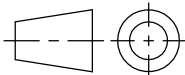


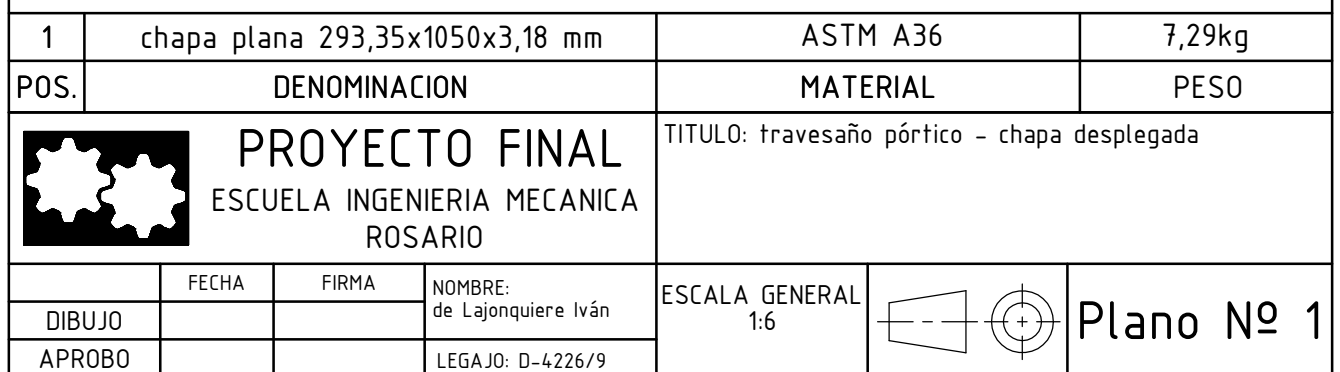
Notas generales:
Los cordones de soldadura seran continuos y con un espesor de $a = 0,7$ del espesor minimo a soldar. Segun norma AWS D.1.1. Proceso MIG-MAG.

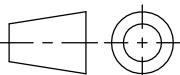
3	Refuerzo inferior medio	ASTM A36	1	0,37kg	Ver plano 6
2	C sargento	ASTM A36	1	1,77kg	Ver plano 4
1	Placa base	ASTM A36	1	1,91kg	Ver plano 3
POS.	DENOMINACION	MATERIAL	CANT.	PESO	OBSERVACIONES
 PROYECTO FINAL ESCUELA INGENIERIA MECANICA ROSARIO		TITULO: plano de conjunto soldado - base			
	FECHA	FIRMA	NOMBRE: de Lajonquiere Iván		ESCALA GENERAL 1:3
DIBUJO			LEGAJO: D-4226/9		
APROBO					 Plano Nº 10

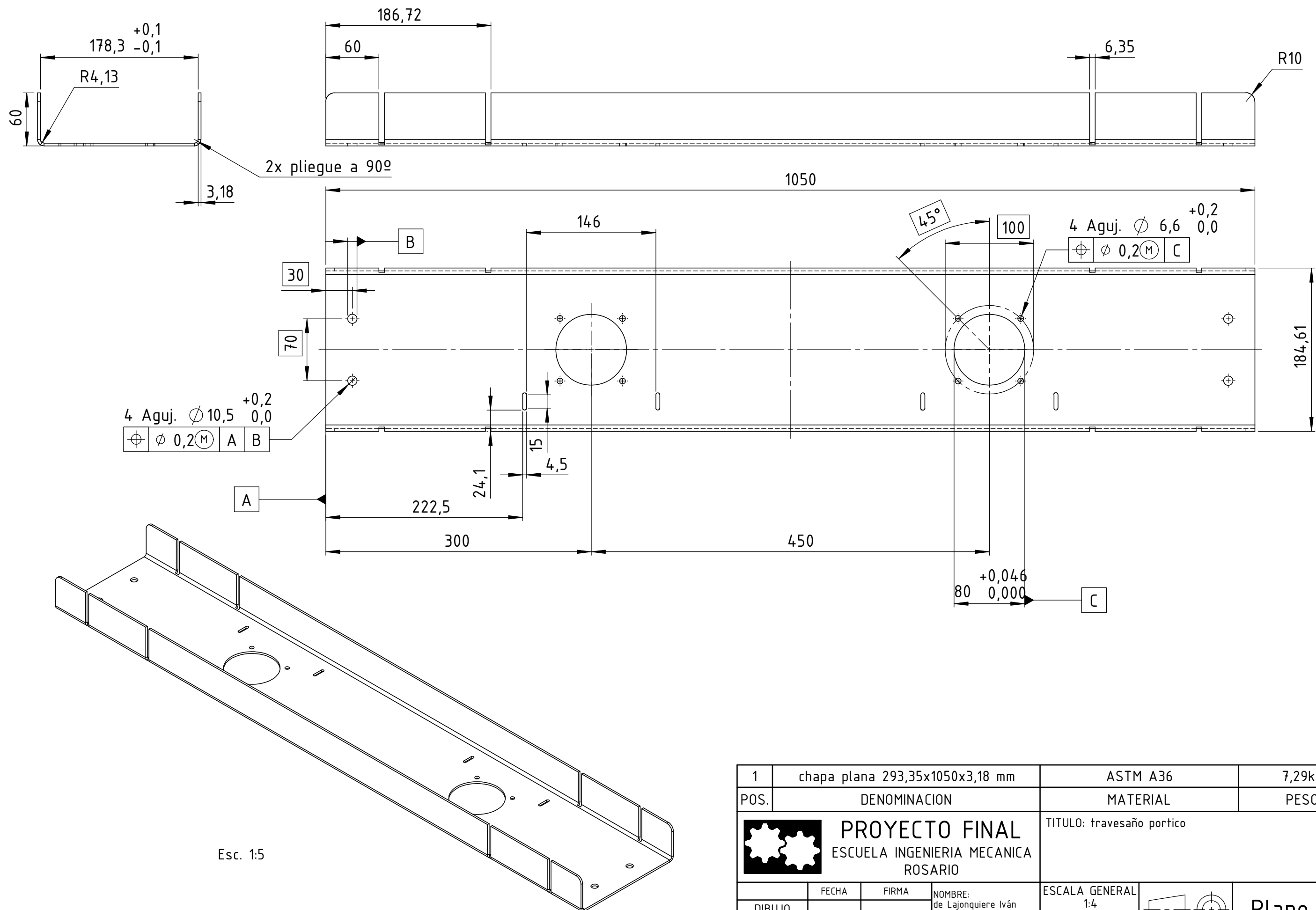


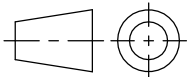
Notas generales:
Los cordones de soldadura seran continuos y con un espesor de $a = 0,7$ del espesor minimo a soldar. Segun norma AWS D.1.1. Proceso MIG-MAG.

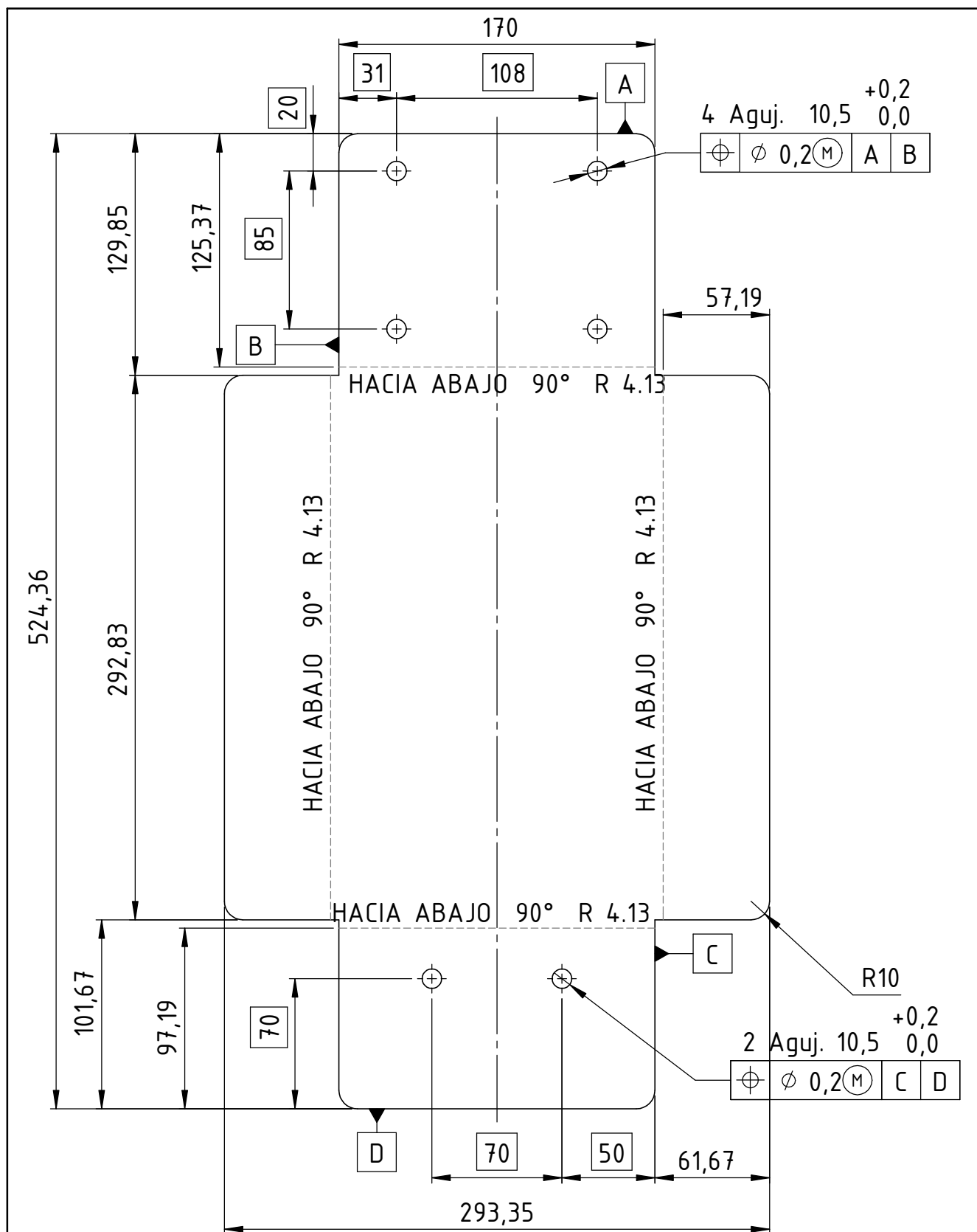
2	Columna	ASTM A36	1	3,08kg	Ver plano 2
1	Nervio	ASTM A36	2	0,4kg	Ver plano 7
POS.	DENOMINACION	MATERIAL	CANT.	PESO	OBSERVACIONES
 PROYECTO FINAL ESCUELA INGENIERIA MECANICA ROSARIO		TITULO: plano de conjunto soldado - columna			
	FECHA	FIRMA	NOMBRE:	ESCALA GENERAL 1:3 	Plano Nº 11
DIBUJO			de Lajonquiere Iván		
APROBO			LEGAJO: D-4226/9		

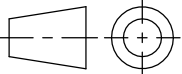


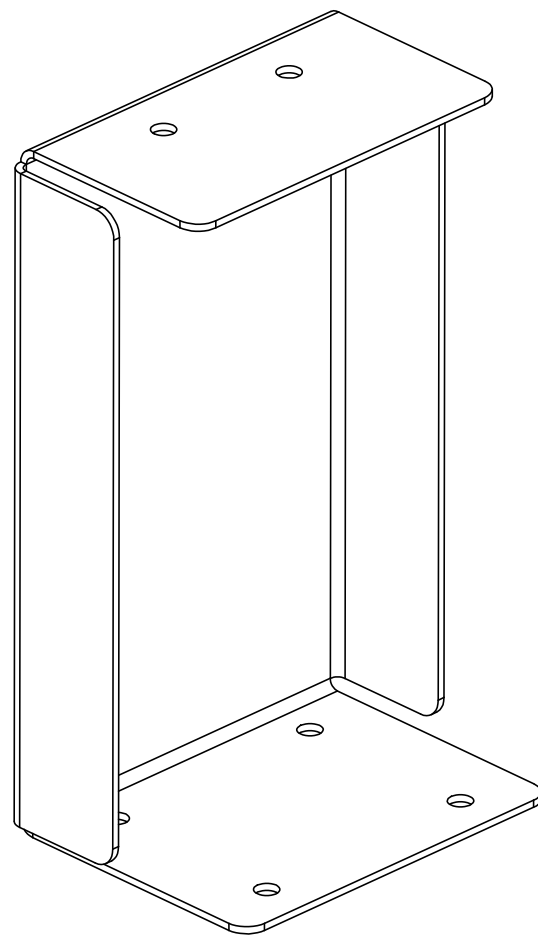
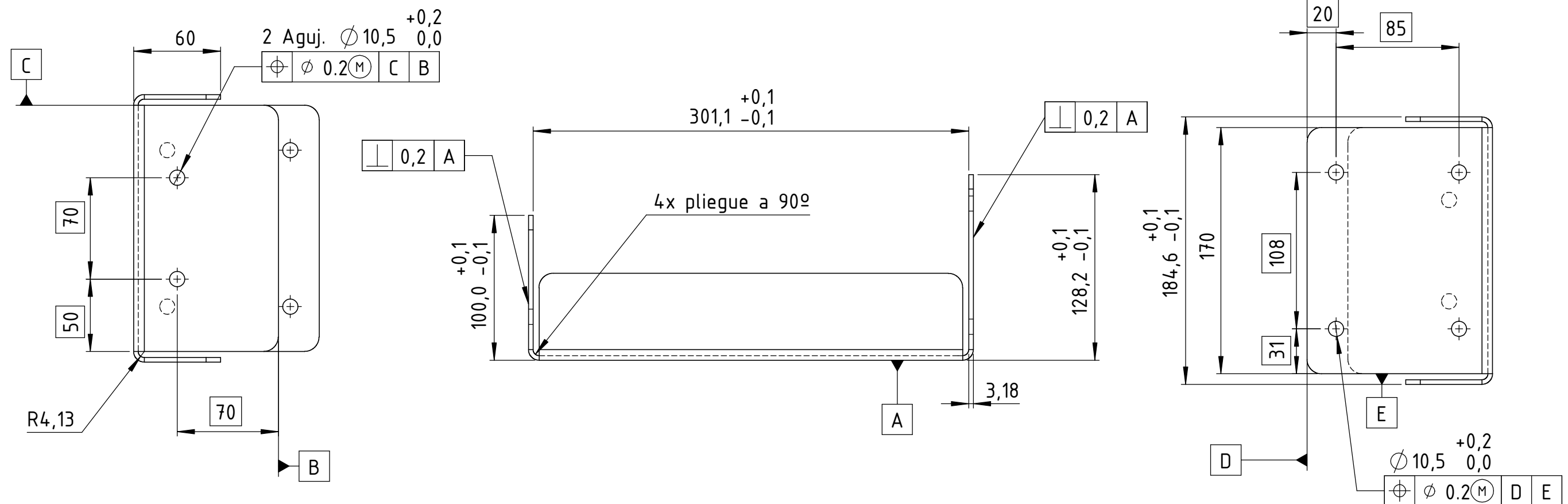
1	chapa plana 293,35x1050x3,18 mm	ASTM A36	7,29kg
POS.	DENOMINACION	MATERIAL	PESO
 PROYECTO FINAL ESCUELA INGENIERIA MECANICA ROSARIO		TITULO: travesaño pórtico - chapa desplegada	
	FECHA	FIRMA	NOMBRE:
DIBUJO			de Lajonquiere Iván
APROBO			LEGAJO: D-4226/9
ESCALA GENERAL 1:6			Plano Nº 1

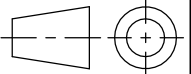


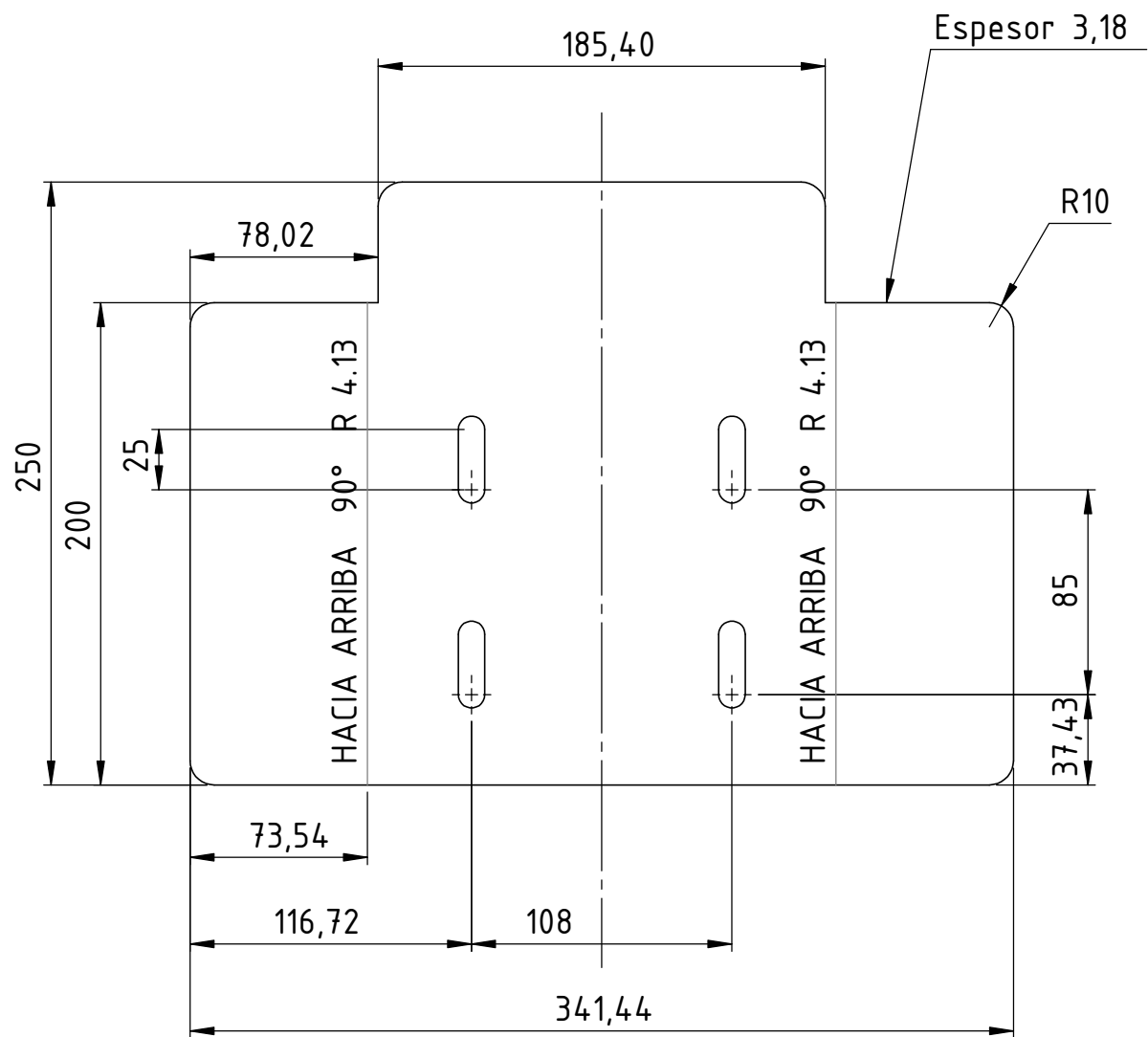
1	chapa plana 293,35x1050x3,18 mm	ASTM A36	7,29kg
POS.	DENOMINACION	MATERIAL	PESO
 PROYECTO FINAL ESCUELA INGENIERIA MECANICA ROSARIO		TITULO: travesaño portico	
	FECHA	FIRMA	NOMBRE:
DIBUJO			de Lajonquiere Iván
APROBO			LEGAJO: D-4226/9
ESCALA GENERAL 1:4			Plano 1b

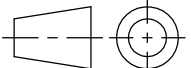


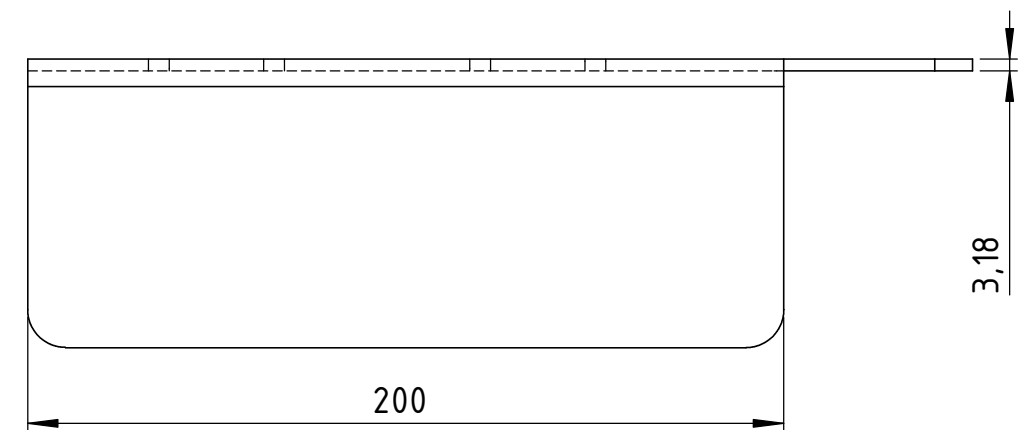
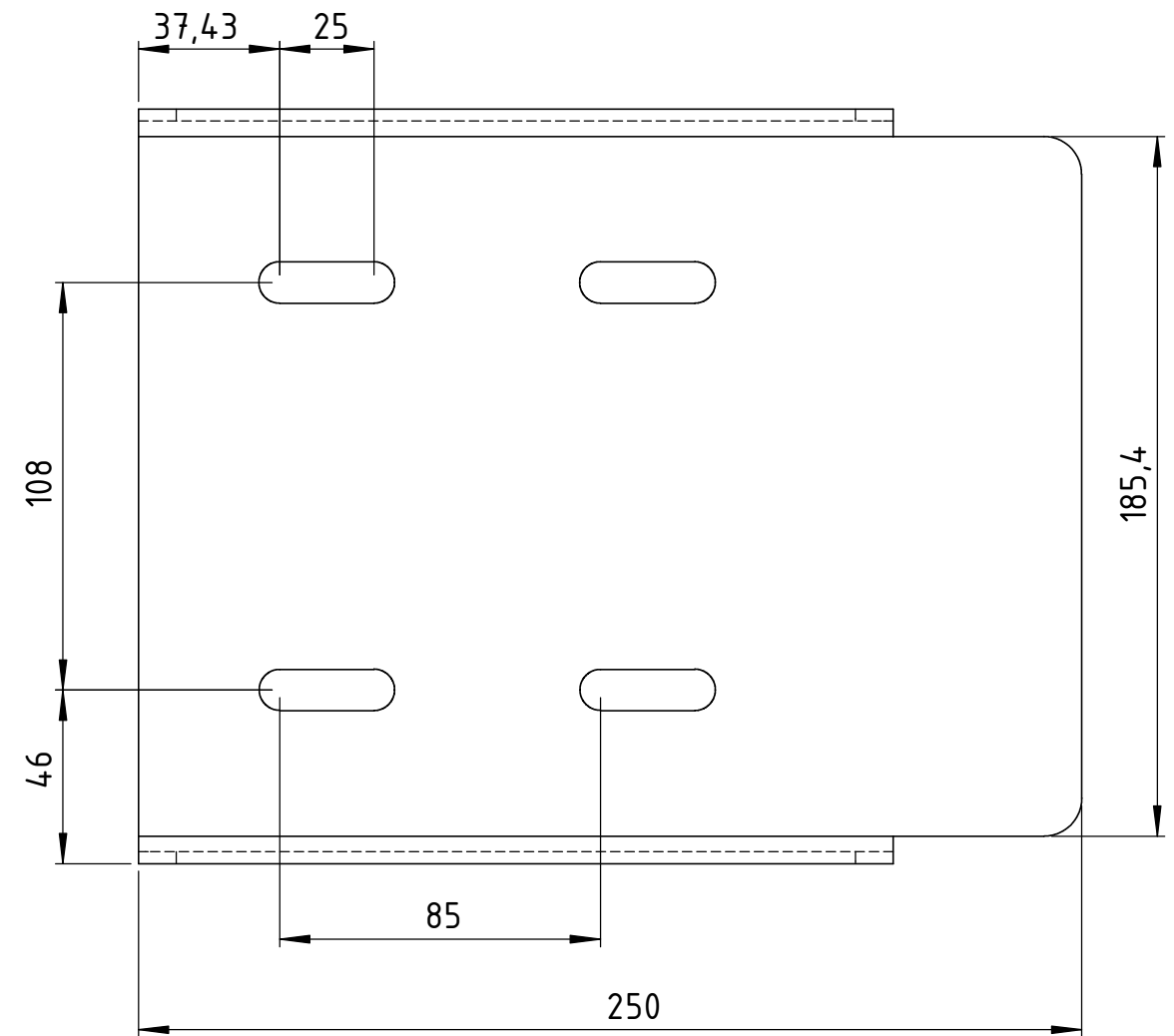
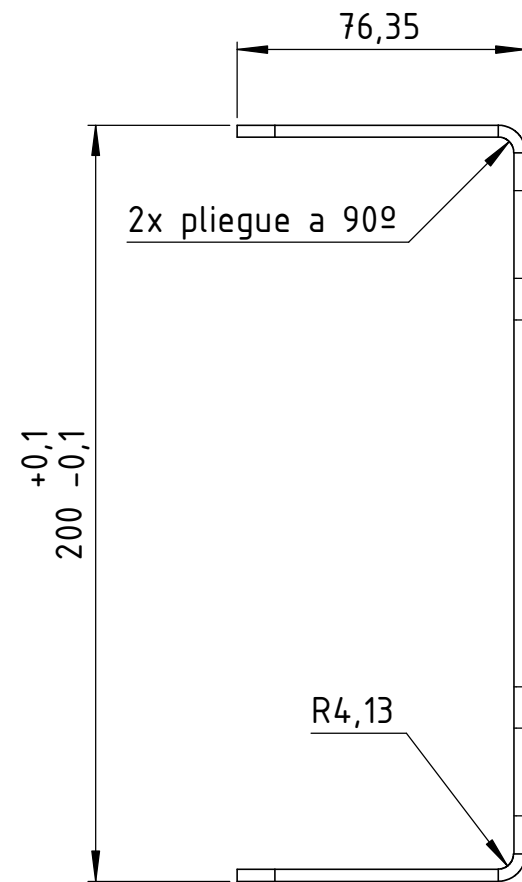
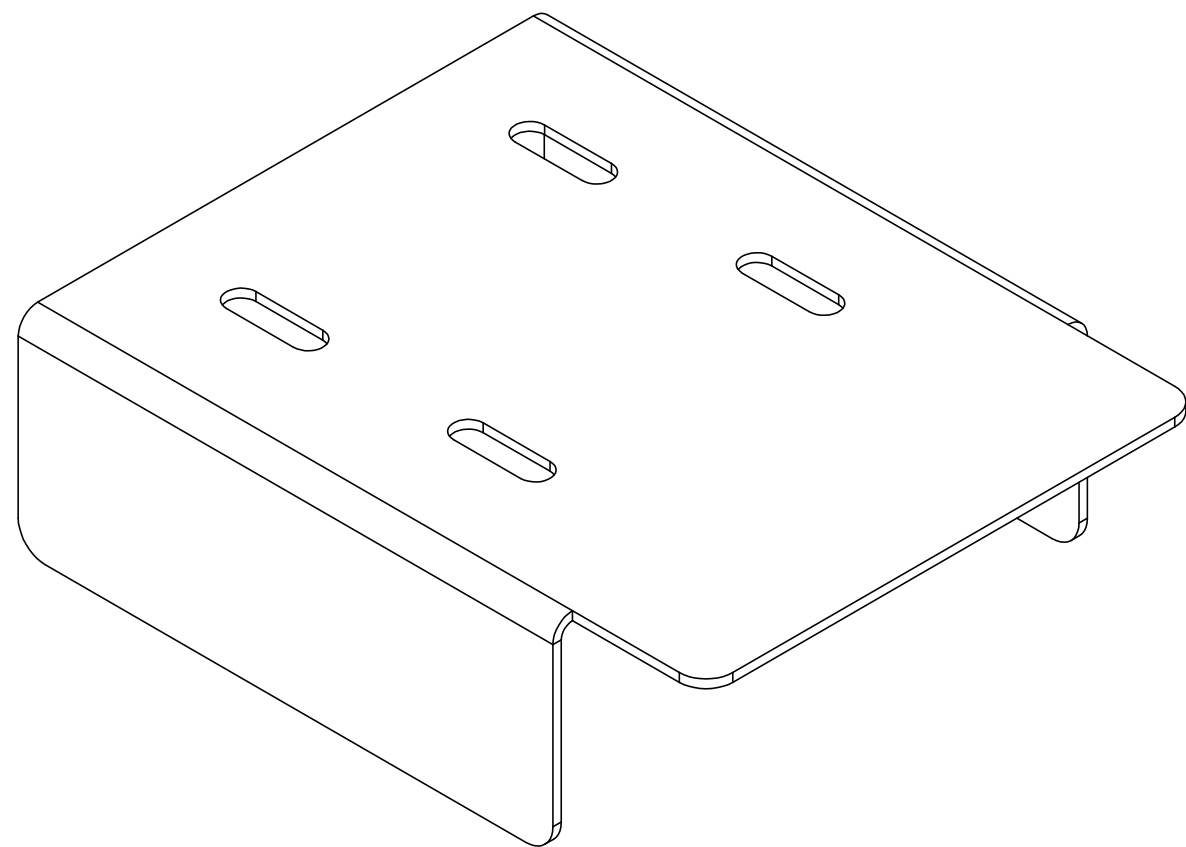
1	chapa plana 293,35x524,36x3,18 mm	ASTM A36	3,08kg
POS.	DENOMINACION	MATERIAL	PESO
 PROYECTO FINAL ESCUELA INGENIERIA MECANICA ROSARIO		TITULO: columna pórtico - chapa desplegada	
	FECHA	FIRMA	NOMBRE: de Lajonquiere Iván
DIBUJO			ESCALA GENERAL 1:3
APROBO			LEGAJO: D-4226/9
			Plano Nº 2

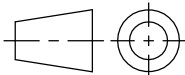


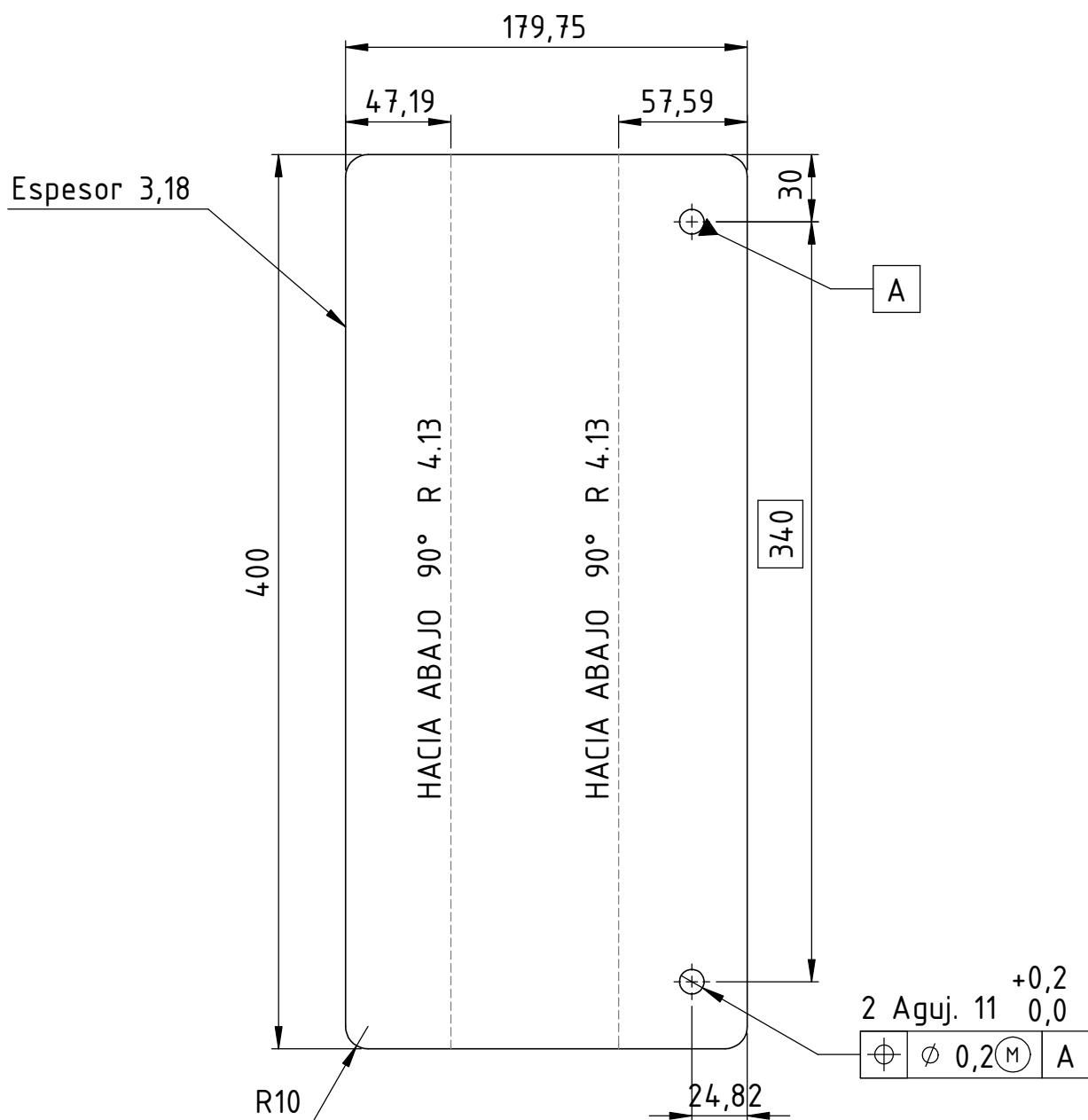
1	chapa plana 293,35x524,36x3,18 mm	ASTM A36	3,08kg
POS.	DENOMINACION	MATERIAL	PESO
 PROYECTO FINAL ESCUELA INGENIERIA MECANICA ROSARIO		TITULO: columna pórtico	
	FECHA	FIRMA	NOMBRE:
DIBUJO			de Lajonquiere Iván
APROBO			LEGAJO: D-4226/9
ESCALA GENERAL 1:3		 Plano 2b	

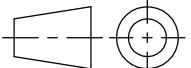


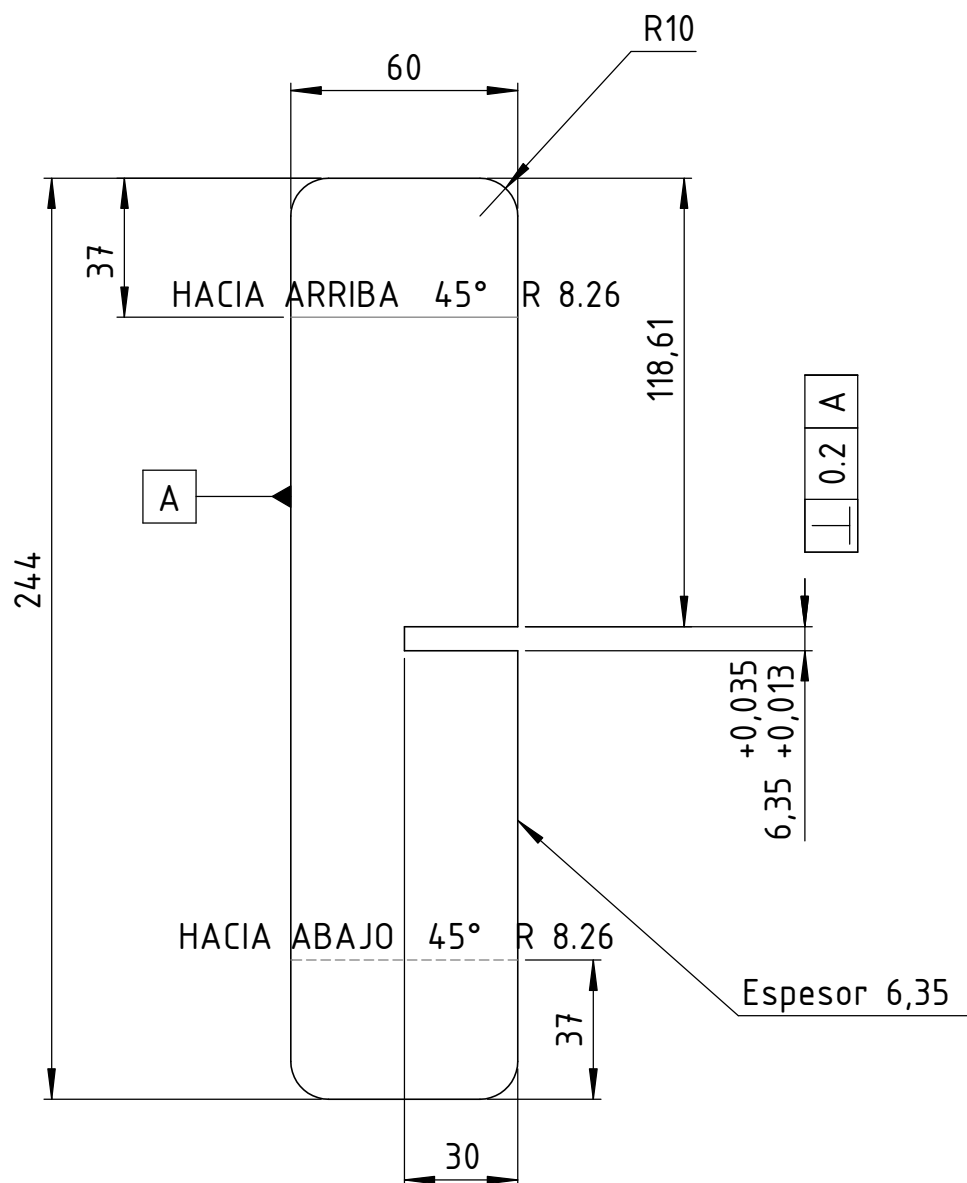
1	chapa plana 341,44x250x3,18 mm	ASTM A36	1,91kg
POS.	DENOMINACION	MATERIAL	PESO
 PROYECTO FINAL ESCUELA INGENIERIA MECANICA ROSARIO		TITULO: chapa sosten - chapa desplegada	
	FECHA	FIRMA	NOMBRE: de Lajonquiere Iván
DIBUJO			ESCALA GENERAL 1:3
APROBO			 Plano Nº 3
			LEGAJO: D-4226/9

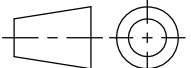


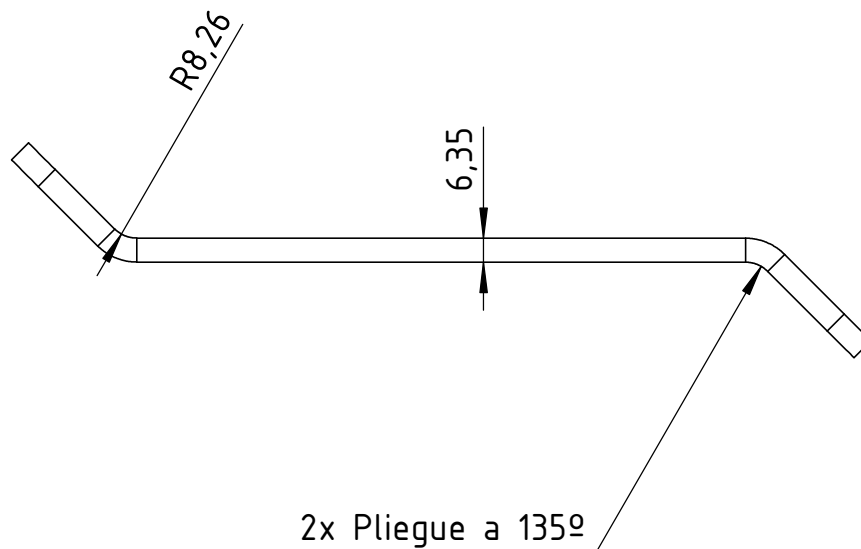
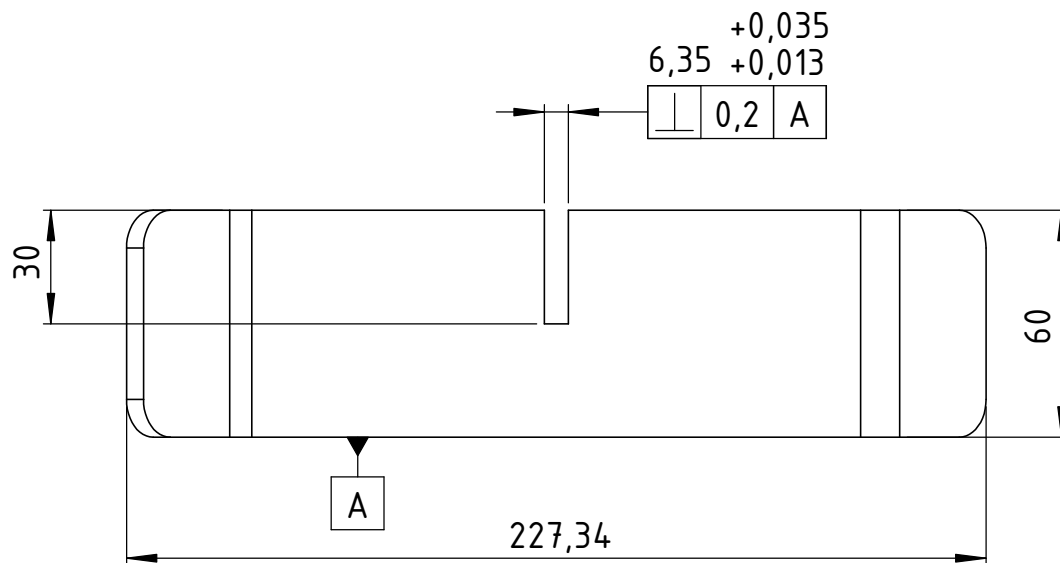
1	chapa plana 341,44x250x3,18 mm			ASTM A36	1,91kg
POS.	DENOMINACION			MATERIAL	PESO
 PROYECTO FINAL ESCUELA INGENIERIA MECANICA ROSARIO				TITULO: chapa sosten	
	FECHA	FIRMA	NOMBRE: de Lajonquiere Iván	ESCALA GENERAL 1:2	 Plano 3b
DIBUJO					
APROBO			LEGAJO: D-4226/9		

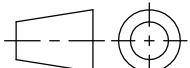


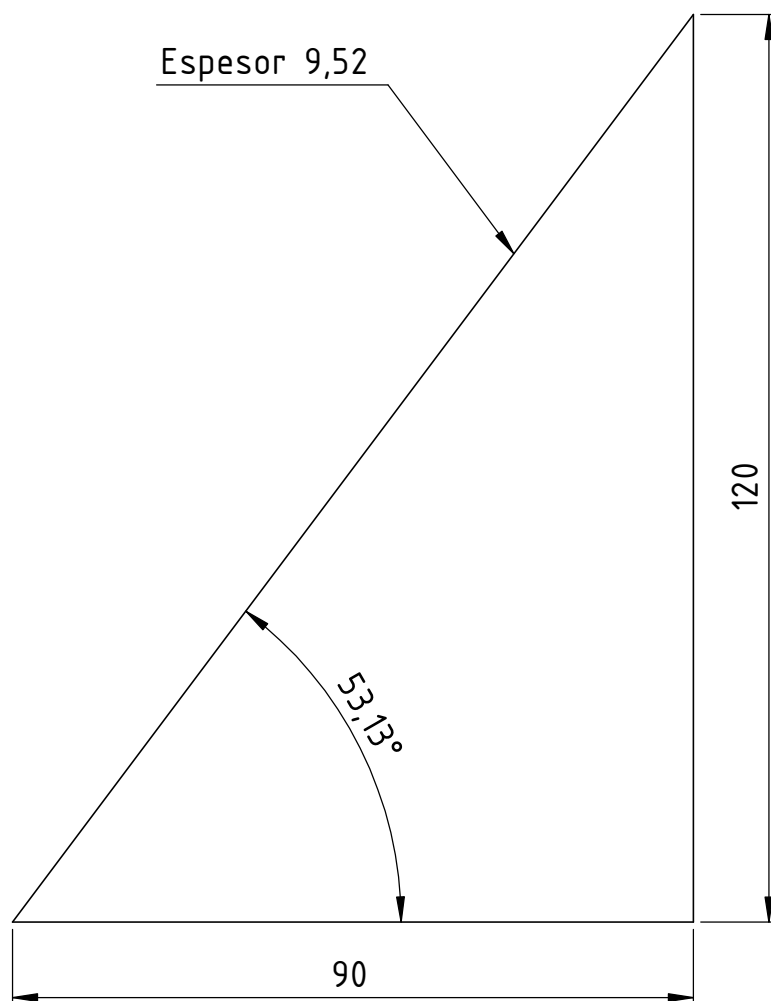
1	chapa plana 179,75x400x3,18 mm	ASTM A36	1,77kg
POS.	DENOMINACION	MATERIAL	PESO
 PROYECTO FINAL ESCUELA INGENIERIA MECANICA ROSARIO		TITULO: c sargento - chapa desplegada	
	FECHA	FIRMA	NOMBRE: de Lajonquiere Iván
DIBUJO			ESCALA GENERAL 1:3
APROBO			LEGAJO: D-4226/9
			Plano Nº 4

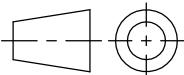


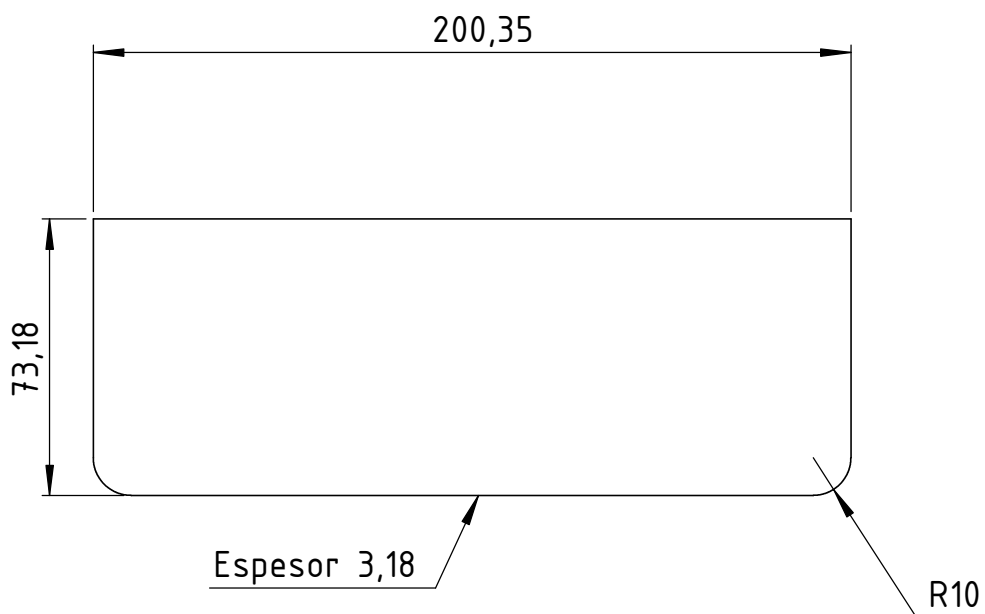
1	chapa plana 60x244x6,35 mm	ASTM A36	0,71kg
POS.	DENOMINACION	MATERIAL	PESO
 PROYECTO FINAL ESCUELA INGENIERIA MECANICA ROSARIO		TITULO: chapa refuerzo X - chapa desplegada	
	FECHA	FIRMA	NOMBRE: de Lajonquiere Iván
DIBUJO			ESCALA GENERAL 1:2
APROBO			LEGAJO: D-4226/9
			Plano Nº 5

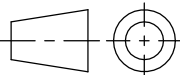


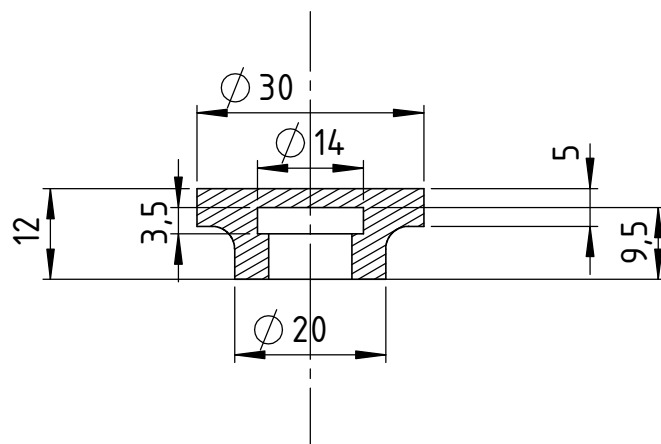
1	chapa plana 60x244x6,35 mm			ASTM A36	0,71kg	
POS.	DENOMINACION			MATERIAL	PESO	
 PROYECTO FINAL ESCUELA INGENIERIA MECANICA ROSARIO				TITULO: chapa refuerzo x		
	FECHA	FIRMA	NOMBRE: de Lajonquiere Iván	ESCALA GENERAL 1:2		Plano 5b
DIBUJO						
APROBO			LEGAJO: D-4226/9			

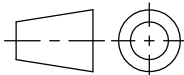


1	chapa plana 90x120x9,52 mm			ASTM A36	0,4kg
POS.	DENOMINACION			MATERIAL	PESO
 PROYECTO FINAL ESCUELA INGENIERIA MECANICA ROSARIO				TITULO: nervio	
	FECHA	FIRMA	NOMBRE: de Lajonquiere Iván	ESCALA GENERAL 1:1	 Plano Nº 7
DIBUJO					
APROBO			LEGAJO: D-4226/9		



1	chapa plana 200,35x73,18x3,18 mm			ASTM A36	0,37kg
POS.	DENOMINACION			MATERIAL	PESO
 PROYECTO FINAL ESCUELA INGENIERIA MECANICA ROSARIO				TITULO: refuerzo base	
	FECHA	FIRMA	NOMBRE: de Lajonquiere Iván	ESCALA GENERAL 1:2	 Plano Nº 6
DIBUJO					
APROBO			LEGAJO: D-4226/9		



1						
POS.	DENOMINACION			MATERIAL	CANT.	PESO
 PROYECTO FINAL ESCUELA INGENIERIA MECANICA ROSARIO				TITULO: tope		
	FECHA	FIRMA	NOMBRE: de Lajonquiere Iván	ESCALA GENERAL 1:1		Plano Nº 9
DIBUJO						
APROBO			LEGAJO: D-4226/9			