UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK & PLACE.

TRABAJO FINAL DEL

Grado en Ingeniería Electrónica Industrial y Automática

REALIZADO POR

Jorge Doménech Jara

TUTORIZADO POR

Ángel Valera Fernández

Carlos Ricolfe Viala

CURSO ACADÉMICO: 2019/2020

Resumen:

En este proyecto de fin de grado se va a diseñar en 3D, imprimir y montar un brazo robótico de 6 grados de libertad que tiene una longitud de 50 cm estirado. El brazo utiliza servomotores controlados con un módulo PCA9685. La configuración del robot es angular. A su vez se diseña una cinta transportadora que dispone de una cámara y un sensor infrarrojo el cual detecta la pieza. A través de la cinta circula una pequeña pieza. El sensor detecta la posición de la pieza y cuando está centrada en el plano de la cámara, para la cinta y realiza una fotografía. Se hace un preprocesado de la imagen y se obtienen las características de la pieza (orientación y centro de masas). Finalmente, la cinta avanza para que el brazo robótico la coja y la deposite en el lugar que corresponda según el tipo de pieza (rectangular o cuadrada). Se usa para la cámara un ESP32cam y para el microcontrolador un ESP32. Se utiliza Arduino para el entorno de desarrollo, MATLAB para realizar cálculos y pruebas de funcionamiento y SolidWorks para diseñar las piezas en 3D. En cuanto al lenguaje de programación es C++ para Arduino y lenguaje M para MATLAB.

Palabras clave: grado en Electrónica Industrial y Automática, TFG, Arduino, impresión 3D, brazo robótico, cinta transportadora, ESP32cam, ESP32, Matlab, 6 grados de libertad, visión artificial.

Resumen:

Aquest projecte de fi de grau tracta de dissenyar en 3D, imprimir i muntar un braç robòtic de 6 graus de llibertat que té una longitud de 50 cm estirat. El braç utilitza servomotors controlats amb un mòdul PCA9685. La configuració del robot és angular. També es dissenya una cinta transportadora que disposa d'una camera i un sensor infraroig el qual detecta la peça. A través de la cinta circula una xicoteta peça. El sensor detecta la posició de la peça i quan està centrada en el pla de la camera, para la cinta per a realitzar una fotografia. Es fa un preprocessat de la imatge i s'obtenen les característiques de la peça (orientació i centre de masses). Posteriorment la cinta avança perquè, finalment, el braç robòtic la coixa i la deposite en el lloc que corresponga segons el tipus de peça (rectangular o quadrada). S'usa per a la camera un ESP32cam i per al microcontrolador un ESP32. S'utilitza Arduino per a l'entorn de desenrotllament, MATLAB per a realitzar cálculs i proves de funcionament i SolidWorks per a dissenyar les peces en 3D. Quant al llenguatge de programació és C++ per a Arduino i llenguatge M per a MATLAB.

Paraules clau: grau en Electrònica Industrial i Automàtica, TFG, Arduino, impressió 3D, braç robòtic, cinta transportadora, ESP32cam, ESP32, Matlab, 6 graus de llibertat, visió artificial.

Abstract:

This end-of-degree project is about 3D designing, printing, and mounting a 6-degree-of-freedom robotic arm that has a length of about 50 cm stretched. The arm uses servo motors controlled with a PCA9685 module. The robot configuration is angular.

Also, a conveyor belt is designed that has a camera and an infrared sensor which detects the piece. A small piece runs through the tape. The sensor detects the position of the part and when it is centered in the plane of the camera, stops the tape to take a photograph. The image is preprocessed and the characteristics of the piece (orientation and center of mass) are obtained. Later the tape advances so that, finally, the robotic arm takes it and deposits it in the corresponding place according to the type of piece (rectangular or square). An ESP32cam is used for the camera and an ESP32 for the microcontroller. I use Arduino for the development environment and MATLAB to perform calculations and test runs. To design the 3D parts, I use SolidWorks. As for the programming language, it is C++ for Arduino and M language for MATLAB.

Keywords: Degree in Industrial and Automatic Electronics, TFG, Arduino, 3D printing, robotic arm, conveyor belt, ESP32cam, ESP32, Matlab, 6 gdl, machine vision.

Agradecimientos:

Hoy finaliza una etapa de mi vida tanto de aprendizaje como de crecimiento personal. Por ello quiero mostrar mi agradecimiento a todas aquellas personas que me han ayudado y apoyado a lo largo de mis estudios de grado.

A mis tutores, Angel Valera y Carlos Ricolfe que me han guiado, animado y aconsejado durante la realización de este proyecto. Gracias por el tiempo que me habéis dedicado, con vosotros ha sido fácil continuar aprendiendo.

Además, me gustaría dar las gracias a mis compañeros de grado con los que poco a poco ha ido surgiendo una gran amistad. Gracias por compartir estos cuatro años conmigo y por haber estado siempre ahí, tanto en los buenos como en los malos momentos.

También deseo agradecer a mis padres el apoyo moral y los consejos que me han dado a lo largo de toda mi etapa académica. Gracias por la confianza que habéis depositado siempre en mí.

Índice de contenido

Índice de ilustraciones

Índice de tablas

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Documento número 1:

MEMORIA TÉCNICA

DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK & PLACE.

TRABAJO FINAL DEL

Grado en Ingeniería Electrónica Industrial y Automática

REALIZADO POR

Jorge Doménech Jara

TUTORIZADO POR

Ángel Valera Fernández

Carlos Ricolfe Viala

CURSO ACADÉMICO: 2019/2020

1.1 Objetivos, Motivación y Justificación.

El motivo de la realización de este proyecto es un encargo del departamento de Ingeniería de Sistemas y Automática (DISA) de la UPV, Camí de Vera, s/n. Este departamento docente está especializado en los campos de la automatización, el control y optimización de procesos, la robótica y visión artificial, la simulación de sistemas dinámicos, la fabricación flexible...

El departamento solicita el diseño, construcción e implementación de un brazo robótico de 6 grados de libertad. El brazo debe coger una pieza que circula a través de una cinta en movimiento y depositarla en un lugar prefijado. Además, se utilizan dos piezas de distinta forma y el brazo debe ser capaz de diferenciarlas y depositarlas en lugares diferentes.

La finalidad del proyecto es meramente educativa pues se trata de crear una maqueta a escala reducida para exponerla en un laboratorio.

A lo largo del diseño y construcción del brazo robótico me he ido encontrando con pequeños problemas a los que he ido buscando soluciones. También he aprendido entre otras cosas a programar una cámara de fotos y modelar en 3D. Por todo ello, la realización de este trabajo me ha ayudado tanto a adquirir nuevos conocimientos como a afianzar y poner en práctica los ya adquiridos en el grado en Ingeniería Electrónica Industrial y Automática.

1.2 Desarrollo Teórico.

1.2.1 Evolución de la robótica y de la visión artificial.

Evolución de la robótica.

La primera pregunta que nos hacemos es cuándo surgen los términos robot y robótica.

En 1917 el checoslovaco Karel Capek escribió la obra Rossum’s Universal Robots. En ella utilizó por primera vez el vocablo “Robota” que se tradujo al inglés como Robot. Su significado es servidumbre o trabajo forzado. En la obra el protagonista crea criaturas mecánicas con forma humana para utilizarlas como mano de obra barata.

Sin embargo, el término robótica no se empezó a emplear hasta 1939. Se le atribuye al escritor de ciencia ficción Isaac Asimov que la utilizó en varias narrativas relativas a robots. Para él un robot es una máquina que cumple con lo que llamó las Tres Leyes de la Robótica: un robot nunca puede hacer daño a un ser humano, siempre obedece las órdenes del ser humano y un robot protege su propia existencia siempre que no entre en conflicto con las leyes anteriores.

La robótica tal y como la conocemos hoy en día no aparece hasta el siglo XX, pero ya desde la prehistoria el hombre ha tenido afán por inventar herramientas que hicieran más fácil la vida cotidiana.

Ya en la época de los antiguos egipcios, los sacerdotes fabricaron los primeros brazos mecánicos que unieron a las estatuas de sus dioses. El motivo era puramente religioso.

A lo largo de la historia siempre han habido genios que se han adelantado a su época. Una muestra de ello son los griegos que diseñaron estatuas con movimiento utilizando sistemas hidráulicos. Entre ellos se encuentra el filósofo griego Archytas quien construyó artefactos dotados de movimiento e impulsados por vapor de agua a los que describió con el término "Automatón". También se puede mencionar al ingeniero Herón de Alejandría quien diseñó máquinas impulsadas por presión de aire, vapor y agua.

Ya en el Renacimiento, nos encontramos con el genio Leonardo Da Vinci. En cuanto a su ingenio robótico podemos destacar la creación de armaduras de caballeros con forma humana a las que dotó de movimiento mediante un sistema de poleas.

A partir del siglo XVII, en Europa se empezaron a construir muñecos mecánicos. Pero fue un siglo más tarde cuando el ingeniero francés Jacques Vauncanson diseñó muñecos de tamaño humano que eran capaces de tocar instrumentos. En 1805 Henri Maillardert diseñó una muñeca capaz de hacer dibujos utilizando levas como programas para que pudiera realizar el proceso de escribir y dibujar.

Con la revolución industrial se buscan aspectos más prácticos, construyendo dispositivos automáticos que ayuden o sustituyan al hombre en sus tareas. Se diseñan máquinas robotizadas a las que se introducen distintos programas dependiendo del trabajo que se quiere que realicen. Es el caso de Joseph Marie Jacquard que dotó a los telares de la época de una programación que se basaba en tarjetas perforadas, consiguiendo de esta forma la primera máquina programada de la historia. Esto supuso un gran avance para la industria textil. A partir de estos conocimientos, Charles Babbage diseñó la Máquina Diferencial, primera calculadora mecánica que realizaba cálculos automáticos de tablas de logaritmos. Con estos inventos ya se empezó a dotar a las máquinas de cierta "inteligencia".

En 1948 Grey Walter diseñó las tortugas de Bristol que eran robots móviles dotados de dos ruedas motrices y un foto-tubo que hacía las veces de ojo. Para ello utilizó detectores de contacto, válvulas y sensores de luz. Estos robots eran atraídos por la luz y con ello surge el concepto comportamiento. Hasta este momento los robots no tenían un sistema de control propio.

En 1954 el ingeniero George Devol concibe la idea de diseñar un dispositivo de transferencia programada de artículos. Años más tarde, junto al ingeniero estadounidense Joseph Engelberger funda Unimation que fue la primera empresa dedicada a la fabricación de robots. Devon inventó lo que se considera el primer robot industrial programable de la historia, el Unimate. Su función era levantar y apilar grandes piezas de metal caliente. En 1960 consiguen un contrato con Generals Motors para implantar el Unimate en una de sus plantas de producción. Continuó trabajando en él hasta llegar a un prototipo mejorado, el robot PUMA que era capaz de mover un objeto y colocarlo en cualquier orientación en el lugar deseado. Su concepto básico multiarticulado es la base de la mayoría de los robots actuales.

Ilustración 1. Robot Unimate. Uno de los primeros brazos robóticos industriales.

En la actualidad, la robótica se ha hecho imprescindible en multitud de sectores entre los que podemos citar la industria, minería, medicina, electrónica o ingeniería de control.

Los robots se usan en plantas de manufactura, montaje y embalaje, transporte, exploraciones en la Tierra y en el espacio, cirugía, armamento, investigación en laboratorios…

Otras aplicaciones incluyen las tareas de búsqueda y rescate de personas, automatización de laboratorios, del fondo oceánico, misiones espaciales y un largo etcétera.

Su implantación tiene múltiples y variadas ventajas: ha terminado con las tareas repetitivas y tediosas de los trabajadores, ha optimizado la productividad y realizan trabajos peligrosos como la limpieza de residuos tóxicos, la localización de minas terrestres o desactivar bombas. Otra ventaja es que poseen una gran precisión casi imposible de realizar por el hombre como en el caso de la cirugía invasiva mínima o la fabricación de elementos electrónicos.

Evolución de la visión artificial.

La visión artificial o visión por computador es un campo de la inteligencia artificial que utilizando ciertas técnicas permite obtener, procesar y analizar la información que se obtiene a través de imágenes digitales. Se centra en la aplicación de sistemas de análisis de imagen y cámaras de visión.

La evolución de la visión artificial está muy relacionada tanto con las cámaras fotográficas como con las imágenes telescópicas y las radiografías obtenidas a lo largo del siglo XIX. En 1838 el químico francés Daguerre realizó una fotografía y para ello utilizó una placa fotográfica que reveló con vapor de mercurio y la fijó con trisulfato de sodio.

Ya en la década de los 50 la fotografía a color, los barredores multiespectrales y el radar permitieron aumentar las zonas del espectro electromagnético para la prospección.

Pero hasta los años 60 no se empezó a utilizar para el guiado de robots, manipulación y control de calidad y automatización de sistemas. En 1961 el científico Larry Roberts realizó un trabajo que marcó el inicio de la visión artificial. Creó un programa en el cual un robot

podía "ver" una estructura de bloques sobre una mesa, analizar su contenido y reproducirlo. De esta forma demostró que la imagen que envió la cámara al ordenador había sido procesada por él.

La visión artificial recibió su mayor impulso en la década de los 80 cuando se crearon procesadores mucho más rápidos y sofisticados. Desde este momento se comenzó a investigar cómo se podía usar la visión artificial para tomar imágenes de forma automatizada, extraer las características visuales de objetos e interpretar los datos mediante un software específico. En esta época se produjo el desarrollo machine learning y el deep learning que unido a la visión artificial abrió un amplio campo de nuevas aplicaciones y se pudo dotar a los sistemas automatizados de mayor inteligencia y capacidad de procesamiento.

Las técnicas que se han utilizado en visión por computador han evolucionado en las últimas décadas. Los primeros sistemas se basaban en imágenes binarias que se procesaban en bloques o píxeles. Más tarde se desarrollaron nuevos algoritmos con los que se podía reconocer el contorno de los objetos y su posición, pero aún no podíran operar en distintos tipos de iluminación. Esto se solucionó más tarde con la introducción de los sistemas de intensidad de gris. Actualmente la visión artificial ha ayudado bastante a aumentar la autonomía en robótica.

Hoy en día se aplica en múltiples sectores como el químico, alimentario, sanitario o industrial.

En el sector industrial podemos destacar el seguimiento de objetos, la localización y reconocimiento de objetos por sus características, la obtención de modelos tridimensionales o la creación de modelos y patrones para dar respuestas automatizadas mediante los archivos de imágenes digitales.

Otro de los campos es el de la medicina en el cual el procesamiento de imágenes médicas ayuda a detectar tumores o medir las dimensiones de un órgano ayudando con ello a diagnosticar a un paciente.

Actualmente también se utiliza para realizar controles de calidad de productos, identificación de especies, procesos de control (robots industriales), vigilancia visual o conteo de personas, modelado topográfico, análisis de imágenes médicas o guías de misiles.

1.2.2 Modelo cinemático de robots.

Articulaciones y configuraciones de brazos robóticos.

En esencia los brazos robóticos industriales están formados por eslabones que están unidos mediante articulaciones. A continuación, se detallan los tipos de articulaciones:

- Articulación de revolución (R): el movimiento es de rotación alrededor de un eje y posee un grado de libertad.
- Articulación prismática (P): se diferencia de la anterior en que el movimiento es de tipo lineal y a lo largo de un eje.

- Articulación cilíndrica: como se puede apreciar en la imagen, es una combinación de las dos articulaciones anteriores y posee dos grados de libertad.
- Por último la articulación esférica o rótula: es la articulación que más grados de libertad tiene, en concreto tres.

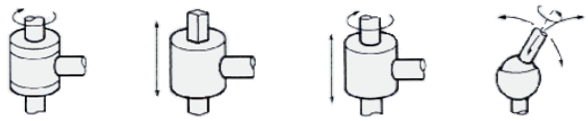


Ilustración 2. Tipos de articulaciones de brazos robóticos.

Según como se combinen las distintas articulaciones en un brazo robótico, se pueden obtener las siguientes configuraciones:

- Configuración cartesiana: posee tres articulaciones prismáticas y tres grados de libertad pudiendo moverse en las tres coordenadas espaciales.
- Configuración cilíndrica: formada por dos articulaciones prismáticas y una de revolución por lo que tiene tres grados de libertad.
- Configuración angular o brazo articulado: dispone de una articulación de revolución y dos angulares.
- Configuración o robot SCARA: posee dos articulaciones rotacionales y una prismática.
- Configuración polar: formada por dos articulaciones angulares y una prismática.

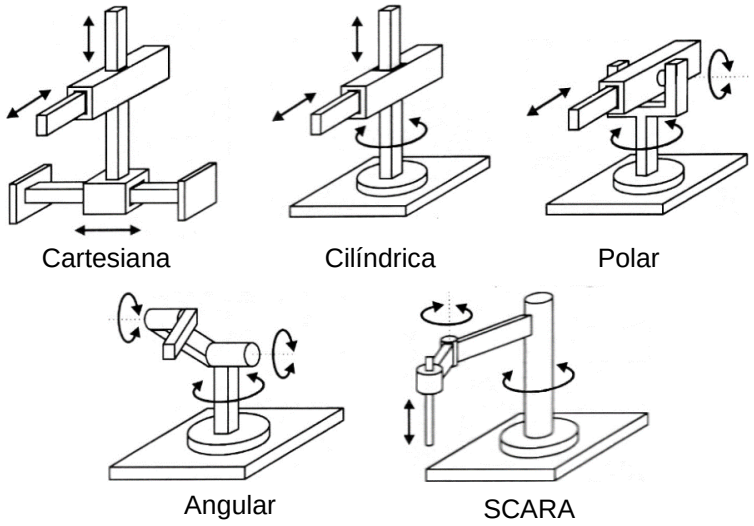


Ilustración 3. Configuraciones de brazos robóticos.

En este proyecto se ha utilizado la configuración angular y se han usado seis articulaciones de revolución.

Matriz de transformación homogénea:

La matriz de transformación ${}^{i-1}A_i$ representa la posición y orientación relativa entre los sistemas de coordenadas asociados a dos eslabones consecutivos del robot. La representación total del robot se realiza encadenando transformaciones:

$$^0A_3 = {}^0A_1 \cdot {}^1A_2 \cdot {}^2A_3$$

$$T = {}^0A_6 = {}^0A_1 \cdot {}^1A_2 \cdot {}^2A_3 \cdot {}^3A_4 \cdot {}^4A_5 \cdot {}^5A_6$$

La matriz de transformación homogénea representa la transformación de un vector de coordenadas homogéneas de un sistema de coordenadas a otro. Está formada por 4 filas y 4 columnas.

$$T = \begin{bmatrix} \vec{R}_{3x3} & \vec{p}_{3x1} \\ \vec{f}_{1x3} & \vec{w}_{1x1} \end{bmatrix} = \begin{bmatrix} Rotación & Traslación \\ Perspectiva & Escalado \end{bmatrix}$$

En nuestro caso la submatriz f_{1x3}, que representa una transformación de perspectiva, es nula mientras que la submatriz de escalado global w_{1x1} es la unidad. De esta forma se obtiene la siguiente matriz:

$$T = \begin{bmatrix} \vec{R}_{3x3} & \vec{p}_{3x1} \\ \vec{0}_{1x3} & 1 \end{bmatrix} = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Donde T representa la orientación y posición de un sistema OUVW rotado y trasladado con respecto al sistema de referencia OXYZ.

Si se hace la inversa a la matriz anterior se obtiene la siguiente matriz:

$$T^{-1} = \begin{bmatrix} n_x & n_y & n_z & -n^T p_x \\ o_x & o_y & o_z & -o^T p_y \\ a_x & a_y & a_z & -a^T p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Método Denavit-Hartenberg.

Este método se ha utilizado para obtener la cinemática directa e inversa del brazo robótico.

Es un modelo sistemático usado para situar los sistemas de coordenadas asociados a cada eslabón y obtener la cadena cinemática del robot.

El sistema de coordenadas SC_i se define a partir del sistema de coordenadas anterior, SC_{i-1}, y de los ejes de articulación i e i+1. Permite el paso de un eslabón al siguiente mediante 4 transformaciones básicas que dependen exclusivamente de las características constructivas del robot. Dichas transformaciones que relacionan el sistema de referencia del elemento i con el del elemento i-1 son las siguientes:

- Rotación θ_i alrededor del eje Z_{i-1}
- Traslación d_i a lo largo del eje Z_{i-1}
- Traslación a_i a lo largo del eje X_i
- Rotación α_i alrededor del eje X_i

A continuación, se expresa la matriz de transformación de forma genérica:

$$^{i-1}A_i = T(z,\theta_i)\cdot T(0,0,d_i)\cdot T(a_i,0,0)\cdot T(x,\alpha_i) =$$

$$= \begin{bmatrix} Cos(\theta_i) & -Sen(\theta_i) & 0 & 0 \\ Sen(\theta_i) & Cos(\theta_i) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & Cos(\alpha_i) & -Sen(\alpha_i) & 0 \\ 0 & Ses(\alpha_i) & Cos(\alpha_i) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} =$$

$$= \begin{bmatrix} Cos(\theta_i) & -Cos(\alpha_i)\cdot Sen(\theta_i) & Sen(\alpha_i)\cdot Sen(\theta_i) & a_i\cdot Cos(\theta_i) \\ Sen(\theta_i) & Cos(\alpha_i)\cdot Cos(\theta_i) & -Sen(\alpha_i)\cdot Cos(\theta_i) & a_i\cdot Sen(\theta_i) \\ 0 & Sen(\alpha_i) & Cos(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Los pasos genéricos para poner los sistemas de referencia de cada eslabón y obtener los parámetros característicos de cada robot son los siguientes:

1. Numerar los eslabones empezando por el 1 (primer eslabón móvil) y posteriormente numerar y dibujar líneas donde están los ejes de las articulaciones empezando por la 1. La base fija se numera como eslabón 0.

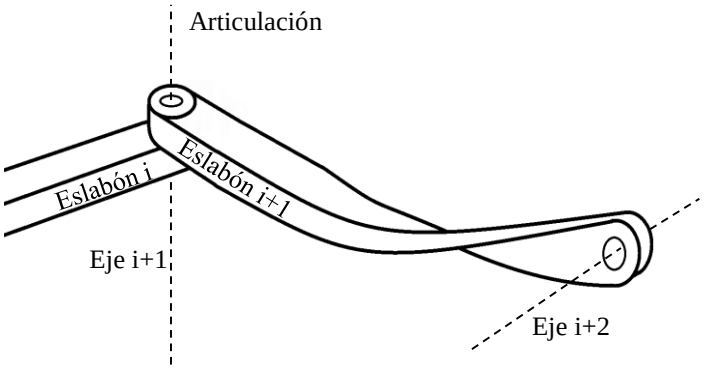


Ilustración 4. Enumeración de ejes y articulaciones.

2. El origen O_i siempre se sitúa en Z_i pero la posición exacta depende del eje Z_{i-1}. Si existe intersección entre Z_{i-1} y Z_i se sitúa en la intersección, si no existe la intersección se traza una normal a los dos ejes y se sitúa el origen en la intersección entre la normal y Z_i. En el caso de que Z_{i-1} y Z_i sean paralelos o coincidentes, el origen se sitúa sobre Z_i. El origen de la base fija, O_0, se puede situar en cualquier parte del eje Z_0.

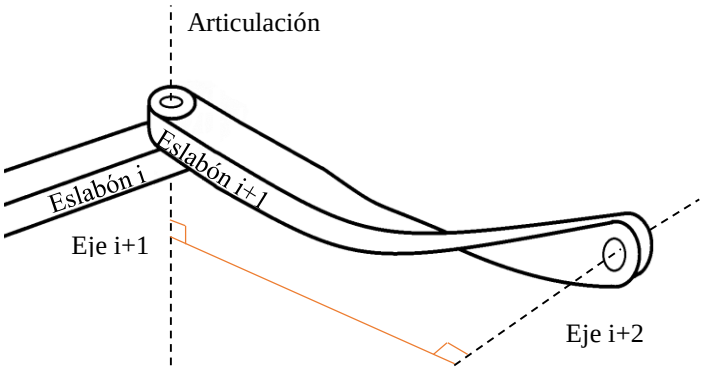


Ilustración 5. Colocación de orígenes de sistemas de referencia de cada eslabón.

3. Poner el eje Z_i sobre el eje de la articulación i+1 (excepto para $i = n - 1$).

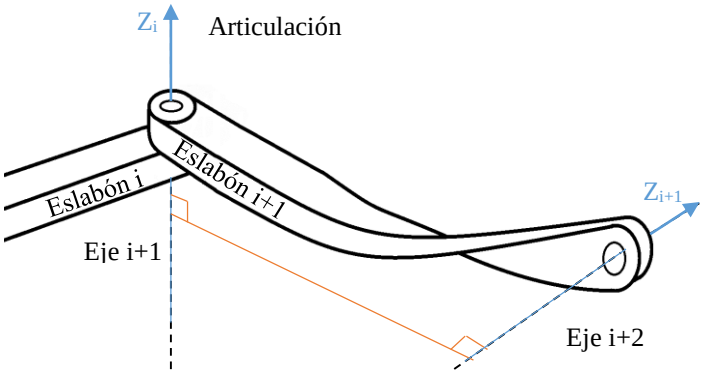


Ilustración 6. Colocación de los ejes Z_i.

4. Trazar el eje X_i perpendicular al plano que forman los ejes Z_{i-1} y Z_i. El eje trazado debe pasar por la intersección de los ejes Z_{i-1} y Z_i pero si no intersectan debe estar sobre su perpendicular común.

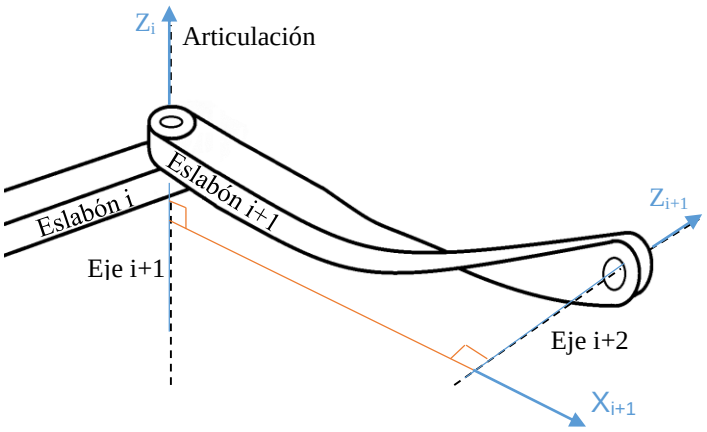


Ilustración 7. Colocación de los ejes X_i.

5. Colocar el eje Y_i de forma que el sistema de referencia sea dextrógiro.
6. El sistema de coordenas SC_n con origen O_n asociado al último elemento, puede tener su origen en cualquier parte de éste siempre que la dirección de Z_n sea equivalente a la de Z_{n-1}.

A continuación se explica cómo se obtienen los parámetros de Denavit-Hartenberg:

- θ_i: ángulo de X_{i-1} a X_i medido sobre el eje Z_{i-1} para que X_{i-1} y X_i queden paralelos (regla mano derecha).
- d_i: distancia de X_{i-1} a X_i medida a lo largo del eje Z_{i-1}
- a_i: distancia de Z_{i-1} a Z_i medida a lo largo del eje X_i
- α_i: ángulo de Z_{i-1} a Z_i medido sobre el eje X_i (regla mano derecha).

La cinemática directa.

Una vez conocidos los valores de las articulaciones y las longitudes de los eslabones, se utiliza el cálculo de la cinemática directa para poder determinar la posición y orientación del extremo del robot. Todo ello con respecto a un sistema de coordenadas de referencia.

Existen dos formas de resolver la cinemática directa.

Utilizando las matrices de transformación homogénea descritas anteriormente o bien utilizando el método geométrico en el caso de que el brazo robótico presente pocos grados de libertad.

En este proyecto se va a resolver la cinemática directa mediante las matrices de transformación homogénea. Para ello en primer lugar se realiza el algoritmo de Denavit-Hartenberg, situando los ejes de referencia de cada eslabón y obteniendo los parámetros $\theta_i, d_i, a_i \; y \; \alpha_i$.

A continuación, a partir de los ángulos $q_1, q_2, q_3, q_4, q_5 \; y \; q_6$ se calculan las matrices de transformación ${}^{i-1}A_i$ y se calcula la matriz de transformación que relaciona el sistema de la base con el del extremo del robot:

$$T = {}^0A_6 = {}^0A_1 \cdot {}^1A_2 \cdot {}^2A_3 \cdot {}^3A_4 \cdot {}^4A_5 \cdot {}^5A_6 = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

La cinemática inversa.

Una vez conocidos los valores de la posición y orientación del extremo del robot, se utiliza la cinemática inversa para determinar los ángulos de las articulaciones con respecto a un sistema de coordenadas de referencia. Al resolverlo matemáticamente se obtienen varias soluciones válidas, se elige una cualquiera.

Para resolver la cinemática inversa de un brazo robótico de seis ejes, se suelen desacoplar por un lado los tres primeros para ajustar la posición y por otro los tres últimos para ajustar la orientación. El desacoplo cinemático se realiza de la siguiente forma:

$$\vec{p}_m = \vec{p}_{TCP} - d_6 \cdot {}^0\vec{a}_6, donde \; {}^0\vec{a}_6 \; y \; \vec{p}_{TCP} \; procede \; de \; {}^0A_6 = \begin{bmatrix} \vec{n} & \vec{o} & \vec{a} & \vec{p} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

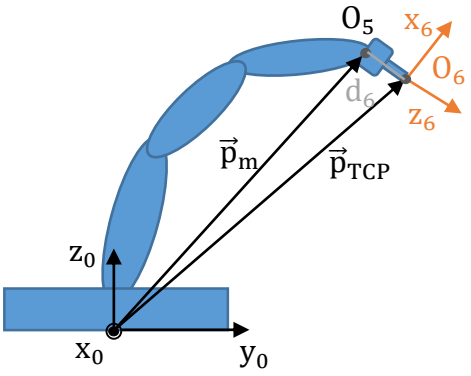


Ilustración 8. Desacople cinemático.

Para obtener los ángulos de las tres primeras articulaciones se puede utilizar el método geométrico o las matrices homogéneas. En el proyecto se utiliza el método geométrico.

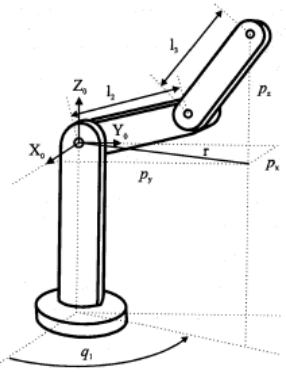


Ilustración 9. Brazo robótico desacoplado.

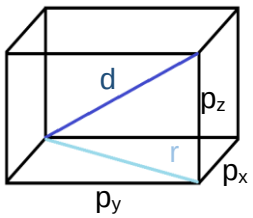
El primer ángulo se calcula de forma inmediata:

$$q_1 = arctg\left(\frac{p_y}{p_x}\right)$$

Para calcular q_3 se utiliza el teorema del coseno y el de Pitágoras:

$$r = \sqrt{{p_x}^2 + {p_y}^2}$$

$$d = \sqrt{r^2 + {p_z}^2} = \sqrt{\left(\sqrt{{p_x}^2 + {p_y}^2}\right)^2 + {p_z}^2} = \sqrt{{p_x}^2 + {p_y}^2 + {p_z}^2}$$

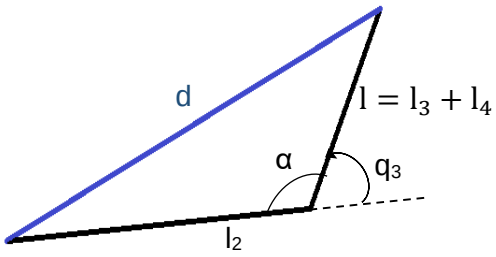


$$d^2 = {l_2}^2 + l^2 - 2 \cdot l_2 \cdot l \cdot \cos(180° - q_3) = {l_2}^2 + l^2 + 2 \cdot l_2 \cdot l \cdot \cos(q_3) \Rightarrow$$

$$\Rightarrow \left(\sqrt{{p_x}^2 + {p_y}^2 + {p_z}^2}\right)^2 = {l_2}^2 + l^2 + 2 \cdot l_2 \cdot l \cdot \cos(q_3) \Rightarrow {p_x}^2 + {p_y}^2 + {p_z}^2 =$$

$$= {l_2}^2 + l^2 + 2 \cdot l_2 \cdot l \cdot \cos(q_3), donde\ p_x = P_x, p_{xy} = P_y\ y\ p_z = P_z - l_1$$

$$q_3 = arcos\left(\frac{{p_x}^2 + {p_y}^2 + {p_z}^2 - {l_2}^2 - l^2}{2 \cdot l_2 \cdot l}\right)$$



A continuación, se calcula q_2:

$$q_2 = \beta - \alpha$$

$$\beta = arctg\left(\frac{p_z}{r}\right) = arctg\left(\frac{p_z}{\pm\sqrt{{p_x}^2 + {p_y}^2}}\right)$$

$$\alpha = arctg\left(\frac{l \cdot sen(q_3)}{l_2 + l \cdot \cos(q_3)}\right)$$

$$q_2 = \beta - \alpha = arctg\left(\frac{p_z}{\pm\sqrt{{p_x}^2 + {p_y}^2}}\right) - arctg\left(\frac{l \cdot sen(q_3)}{l_2 + l \cdot \cos(q_3)}\right)$$

Tal y como se puede ver en la siguiente imagen q_2 posee dos soluciones, en una el codo del brazo robótico está posicionado en la parte alta y en la otra se posiciona en la parte baja:

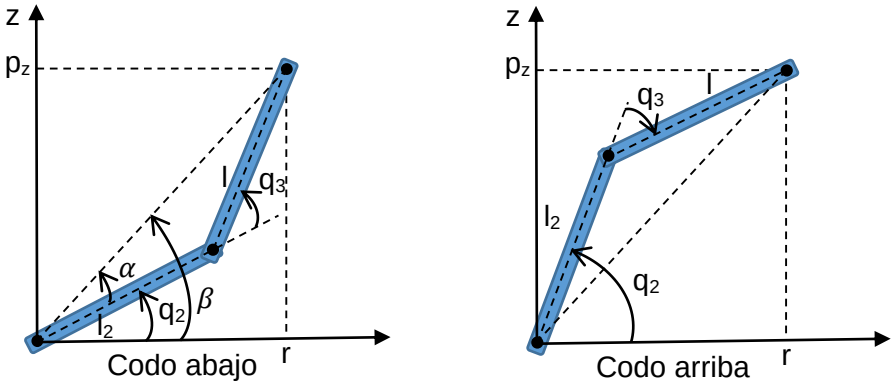


Ilustración 10. Posiciones para el ángulo de las articulaciones 2 y 3.

Para obtener los ángulos de las tres últimas articulaciones se utilizan las matrices homogéneas.

Primero se realiza el algoritmo del método Denavit-Hartenberg situando los ejes de referencia de cada eslabón y obteniendo los parámetros $\theta_i, d_i, a_i \; y \; \alpha_i$.

A continuación se calculan todas las matrices de transformación, ${}^{i-1}A_i$, y la matriz ${}^{0}R_3$ de la siguiente forma:

$${}^{0}R_3 = {}^{0}R_1 \cdot {}^{1}R_2 \cdot {}^{2}R_3$$

Posteriormente se obtiene la inversa de la matriz de rotación de la articulación 3 respecto la base, ${}^{0}R_{3}{}^{-1}$.

$$ {}^{0}R_{3}{}^{-1} = {}^{0}R_{3}{}^{T} $$

Los ángulos de las articulaciones se obtienen a partir de la orientación deseada, $\vec{n}, \vec{o}\ y\ \vec{a}$:

$$ {}^{0}R_{6} = {}^{0}R_{3} \cdot {}^{3}R_{6} $$

$$ {}^{3}R_{6} = r_{ij} = {}^{0}R_{3}{}^{-1} \cdot {}^{0}R_{6} = {}^{0}R_{3}{}^{-1} \cdot [\vec{n} \quad \vec{o} \quad \vec{a}] $$

Por último se resuelve el sistema de ecuaciones obteniendo los siguientes ángulos:

$$ q_4 = arctg\left(\frac{r_{23}}{r_{13}}\right) $$

$$ q_5 = arctg(r_{33}) $$

$$ q_6 = arctg\left(\frac{r_{32}}{-r_{31}}\right) $$

1.2.3 Detección de regiones mediante visión artificial.

El principal problema de obtener imágenes es que a mayor resolución la cantidad de información que se obtiene también es mayor. Esto origina que el procesador trabaje saturado y el tiempo de respuesta sea mayor.

Para reducir o quitar la información se puede segmentar la imagen. Esta segmentación consiste en identificar regiones homogéneas que presenten objetos o partes. Dichas regiones deben poder ser diferenciables de las regiones adyacentes y del fondo.

A su vez los píxeles de una región deben estar conectados entre sí, tener unos bordes definidos y diferenciables del fondo y unos valores parecidos de color, textura, nivel de gris…

Existen múltiples algoritmos para segmentar imágenes. En el proyecto se utiliza el de umbralización de regiones (Threshold). Este algoritmo consiste en aplicar una condición a cada píxel de la imagen, si la cumple se le asigna un valor determinado y si no otro distinto. Este método es muy simple y eficiente en términos computacionales.

Existen otras variantes como Threshold Band que se diferencia en que son dos condiciones y se selecciona la banda intermedia. Es útil para separar objetos de otros que contrasten con el fondo.

Para determinar los valores de umbral se utiliza el histograma. A partir de los umbrales del histograma un objeto se puede segmentar en color o bien transformando la imagen a escala de grises.

A continuación, se pone un ejemplo de segmentación:

Se va a segmentar la cartulina rosa. En nuestro caso se va a transformar la imagen a escala de grises ya que es más sencillo de apreciar los objetos en ella.

Ilustración 11. Cartulinas originales y a escala de grises.

Tras realizar el escalado a grises se representan los valores de cada píxel en función de la cantidad total, obteniendo el siguiente histograma:

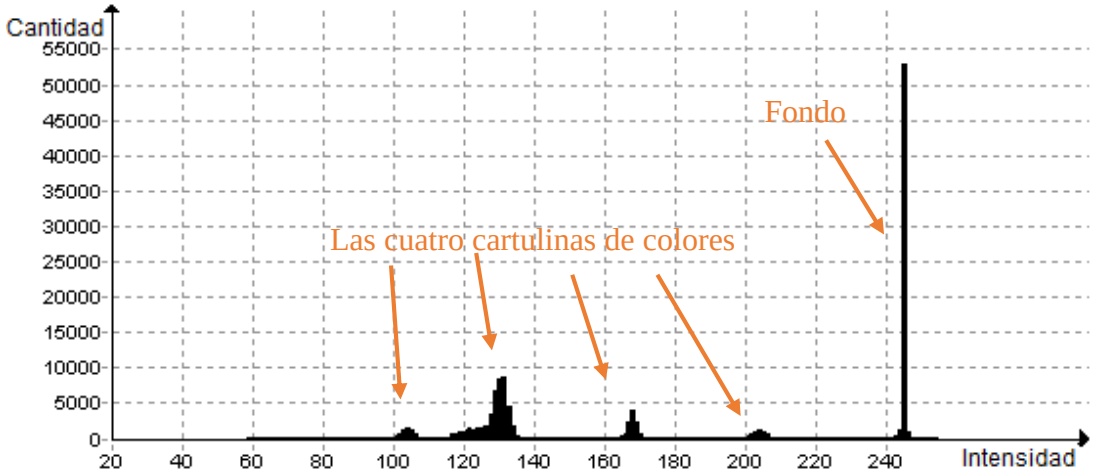


Gráfico 1. Histograma de la ilustración gris anterior.

De este histograma se puede extraer que los 5 picos que aparecen están formados por las 4 cartulinas más el fondo.

Como se puede apreciar en el histograma ampliado se pueden diferenciar claramente los valores límite o umbrales de cada objeto (señalados con una flecha).

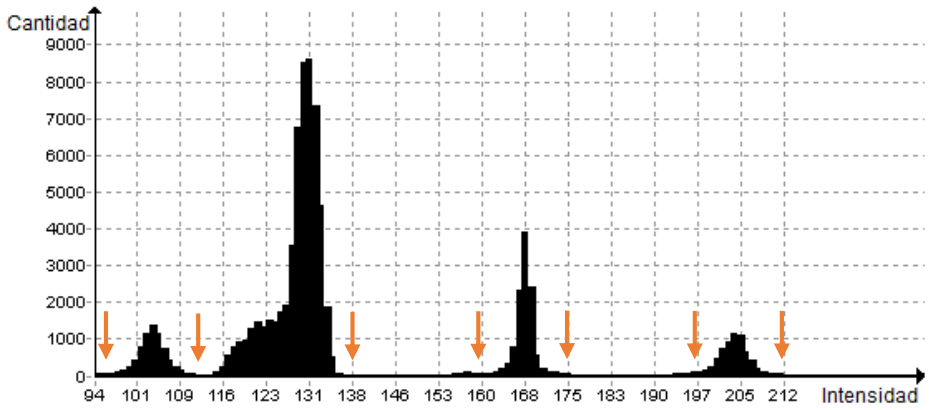


Gráfico 2. Histograma ampliado de la ilustración gris anterior.

Se realiza un Threshold Band para obtener la cartulina rosa y poder eliminar el resto de información. Se escogen como umbral mínimo el 94 y como máximo el 113, obteniendo la siguiente imagen segmentada.

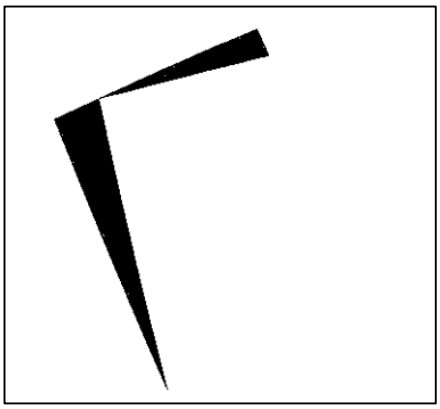


Ilustración 12. Imagen segmentada mediante Threshold Band con umbrales 94 y 113.

1.3 Trabajo Práctico.

1.3.1 Estudio de necesidades, factores a considerar: limitaciones y condicionantes.

A continuación, se detallan las especificaciones del encargo y la normativa tenida en cuenta en la redacción del proyecto.

1.3.1.1 Especificaciones del encargo.

El cliente nos encarga el diseño, desarrollo, programación e impresión 3D de un brazo robot de 6 grados de libertad. Su utilización es para aplicaciones de Pick and Place.

El proyecto debe tener un brazo robótico estirado de una longitud de 40 a 50 cm y una cinta de entre 50 y 60 cm. Debe estar construido con piezas ligeras para facilitar su transporte y tener un aspecto minimalista.

El brazo debe poder coger piezas de un tamaño de 4 x 2,5 cm y 2,5 x 2,5 cm y dejarlas en su correspondiente lugar en función de su forma.

El cliente pide que por seguridad las piezas no presenten aristas cortantes ni punzantes y el material debe ser plástico y no tóxico.

El proyecto debe estar alimentado por la red eléctrica del laboratorio y su mantenimiento debe ser mínimo.

El plazo de entrega del proyecto no debe exceder el mes de julio, si no se cumple tendrá una penalización del 10 % del coste total del proyecto. Dicho coste total no debe exceder de 8.000€.

En la última reunión mantenida con el cliente, éste manifiesta que se utilice nuestro criterio en los aspectos no definidos anteriormente.

1.3.1.2 Normativa tenida en cuenta en la redacción.

El presente diseño del producto se redacta cumpliendo la siguiente normativa:

- Real Decreto 186/2016, de 6 de mayo, por el que se regula la compatibilidad electromagnética de los equipos eléctricos y electrónicos. (BOE nº 113, de 10/05/2016). Un equipo debe tener capacidad para funcionar de forma satisfactoria en su entorno electromagnético sin introducir perturbaciones electromagnéticas intolerables para otros equipos en ese entorno.

- REGLAMENTO (CE) nº 1907/2006 DEL PARLAMENTO EUROPEO Y DEL CONSEJO de 18 de diciembre de 2006 relativo al registro, la evaluación, la autorización y la restricción de las sustancias y preparados químicos (REACH). (DOUE L 396, de 30/12/2006). Para la fabricación del producto se deben utilizar productos no infecciosos ni alérgicos. Por ello, para la creación de las piezas se utiliza acrilonitrilo butadieno estireno, ABS.

- UNE-EN 61558-1:2007, por el que regula la seguridad de los transformadores, fuentes de alimentación, bobinas de inductancia y productos análogos.

- ISO 80000, donde se regula la nomenclatura para la escritura de magnitudes. A la hora de redactar la unidades, prefijos y magnitudes se hace teniendo en cuenta esta norma.

1.3.1.3 Análisis de hojas de especificaciones.

La siguiente hoja de especificaciones corresponde al microcontrolador utilizado, el ESP32. En primer lugar, hay que destacar su voltaje de alimentación 3,6 V y el consumo de corriente media de 80 mA. Esta placa utiliza el núcleo ESP32-D0WDQ6. Las dimensiones para el diseño del soporte son 18 x 25,5 x 3,1 mm.

Categorías	Características	Especificaciones
Hardware	Tensión de alimentación	3,6 V
	Corriente de alimentación	80 mA
	Dimensiones	18 x 25,5 x 3,1 mm

Tabla 1. Especificaciones del ESP32.

A continuación, se muestran las características de cada pin del ESP32.

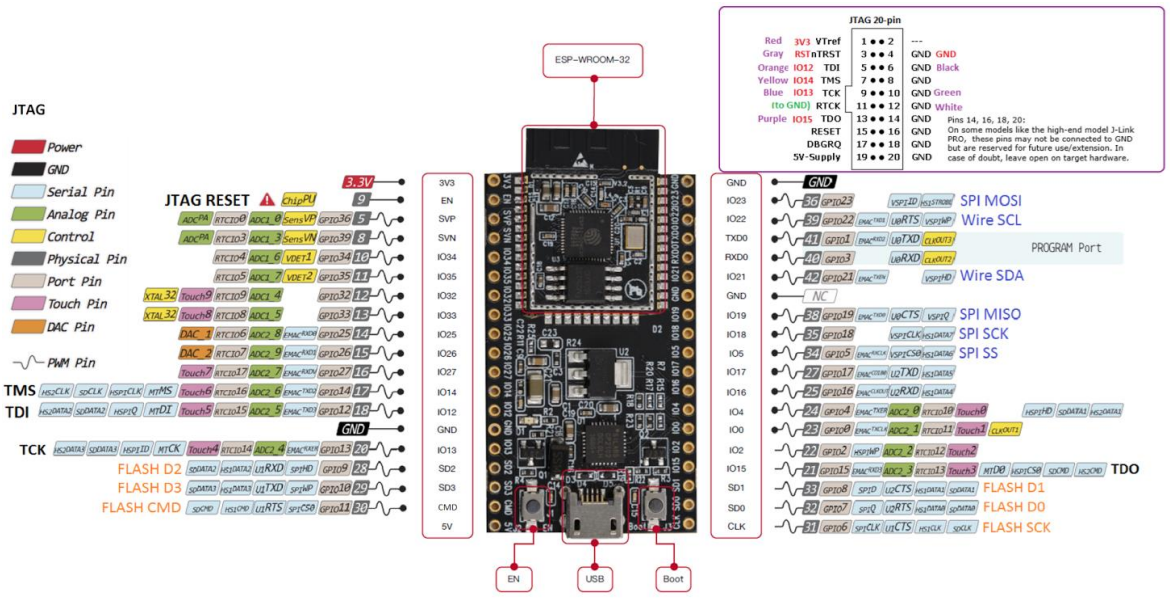


Ilustración 13. Características de los pines del ESP32.

La siguiente hoja de especificaciones corresponde al microcontrolador ESP32cam utilizado. Cabe destacar su voltaje de alimentación 5 V y el consumo de corriente de 20 mA. Esta placa utiliza el núcleo ESP32-D0WDQ6. Las dimensiones para el diseño del soporte son 40,5 x 27 x 4,5 mm.

Especificaciones	
Tensión de alimentación	5 V
Corriente de alimentación	20 mA
Formato de salida de la imagen	BMP
Dimensiones	40,5 x 27 x 4,5 mm

Tabla 2. Especificaciones del ESP32cam.

A continucaión, se muestran las características de cada pin del ESP32cam.

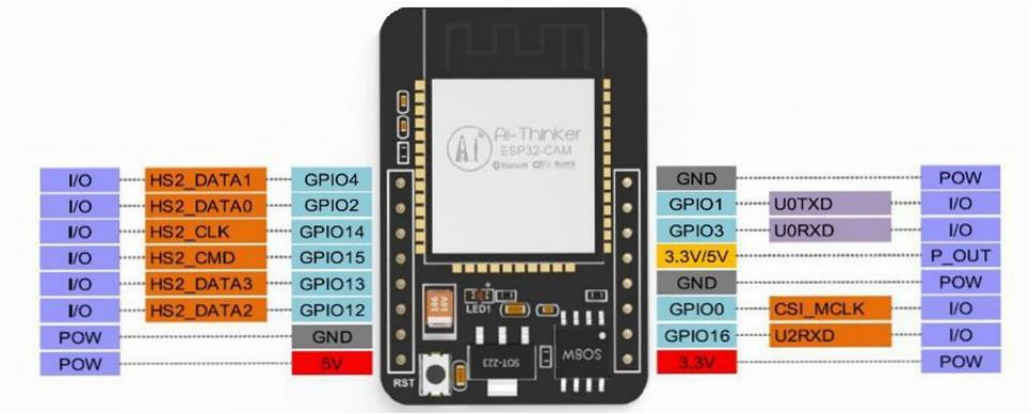


Ilustración 14. Características de los pines del ESP32cam.

En la siguiente tabla se muestran las especificaciones de los servomotores utilizados:

	Dimensiones (mm)	Par (kg·cm)	Tensión de alimentación (V)	Consumo de corriente (mA)	Masa (g)
SG90	22,2 x 11,8 x 31	1,8	5	20	9
MG996R	40,7 x 19,7 x 42,9	11	6	900	55
DS04-NFC	54 x 20 x 44	5,5	6	25	38

Tabla 3. Valores más representativos de los diferentes servomotores del proyecto.

La siguiente hoja de especificaciones corresponde al material acrilonitrilo butadieno estireno, ABS. Con relación a las propiedades mecánicas, la capacidad de alargamiento de este material hasta su rotura es de un 4,8 % y la de su alargamiento hasta su deformación es de un 3,5 %. En cuanto a sus propiedades térmicas, la temperatura de reblandecimiento tiene un valor mínimo de 97 ºC.

Propiedades mecánicas	Impresión 3D
Alargamiento a la deformación	2,5 %
Alargamiento a la rotura	4,8 %
Propiedades térmicas	Impresión 3D
Temperatura de reblandecimiento Vicat a 10 N	97 ºC

Tabla 4. Especificaciones del acrilonitrilo butadieno estireno, ABS.

A continuación, se muestra las especificaciones de la PCA9685.

Parámetro	Valores
Tensión de alimentación	2,2 – 5,5 V
Corriente de alimentación	6 – 10 mA

Tabla 5. Características de la PCA9685.

Por último se muestran las especificaciones del convertidor buck HW636:

- Tensión de alimentación: 6,5 - 60 V.
- Tensión de salida: 1,25 - 30 V.
- Eficiencia máxima: 97 %
- Dimensiones: 75,7 x 22,2 x 20 mm.

1.3.2 Planteamiento de soluciones alternativas y justificación de la solución adoptada.

- Piezas del brazo robótico y de la cinta:

Se contemplan tres posibilidades para fabricar las piezas.

  - Diseñar moldes para imprimirlas por inyección.
  - Diseñarlas para poder ser cortadas con láser, utilizando como material el aluminio o un metal similar.
  - Diseñarlas para imprimirlas en una impresora 3D, utilizando como material el plástico.

En el primer caso, el precio final no se ajusta al presupuesto exigido y en el segundo el peso de las piezas es mayor. Se ha optado por imprimir las piezas mediante impresión 3D y de esta forma se consigue abaratar el coste y disminuir su peso para que se ajuste a las especificaciones del cliente.

- Materiales de las piezas:

En cuanto a los materiales para imprimir las piezas existen principalmente el ABS y el PLA. Se opta por el primero porque presenta una mayor resistencia a la tracción y a la compresión. Además, está libre de alérgenos que es un requisito del cliente.

- Elección de los servomotores:

Para seleccionar los servomotores se han calculado las fuerzas a las que tienen que hacer frente. Los servomotores SG90 no son aptos para todas las articulaciones del brazo robótico puesto que el par mínimo que tienen que ejercer no es suficiente. Por ello se decide utilizar servomotores MG996R porque poseen mayor potencia y disponen de engranajes internos de metal.

- Elección de la cámara:

En el mercado existe una gran variedad de cámaras con distintos precios y prestaciones. Se decide utilizar el ESP32cam por su buena relación calidad precio y porque contiene todas las prestaciones requeridas por el proyecto: una resolución ajustable máxima de 740 x 680 píxeles, un pequeño tamaño y además es compatible con varios lenguajes de programación incluido el Arduino.

- Elección de la cinta transportadora:

  En un principio se plantea utilizar una cinta formada por pequeños eslabones de plástico o similar, pero es descartada por su complejidad de montaje y coste económico superior. En cuanto al material se decide usar la goma eva (etileto y acetato de vinilo) porque es más económica, sencilla de montar y posee una gran flexibilidad.

- Elección de la forma de alimentar el proyecto:

  Se puede utilizar una fuente de alimentación conectada directamente a la instalación eléctrica o bien utilizar una batería de ion litio. Debido a los requisitos del cliente se opta por alimentar el proyecto con la instalación eléctrica y se descarta la batería de ion litio.

- Elección del sensor de proximidad:

  Entre las opciones para detectar la pieza se puede usar un sensor de ultrasonidos, un sensor de infrarrojos o un sensor láser. El primero es descartado porque por posibles rebotes puede tomar falsas lecturas, el láser también se descarta porque pese a tener una mayor resolución su coste económico también es mayor. Se opta por el sensor de infrarrojos debido a que su resolución es suficiente y su precio menor.

1.3.3 Descripción y justificación detallada de la solución adoptada

El producto está desglosado en 5 subsistemas. A continuación, se describen los elementos de cada uno de ellos.

El primer subsistema es el electrónico y está formado por:

- La cámara que se encarga de tomar las imágenes, procesar y extraer la información útil.
- El microprocesador que gestiona todas las entradas y salidas y aplica la lógica necesaria.
- La fuente de alimentación suministra energía a los componentes desde una toma de luz.
- El convertidor BUCK encargado de reducir la tensión de la fuente de alimentación a una adecuada para alimentar al circuito.
- La PCA9685 coordina la comunicación entre el microprocesador y el conjunto de servomotores pudiendo suministrar una mayor potencia.
- El sensor infrarrojo encargado de detectar la pieza y enviar la señal al microprocesador para que actúe en consecuencia.
- El encoder tiene la misión de obtener una medida cuantitativa de forma indirecta del avance de la cinta.

El segundo subsistema es el eléctrico compuesto por el cableado de “baja tensión” que cablea el circuito interno y el de “alta tensión” que enlaza la fuente de alimentación con el suministro eléctrico del edificio. También lo forma una tira de LEDs encargada de iluminar de forma controlada el encuadre de la cámara.

El tercer subsistema es el de la fuerza motriz formado por:

- El Motor DS04-NFC de rotación continua que se encarga de hacer girar la cinta transportadora.
- Servomotores:
    - Los SG90 se usan para las articulaciones 4, 5 y 6 y para el accionamiento de la pinza en el efector final.
    - Los MG996R se utilizan para las articulaciones 1, 2 y 3 del brazo robótico.

El cuarto subsistema es el de materiales de plástico que están formados en su totalidad por ABS y posee los siguientes componentes:

- Piezas que coge el brazo robótico. Una con forma rectangular de 40 x 25 x 7 mm y la otra cuadrada de 25 x 25 x 7 mm (largo x ancho x alto).
- Los soportes del sensor infrarrojo del motor DS04-NFC, de la cinta, de la cámara, del ESP32 y de la PCA. Todos estos soportes se encargan de sujetar y fijar adecuadamente los distintos elementos al tablero de madera.
- Los ejes de la cinta que se encargan de hacer girar de forma sincronizada los engranajes de la cinta.
- La rueda con un patrón de agujeros simétricos usada junto al encoder para medir la distancia lineal de la cinta.
- Los eslabones del brazo robótico:
    - El eslabón 0 o base del robot que parte desde donde se apoya el brazo y hace de unión con el tablero de madera.
    - Los eslabones 1, 2, 3 y 4 que se encargan de sostener la pinza y darle mayor libertad de movimiento.
    - La pinza cuyo mecanismo está diseñado para poder coger el objeto que circula por la cinta.

Por último, el quinto subsistema está formado por:

- Los engranajes metálicos de aluminio utilizados para mover la cinta.
- Goma eva usada para tapar el lugar donde se realiza la captura de imágenes y para fabricar la cinta.
- Tabla de madera de 600 x 400 x 2,5 mm usada como base para montar el proyecto.
- Correas dentadas que van pegadas a la cinta y sincronizan los engranajes con dicha cinta.
- Canaleta de PVC para canalizar y tapar los cables mejorando su estética.
- La placa para impresión PCB usada para optimizar y fijar el espacio ocupado por los elementos electrónicos necesarios.

A continuación, se muestra el organigrama técnico del proyecto:

Ilustración 15. Organigrama técnico del proyecto.

Justificación detallada de los elementos o componentes de la solución adoptada.

A continuación, se detallan los cálculos realizados para la elección de los componentes:

- Selección de los servomotores: Se han calculado los pares que ejercen las diferentes fuerzas en cada articulación del brazo robótico. Abajo se muestran detalladamente todos los cálculos de una articulación.

  Dado el siguiente diagrama de fuerzas, donde d es la distancia entre articulaciones, m las masas de los elementos y F las fuerzas que ejerce la gravedad sobre las masas

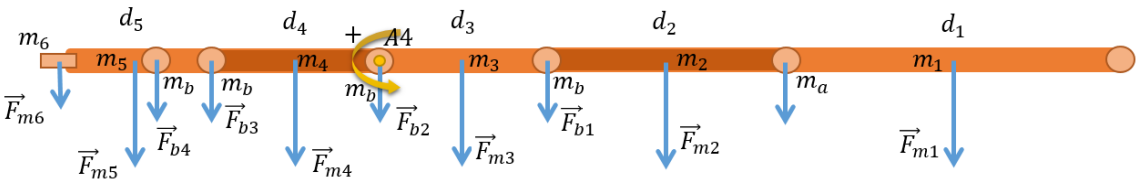


Ilustración 16. Diagrama de fuerzas de la articulación 4.

  junto con las longitudes de los eslabones del brazo robótico

Eslabones	Longitud (cm)
d_1	16,5
d_2	12,5
d_3	7,8
d_4	7,1
d_5	6,2
d_{total}	50,1

Tabla 6. Longitudes de cada eslabón.

  y las masas de cada elemento

Componentes		Masa (k)
Motores	m_a	0,055
	m_b	0,010
Eslabones	m_1	0,022
	m_2	0,019
	m_3	0,011
	m_4	0,011
	m_5	0,017
Pieza rectangular	m_6	0,005

Tabla 7. Masas de cada componente o elemento.

se calcula el momento generado en A4, que es el sumatorio de las masas de cada elemento multiplicado por la distancia al eje A4. Para calcular este momento solo se toman en consideración las masas que hay a la izquierda de A4.

$$M_{A4} = \sum_{i}^{n} (m_i \cdot d_{A4,i})$$

$$M_{A4} = m_4 \cdot \frac{d_4}{2} + m_5 \cdot \left(d_4 + \frac{d_5}{2}\right) + m_6 \cdot (d_4 + d_5) + m_b \cdot d_4 + m_b \cdot \left(d_4 + \frac{d_5}{10}\right) =$$

$$= 0{,}011 \cdot \frac{7{,}1}{2} + 0{,}017 \cdot \left(7{,}1 + \frac{6{,}2}{2}\right) + 0{,}005 \cdot (7{,}1 + 6{,}2) + 0{,}01 \cdot 7{,}1 +$$

$$+ 0{,}01 \cdot \left(7{,}1 + \frac{6{,}2}{10}\right) = 0{,}9615\ kg \cdot cm$$

En este caso para simplificar se ha calculado el módulo del momento de fuerzas en lugar del vector y se ha optado directamente por calcularlo en kg·cm.

El resultado del resto de articulaciones se muestra en la siguiente tabla:

Articulación	Par (kg·cm)
2	4,919
3	2,130
4	0,9615
5	0,4272

Tabla 8. Momento de fuerzas de cada articulación o par motor mínimo para soportar el peso del propio brazo robótico y la pieza en su extremo.

Como se puede apreciar en la tabla anterior, los pares para las articulaciones 4, 5 y extrapolando a la 6 es necesario usar un servomotor de un par cercano a 1 kg·cm. Dado este valor se ha optado por el SG90 que tiene un par de 1,8 kg·cm.

En cuanto al resto de las articulaciones no se tienen en cuenta las aceleraciones porque son muy pequeñas debido a la aceleración reducida del brazo. Es necesario un motor que pueda soportar un par de 5 kg·cm o más y por ello se escoge el servomotor MG996R que soporta un par de 11 kg·cm.

- En cuanto al diseño del convertidor, se busca en el mercado el que más se adapta a las siguientes restricciones:
  - El tipo de convertidor debe ser un DC - DC.
  - La tensión de entrada es de 12 V y coincidir con la tensión de salida de la fuente de alimentación.
  - La tensión de salida del convertidor debe estar dentro del rango de 5 y 6,5 V.

- La corriente máxima admisible es la calculada con la suma de la corriente de consumo máxima de cada elemento del circuito:

$$I_{máx,convertidor} = I_{DS04} + 4 \cdot I_{SG90} + 4 \cdot I_{MG996} + I_{cámara} +$$

$$+ I_{procesador} + I_{sensores} + I_{PCA} = 0{,}25 + 4 \cdot 0{,}20 + 4 \cdot 0{,}90 + 0{,}055 +$$

$$+ 0{,}08 + 0{,}085 + 0{,}01 = 4{,}975 \approx 5\ A$$

Una vez realizados los cálculos anteriores se opta por la utilización del convertidor HW636.

- Para el diseño de la fuente de alimentación se calcula el consumo de corriente máximo:

$$P_{LEDs} = 15\ W/m$$

$$I_{LEDs} = \frac{P_{LEDs}}{V_{LEDs}} \cdot longitud = \frac{15}{12} \cdot 0{,}6 = 750\ mA$$

$$I_{máx,fuente\ alim} = I_{máx,convertidor} + I_{LEDs} = 4{,}975 + 0{,}75 = 5{,}725\ A$$

Otras limitaciones:

$$V_{in} = 220\ V, 50\ Hz$$

$$V_{out} = 12\ V$$

$$I_{out} > 6\ A$$

A partir de los cálculos anteriores se elige una fuente de alimentación AC-DC de 220 V de entrada y 12 V de salida con una potencia máxima de 120 W.

1.3.4 Resolución del problema.

1.3.4.1 Resolución de la cinemática del robot.

Uso del algoritmo de Denavit-Hartenberg para el brazo robótico diseñado.

Para resolver la cinemática directa e inversa se utiliza el modelo de Denavit-Hartenberg. Tras seguir los pasos descritos en el apartado 1.2.2 se llega a la siguiente resolución:

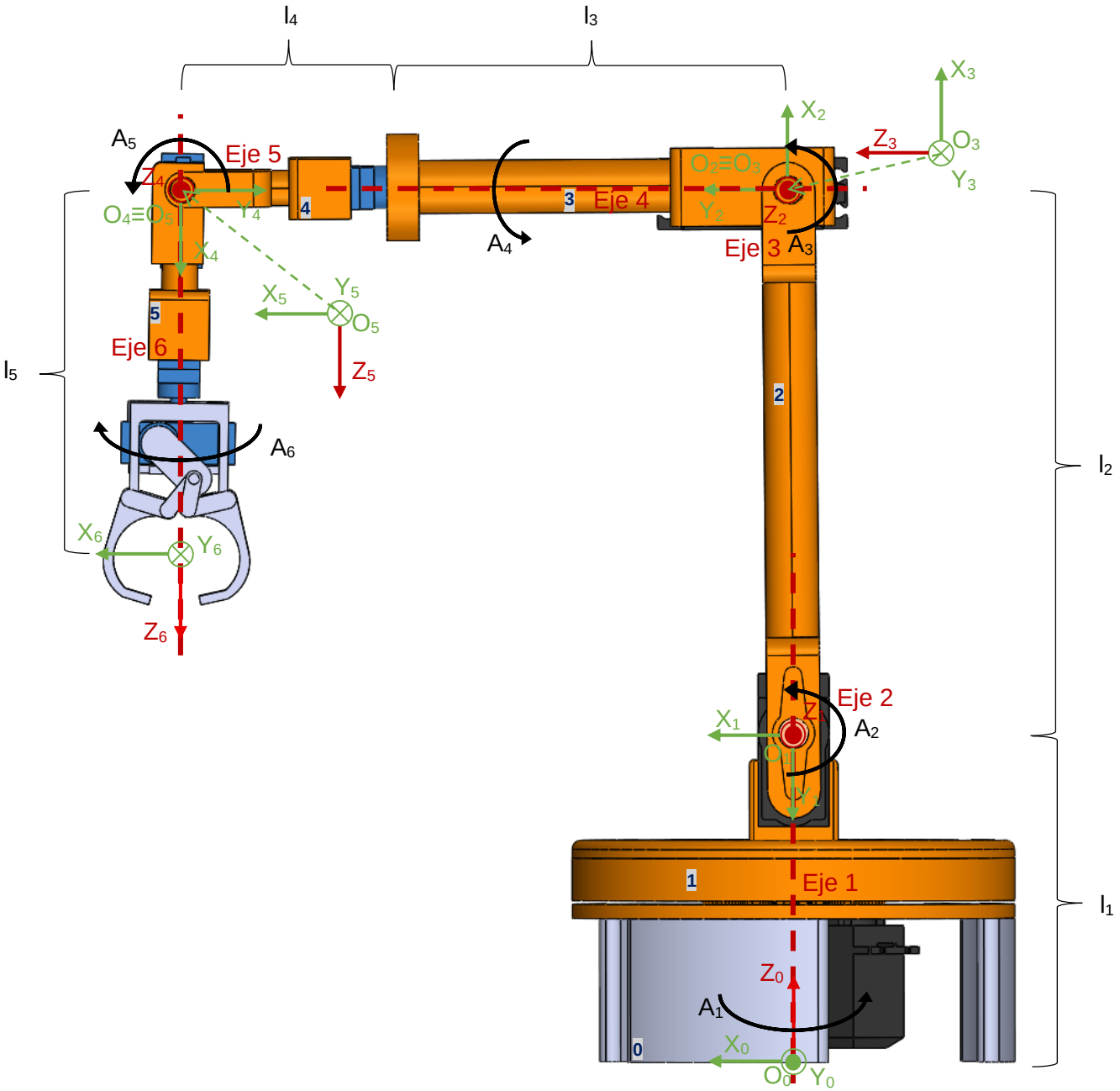


Ilustración 17. Resolución del modelo de Denavit-Hartenberg.

Tras haber posicionado y numerado los ejes y sistemas de referencia de cada eslabón se obtienen los parámetros de Denavit-Hartenberg:

i	θ_i	d_i	a_i	α_i
1	q_1	l_1	0	$-90°$
2	$q_2 - 90°$	0	l_2	0º
3	q_3	0	0	$-90°$
4	$180° - q_4$	$l_3 + l_4$	0	$-90°$
5	$q_5 + 90°$	0	0	$-90°$
6	q_6	l_5	0	0º

Tabla 9. Parámetros de Denavit-Hartenberg.

Tras imprimir y montar los eslabones junto con los servomotores, se miden las longitudes de dichos eslabones obteniendo los siguientes valores:

$l_1 = 102{,}25$ mm
$l_2 = 153$ mm
$l_3 = 112{,}4$ mm
$l_4 = 58{,}5$ mm
$l_5 = 120$ mm

Resolución de la cinemática directa.

En primer lugar se realiza el algoritmo de Denavit-Hartenberg, situando los ejes de referencia de cada eslabón y obteniendo los parámetros $\theta_i, d_i, a_i \; y \; \alpha_i$ cuyos valores aparecen en la Tabla 9.

En segundo lugar, una vez conocidos los parámetos se calculan las matrices de transformación ${}^{i-1}A_i$ y se deja la ecuación en función de los ángulos ($q_1, q_2, q_3, q_4, q_5 \; y \; q_6$). La ecuación genérica a aplicar es:

$$ {}^{i-1}A_i = \begin{bmatrix} Cos(\theta_i) & -Cos(\alpha_i)\cdot Sen(\theta_i) & Sen(\alpha_i)\cdot Sen(\theta_i) & a_i \cdot Cos(\theta_i) \\ Sen(\theta_i) & Cos(\alpha_i)\cdot Cos(\theta_i) & -Sen(\alpha_i)\cdot Cos(\theta_i) & a_i \cdot Sen(\theta_i) \\ 0 & Sen(\alpha_i) & Cos(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Se sustituyen los valores de la Tabla 9 en la ecuación genérica y se obtienen las siguientes matrices:

$$ {}^{0}A_1 = \begin{bmatrix} Cos(q_1) & -Cos(-90°)\cdot Sen(q_1) & Sen(-90°)\cdot Sen(q_1) & 0\cdot Cos(q_1) \\ Sen(q_1) & Cos(-90°)\cdot Cos(q_1) & -Sen(-90°)\cdot Cos(q_1) & 0\cdot Sen(q_1) \\ 0 & Sen(-90°) & Cos(-90°) & 102{,}25 \\ 0 & 0 & 0 & 1 \end{bmatrix} = $$

$$ = \begin{bmatrix} Cos(q_1) & 0 & -Sen(q_1) & 0 \\ Sen(q_1) & 0 & Cos(q_1) & 0 \\ 0 & -1 & 0 & 102{,}25 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{1}A_2 = \begin{bmatrix} Sen(q_2) & Cos(q_2) & 0 & 153\cdot Sen(q_2) \\ -Cos(q_2) & Sen(q_2) & 0 & -153\cdot Cos(q_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{2}A_3 = \begin{bmatrix} Cos(q_3) & 0 & -Sen(q_3) & 0 \\ Sen(q_3) & 0 & Cos(q_3) & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{3}A_4 = \begin{bmatrix} -Cos(q_4) & 0 & Sen(q_4) & 0 \\ Sen(q_4) & 0 & -Cos(q_4) & 0 \\ 0 & -1 & 0 & 170{,}9 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{4}A_5 = \begin{bmatrix} Sen(q_5) & 0 & Cos(q_5) & 0 \\ -Cos(q_5) & 0 & Sen(q_5) & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{5}A_6 = \begin{bmatrix} Cos(q_6) & -Sen(q_6) & 0 & 0 \\ Sen(q_6) & Cos(q_6) & 0 & 0 \\ 0 & 0 & 1 & 120 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Posteriormente se realiza la matriz de transformación que relaciona el sistema de referencia de la base del brazo robótico con el sistema de referencia del extremo:

$$ T = {}^{0}A_6 = {}^{0}A_1 \cdot {}^{1}A_2 \cdot {}^{2}A_3 \cdot {}^{3}A_4 \cdot {}^{4}A_5 \cdot {}^{5}A_6 = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Los cálculos finales se realizan utilizando el programa Matlab cuyo código está en el Anejo A-3. En dicho programa se escriben los valores deseados de la configuración del robot y las matrices de transformación anteriores. Posteriormente se multiplican redondeando el valor obtenido:

round(A01*A12*A23*A34*A45*A56)

Resolución de la cinemática inversa.

El primer paso para resolver la cinemática inversa es realizando el algoritmo de Denavit-Hartenberg, situando los ejes de referencia de cada eslabón y obteniendo los parámetros $\theta_i, d_i, a_i \; y \; \alpha_i$ cuyos valores aparecen en la Tabla 9.

A continuación, se procede a hacer el desacople cinemático:

$$ \vec{p}_m = \vec{p}_{TCP} - d_6 \cdot {}^{0}\vec{a}_6 = \begin{bmatrix} p_{x,TCP} \\ p_{y,TCP} \\ p_{z,TCP} \end{bmatrix} - 120 \cdot \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} = \begin{bmatrix} p_{x,TCP} \\ p_{y,TCP} \\ p_{z,TCP} + 120 \end{bmatrix} $$

Se resuelven los tres primeros ángulos del brazo usando el método geométrico. El desarrollo de las fórmulas empleadas está descrito en el apartado 1.2.2.

Para calcular el ángulo de la primera articulación se utiliza la fórmula siguiente:

$$ q_1 = arctg\left(\frac{p_y}{p_x}\right) $$

A continuación se calcula q_3:

$$q_3 = arcos\left(\frac{{p_x}^2 + {p_y}^2 + {p_z}^2 - 52615{,}81}{52295{,}4}\right)$$

Matemáticamente existen dos soluciones para obtener q_2, se elige la que está a continuación ya que hace que el codo del brazo robótico esté arriba.

$$q_2 = \beta - \alpha = arctg\left(\frac{p_z}{-\sqrt{{p_x}^2 + {p_y}^2}}\right) - arctg\left(\frac{170{,}9 \cdot sen(q_3)}{153 + 170{,}9 \cdot \cos(q_3)}\right)$$

A partir de todas las matrices de transformación que ya se han calculado anteriormente se obtiene ${}^0R_3$:

$${}^0R_3 = {}^0R_1 \cdot {}^1R_2 \cdot {}^2R_3$$

Se utiliza Matlab para resolverlo mediante el siguiente código:

```
A03 = A01 * A12 * A23;
R03 = A03(1:3,1:3);
```

Posteriormente se calcula la inversa de la matriz de rotación de la articulación 3 respecto a la base, ${}^0{R_3}^{-1}$, usando el código siguiente:

```
R30 = inv(R03);
```

A continuación, se calculan los tres últimos ángulos en función de la orientación deseada: $\vec{n}, \vec{o}\ y\ \vec{a}$;

$${}^3R_6 = r_{ij} = {}^0{R_3}^{-1} \cdot {}^0R_6 = {}^0{R_3}^{-1} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

Mediante el siguiente código se obtienen los ángulos en funcion de la orientación y posición del extremo del brazo robótico. Las ecuaciones obtenidas se utilizan para que el microcontrolador ESP32 pueda obtener la solución a la cinemática inversa a partir de la posición y orientación.

```
q4 = atan(R(2,3) / R(1,3));
q5 = acos(R(3,3)) - (pi/2);
q6 = atan(R(3,2) / - R(3,1));
```

1.3.4.2 Calibración de la cámara. Obtención de la matriz de homografía.

A continuación, se explica el procedimiento seguido para calibrar la cámara OV7670. El código de Arduino empleado está dispuesto en el Anejo 2.

Primero se dispone en la cinta una pieza cuadrada y se hace una fotografía, obteniendo la siguiente imagen:

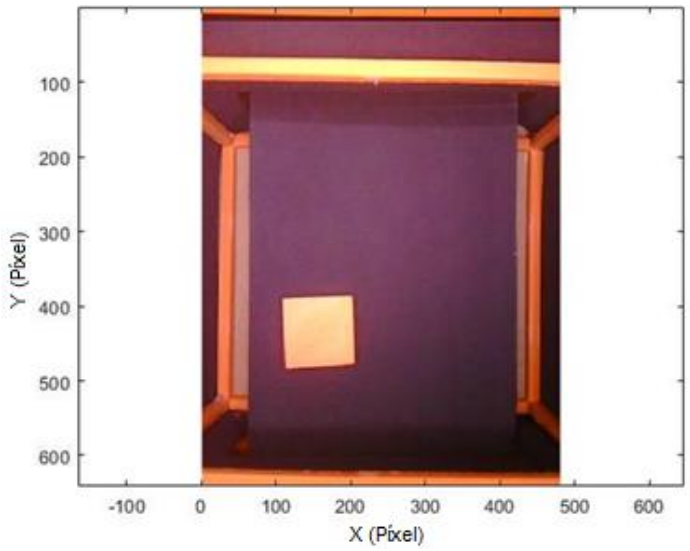


Ilustración 18. Pieza situada en la esquina inferior izquierda de la cinta.

A continuación, mediante el uso de un algoritmo que extrae la parte central de la imagen se puede segmentar correctamente. Destacar que dicho algoritmo extrae siempre el mismo fragmento de la cinta independientemente de dónde esté situada la pieza.

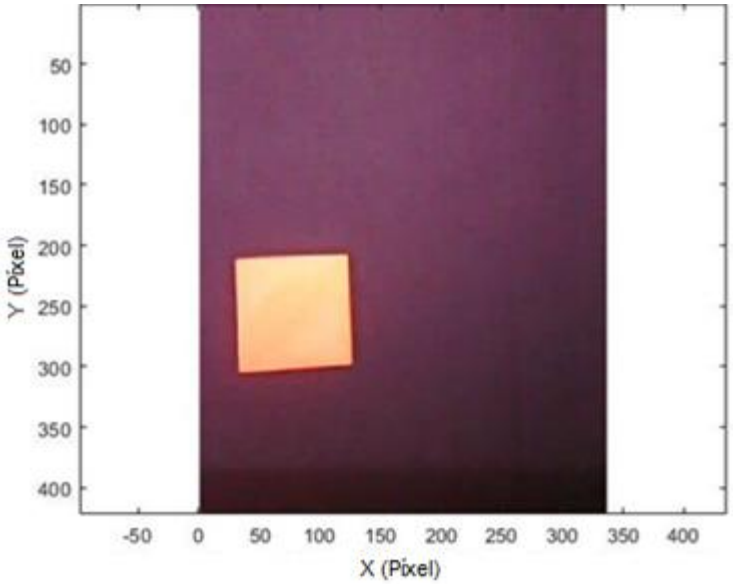


Ilustración 19. Imagen de la pieza ya recortada.

Una vez recortada se procede a su segmentación. Para segmentar la imagen se ha utilizado el algoritmo de umbralización de regiones (Threshold) debido a que es un método muy simple y eficiente en términos computacionales. Se ha utilizado el histograma de intensidades correspondiente al color rojo cuyo umbral se ha determinado a partir del histograma siguiente:

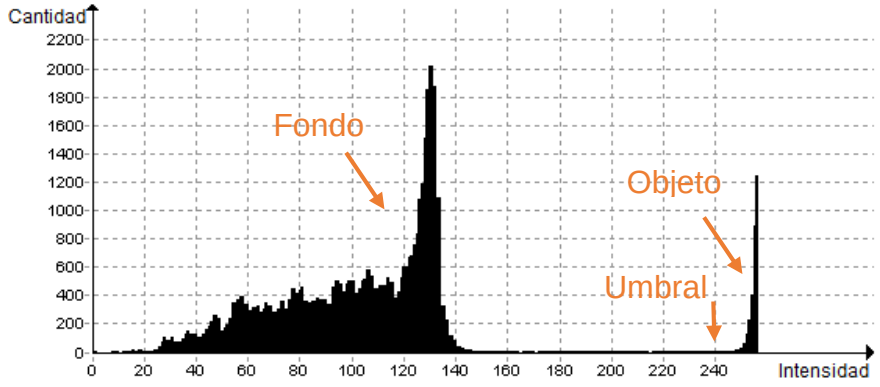


Gráfico 3. Histograma de la capa roja de la ilustración anterior

Como se puede apreciar el umbral adquiere un valor de 240. A continuación, se muestra la imagen segmentada y se calculan sus vértices y su centroide:

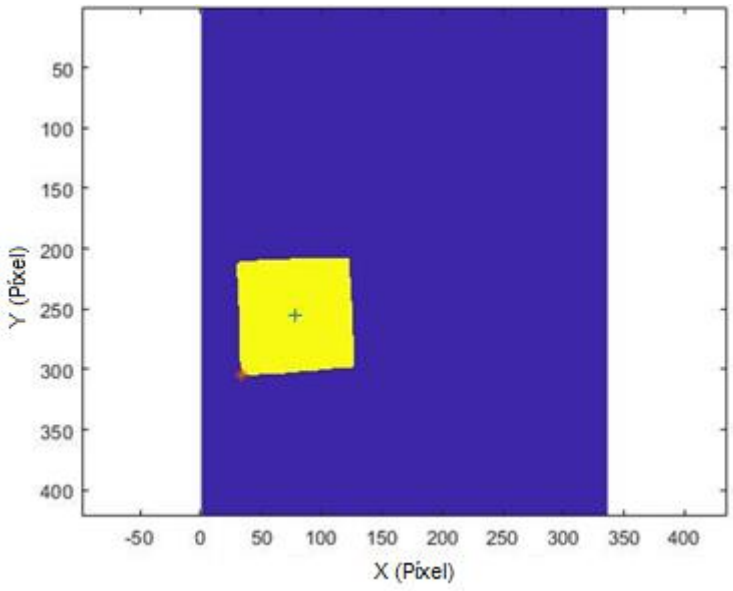


Ilustración 20. Imagen de la pieza segmentada donde se muestra su centroide "+" y su vértice más lejano al centroide "*".

Para el cálculo del centroide se utilizan las siguientes fórmulas:

$$\text{Centroide}_x = \frac{\sum_{i=0}^{n} x_i}{n}$$

$$\text{Centroide}_y = \frac{\sum_{i=0}^{n} y_i}{n}$$

Donde x_i es la componente x del píxel i, y_i es la componente y del píxel i (éstos solo amarillos) y n es el número total de píxeles amarillos. La variable n se usa para diferenciar si la pieza es cuadrada o rectangular, siendo un rectángulo si n es mayor de 1000.

La segmentación se realiza mediante un threshold, es decir, se aplica una condición "if" a todos los pixeles de la Ilustración 19. Si el color del pixel es mayor a un umbral, en este caso 240, se le asigna el color amarillo y si es menor se le asigna el azul.

A continuación, se hace avanzar la pieza a través de la cinta una distancia de 34 cm hasta posicionarse delante del brazo robótico. Se le suministran los ángulos a mano y de esta forma el brazo se mueve hasta la posición donde debe coger la pieza. Posteriormente se obtiene el punto donde se posiciona el brazo mediante la cinemática directa utilizando el código que se adjunta en el Anejo 3. Una vez conocidos los centroides de la pieza en función del sistema de coordenadas de la cámara y del brazo robótico, se anotan ambas medidas y se realiza el proceso de nuevo un total de 9 veces en diferentes posiciones de la cinta, obteniendo los siguientes puntos:

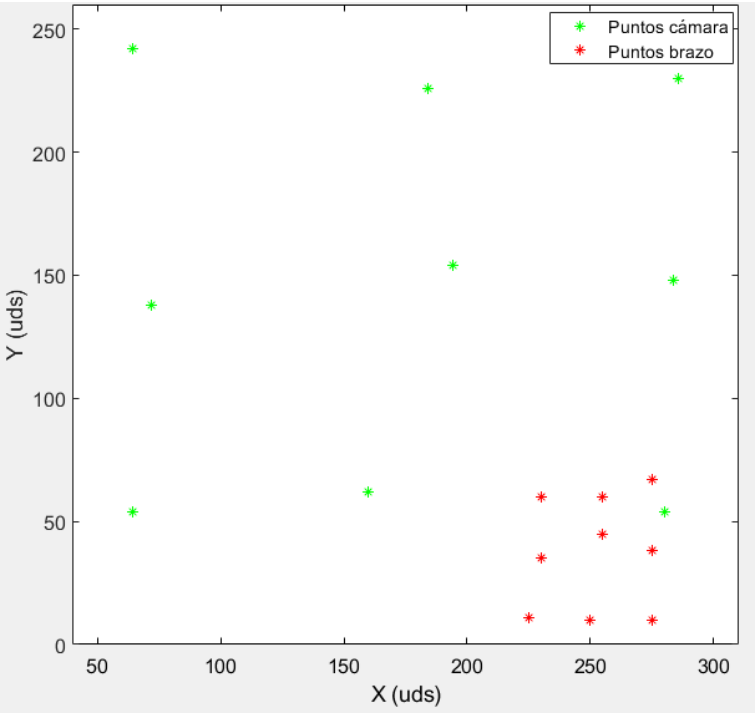


Ilustración 21. Valores de posición de los centroides. Rojo en mm y verde en píxeles.

Los centroides de color verde están en función de la referencia de la cámara y medidos en píxeles. Los de color rojo están en función de la referencia del brazo robótico y medidos en milímetros.

```
X=[ 64 242 1;   U=[275 10 30;
   280  54 1;      230 60 30;
   194 154 1;      255 45 30;
   284 148 1;      255 60 30;
    72 138 1;      250 10 30;
    64  54 1;      225 11 30;
   160  62 1;      230 35 30;
   286 230 1;      275 67 30;
   184  226 1];    275 38 30];
```

Ilustración 22. Valores de las componentes de los centroides de la pieza. X son los valores con la referencia de la cámara y U con la referencia del brazo.

Mediante el código facilitado en el Anejo 3 y los valores de la ilustración anterior, se obtiene la matriz de transformación homográfica:

```
H =

    0.0438    0.1124  212.4988
    0.2294   -0.0149   -3.0303
    0.0001   -0.0005    1.0000
```

Ilustración 23. Valores de la matriz homográfica.

A partir de esta matriz se calculan los centroides U usando los valores de la matriz X y así se obtiene el error. En nuestro caso se obtienen los siguientes errores:

```
error =

   -1.6815    0.8112         0
   -0.5503   -0.3160         0
    0.9423    3.2308         0
    0.9758   -2.9988         0
    2.3435   -2.2499         0
   -1.3926   -0.0933         0
   -0.1792    1.7167         0
   -1.6024    1.7830         0
    1.1324   -1.8798         0
```

Ilustración 24. Error en mm de cada centroide producido por el uso de la matriz homográfica.

El error máximo producido por la matriz homográfica es de 3,23 mm.

Esta matriz se utiliza para cambiar la referencia de los centroides, su escala y su unidad de medida. De esta forma cuando la cámara realice una imagen y extraiga las características, centroides, puede transformar sus valores y dar la orden de dónde se tiene que posicionar el brazo robótico para cogerla.

1.3.4.3 Programación de los microcontroladores. Utilización de GRAFCET.

Para programar los microcontroladores, Anejo 1 y 2, se ha realizado el siguiente diagrama, con el fin de ayudar a una mejor comprensión de la estructura que tiene el código.

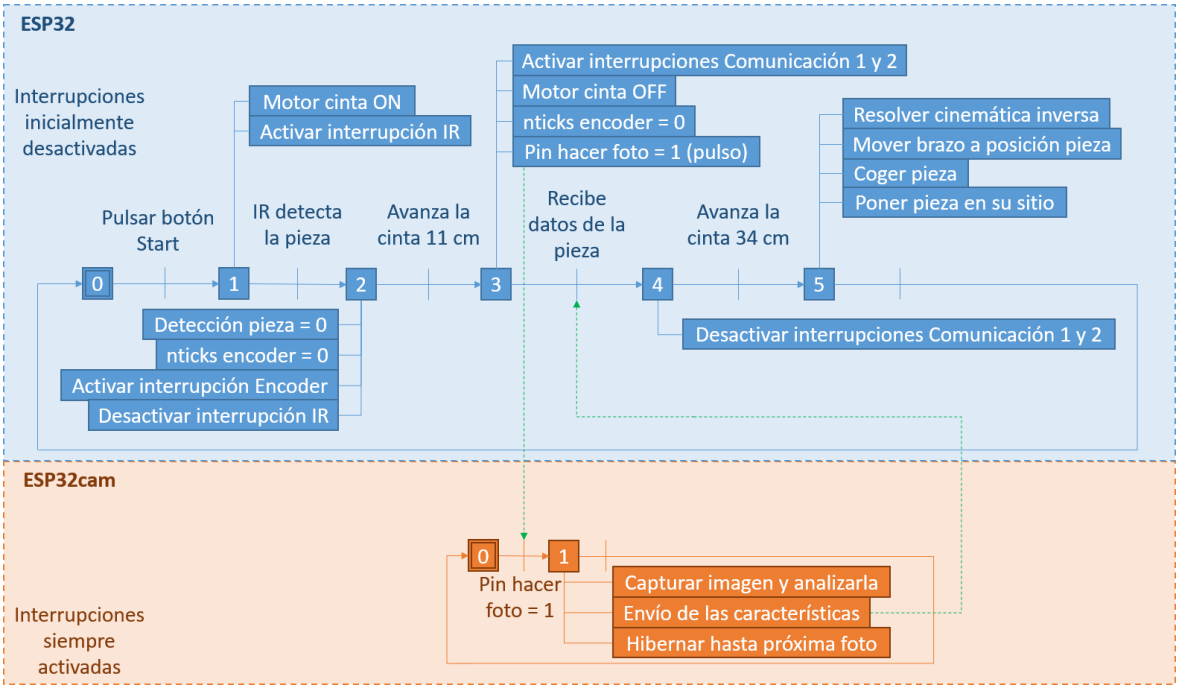


Ilustración 25. GRAFCET simplificado de la programación de los microcontroladores.

En la ilustración se puede apreciar el algoritmo secuencial y cíclico utilizado en ambos dispositivos. En naranja el de ESP32cam y en azul el de ESP32. A la izquierda se detallan ciertos parámetros a tener en cuenta al inicio del programa como que el ESP32 tiene las interrupciones desactivadas y el ESP32cam las tiene activadas.

Las flechas verdes discontinuas de la ilustración anterior muestran la interacción entre microprocesadores.

Dado que el programa del ESP32 se repite cíclicamente se ha ideado el siguiente código muy compacto que permite que el procesador lo ejecute y “le sobre tiempo” para gestionar las interrupciones y no se sobresature. A continuación, se muestra la estructura empleada:

- Se escriben las librerías empleadas:

```
#include <Wire.h>
#include <Adafruit_PWMServoDriver.h>
```

- Se declaran las variables utilizadas y se asignan valores iniciales.

- Se crean las estructuras de las interrupciones:

```
struct Encoder {
  const uint8_t PIN;
  uint32_t NTicks;
};
struct IR {
  const uint8_t PIN;
  bool Detectado;
};
                .
                .
                .
```

  - A continuación, se asignan sus valores:

```
Encoder Encoder1 = {18, 0};
IR IR1 = {19, false};
                .
                .
                .
```

- Funciones:
  - cambiarRefCamaraARefRobot: a partir de la matriz de homografía, se cambian las referencias de los valores de los centroides de los objetos, escala, etc.:

```
void cambiarRefCamaraARefRobot(){
  pz =  Cx*H[2][0]+Cy*H[2][1]+H[2][2];
  px = (Cx*H[0][0]+Cy*H[0][1]+H[0][2])/pz;
  py = (Cx*H[1][0]+Cy*H[1][1]+H[1][2])/pz;
  pz =  30;
}
```

  - Definición de las interrupciones del pin del encoder, del infrarrojo y de los de comunicación:

```
void IRAM_ATTR isrEncoder1() {
  if (millis() - startTimeEncoder > timeThresholdEncoder){
    Encoder1.NTicks += 1;
    startTimeEncoder = millis();
  }
}
                                .
                                .
                                .
```

- IrPosición: la función se encarga de mover el brazo robótico dada la orientación de la pinza, su estado (abierto/cerrado) y el punto de destino del extremo del brazo robótico.

```
void IrPosicion(float px, float py, float pz, int alpha, int pinza){
  ciRRR16(px, py, pz, alpha);                          //Cinemática inversa
  if(q1 <= 180){angulosDeseadosGrados[0] =q 1;}
  if(q2 <= 60){angulosDeseadosGrados[1] = q2;}
  if(q3 <= 60){angulosDeseadosGrados[2] = q3;}
  if(q4 <= 90){angulosDeseadosGrados[3] = q4;}
  if(q5 <= 90){angulosDeseadosGrados[4] = q5;}
  if(q6 <= 90){angulosDeseadosGrados[5] = q6;}
  angulosDeseadosGrados[6] = pinza;                    //Pinza abierta/cerrada
  angulosDeseadosGrados[7] = 0;                        //Cinta encendida/apagada

  ConversionDeGrados(Nmax, Nmin, angulosDeseadosGrados, angulosActualesGrados, angulosDeseados);
  moverMotores(angulosDeseados, angulosActuales);
}
```

- MoverMotores: se encarga de mover los servomotores al ángulo solicitado.

```
void moverMotores(unsigned int angulosDeseados[8], unsigned int angulosActuales[8]){
  //Servo motor 1:
  if(angulosActuales[0] > angulosDeseados[0]){
    for(i = angulosActuales[0]; i > angulosDeseados[0]; i--){
      pwm.setPWM(12, 0, i);    //(canal,valor en que empieza on,valor en que acaba on)
      delay(t);
    }
  }
  if(angulosActuales[0] < angulosDeseados[0]){
    for(i = angulosActuales[0]; i < angulosDeseados[0]; i++){
      pwm.setPWM(12, 0, i);
      delay(t);
    }
  }
  angulosActuales[0] = angulosDeseados[0];
                                                       .
                                                       .
                                                       .
```

- ConversiónDeGrados: cambia el rango de valores angulares al requerido por los servomotores.

```
void ConversionDeGrados(unsigned int Nmax[8], unsigned int Nmin[8], int angulosDeseadosGrados[8], int
angulosActualesGrados[8], unsigned int angulosDeseados[8]){

  //Servo motor 1:
  angulosDeseados[0]=angulosDeseadosGrados[0]*(((Nmax[0]-Nmin[0])/180.0))+Nmin[0];  //conversión de grados
  angulosActualesGrados[0] = angulosDeseadosGrados[0];

                                                       .
                                                       .
                                                       .
```

- ciRRR16: función que calcula la cinemática inversa a partir del punto del extremo del brazo robótico y su orientación. Se obtienen los ángulos necesarios para que el brazo alcance la posición solicitada.

```
void ciRRR16(float pxx, float pyy, float pzz, int anguloPinza){
  float dx=pxx, dy=pyy, dz=pzz+L5-L1;
  float L=L3+L4;
  //Cálculo de q1 considerando el cuadrante en el que esté el robot
  if((dx>0)&&(dy>0)){
     q1=atan(dy/dx);
  }
  else if((dx>0)&&(dy<0)){
     q1=atan(dy/dx);
  }
                                                    :
                                                    :
  //Codo arriba, L2 izquierda
  //Trigonometría para obtener los valores de los ángulos
  float r=-sqrt(pow(dx,2)+pow(dy,2));
  q3=acos((pow(dx,2)+pow(dy,2)+pow(dz,2)-pow(L2,2)-pow(L,2))/(2*L2*L));
  q2=atan(dz/r)-atan((L*sin(q3))/((L2+L*cos(q3))));
  q4=atan(0);
  q5=acos((cos(q2)*sin(q3)+cos(q3)*sin(q2))/(pow(cos(q2),2)*pow(cos(q3),2)+
pow(cos(q2),2)*pow(sin(q3),2)+pow(cos(q3),2)*pow(sin(q2),2)+pow(sin(q2),2)*pow(sin(q3),2)));
  q6=atan(sin(q1)/cos(q1));
  //Conversión de radianes a grados
  q1=q1*(180/PI);
  q2=q2*(180/PI);
  q3=q3*(180/PI);
  q4=q4*(180/PI);
  q5=q5*(180/PI);
  q6=q1+anguloPinza;
}
```

- Setup():
  - Se inicia el monitor serial:

```
Serial.begin(115200);
```

  - Se inicializa la librería PWM y se pone a 50 Hz la frecuencia máxima PWM:

```
pwm.begin();
pwm.setPWMFreq(50);
```

  - Se configuran todos los pines incluidos los de las interrupciones:

```
pinMode(Encoder1.PIN, INPUT_PULLUP);
pinMode(IR1.PIN, INPUT_PULLUP);
pinMode(CambioDigito1.PIN, INPUT_PULLUP);
pinMode(ValorDigito1.PIN, INPUT_PULLUP);

pinMode(EmpezarPIN, INPUT_PULLUP);
pinMode(HacerFotoPIN, OUTPUT);
```

- Bucle principal, Loop():
  - Switch(estado): dado el valor del estado, se encarga de saltar a su caso correspondiente. Dentro de cada caso hay una condición que usa la variable "repetido", ésta se encarga de ejecutar solo una vez las acciones de cada estado.

```
switch (estado) {
                                                    .
                                                    .
                                                    .
 case 1:     //estado 1
  if(repetido==0){
     attachInterrupt(IR1.PIN, isrIR1, FALLING);          //Activar interrupción pin sensor infrarrojo
     pwm.setPWM(0, 0, 120);                              //Encender motor de la cinta
     repetido=1;
    }break;
 case 2:      //estado 2
  if(repetido==0){
     IR1.Detectado = false;
     Encoder1.NTicks=0;
     attachInterrupt(Encoder1.PIN, isrEncoder1, CHANGE);    //Activar interrupción pin encoder
     detachInterrupt(IR1.PIN);                              // Desactivar interrupciones IR
     repetido=1;
    }break;
 case 3:     //estado 3
  if(repetido==0){
     pwm.setPWM(0, 0, 0);                                //Parar cinta
     detachInterrupt(Encoder1.PIN);                      // Desactivar interrupciones del encoder
     float d = 0.28274 * 0.5 * Encoder1.NTicks;          // d: distancia que lleva avanzando la cinta
     Encoder1.NTicks=0;
     //Activar interrupciones para transmisión de datos
     attachInterrupt(CambioDigito1.PIN, isrCOMUNICATION1, FALLING);
     attachInterrupt(ValorDigito1.PIN, isrCOMUNICATION2, FALLING);
     //Pulso para hacer la foto
     digitalWrite(HacerFotoPIN, HIGH);
     delay(110);
     digitalWrite(HacerFotoPIN, LOW);
     delay(1000);
     //Comprobar que hay respuesta
     long trespuesta = millis();
     while(tipo == 0){
      if(millis()-trespuesta>5500){
        //Pulso para hacer la foto de nuevo
        digitalWrite(HacerFotoPIN, HIGH);
        delay(120);
        digitalWrite(HacerFotoPIN, LOW);
        trespuesta = millis();
      }
     }
     repetido=1;
    }break;
```

```
  case 4:    //estado 4
   if(repetido==0){
      if(tipo==2){          //Para coger a pieza cuadrada.
        alpha=alpha-45;     //Convertir Rg  [90,0] -> [45, -45]
      }
      if(tipo==3){          //Para coger a pieza rectangular por el lado estrecho.
        alpha=90-alpha;     //Convertir Rg [180,0] -> [-90, 90]
      }
      //Desactivar interrupciones de transmisión de datos
      detachInterrupt(CambioDigito1.PIN);
      detachInterrupt(ValorDigito1.PIN);

      attachInterrupt(Encoder1.PIN, isrEncoder1, CHANGE); //Habilitar interrupción del encoder
      pwm.setPWM(0, 0, 120);                              //Encender cinta
      repetido=1;
    }break;
 case 5:    //Estado 5
   if(repetido==0){
      pwm.setPWM(0, 0, 0);                                //Parar cinta
      detachInterrupt(Encoder1.PIN);                      //Deshsbilitar interrupción del encoder
      float d = 0.28274 * 0.5 * Encoder1.NTicks;
      Encoder1.NTicks=0;
      Cy=Cy*100;                                          //conversión a pixeles
      Cx=Cx*100;                                          //conversión a pixeles
      cambiarRefCamaraARefRobot();//Matriz homográfica
      IrPosicion(px, py, pz+45, alpha, 1);                //Posicionarse para coger pieza
      IrPosicion(px, py, pz, alpha, 0);                   //coger pieza
      IrPosicion(px, py, pz+45, alpha, 0);                //Separarse de la cinta
      //Posicionarse en el sitio para dejar la pieza y abrir la pinza
      if(tipo == 2){
        IrPosicion(160, -140, 100, 0, 0);
        IrPosicion(100, -150, 40, 0, 0);
        IrPosicion(100, -150, 10, 0, 1);
        IrPosicion(160, -140, 100, 0, 1);
      }
      else{
        IrPosicion(160, -140, 100, 0, 0);
        IrPosicion(10, -150, 40, 0, 0);
        IrPosicion(10, -150, 10, 0, 1);
        IrPosicion(160, -140, 100, 0, 1);
      }
      repetido=1;
    }break;
}
```

- Condicionales: se encargan de permitir el cambio de estado si se cumplen las condiciones necesarias, son las transiciones. También gestionan la variable “repetido”.

```
if((estado==0)&&(digitalRead(EmpezarPIN) == LOW)){
   estado=1, repetido=0;
  }
  else if((estado==1)&&(IR1.Detectado == true)){
   estado=2, repetido=0;
  }
  else if((estado==2)&&(2.8274*0.5*Encoder1.NTicks>=110)){      //distancia en mm
   estado=3, repetido=0;
  }
  else if((estado==3)&&(tipo != 0)){ //información recibida
   estado=4, repetido=0;
  }
  else if((estado==4)&&(2.8274*0.5*Encoder1.NTicks >= 340)){    //distancia donde el robot coge la pieza
   estado=5, repetido=0;
  }
  else if((estado==5)&&(digitalRead(EmpezarPIN)==LOW)){
   estado=1, repetido=0, Encoder1.NTicks=0;
   Cx1=0, Cx2=0, Cy1=0, Cy2=0, Cx=0.0, Cy=0.0, tipo=0;
  }
}
                                                                .
                                                                .
                                                                .
```

La estructura empleada para programar el ESP32cam es la siguiente:

- Se escriben las librerías empleadas:

```
#include <stdint.h>
#include "esp_camera.h"
#include "esp_timer.h"
#include "img_converters.h"
#include "Arduino.h"
#include "fb_gfx.h"
#include "fd_forward.h"
#include "fr_forward.h"
#include "soc/soc.h"             //Desactivar para problemas de brownour
#include "soc/rtc_cntl_reg.h"  //Desactivar para problemas de brownour
#include "driver/rtc_io.h"
```

- Se declaran las variables utilizadas y se asignan valores iniciales.

- Se definen los pines para el modelo de cámara AI_THINKER:

```
#define PWDN_GPIO_NUM    32
#define RESET_GPIO_NUM   -1
#define XCLK_GPIO_NUM     0
#define SIOD_GPIO_NUM    26
#define SIOC_GPIO_NUM    27
                          .
                          .
                          .
```

- Funciones:

  - EnviarDatos: envía los datos al ESP32 (centroide y orientación del objeto):

```
void EnviarDatos(){
  //Enviar pulso al ESP32 para indicar que empieza el envio
  digitalWrite(Comunicacion1Pin, HIGH);
  delay(tiempo);
  digitalWrite(Comunicacion1Pin, LOW);
  delay(tiempo);
  //Enviar componente x del centroide: Cx
  Valor = Cx*100;
  unidad = Valor/100;                                   // Se extrae la unidad
  decima = (Valor - unidad*100)/10;                     // Se extrae la decima
  centesima = (Valor - unidad*100 - decima*10);         // Se extrae la centésima
  // Algoritmo para enviar los valores (explicado en el apartado 1.3.5.6)
  for(g=0;g<3;g++){
    if(g==0){
      comp=unidad;
    }
    else if(g==1){
      comp=decima;
    }
    else{
      comp=centesima;
    }
    for(f=0;f<comp;f++){
      digitalWrite(Comunicacion2Pin, HIGH);
      delay(tiempo);
      digitalWrite(Comunicacion2Pin, LOW);
      delay(tiempo);
    }
    digitalWrite(Comunicacion1Pin, HIGH);
    delay(tiempo);
    digitalWrite(Comunicacion1Pin, LOW);
    delay(tiempo);
  }
                                                        .
                                                        .
                                                        .
```

- Setup():

  - Se inicia el monitor serial:

```
Serial.begin(115200);
```

- Se configura la cámara:

```
WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0); //desactivar detector de  brownout
camera_config_t config;
config.ledc_channel = LEDC_CHANNEL_0;
config.ledc_timer = LEDC_TIMER_0;
config.pin_d0 = Y2_GPIO_NUM;
config.pin_d1 = Y3_GPIO_NUM;
config.pin_d2 = Y4_GPIO_NUM;
config.pin_d3 = Y5_GPIO_NUM;
config.pin_d4 = Y6_GPIO_NUM;
                                                    .
                                                    .
                                                    .
if(psramFound()){
 //UXGA FRAMESIZE_ + QVGA|CIF|VGA|SVGA|XGA|SXGA|UXGA
 config.frame_size = FRAMESIZE_VGA;
 config.jpeg_quality = 10;          //resolución: 640 x 480
  config.fb_count = 2;
} else {
  config.frame_size = FRAMESIZE_SVGA;
  config.jpeg_quality = 12;
  config.fb_count = 1;
}
```

- Se inicializa la cámara:

```
esp_err_t err = esp_camera_init(&config);
if (err != ESP_OK) {
Serial.printf("Camera init failed with error 0x%x", err);
    return;
  }
```

- Se configuran todos los pines de las interrupciones:

```
  pinMode(HacerFotoPIN, INPUT_PULLUP);
  pinMode(Comunicacion1Pin, OUTPUT);
  pinMode(Comunicacion2Pin, OUTPUT);
```

- Loop():

  - Condición: asigna a foto el valor de 1 si detecta un valor alto en la entrada “HacerFotoPIN”:

```
if(digitalRead(HacerFotoPIN) == 1){
foto=1;
Serial.println("Haz foto");
```

- Condición: hacer la foto si “foto = 1”:

```
if(foto==1){
  camera_fb_t * fb = NULL;
  uint8_t * _bmp_buf = NULL;
  size_t _bmp_buf_len = 0;
  fb = esp_camera_fb_get();          //Tomar una imagen con la cámara
  if(!fb) {
    Serial.println("Camera capture failed");
    return;
  }
}
```

- Se convierte la imagen a BMP:

```
bool imagen = frame2bmp(fb,&_bmp_buf,&_bmp_buf_len);
esp_camera_fb_return(fb);
```

- Se realiza la segmentación, se recorta la imagen y se almacenan los píxeles:

```
k=153600;                              //pixels totales
for(int i=0;i<168;i++){
k=k+540;
  for(int l=0;l<210;l++){
    if(_bmp_buf[k]>240){
      imagenRoja[i][l]=1;
      SumCoordX=SumCoordX+l;
      SumCoordY=SumCoordY+i;
      nPixeles=nPixeles+1;
    }
    if(_bmp_buf[k]<=140){
      imagenRoja[i][l]=0;
    }
    k=k+6;
  }
  k=k+120;
  k=k+1920;
}
```

- Se realiza el cálculo el tipo de pieza y se obtienen los centroides:

```
if(nPixeles>=3000){
  tipo=3;
}
else tipo=2;

Cx=SumCoordX/nPixeles;
Cy=SumCoordY/nPixeles;
```

- Se obtiene la orientación de la pieza en función de si es un cuadrado o un rectángulo:

```
for(int j=0;j<168;j++){
   for(int i=0;i<210;i++){
      if(imagenRoja[j][i] == 1){
         distACentroide=sqrt(pow(Cx-j,2)+pow(Cy-i,2));
         if(distACentroideMayor < distACentroide){
            distACentroideMayor=distACentroide;
            CoordenadasPtoMasLejano[0]=j;
            CoordenadasPtoMasLejano[1]=i;
         }
      }
   }
}
if(tipo==2){       //Cuadrado diagonal-horizontal -> 45º
  float valorTan=abs((Cy-CoordenadasPtoMasLejano[1])/(Cx-CoordenadasPtoMasLejano[0]));
  angOrientac=abs(atan(valorTan)*180/PI)-45;
  alpha=(int)angOrientac;
}
if(tipo==3){     //Rectángulo diagonal-horizontal -> 32º
  float valorTan=abs((Cy-CoordenadasPtoMasLejano[1])/(Cx-CoordenadasPtoMasLejano[0]));
  angOrientac=abs(atan(valorTan)*180/PI);
  alpha=(int)angOrientac;
```

- Se obtienen los vértices del rectángulo:

```
int v1[2]={0,0}, v2[2]={0,0}, v3[2]={0,0}, v4[2]={0,0};
for(int j=0;j<168;j++){
  for(int i=0;i<210;i++){
     if((imagenRoja[j][i] == 1)&&(v1[0] == 0)){
        v1[0]=j, v1[1]=i;
     }
  }
}
                                        .
                                        .
                                        .
//Se selecciona V1 y se busca el 2º vértice más cercano o lejano:
float d12=sqrt(pow(v1[0]-v2[0],2)+pow(v1[1]-v2[1],2));
float d13=sqrt(pow(v1[0]-v3[0],2)+pow(v1[1]-v3[1],2));
float d14=sqrt(pow(v1[0]-v4[0],2)+pow(v1[1]-v4[1],2));
float al=0.0, dif1=0.0, dif2=0.0;
//Se van haciendo comparaciones para encontrarlo
if((d12>d13)&&(d12<d14)){
  dif1=v1[1]-v2[1];
  dif2=v1[0]-v2[0];
  al=atan2(abs(v1[1]-v2[1]), abs(v1[0]-v2[0]))*180/PI;
}
else if((d12>d14)&&(d12<d13)){
  dif1=v1[1]-v2[1];
  dif2=v1[0]-v2[0];
  al=atan2(abs(v1[1]-v2[1]), abs(v1[0]-v2[0]))*180/PI;
}
                                        .
                                        .
                                        .
if(dif1*dif2>=0){
  alpha=(int)al;
}
else if(dif1*dif2<0){
  alpha=(-1)*(int)al;
}
}
```

- Se borra la imagen, se cambia la escala para poder enviar los datos y se pone a cero la variable foto:

```
for(int i=0;i<168;i++){
  for(int l=0;l<210;l++){
    imagenRoja[i][l]=0;
  }
}
Cx=(2*Cx)/100;
Cy=(2*Cy)/100;

EnviarDatos();
foto=0;
```

1.3.4.4 Programación de la comunicación entre microcontroladores.

Para poder comunicarse ambos microcontroladores entre sí y de forma rápida se ha optado por usar las interrupciones externas que disponen ambos.

Éstos se van a comunicar entre sí dos veces, la primera para ordenar que se haga una fotografía y la segunda para enviarse información.

Para que el controlador ESP32 envíe la petición al ESP32cam se ha enlazado la salida del primero con una entrada del segundo. Para ello se habilita una interrupción externa del ESP32cam.

Una vez que el ESP32cam tiene la información del centroide, la orientación y el tipo de pieza que extrae de la imagen, dicha información debe ser enviada al ESP32. Para ello se han conectado entre si dos pines, uno se encarga de marcar el cambio de dígito y el otro de contar el número de pulsos. Cuando ambos están en alto indica que empieza a enviar información, la cual debe estar estructurada siempre de la misma forma para que pueda ser entendida por el receptor. Por ello las variables toman siempre la siguiente forma:

- Centroide $= [x_1\ x_2\ x_3,\ x_4\ x_5\ x_6]$, donde $x \in [9, 0]$
- Orientación $= y\ x_1\ x_2$, donde $x \in [9, 0]$ e $y \in [1, 0]$
- Tipo $= z$, donde $z \in [2, 0]$

Para poder enviar datos se ha creado un algoritmo que los transforma en pulsos, Anejo 1 y 2. A continuación, se muestra un ejemplo del funcionamiento de la transferencia de un número. Se pretende enviar el número 132º correspondiente a la orientación de la pieza. La solución que aporta el algoritmo tiene el siguiente aspecto:

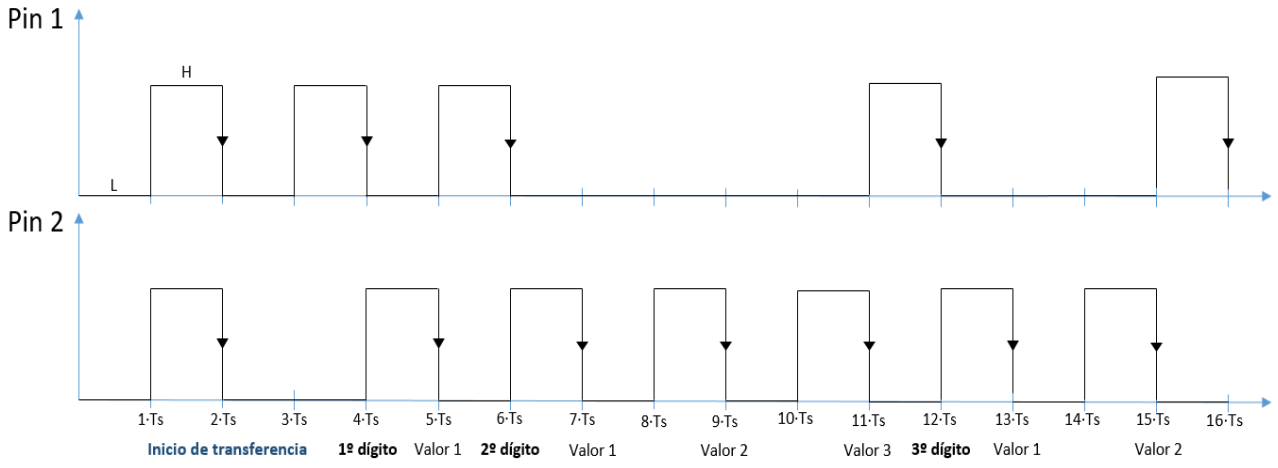


Ilustración 26. Cronograma. La flecha indica que el controlador solo detecta flancos de bajada.

1.3.4.5 Diseño e impresión de las piezas en 3D.

Una vez realizado el croquis de las piezas se ha utilizado SolidWorks, un software para diseño y simulación de piezas 3D. En dicho software se procede a crearlas en 3D y posteriormente se ensamblan virtualmente para comprobar su correcto funcionamiento y se añaden algunas relaciones con el fin de otorgarle movimiento y poder comprobar que las piezas están bien diseñadas y dimensionadas.

Tras hacer los diseños se guardan las piezas con el formato .SLDPRT y los ensamblajes con el formato .SLDASM. El archivo también se guarda con extensión .STL para que sea reconocido por la impresora 3D que utiliza el software Cura.

Ilustración 27. Creality Ender 3, impresora 3D utilizada para imprimir las piezas.

A la hora de imprimir las piezas se configuran las opciones de la impresora 3D de la siguiente forma:

- Altura de capa: 0,2 mm.
- Altura de capa inicial: 0,2 mm.
- Ancho de línea: 0,4 mm.
- Ancho de pared: 0,4 mm.
- Densidad del soporte: 15 %.
- Relleno: 15 - 20 %.
- Tipo de adherencia de la placa de impresión: Falda.
- Boquilla: 0,4 mm.
- Temperatura del extrusor: 240 ºC.
- Temperatura de la cama: 70 ºC la primera capa y 84 ºC el resto.
- Velocidad de impresión: 65 mm/s.

Destacar que existen numerosas opciones, pero sólo se detallan las más relevantes.

Aunque se usan las características de impresión citadas anteriormente, en algunas piezas se varia el valor del relleno. Por ese motivo se pone el rango entre 15 y 20 %.

1.4 Bibliografía.

ai-thinker. (s.f.). Recuperado el 18 de Enero de 2020, de https://loboris.eu/ESP32/ESP32-CAM%20Product%20Specification.pdf

alldatasheet. (s.f.). Recuperado el 16 de Diciembre de 2019, de espressif: https://pdf1.alldatasheet.com/datasheet-pdf/view/1148026/ESPRESSIF/ESP32-WROOM-32.html

alldatasheet. (s.f.). Recuperado el 17 de Diciembre de 2019, de https://pdf1.alldatasheet.net/datasheet-pdf/view/424207/NXP/PCA9685_10.html

Balón, V. (28 de Febrero de 2018). *Flexbot*. Recuperado el 11 de Junio de 2020, de https://www.flexbot.es/origen-de-la-robotica/

electronicoscaldas. (s.f.). Recuperado el 18 de Diciembre de 2019, de https://www.electronicoscaldas.com/datasheet/MG996R_Tower-Pro.pdf

electronicoscaldas. (s.f.). Recuperado el 10 de Diciembre de 2019, de https://www.electronicoscaldas.com/datasheet/SG90_Tower-Pro.pdf

fixokids. (s.f.). Obtenido de https://fixokids.com/wp-content/uploads/2017/02/papel-colores-FLUOR-500-hojas-A3-650091.jpg

García, N. L. (s.f.). *UCO*. Recuperado el 11 de junio de 2020, de http://www.uco.es/users/ma1fegan/2012-2013/vision/Temas/tema-1.pdf

Ingeniería ISA de sistemas y automática. (s.f.). Recuperado el 10 de Junio de 2020, de http://www.isa.uniovi.es/~alonsog/Robotica/02%20Morfologia.pdf

Kucher, D. (20 de Febrero de 2020). *Film-like Story of the First Real Robot Unimate in History*. Obtenido de Somag News: https://www.somagnews.com/film-like-story-first-real-robot-unimate-history/

milinuxblog. (s.f.). Recuperado el 29 de Marzo de 2020, de https://milinuxblog.files.wordpress.com/2019/06/ender_3.jpg?w=1000

Mundo Digital. (s.f.). Recuperado el 11 de Junio de 2020, de http://www.mundodigital.net/origenes-de-la-robotica-mas-de-50-anos-de-historia/

Revolución artificial. (29 de Enero de 2020). Recuperado el 10 de Junio de 2020, de https://blog.infaimon.com/historia-vision-artificial/

Ricolfe Viala, C. (2006). *Caracterización y optimización del proceso de calibrado de cámaras basado en plantilla bidimensional.* Tesis, Universitat Politècnica de València, Valencia. doi:10.4995/Thesis/10251/1858.

Robótica. (s.f.). Recuperado el 2020 de Junio de 12, de http://inteligencia-artificialrobotica.blogspot.com/p/historia-de-la-robotica.html

xbayronx. (s.f.). Recuperado el 10 de Junio de 2020, de https://es.slideshare.net/xbayronx/robotica-industrial-15628055

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Documento número 2:

PLANOS

DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK & PLACE.

TRABAJO FINAL DEL

Grado en Ingeniería Electrónica Industrial y Automática

REALIZADO POR

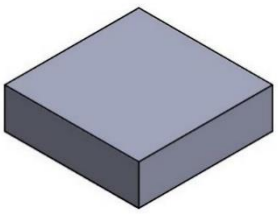
Jorge Doménech Jara

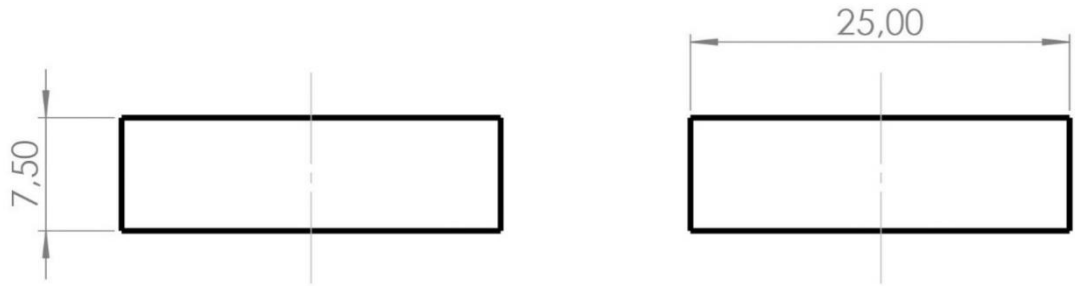
TUTORIZADO POR

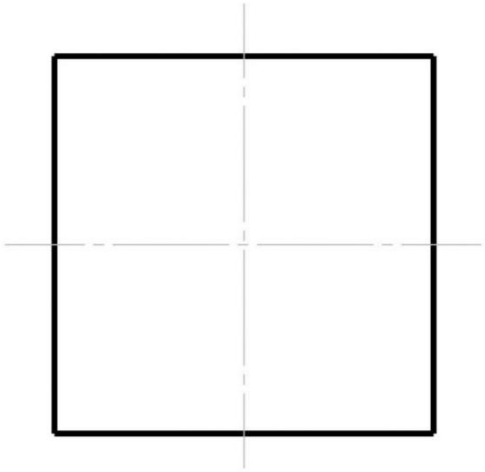
Ángel Valera Fernández

Carlos Ricolfe Viala

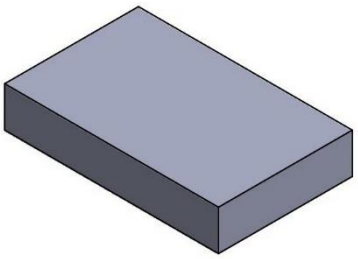
CURSO ACADÉMICO: 2019/2020

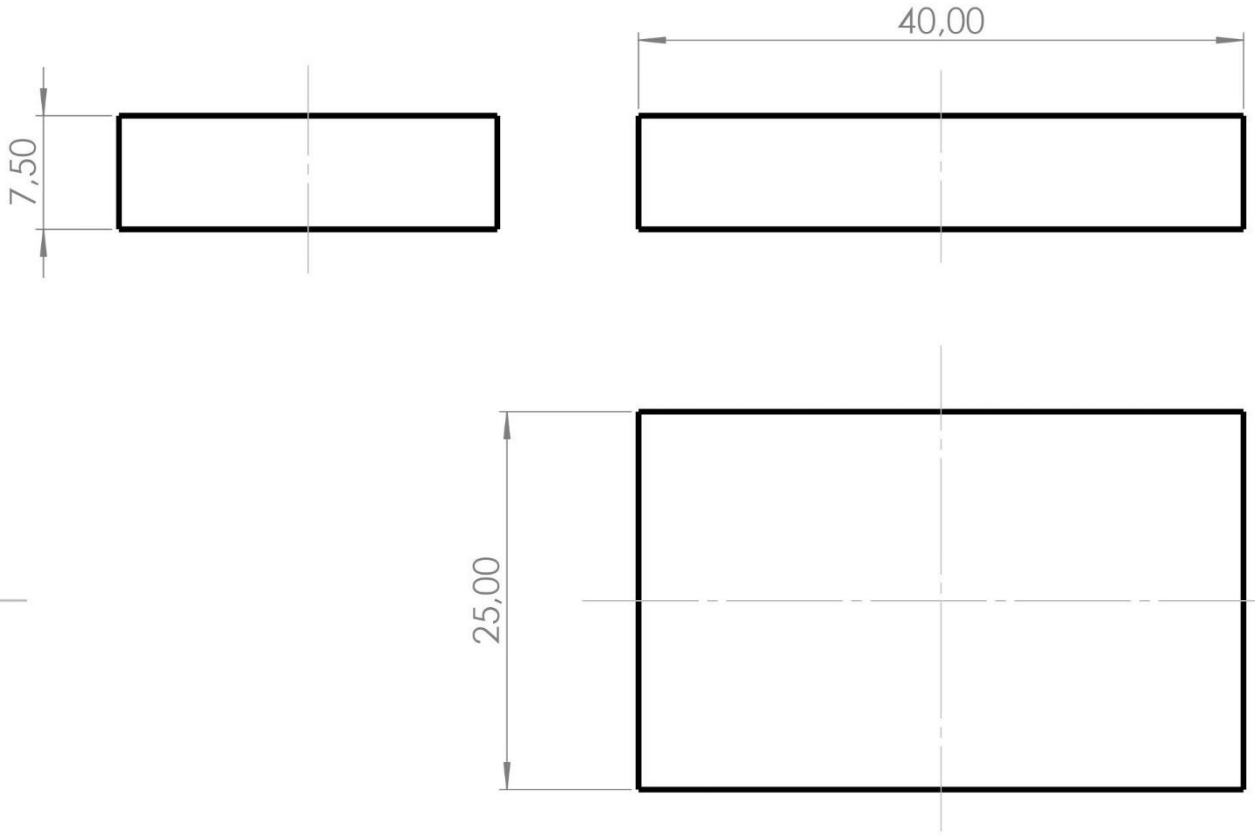




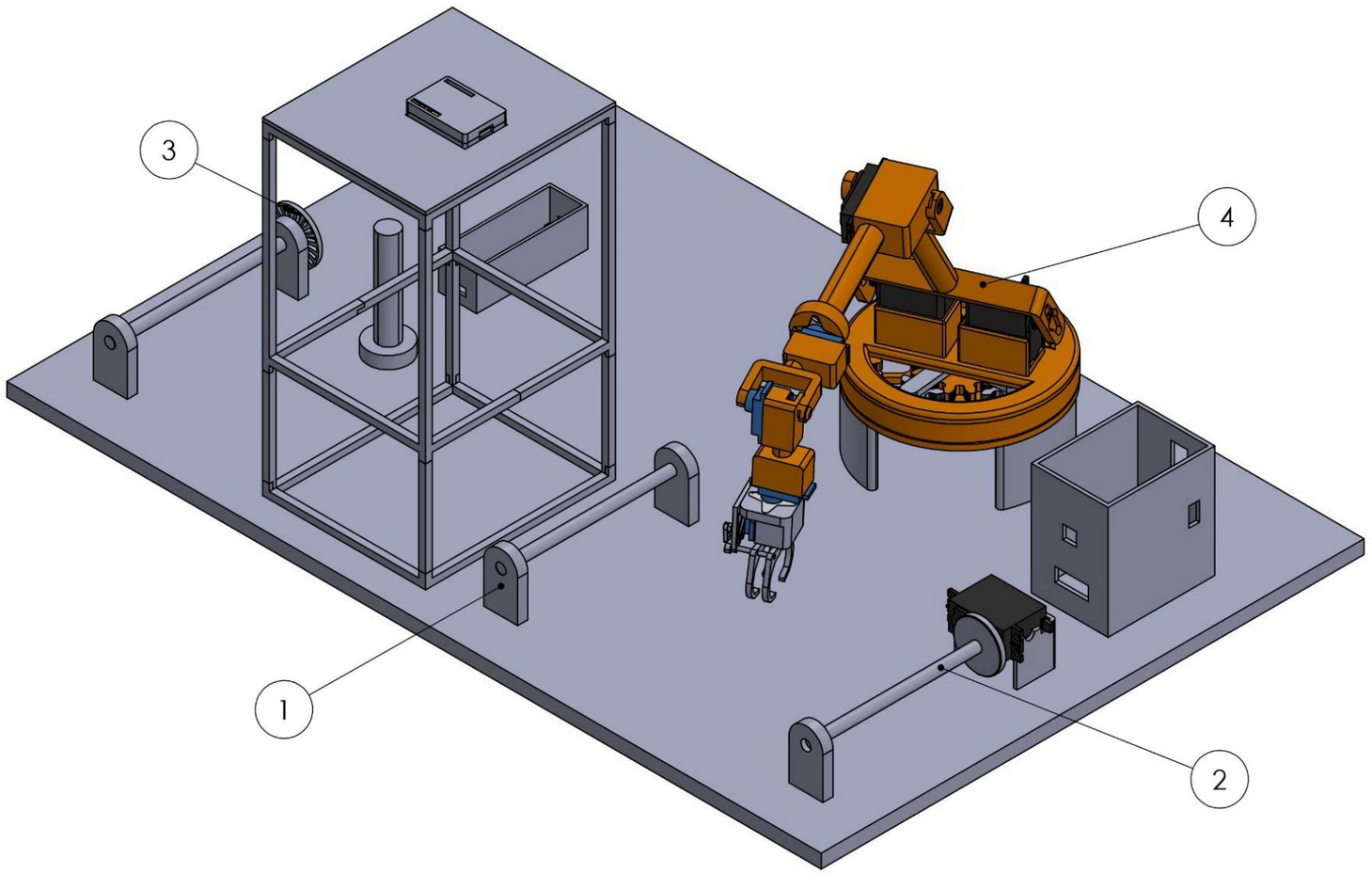


NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.1.2	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 0	REFERENCIA: 220320-7	ORIENTACIÓN:
TÍTULO: PLANTA CUADRADA			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 2:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 22/03/20	HOJA: 1/1 A4





NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.1.1	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 0	REFERENCIA: 220320-6	ORIENTACIÓN:
TÍTULO: PLANTA RECTANGULAR			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 2:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 22/03/20	HOJA: 1/1 A4



Marca	Nº Piezas	Designación			Norma	Material
4	1	Brazo robótico			UNE-EN ISO 15015	ABS
3	1	Rueda del encoder			UNE-EN ISO 15015	ABS
2	2	Ejes de la cinta 1.4.2.2	Eje	1.4.2.2.1	UNE-EN ISO 15015	ABS
	1		Eje con adaptador	1.4.2.2.2	UNE-EN ISO 15015	ABS
1	1	Soportes			UNE-EN ISO 15015	ABS

NOMBRE DEL PROYECTO:
DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE

Nº PLANO: 1.4.2

FIRMA: JORGE DOMÉNECH JARA

TIPO DE DOCUMENTO: Dibujo de subconjunto

REFERENCIA: 120420-4

ORIENTACIÓN:

TÍTULO:

ESTRUCTURAS

CREADO POR: AUTOMA

REVISIÓN: A

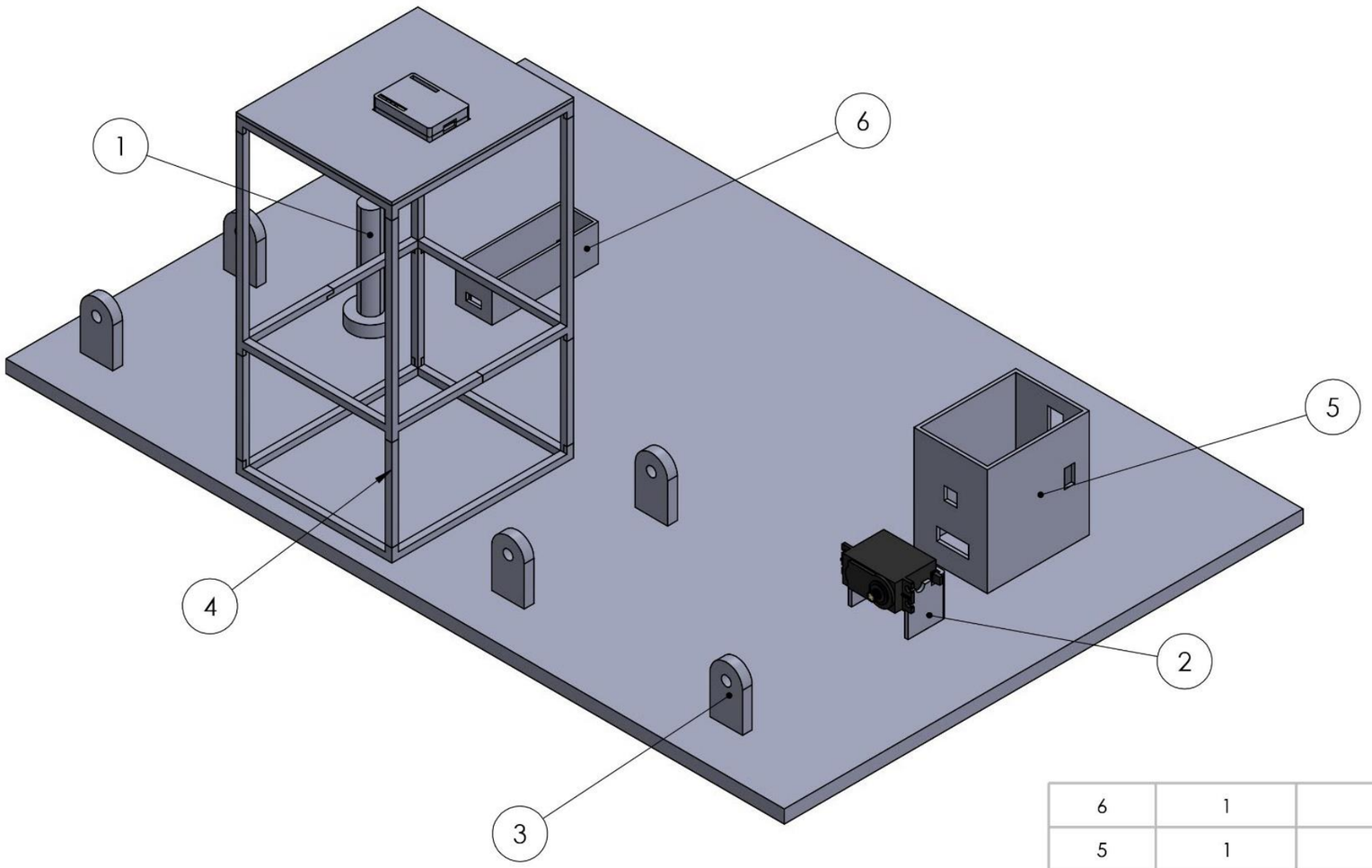
ESCALA: 1:3

MATERIAL: ABS

FECHA DE EDICIÓN: 26/03/20

HOJA: 1/1

A3



Marca	Nº Piezas	Designación	Norma	Material
6	1	Soporte BUCK	UNE-EN ISO 15015	ABS
5	1	Soporte ESP32	UNE-EN ISO 15015	ABS
4	1	Soporte cámara	UNE-EN ISO 15015	ABS
3	5	Soporte cinta	UNE-EN ISO 15015	ABS
2	1	Soporte motor DS04-NFC	UNE-EN ISO 15015	ABS
1	1	Soporte infrarrojos	UNE-EN ISO 15015	ABS

NOMBRE DEL PROYECTO:
DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE

Nº PLANO: 1.4.2.1

FIRMA: JORGE DOMÉNECH JARA

TIPO DE DOCUMENTO: Dibujo de subconjunto

REFERENCIA: 260320-1

ORIENTACIÓN:

TÍTULO:

SOPORTES

CREADO POR: AUTOMA

REVISIÓN: A

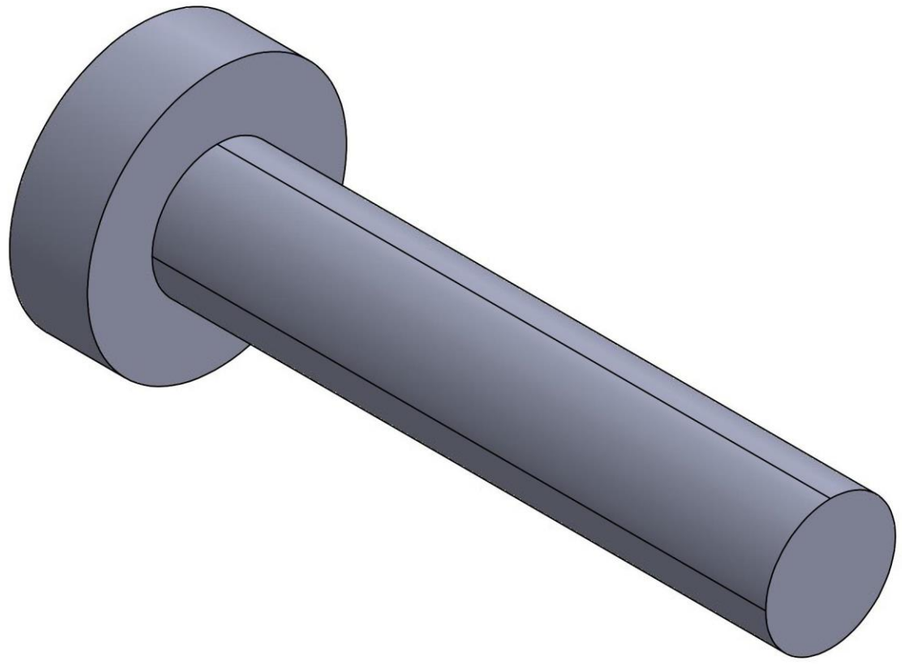
ESCALA: 1:3

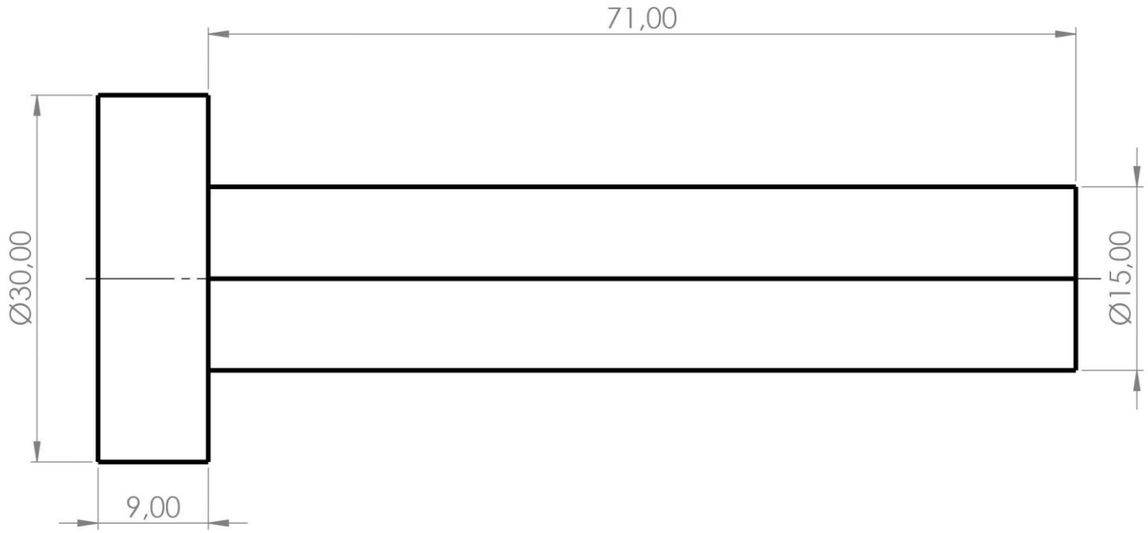
MATERIAL: ABS

FECHA DE EDICIÓN: 26/03/20

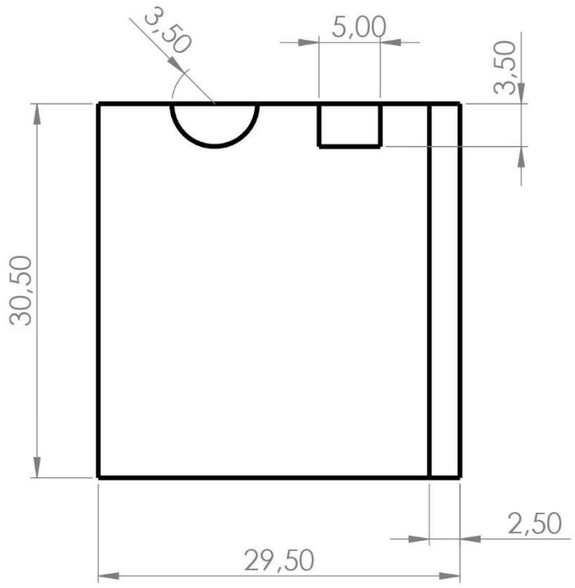
HOJA: 1/1

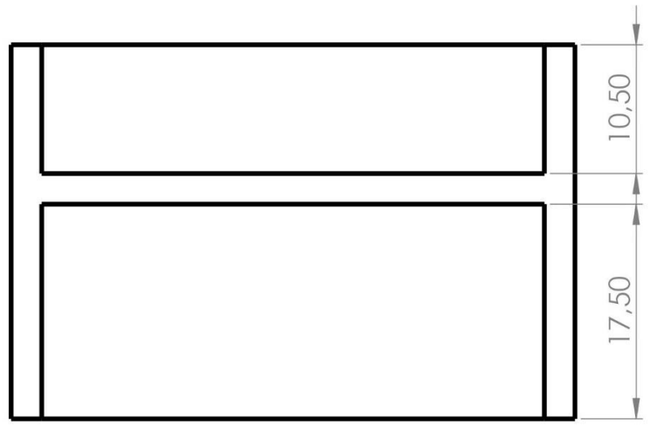
A3

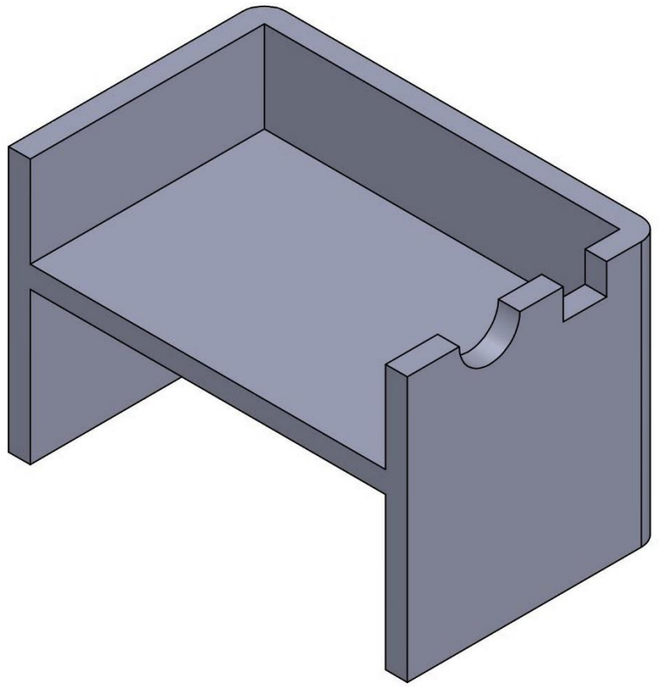


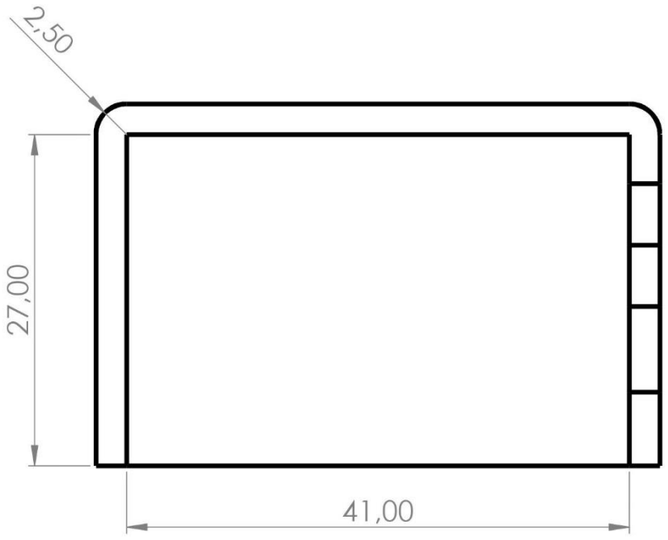


NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.1.1	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 1	REFERENCIA: 140420-2	ORIENTACIÓN:
TÍTULO: SOPORTE INFRARROJOS			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 2:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 22/03/20	HOJA: 1/1 A3

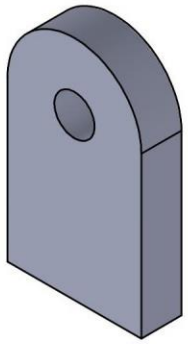


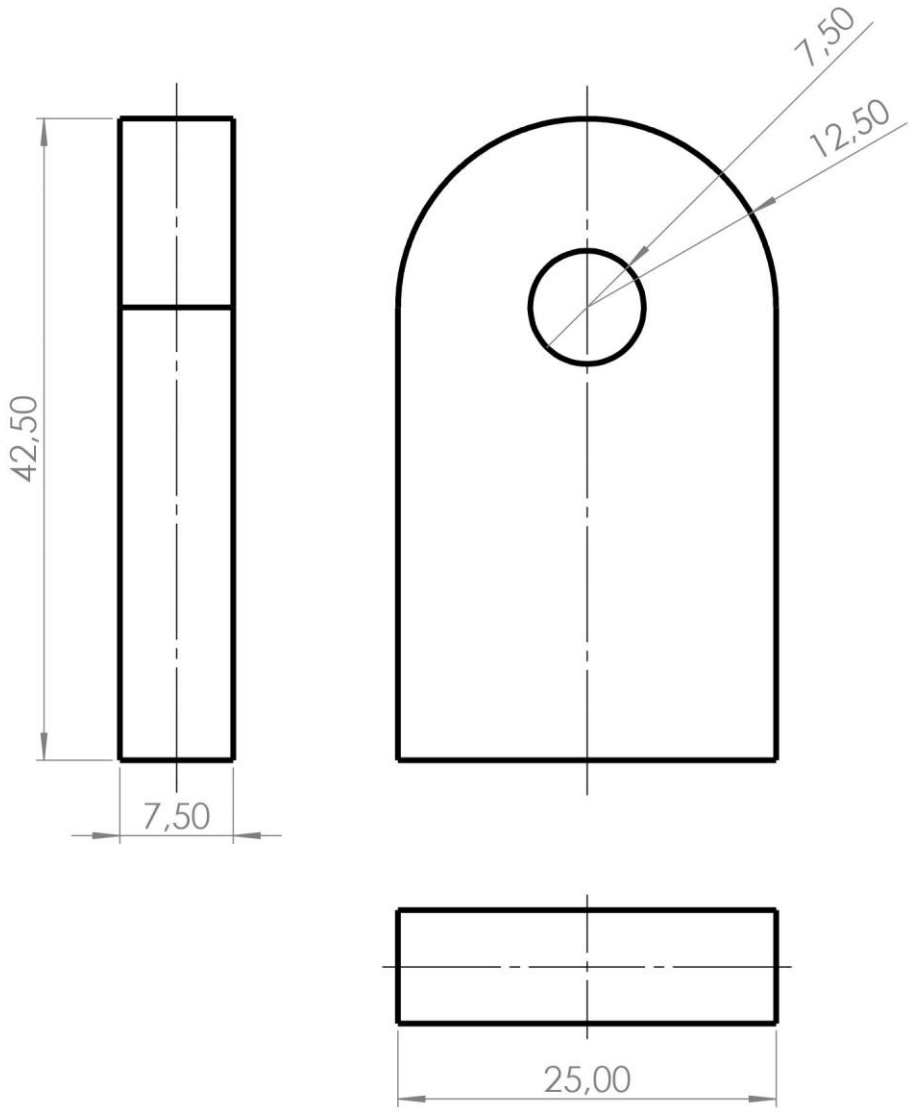




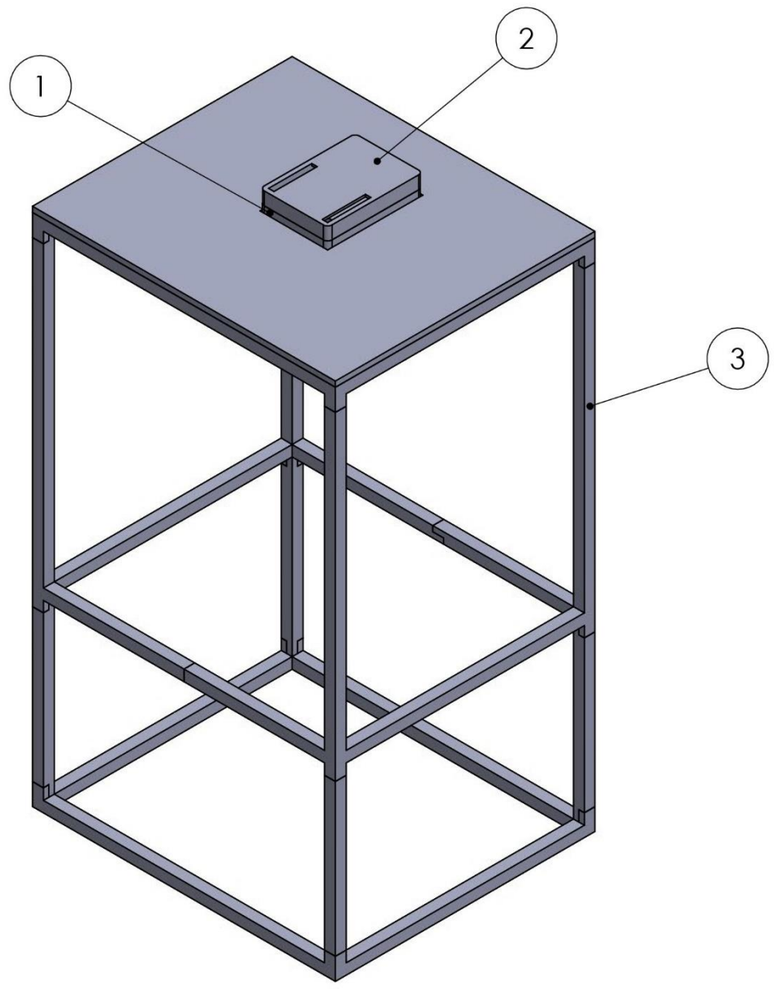


NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.1.2	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 2	REFERENCIA: 220320-4	ORIENTACIÓN:
TÍTULO: SOPORTE MOTOR DS04-NFC			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 2:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 22/03/20	HOJA: 1/1 A3





NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.1.3	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 3	REFERENCIA: 220320-9	ORIENTACIÓN:
TÍTULO: SOPORTE CINTA			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 2:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 22/03/20	HOJA: 1/1 A4



Marca	Nº Piezas	Designación	Norma	Material
3	1	Estructura cámara	UNE-EN ISO 15015	ABS
2	1	Tapa cámara	UNE-EN ISO 15015	ABS
1	1	Base cámara	UNE-EN ISO 15015	ABS

NOMBRE DEL PROYECTO:
DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE

Nº PLANO: 1.4.2.1.4

FIRMA: JORGE DOMÉNECH JARA

TIPO DE DOCUMENTO: VISTA GENERAL

REFERENCIA: 260320-2

ORIENTACIÓN:

TÍTULO:

CÁMARA

CREADO POR: AUTOMA

REVISIÓN: A

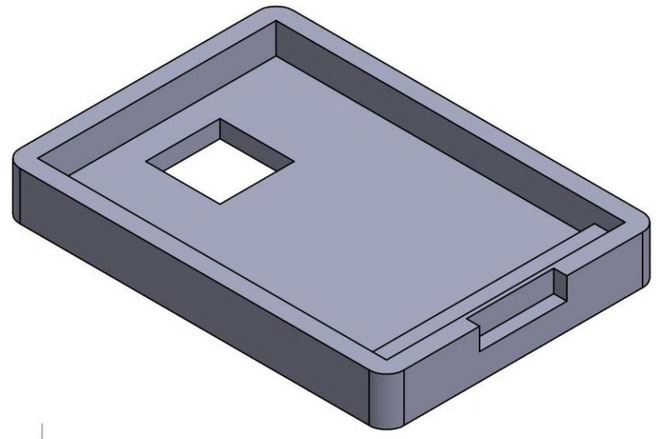
ESCALA: 1:2

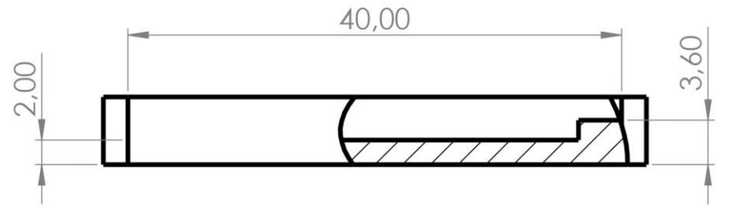
MATERIAL: ABS

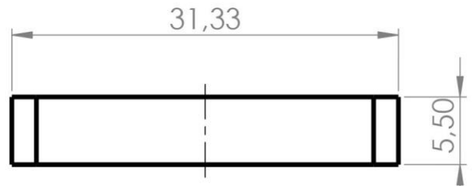
FECHA DE EDICIÓN: 26/03/20

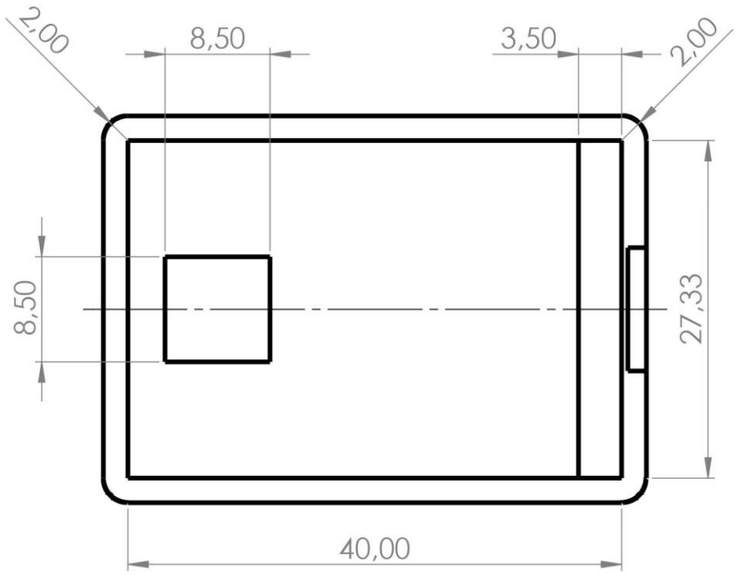
HOJA: 1/1

A3

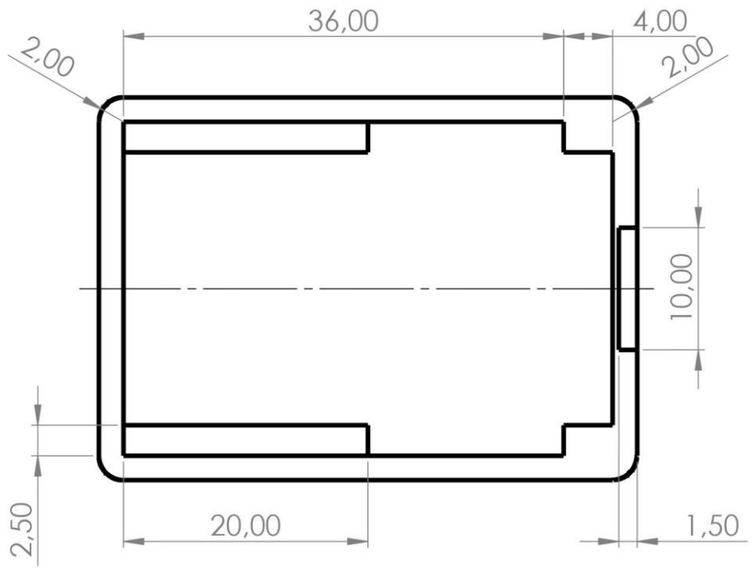


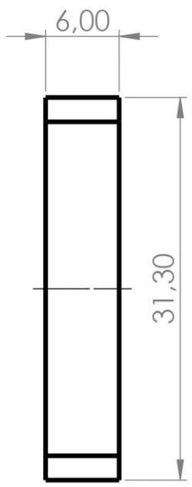


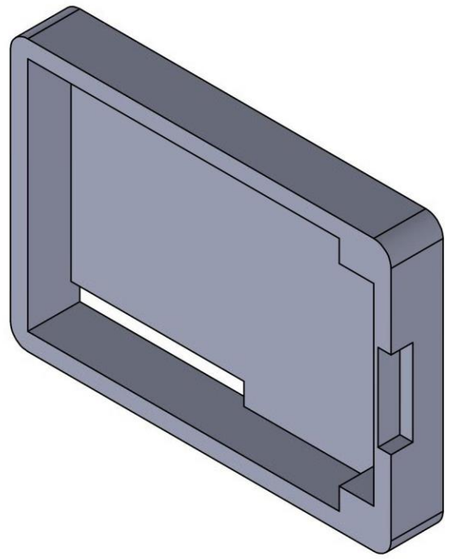


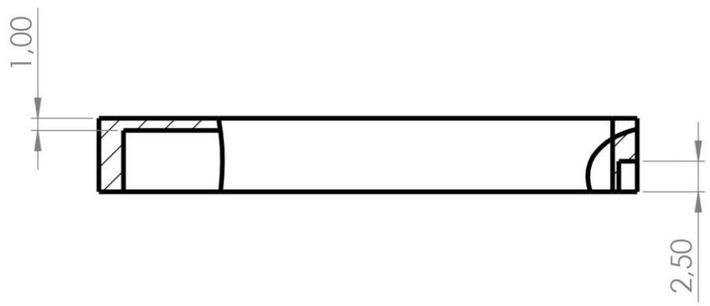


NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.1.4.1	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 1	REFERENCIA: 220320-1	ORIENTACIÓN:
TÍTULO: BASE CÁMARA			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 2:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 22/03/20	HOJA: 1/1 A3

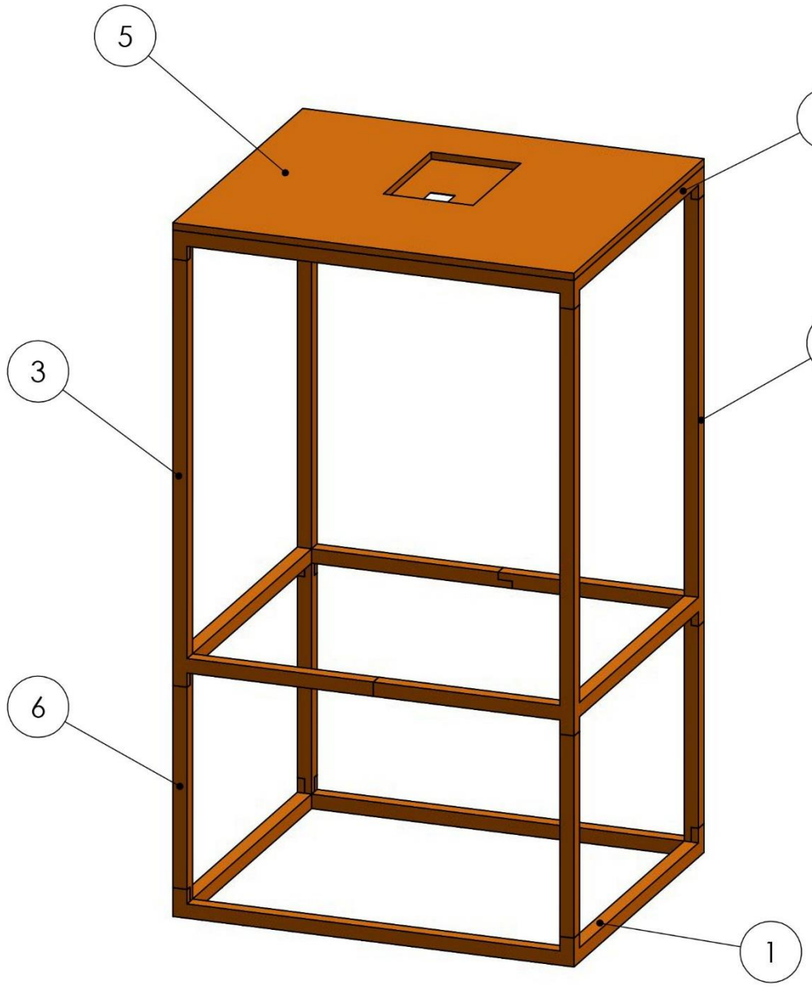








NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.1.4.2	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 2	REFERENCIA: 220320-2	ORIENTACIÓN:
TÍTULO: TAPA CÁMARA			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 2:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 22/03/20	HOJA: 1/1 A3



Marca	Nº Piezas	Designación	Norma	Material
6	4	Pilar	UNE-EN ISO 15015	ABS
5	1	Cubierta	UNE-EN ISO 15015	ABS
4	1	Base alta	UNE-EN ISO 15015	ABS
3	1	Base media 2	UNE-EN ISO 15015	ABS
2	1	Base media 1	UNE-EN ISO 15015	ABS
1	1	Base baja	UNE-EN ISO 15015	ABS

NOMBRE DEL PROYECTO:
DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE

Nº PLANO: 1.4.2.1.4.3

FIRMA: JORGE DOMÉNECH JARA

TIPO DE DOCUMENTO: Dibujo de subconjunto

REFERENCIA: 260320-1

ORIENTACIÓN:

TÍTULO:

ESTRUCTURA CÁMARA

CREADO POR: AUTOMA

REVISIÓN: A

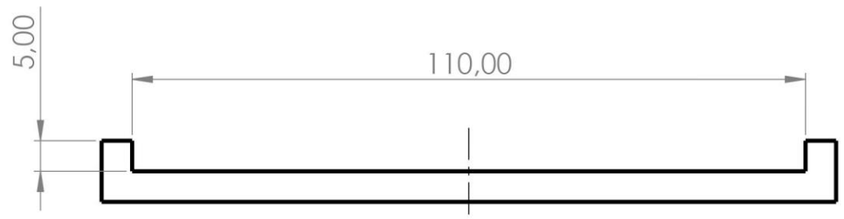
ESCALA: 1:2

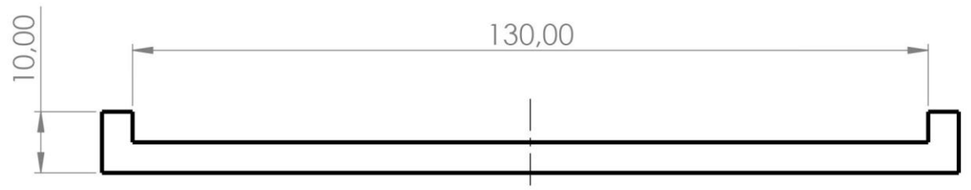
MATERIAL: ABS

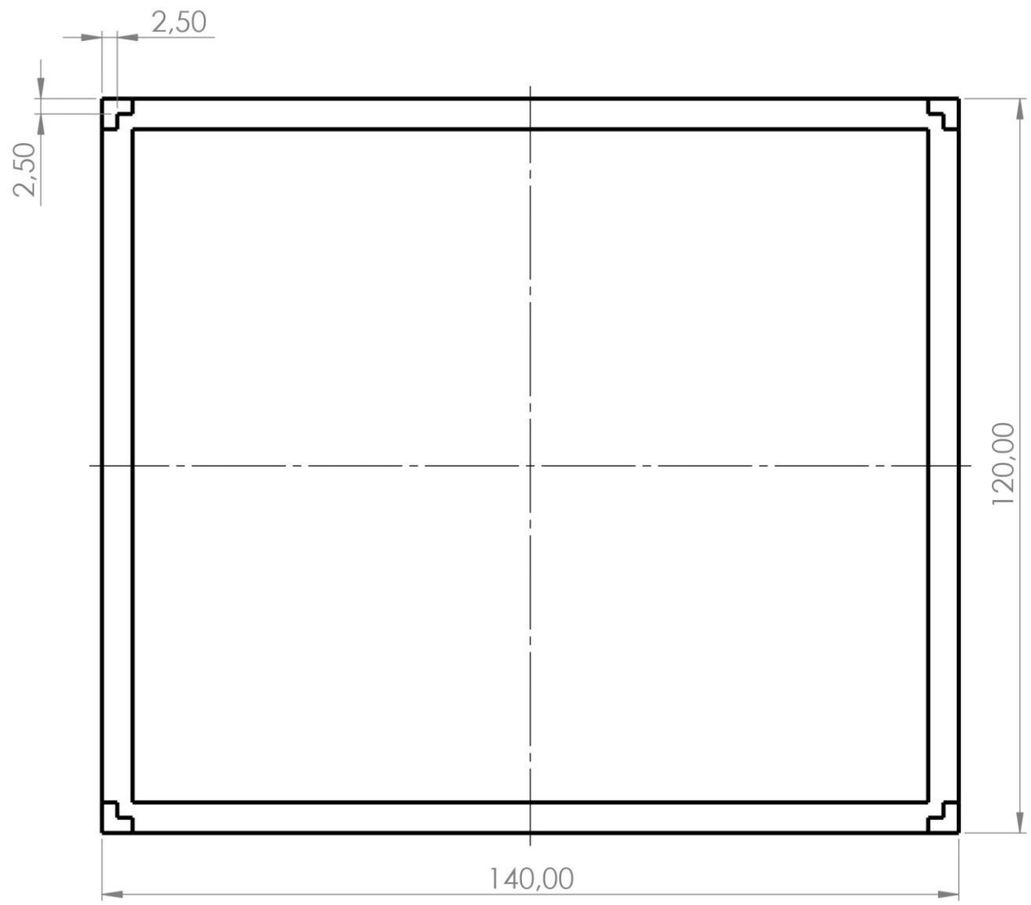
FECHA DE EDICIÓN: 26/03/20

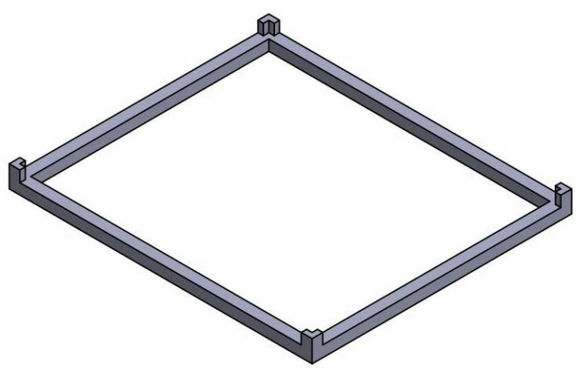
HOJA: 1/1

A3

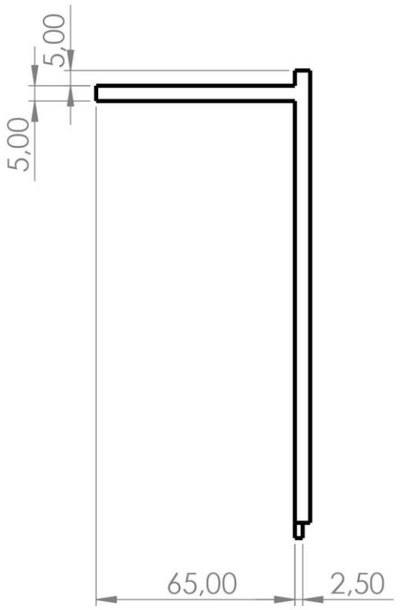


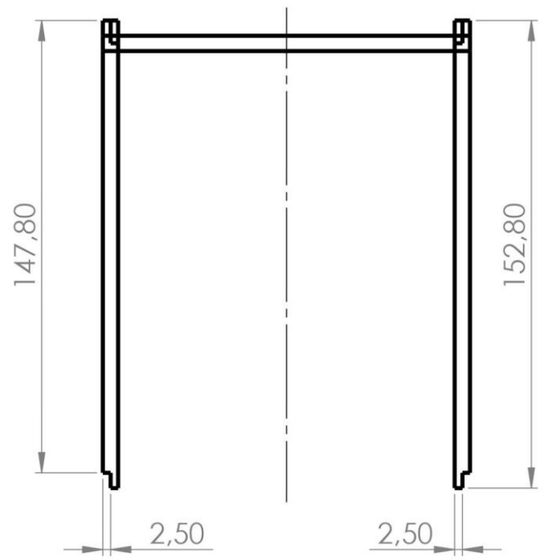


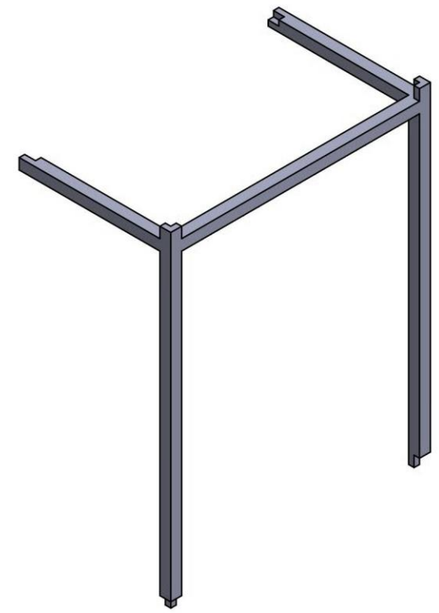


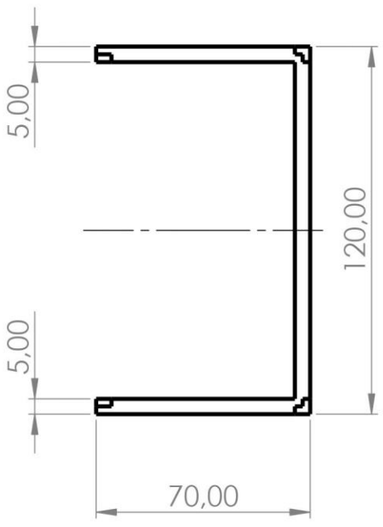


NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.1.4.3.1	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 1	REFERENCIA: 220320-12	ORIENTACIÓN:
TÍTULO: BASE BAJA			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 1:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 22/03/20	HOJA: 1/1 A3

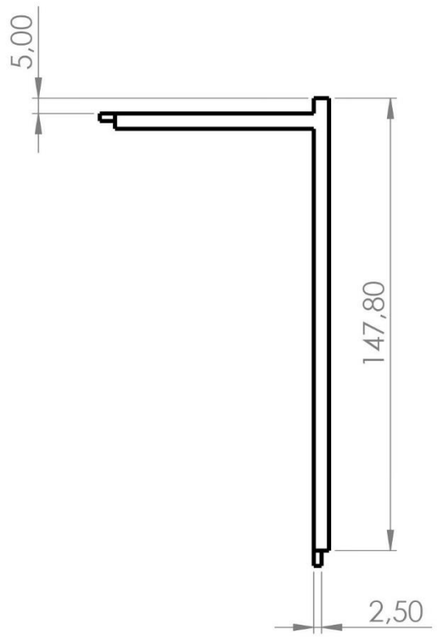


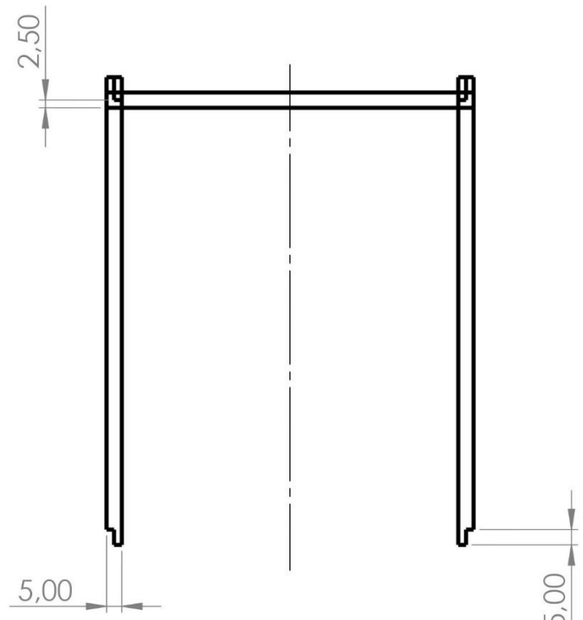


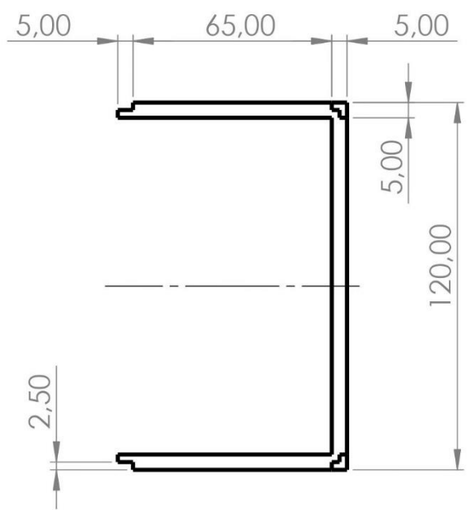




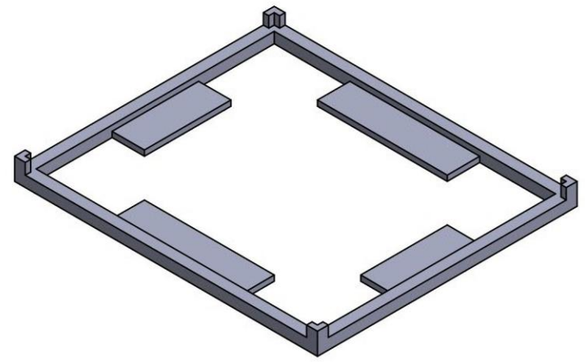
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.1.4.3.2	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 2	REFERENCIA: 220320-13	ORIENTACIÓN:
TÍTULO: BASE MEDIA 1			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 1:2	MATERIAL: ABS
			FECHA DE EDICIÓN: 22/03/20	HOJA: 1/1 A3



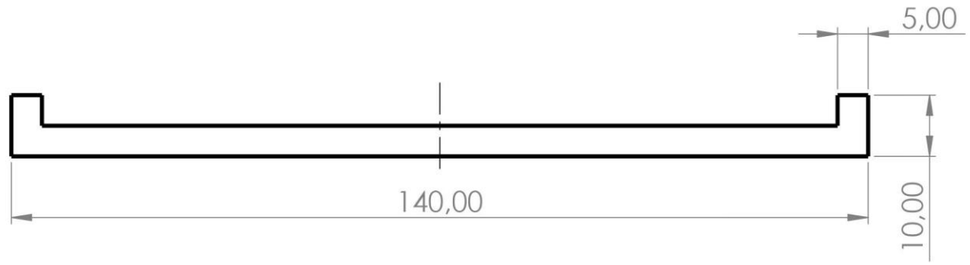


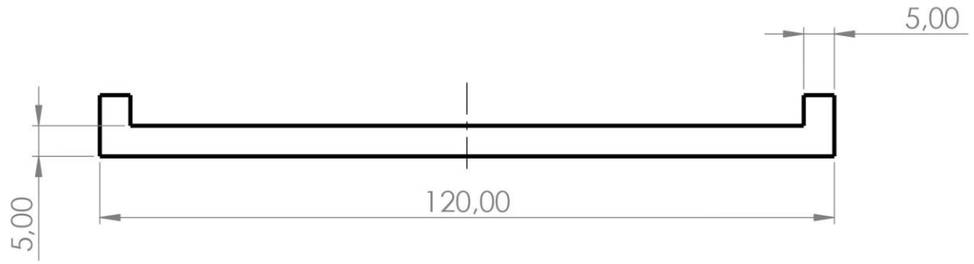


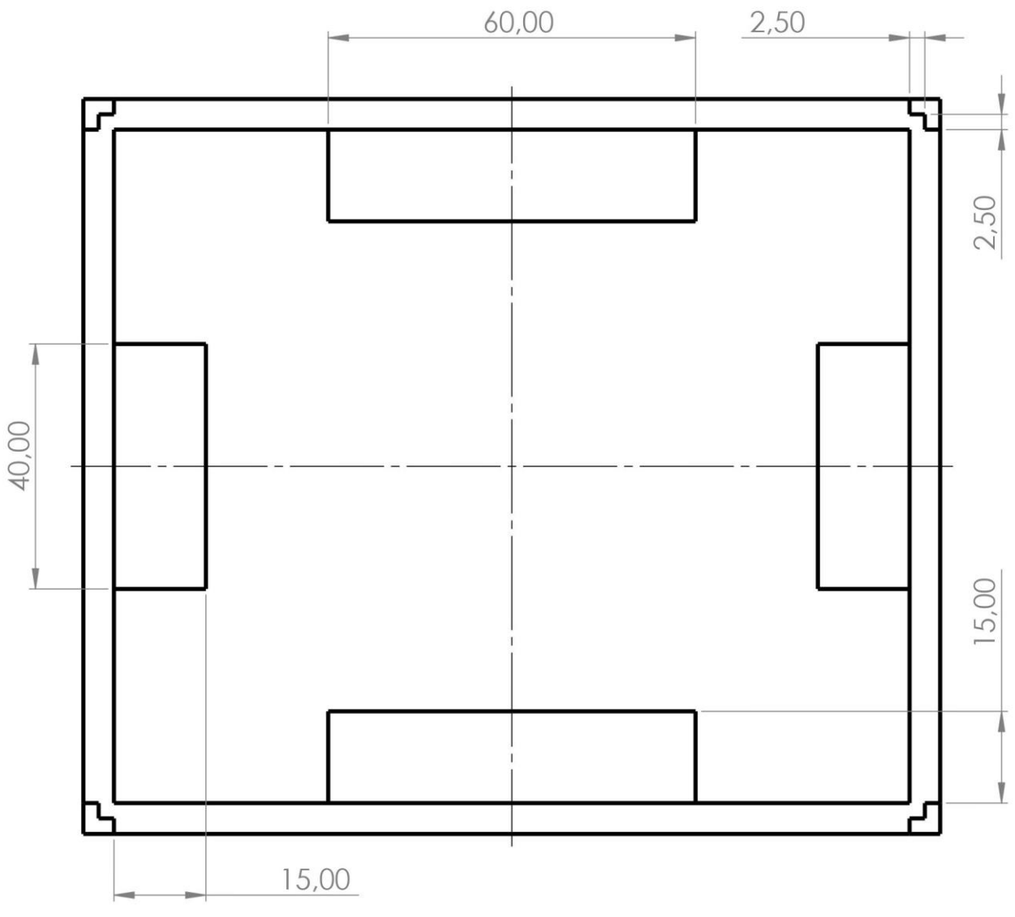
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.1.4.3.3	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 3	REFERENCIA: 220320-14	ORIENTACIÓN:
TÍTULO: BASE MEDIA 2			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 1:2	MATERIAL: ABS
			FECHA DE EDICIÓN: 22/03/20	HOJA: 1/1 A3



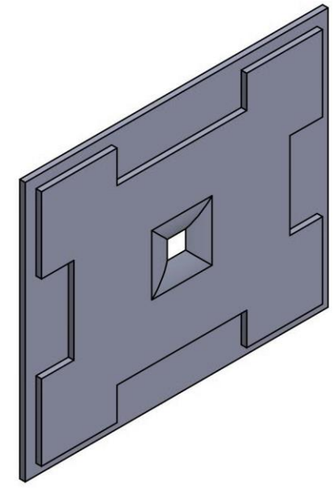
VIEW

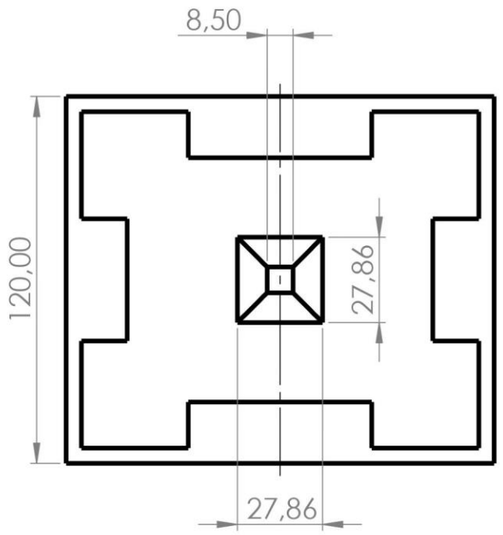


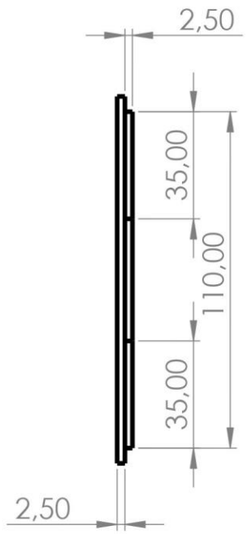


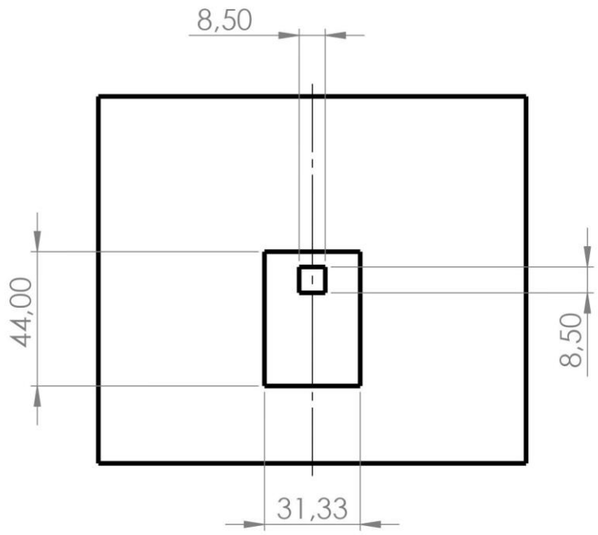


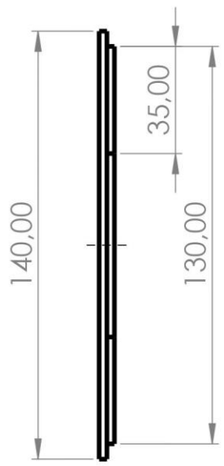
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.1.4.3.4	
FIRMA: JORGE DOMÈNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 4	REFERENCIA: 220320-15	ORIENTACIÓN:
TÍTULO: BASE ALTA			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 1:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 22/03/20	HOJA: 1/1 A3



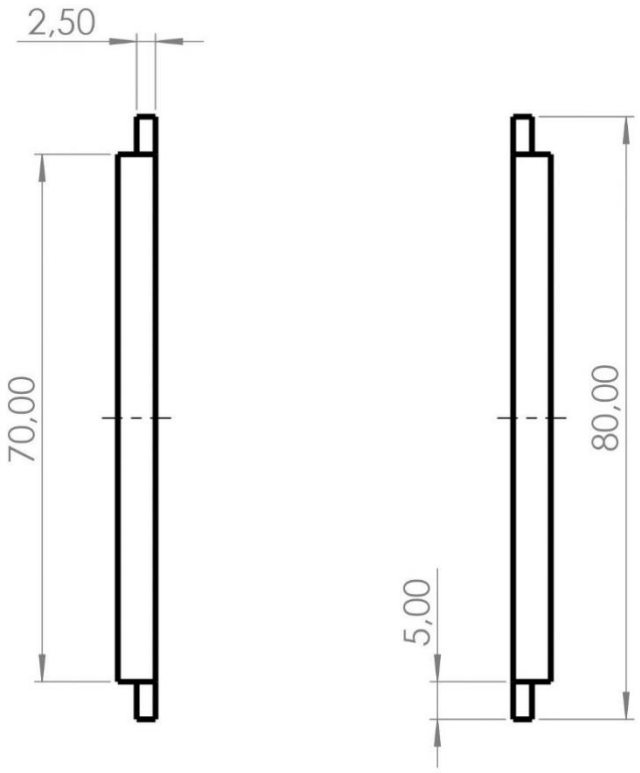


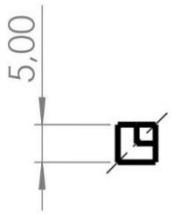




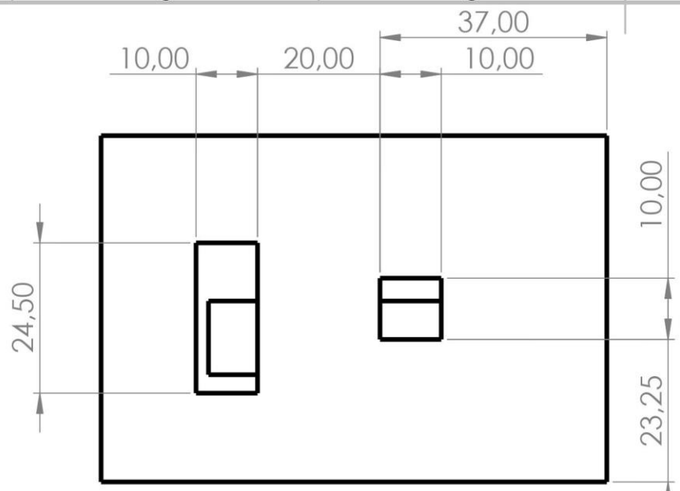


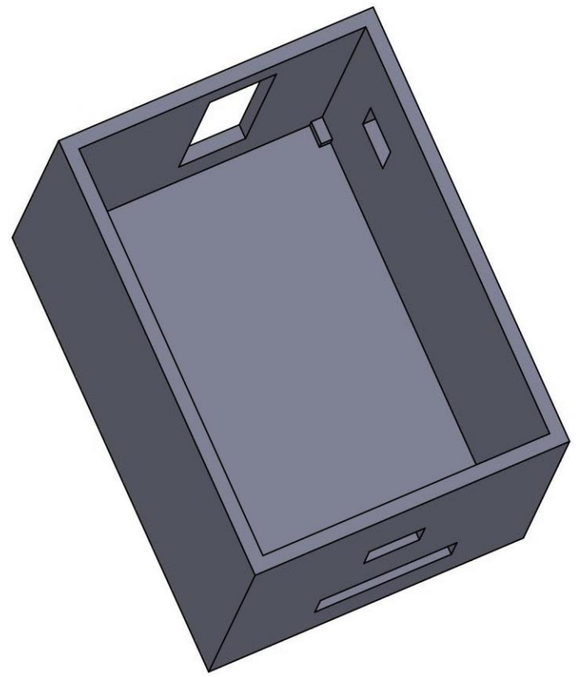
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.1.4.3.5	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 5	REFERENCIA: 220320-16	ORIENTACIÓN:
TÍTULO: CUBIERTA			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 1:2	MATERIAL: ABS
			FECHA DE EDICIÓN: 22/03/20	HOJA: 1/1 A3

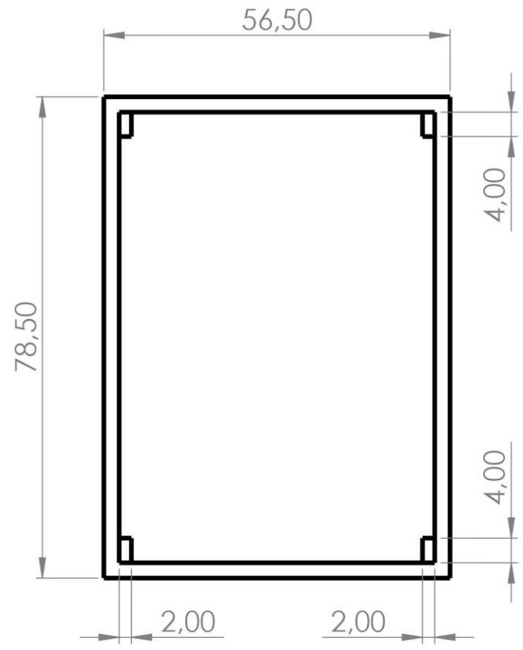


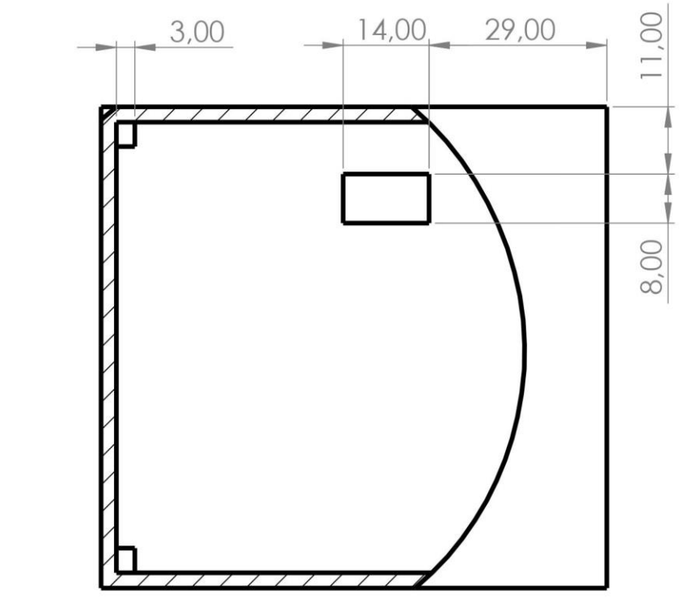


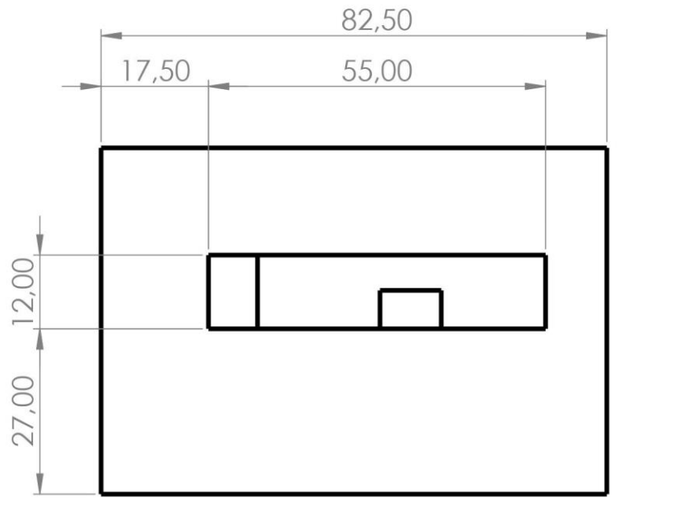
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.1.4.3.6	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 6	REFERENCIA: 220320-17	ORIENTACIÓN:
TÍTULO: PILAR			CREADO POR: AUTOMA	REVISIÓN: B
			ESCALA: 1:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 22/03/20	HOJA: 1/1 A3



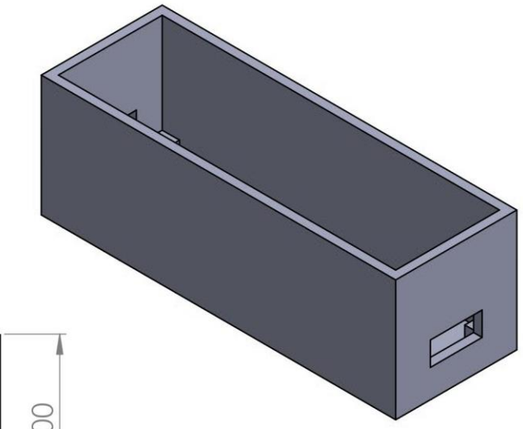


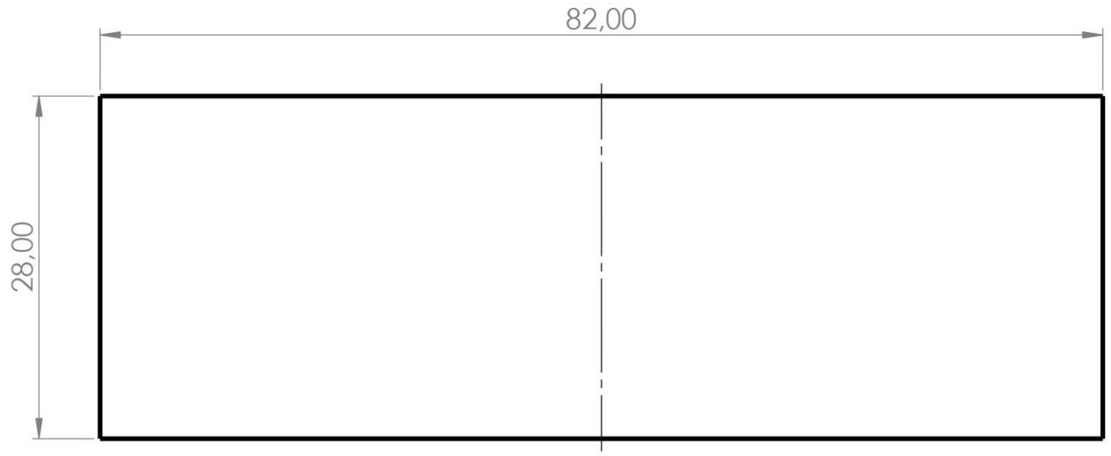


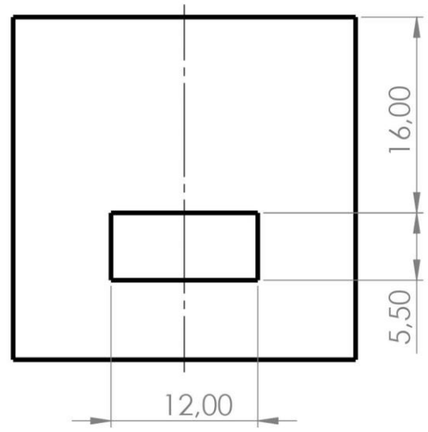


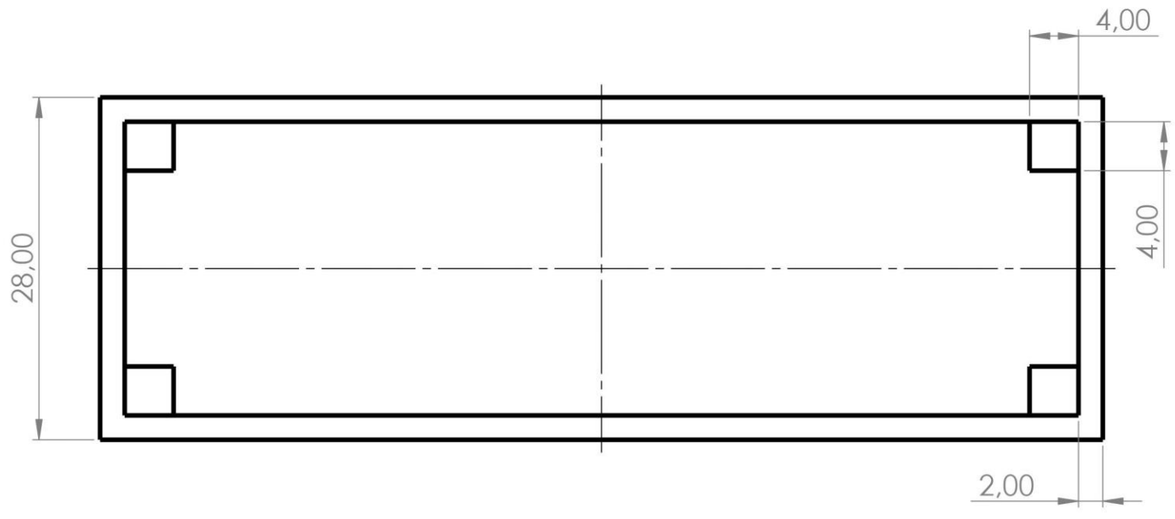


NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.1.5	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 5	REFERENCIA: 120420-1	ORIENTACIÓN:
TÍTULO: SOPORTE ESP32			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 1:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 12/04/20	HOJA: 1/1 A3

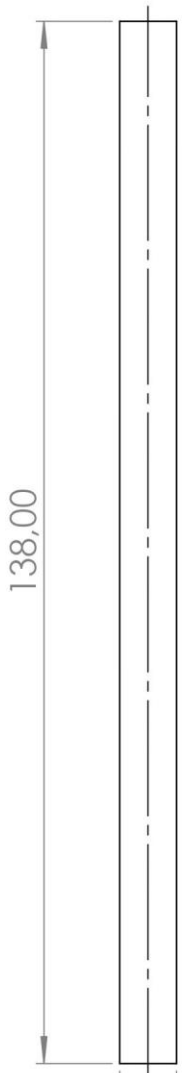




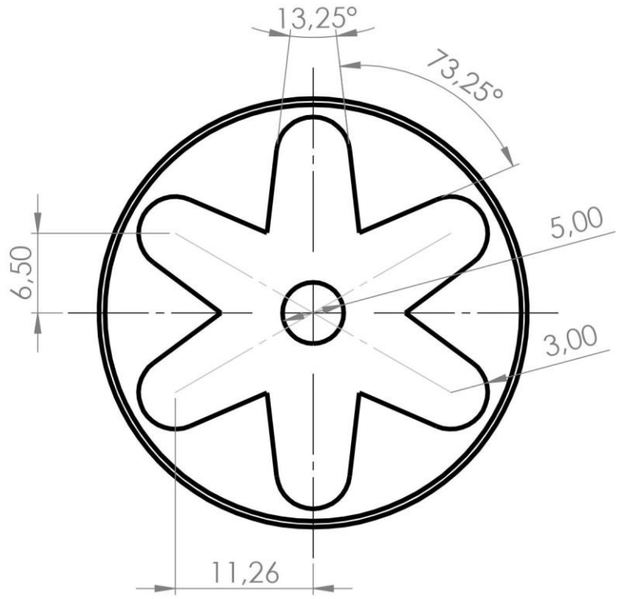


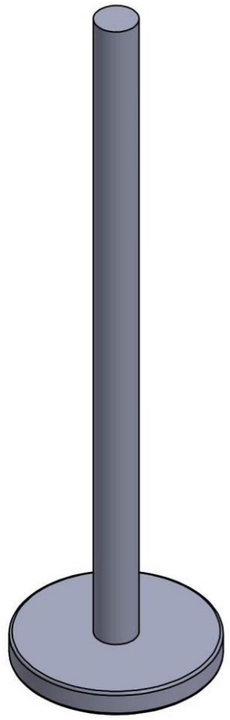


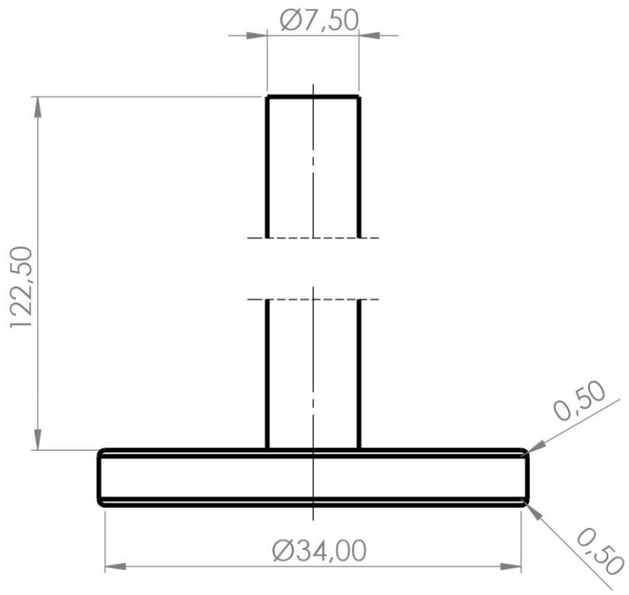
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.1.6	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 6	REFERENCIA: 110420-1	ORIENTACIÓN:
TÍTULO: SOPORTE BUCK			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 2:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 11/04/20	HOJA: 1/1 A3



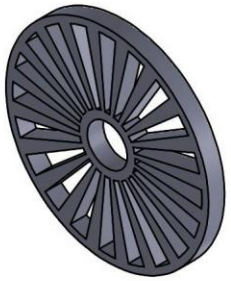
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.2.1	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 1	REFERENCIA: 220320-18	ORIENTACIÓN:
TÍTULO: EJE			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 1:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 22/03/20	HOJA: 1/1 A4

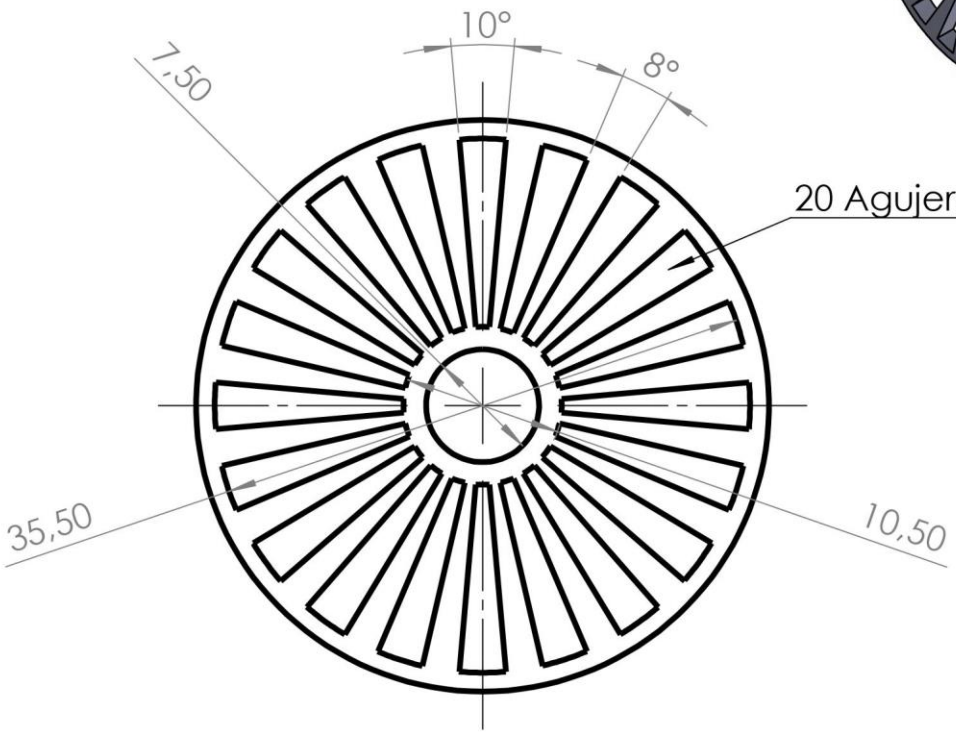


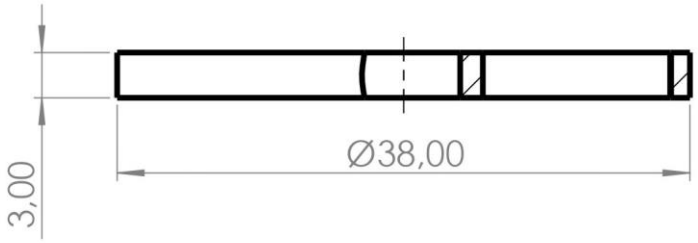




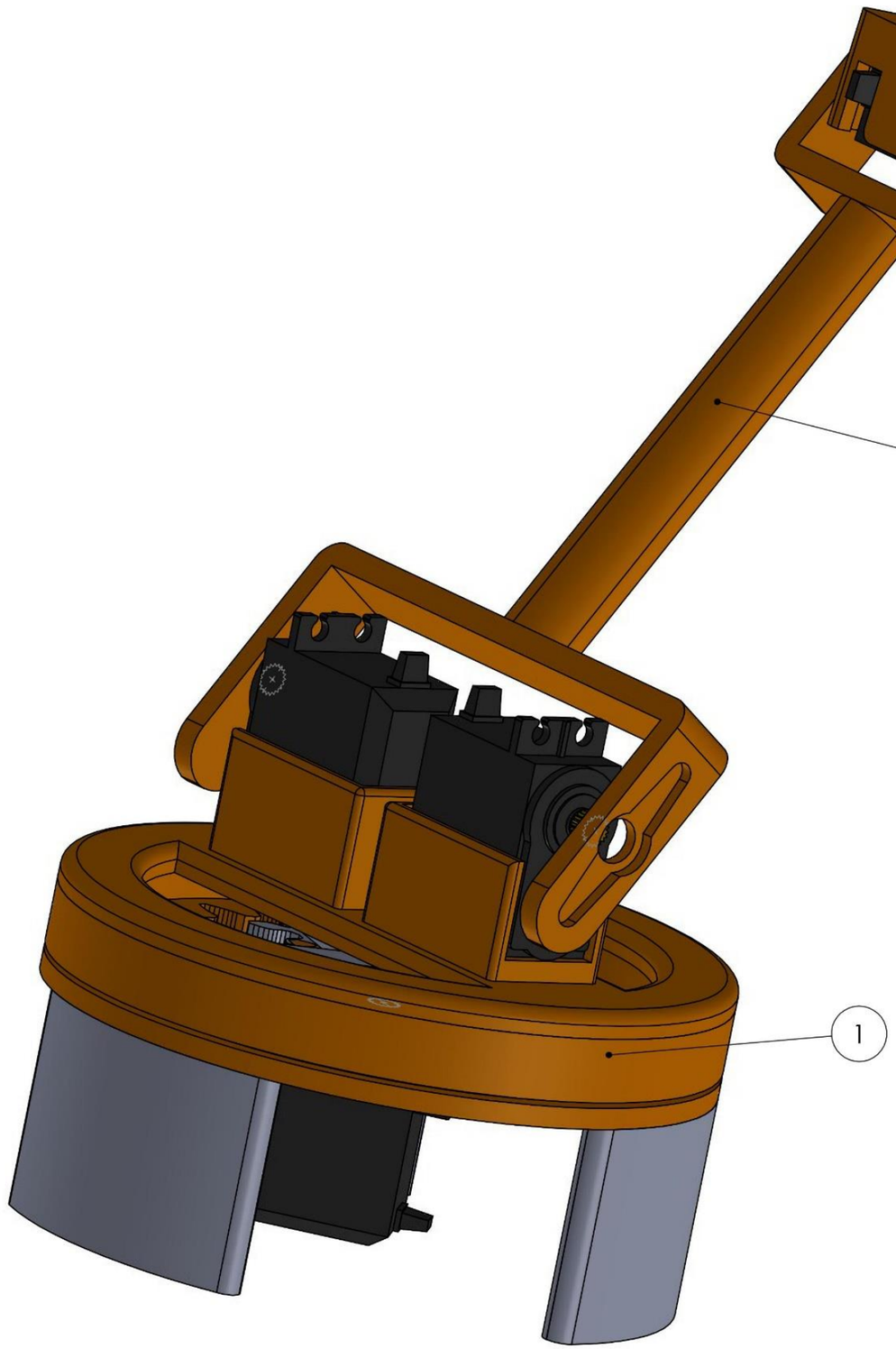
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.2.2	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 2	REFERENCIA: 220320-11	ORIENTACIÓN:
TÍTULO: EJE CON ADAPTADOR			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 2:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 22/03/20	HOJA: 1/1 A3

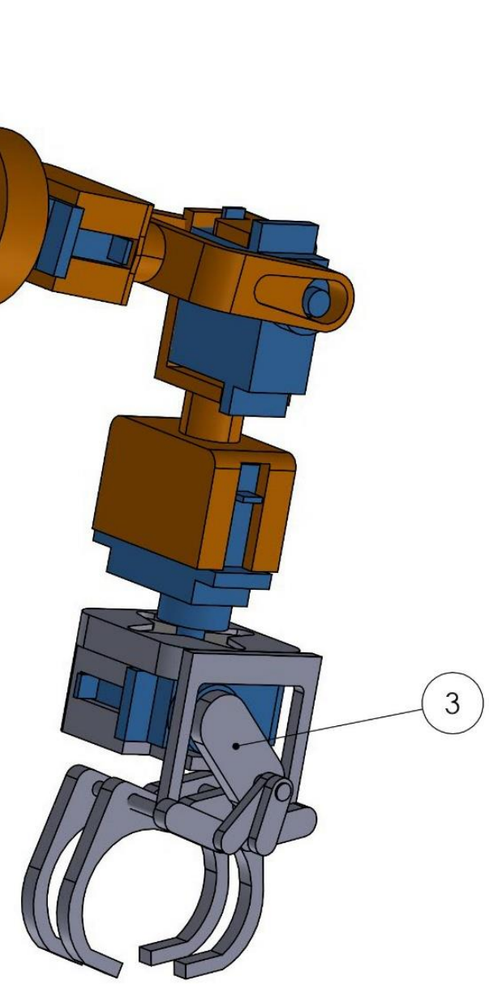






NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.3	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 3	REFERENCIA: 220320-8	ORIENTACIÓN:
TÍTULO: RUEDA DEL ENCODER			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 2:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 22/03/20	HOJA: 1/1 A4





Marca	Nº Piezas	Designación	Norma	Material
3	1	Pinza	UNE-EN ISO 15015	ABS
2	1	Eslabones	UNE-EN ISO 15015	ABS
1	1	Base - planetario	UNE-EN ISO 15015	ABS

NOMBRE DEL PROYECTO:
DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE

Nº PLANO: 1.4.2.4

FIRMA: JORGE DOMÉNECH JARA

TIPO DE DOCUMENTO: Dibujo de subconjunto

REFERENCIA: 120420-2

ORIENTACIÓN:

TÍTULO:

BRAZO ROBÓTICO

CREADO POR: AUTOMA

REVISIÓN: A

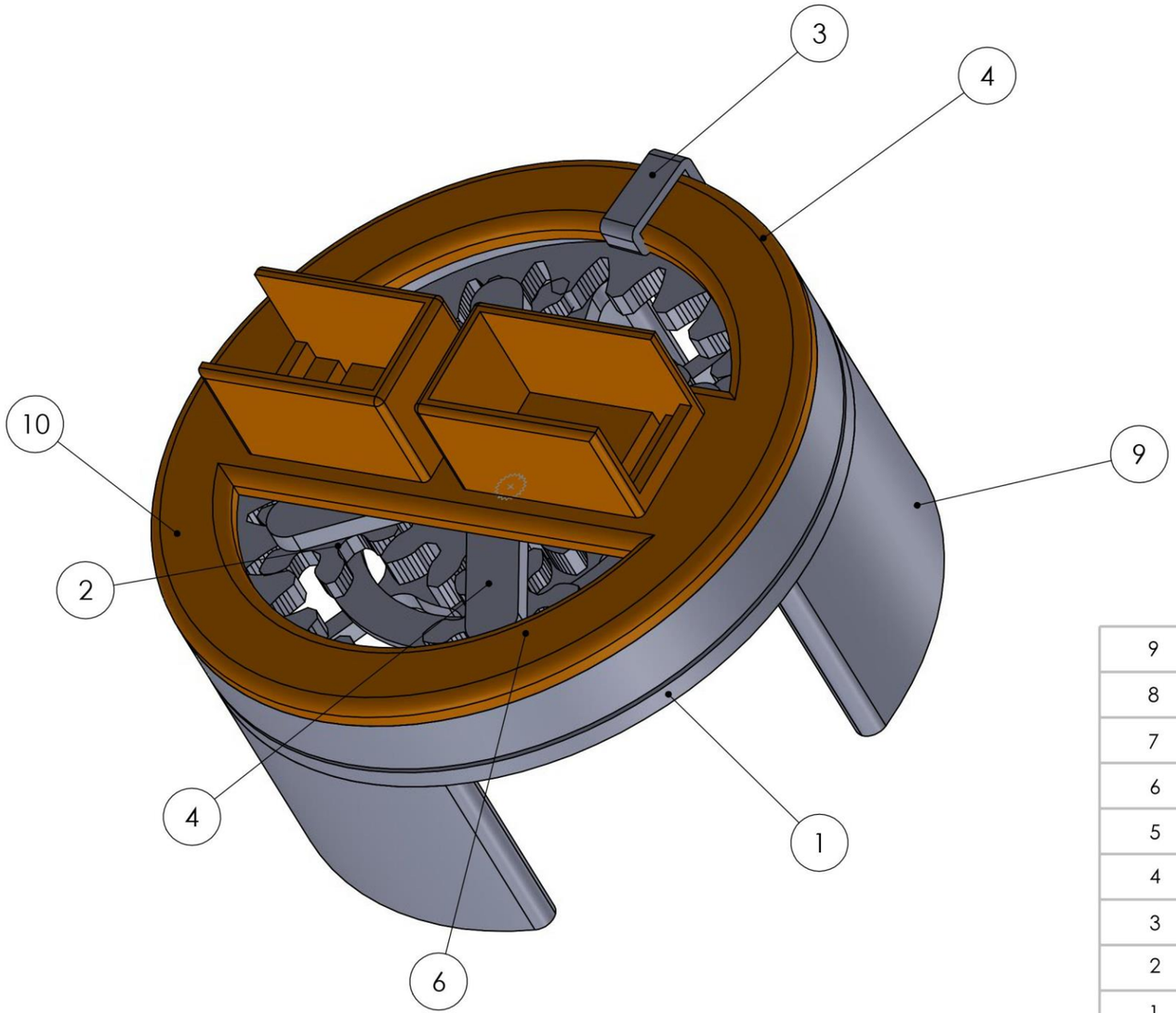
ESCALA: 1:1

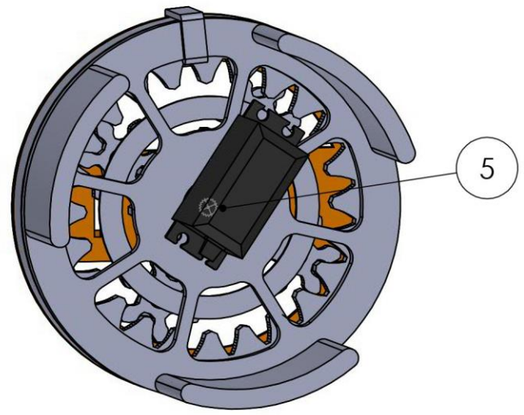
MATERIAL: ABS

FECHA DE EDICIÓN: 26/03/20

HOJA: 1/1

A3





9	3	Soporte	UNE-EN ISO 15015	ABS
8	1	Suelo	UNE-EN ISO 15015	ABS
7	1	Sol	UNE-EN ISO 15015	ABS
6	4	Luna	UNE-EN ISO 15015	ABS
5	1	Extensor	UNE-EN ISO 15015	ABS
4	1	Corona	UNE-EN ISO 15015	ABS
3	1	Abrazadera	UNE-EN ISO 15015	ABS
2	1	Anilla	UNE-EN ISO 15015	ABS
1	1	Base planetario	UNE-EN ISO 15015	ABS
Marca	Nº Piezas	Designación	Norma	Material

NOMBRE DEL PROYECTO:
DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE

Nº PLANO: 1.4.2.4.1

FIRMA: JORGE DOMÉNECH JARA

TIPO DE DOCUMENTO: Dibujo de subconjunto

REFERENCIA: 260320-1

ORIENTACIÓN:

TÍTULO:

BASE - PLANETARIO

CREADO POR: AUTOMA

REVISIÓN: A

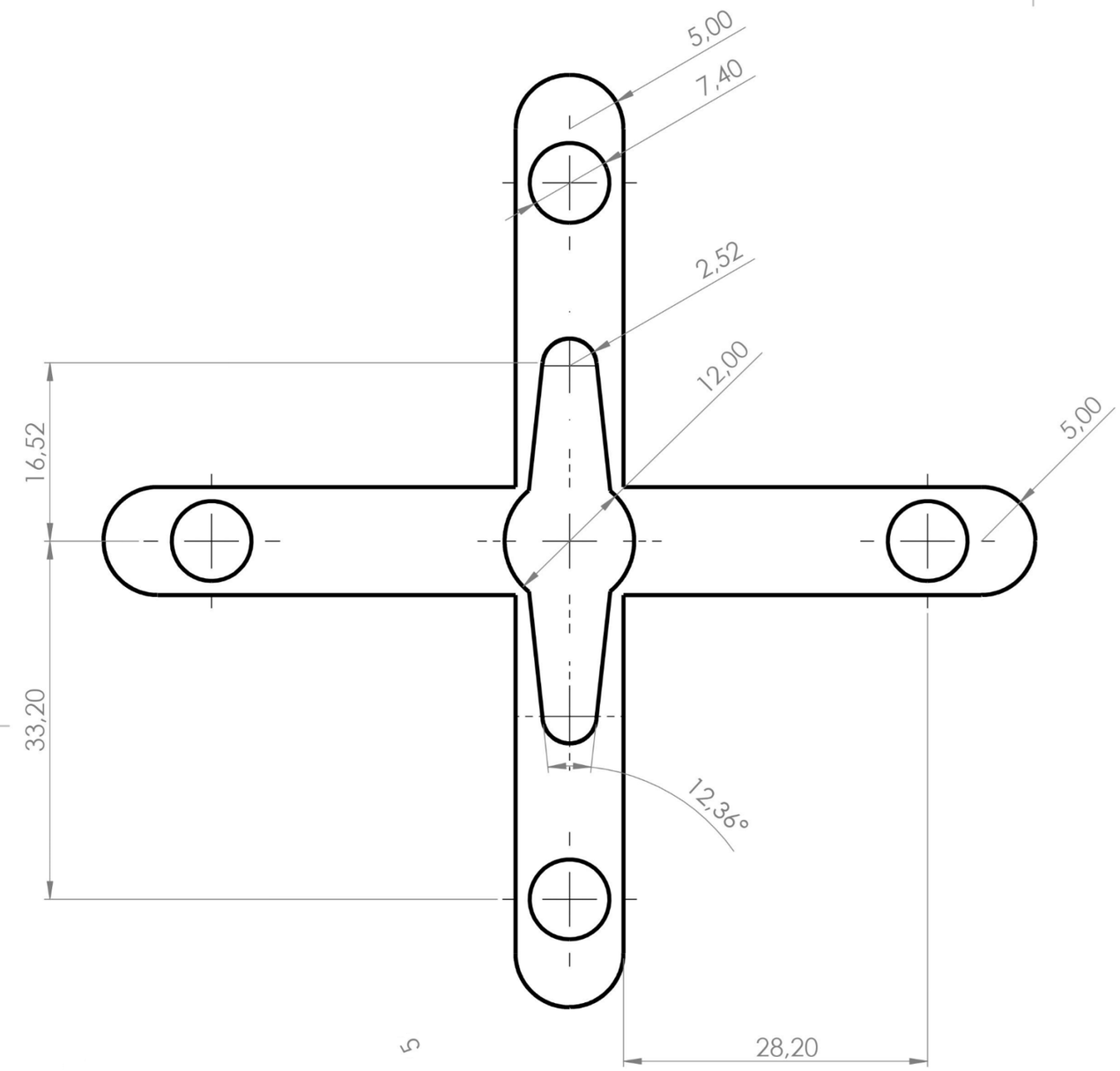
ESCALA: 1:1

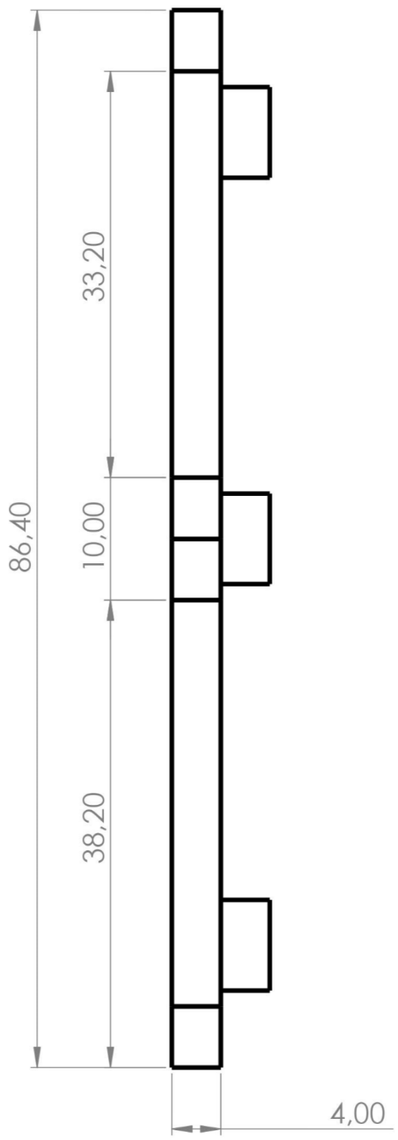
MATERIAL: ABS

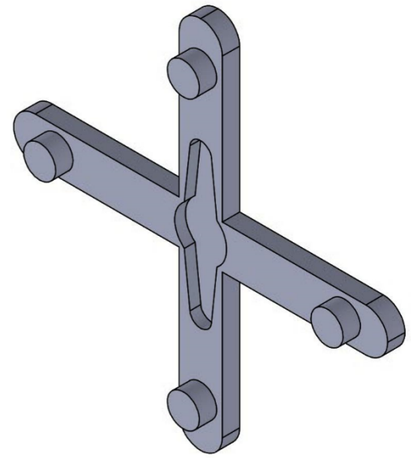
FECHA DE EDICIÓN: 26/03/20

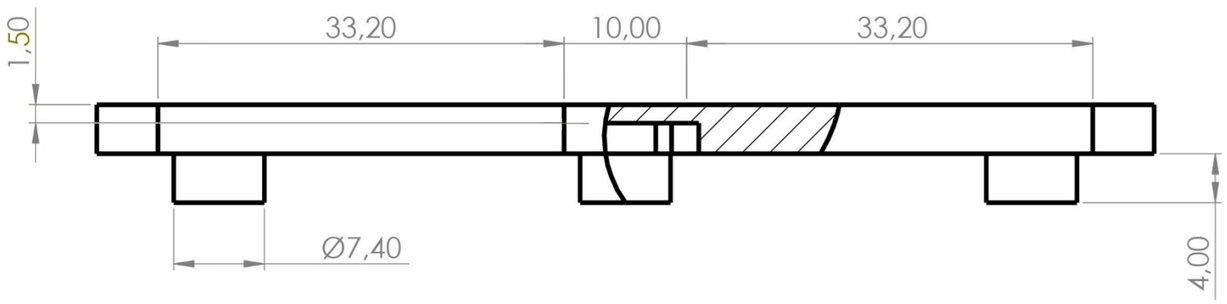
HOJA: 1/1

A3

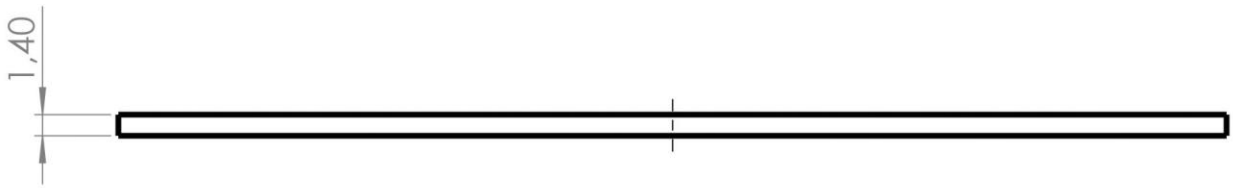




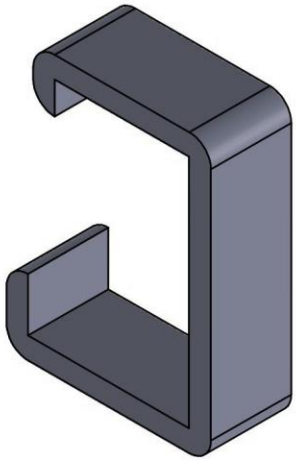


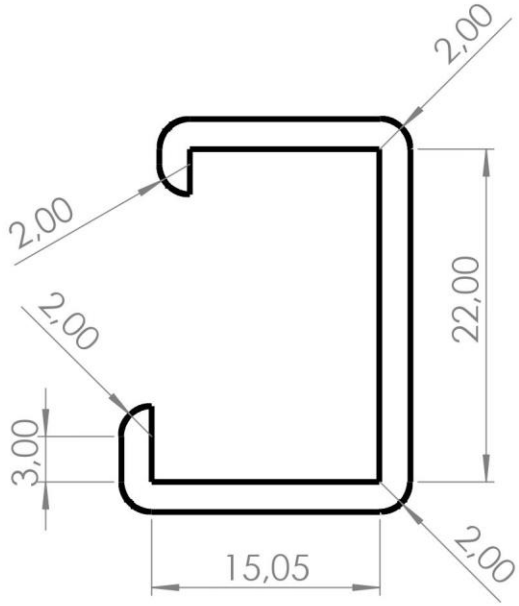


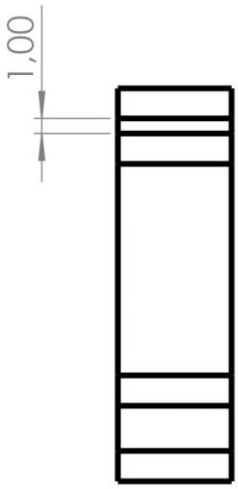
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.4.1.1	
FIRMA: JORGE DOMÊNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 1	REFERENCIA: 230320-1	ORIENTACIÓN:
TÍTULO: BASE PLANETARIO			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 2:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 23/03/20	HOJA: 1/1 A3

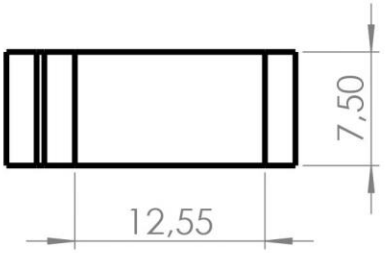


NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.4.1.2	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 2	REFERENCIA: 230320-2	ORIENTACIÓN:
TÍTULO: ANILLA			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 2:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 23/03/20	HOJA: 1/1 A4

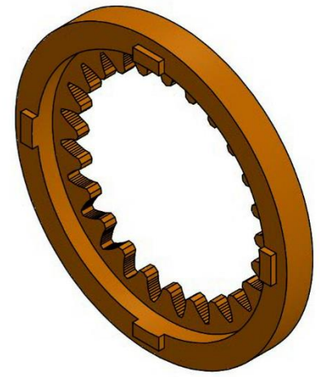


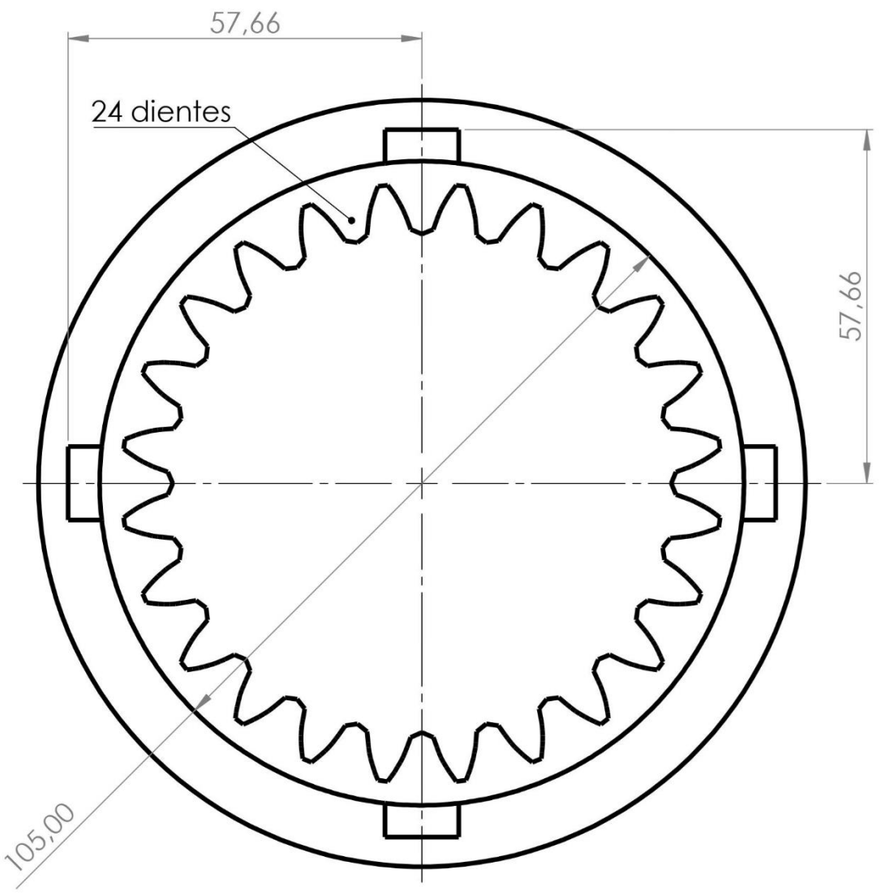


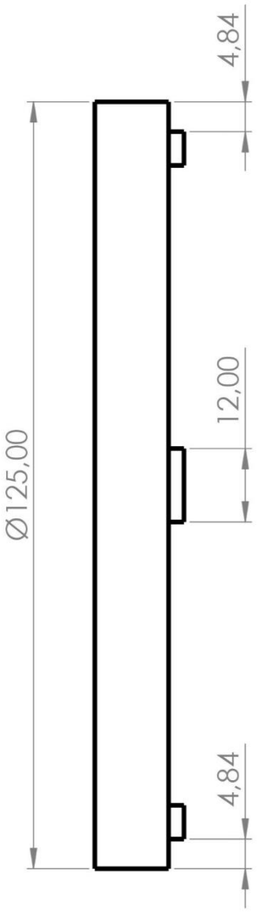


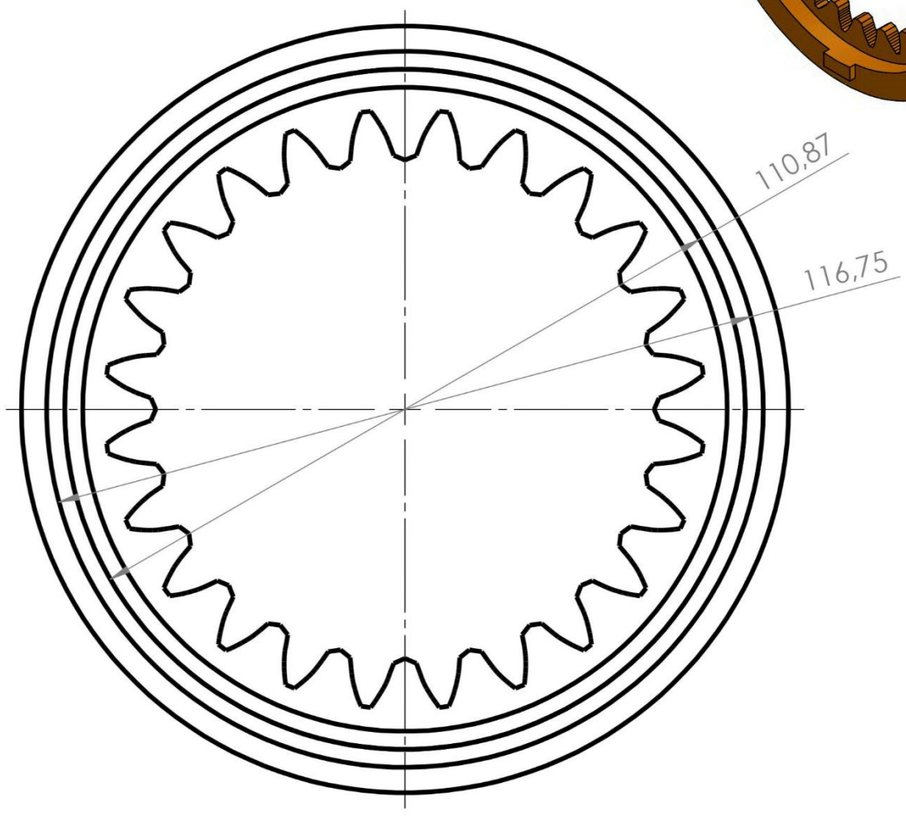


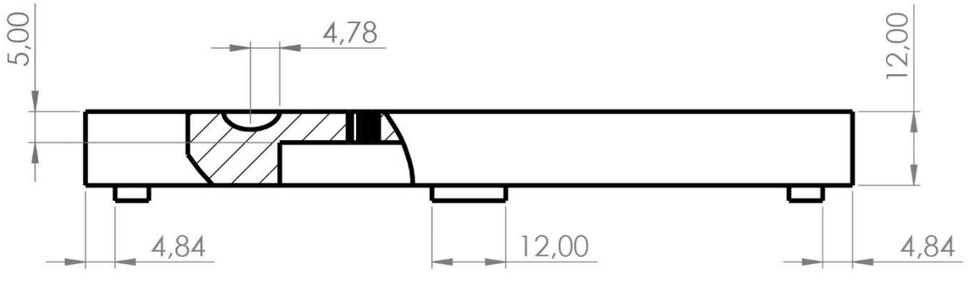
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.4.1.3	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 3	REFERENCIA: 230320-3	ORIENTACIÓN:
TÍTULO: ABRAZADERA			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 2:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 23/03/20	HOJA: 1/1 A4



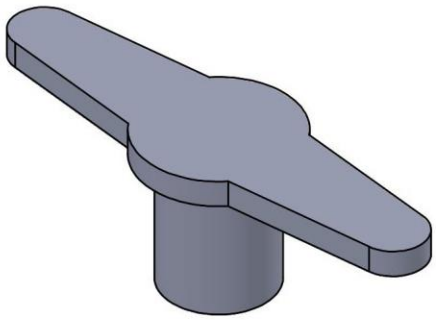


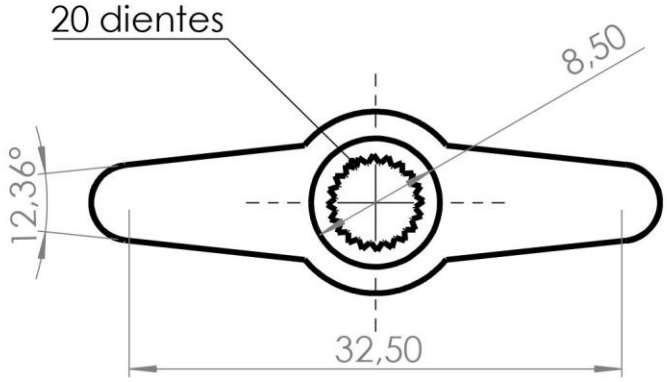


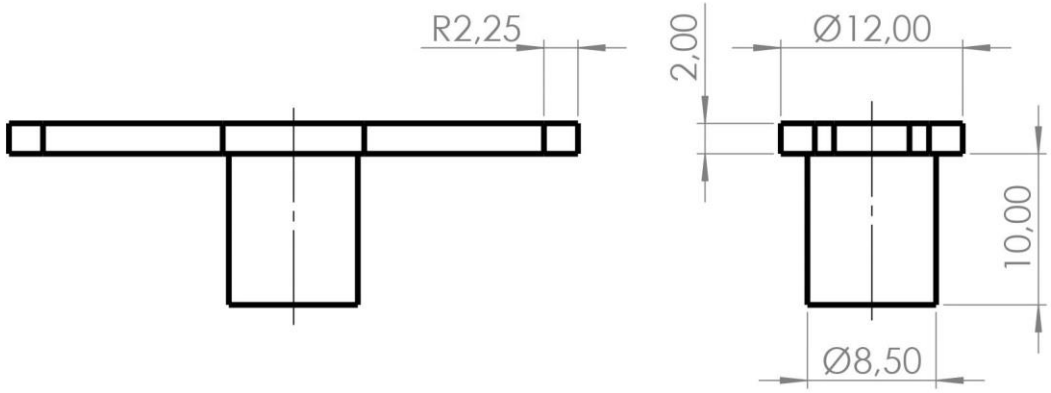




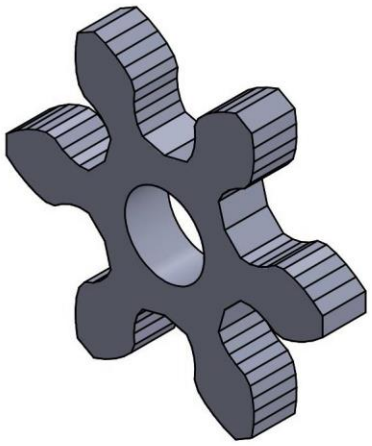
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.4.1.4	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 4	REFERENCIA: 230320-4	ORIENTACIÓN:
TÍTULO: CORONA			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 1:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 23/03/20	HOJA: 1/1 A3

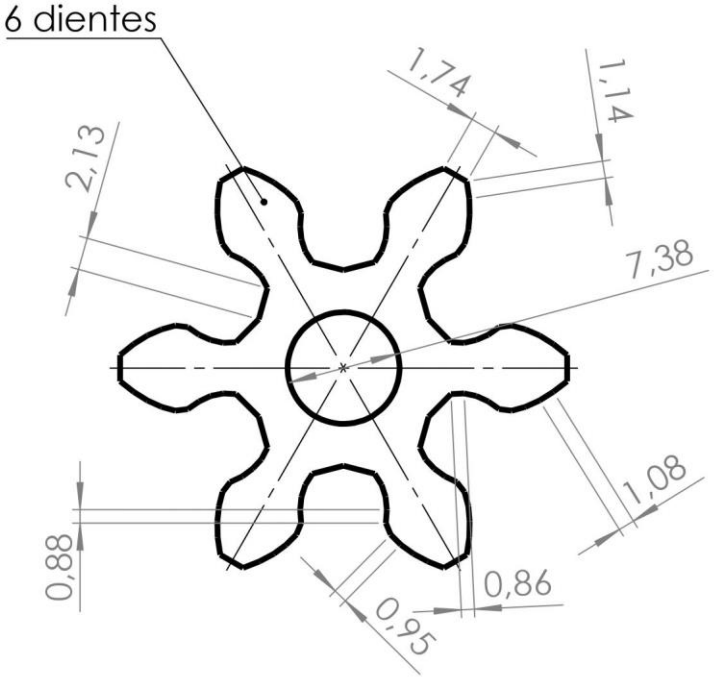


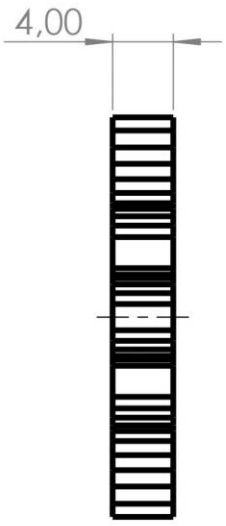




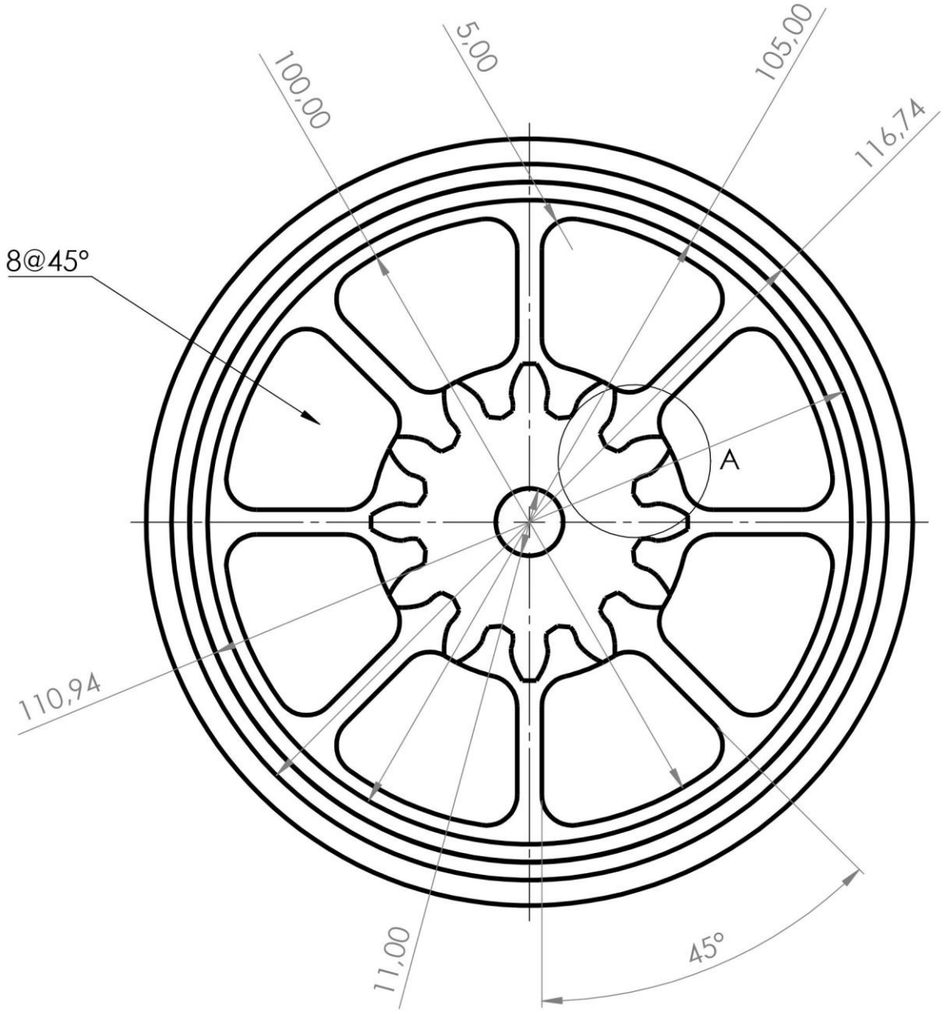
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.4.1.5	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 5	REFERENCIA: 230320-5	ORIENTACIÓN:
TÍTULO: EXTENSOR			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 5:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 23/03/20	HOJA: 1/1 A4

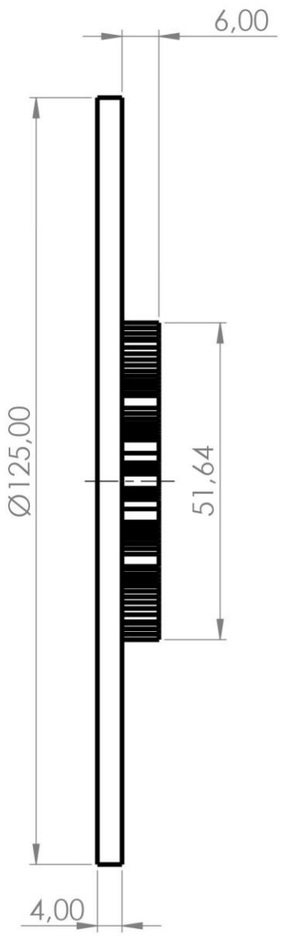


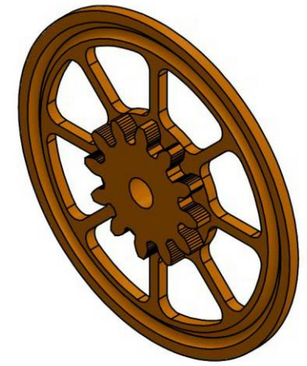


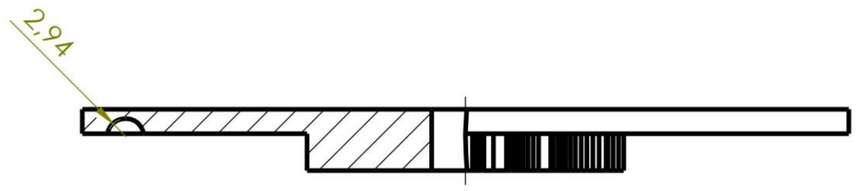


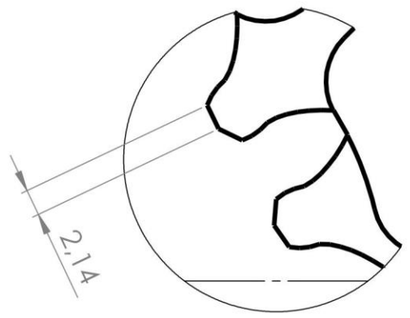
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.4.1.6	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 6	REFERENCIA: 230320-6	ORIENTACIÓN:
TÍTULO: LUNA			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 2:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 23/03/20	HOJA: 1/1 A4





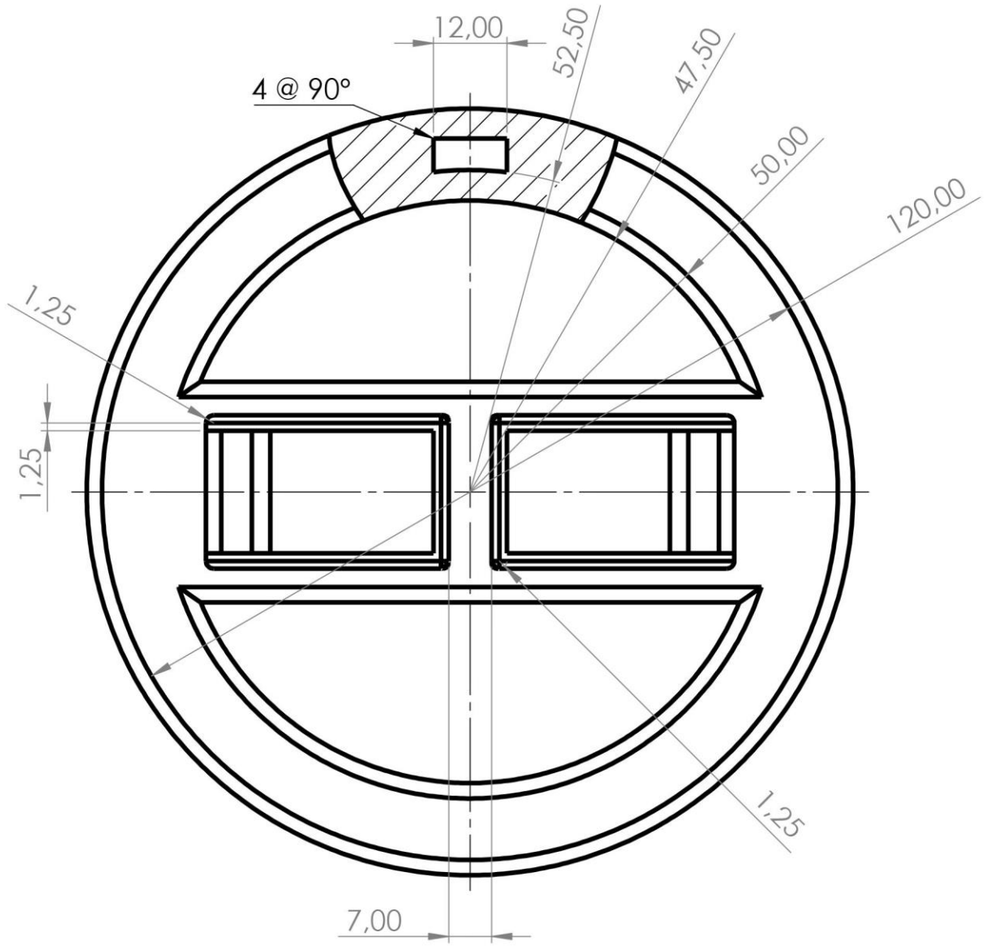


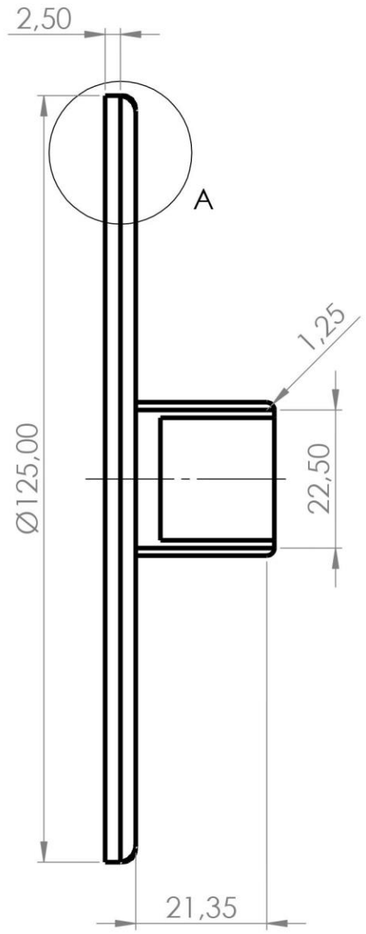


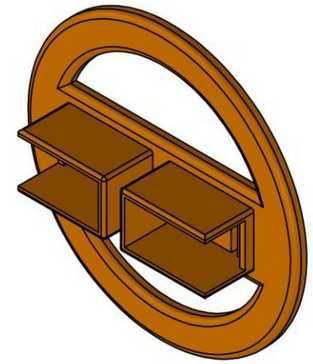


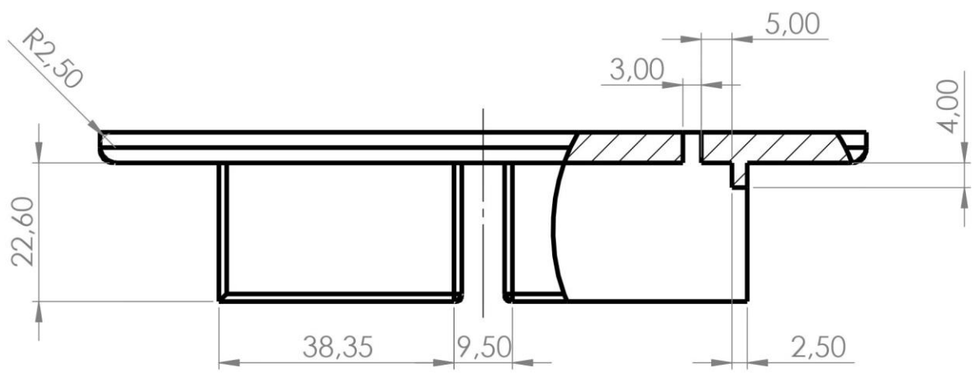
DETALLE A
ESCALA 2 : 1

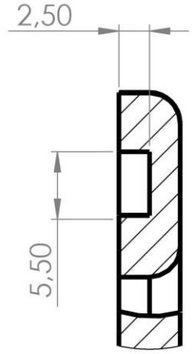
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.4.1.7	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 7	REFERENCIA: 230320-7	ORIENTACIÓN:
TÍTULO: SOL			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 1:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 23/03/20	HOJA: 1/1 A3





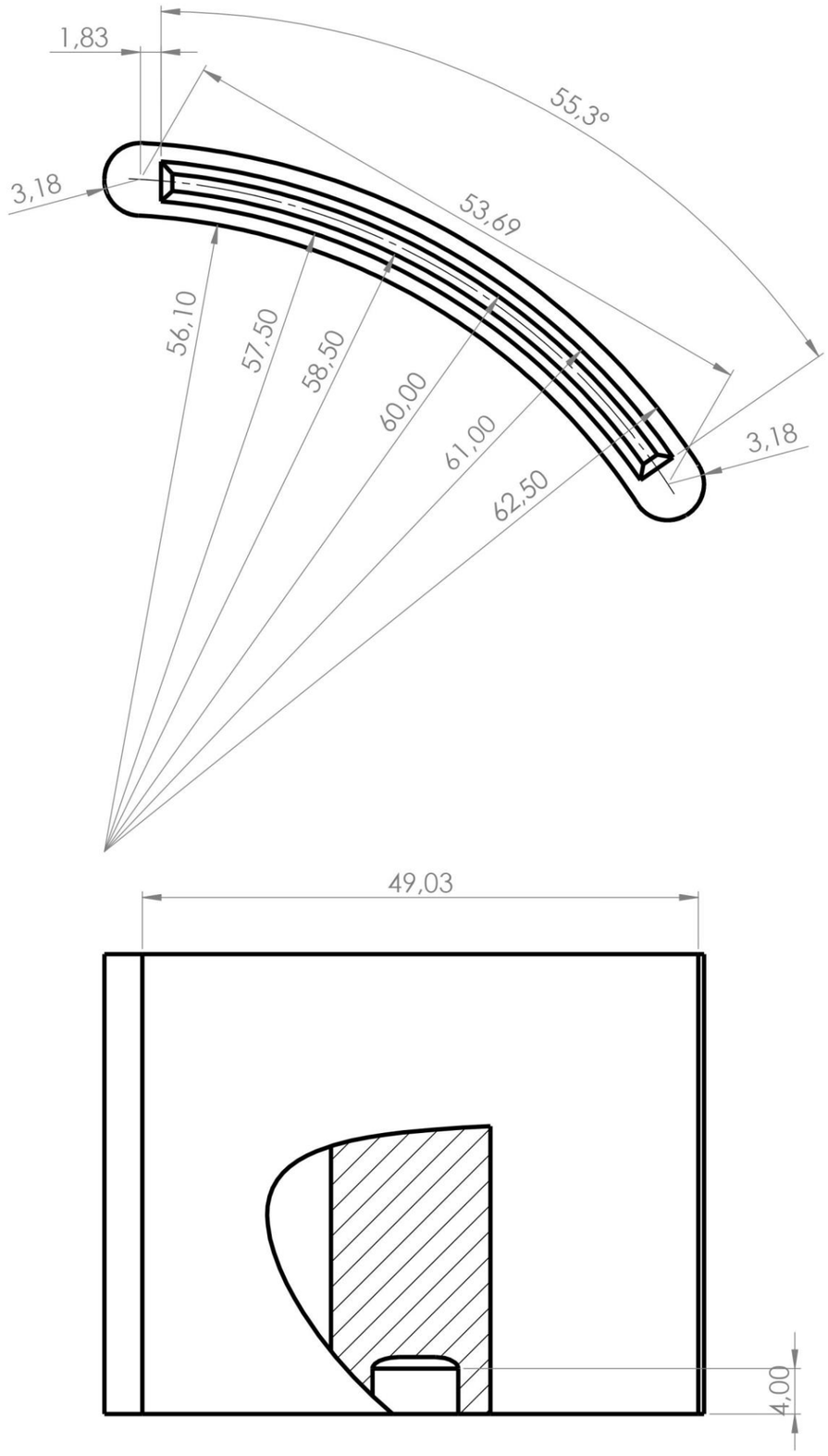


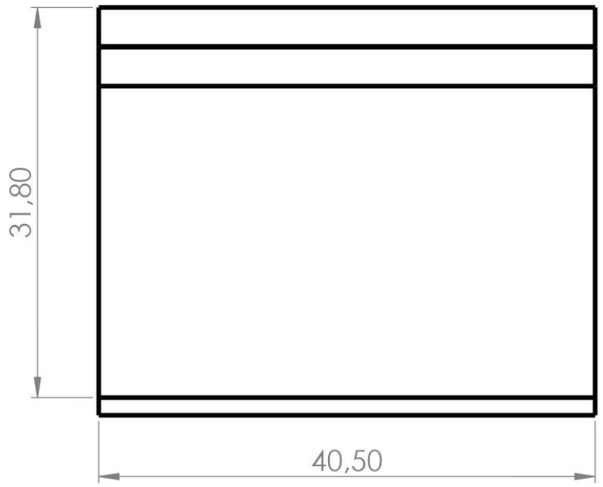


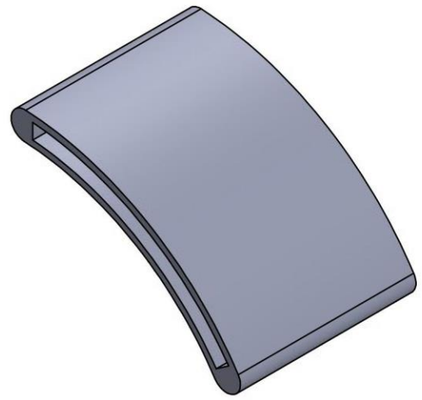


DETALLE A
ESCALA 2 : 1

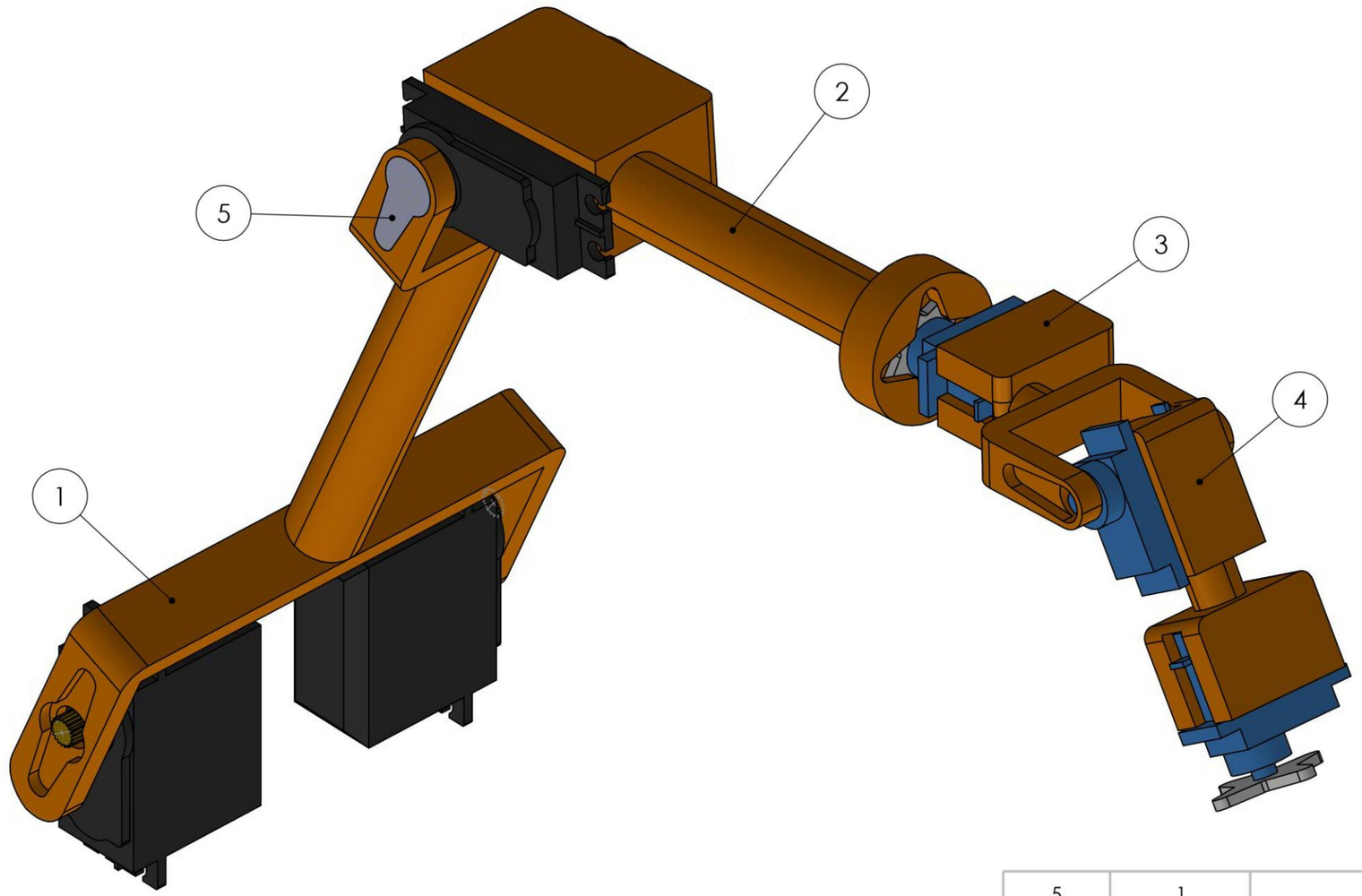
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.4.1.8	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 8	REFERENCIA: 230320-8	ORIENTACIÓN:
TÍTULO: SUELO			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 1:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 23/03/20	HOJA: 1/1 A3







NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.4.1.9	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 9	REFERENCIA: 230320-9	ORIENTACIÓN:
TÍTULO: SOPORTE			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 2:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 23/03/20	HOJA: 1/1 A3



Marca	Nº Piezas	Designación	Norma	Material
5	1	Conector motor	UNE-EN ISO 15015	ABS
4	1	Eslabón 4	UNE-EN ISO 15015	ABS
3	1	Eslabón 3	UNE-EN ISO 15015	ABS
2	1	Eslabón 2	UNE-EN ISO 15015	ABS
1	1	Eslabón 1	UNE-EN ISO 15015	ABS

NOMBRE DEL PROYECTO:
DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE

Nº PLANO: 1.4.2.4.2

FIRMA: JORGE DOMÉNECH JARA

TIPO DE DOCUMENTO: Dibujo de subconjunto

REFERENCIA: 290320-1

ORIENTACIÓN:

TÍTULO:

ESLABONES

CREADO POR: AUTOMA

REVISIÓN: A

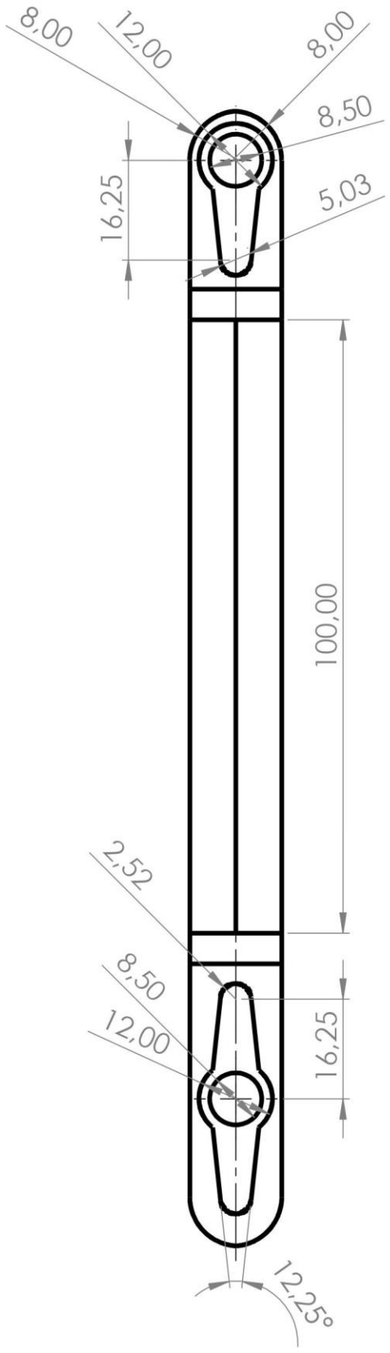
ESCALA: 1:1

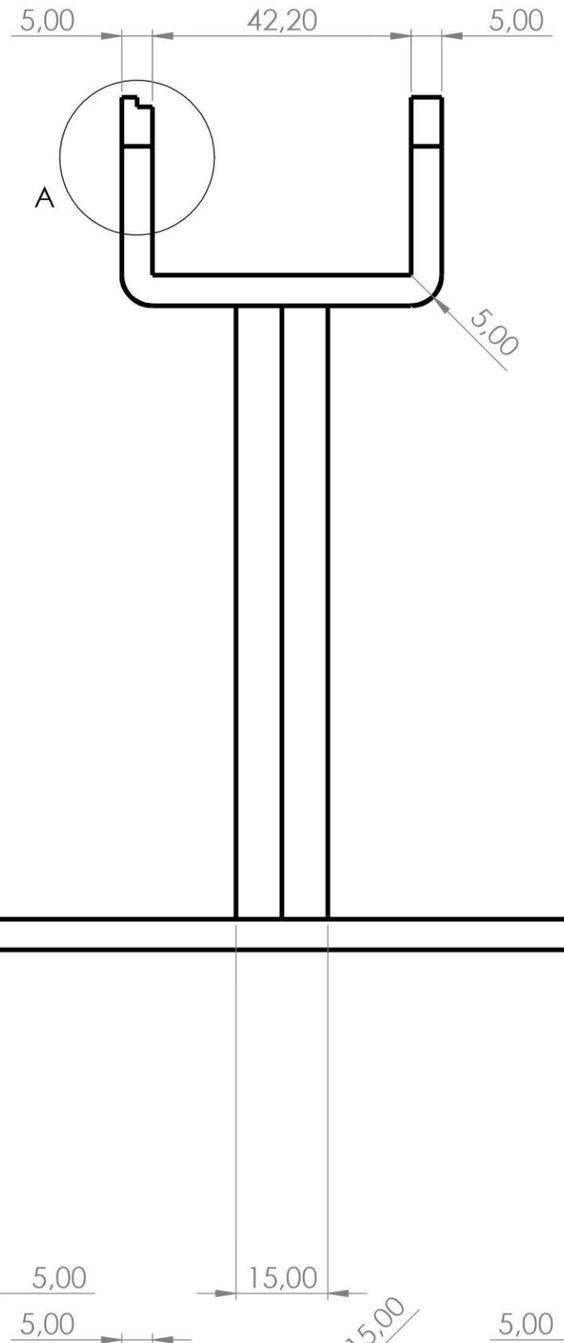
MATERIAL: ABS

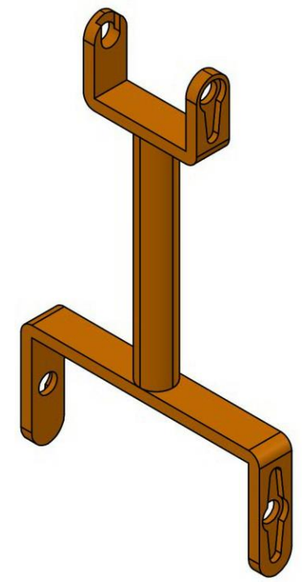
FECHA DE EDICIÓN: 29/03/20

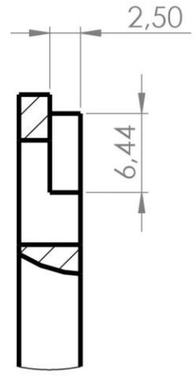
HOJA: 1/1

A3



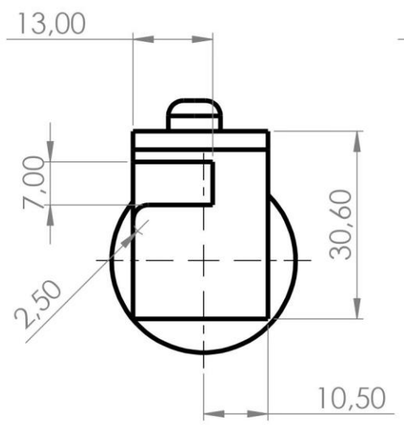


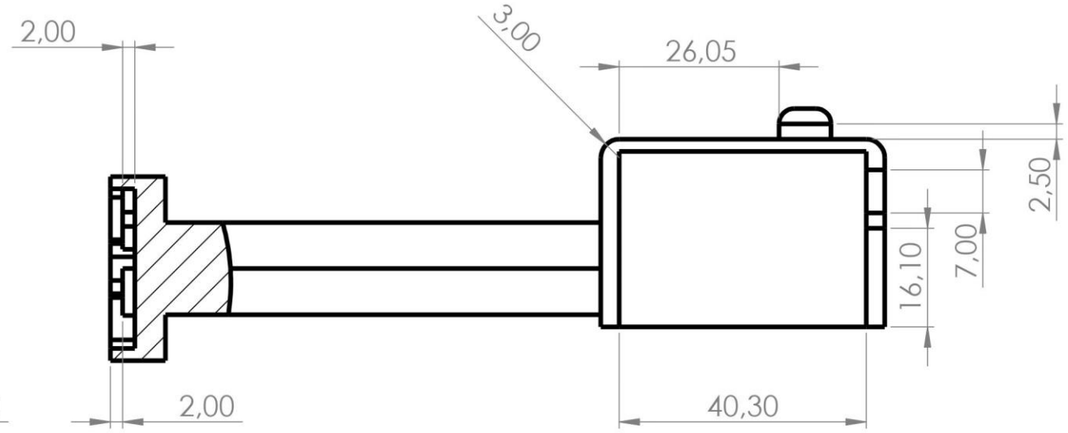


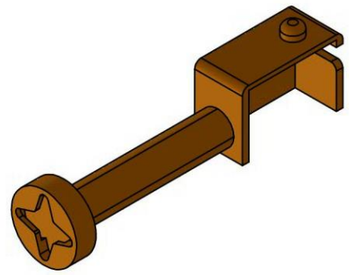


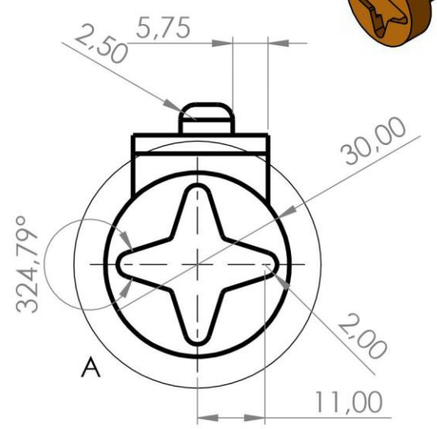
DETALLE A
ESCALA 2 : 1

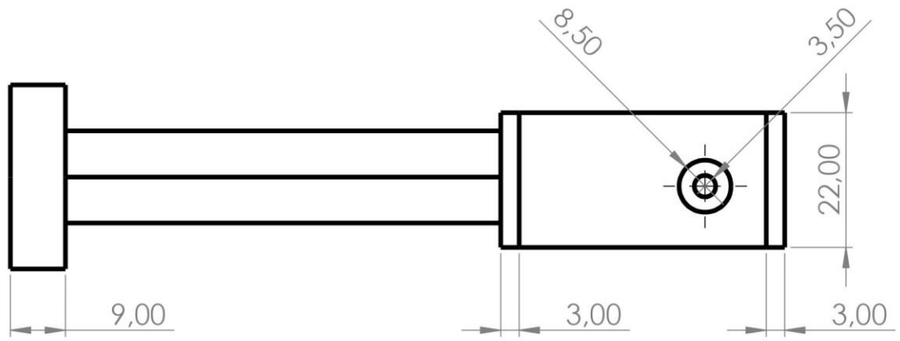
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.4.2.1	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 1	REFERENCIA: 250320-1	ORIENTACIÓN:
TÍTULO: ESLABÓN 1			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 1:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 25/03/20	HOJA: 1/1 A3

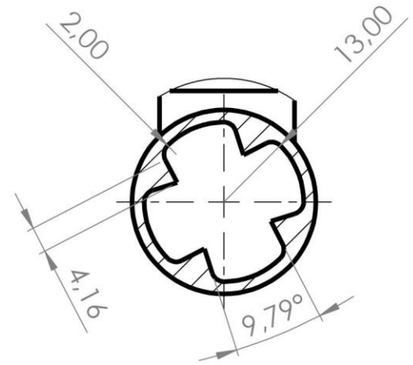






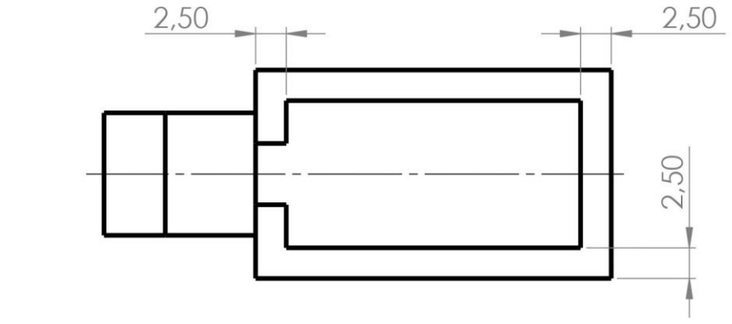


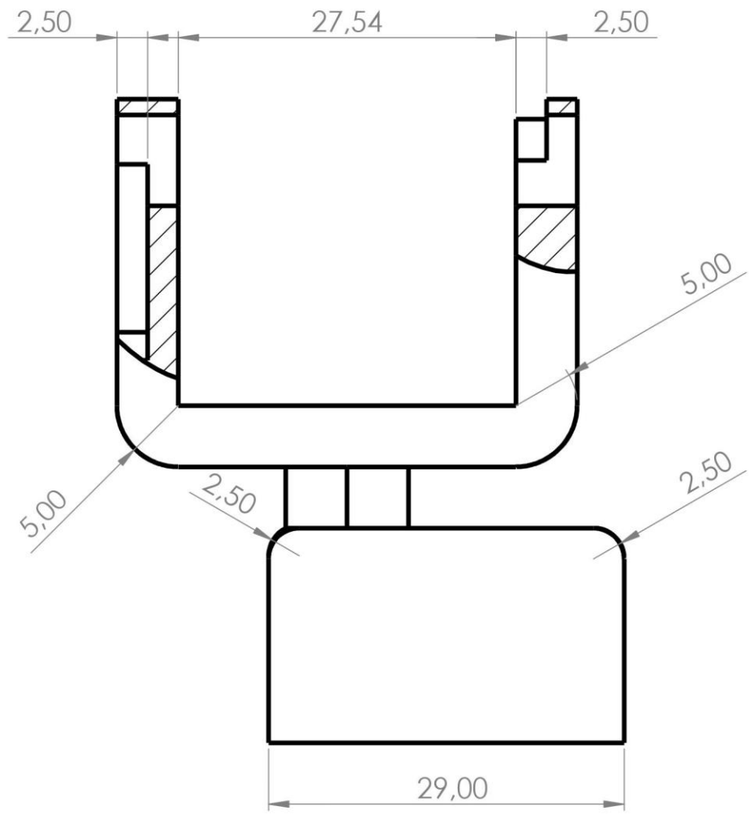


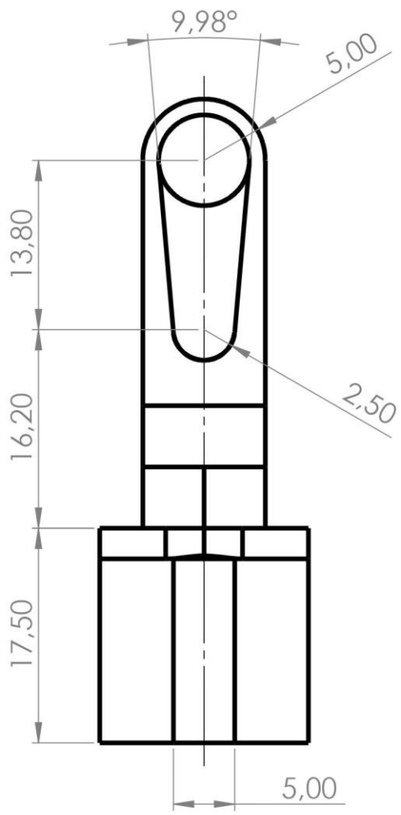


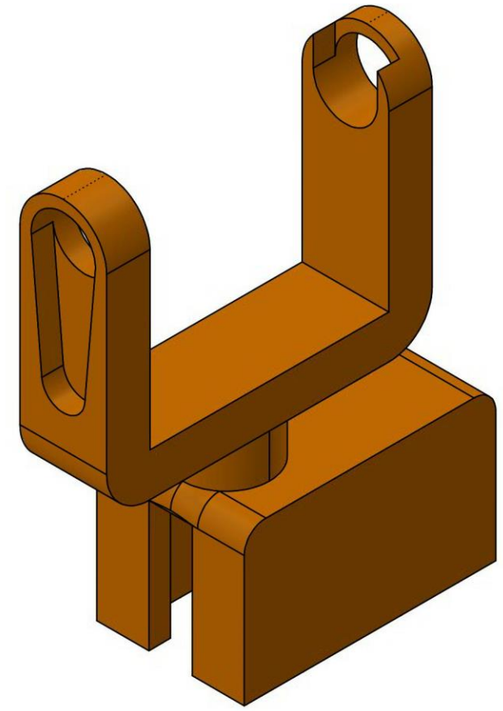
DETALLE A

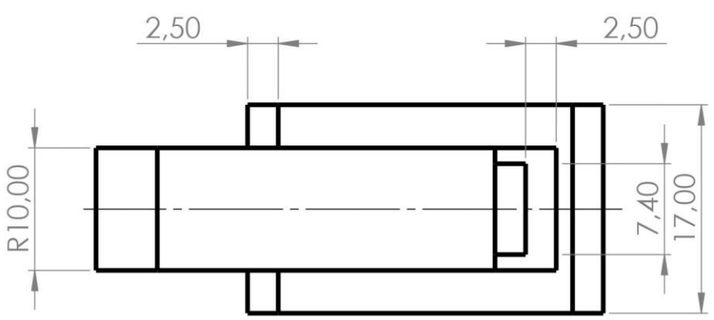
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.4.2.2	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 2	REFERENCIA: 250320-2	ORIENTACIÓN:
TÍTULO: ESLABÓN 2			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 1:2	MATERIAL: ABS
			FECHA DE EDICIÓN: 25/03/20	HOJA: 1/1 A3



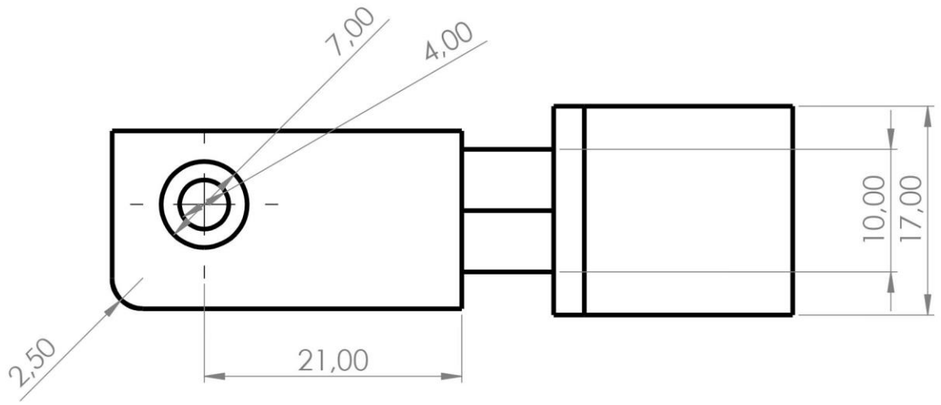


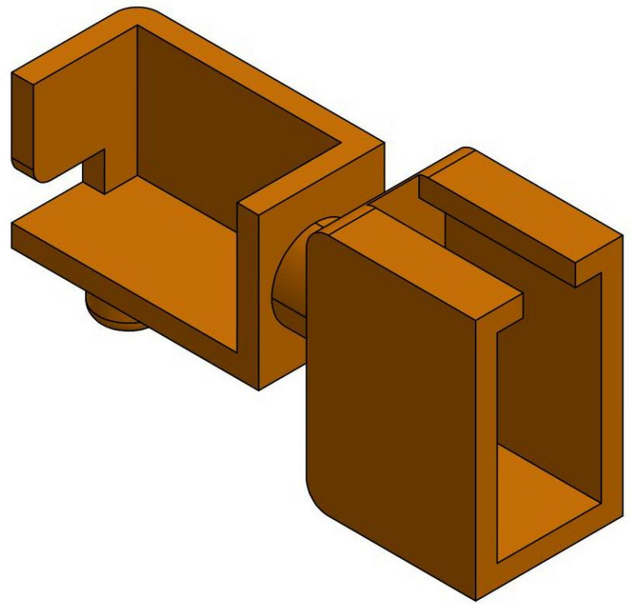


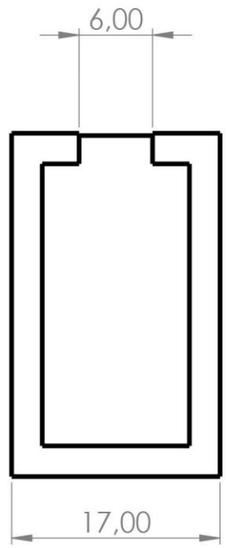


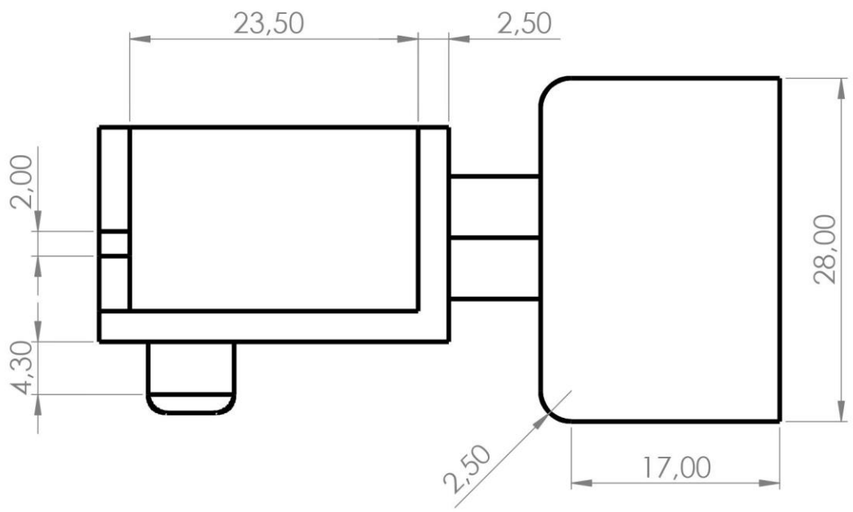


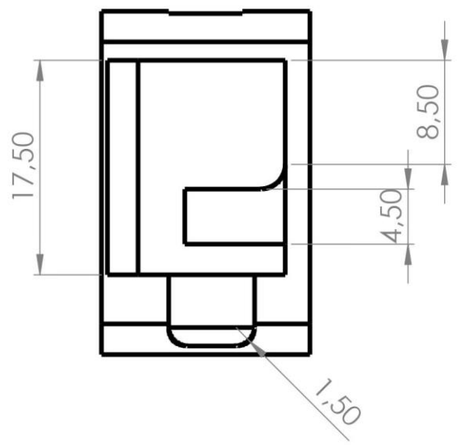
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.4.2.3	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 3	REFERENCIA: 250320-3	ORIENTACIÓN:
TÍTULO: ESLABÓN 3			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 2:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 25/03/20	HOJA: 1/1 A3



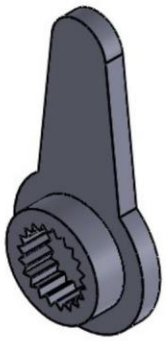


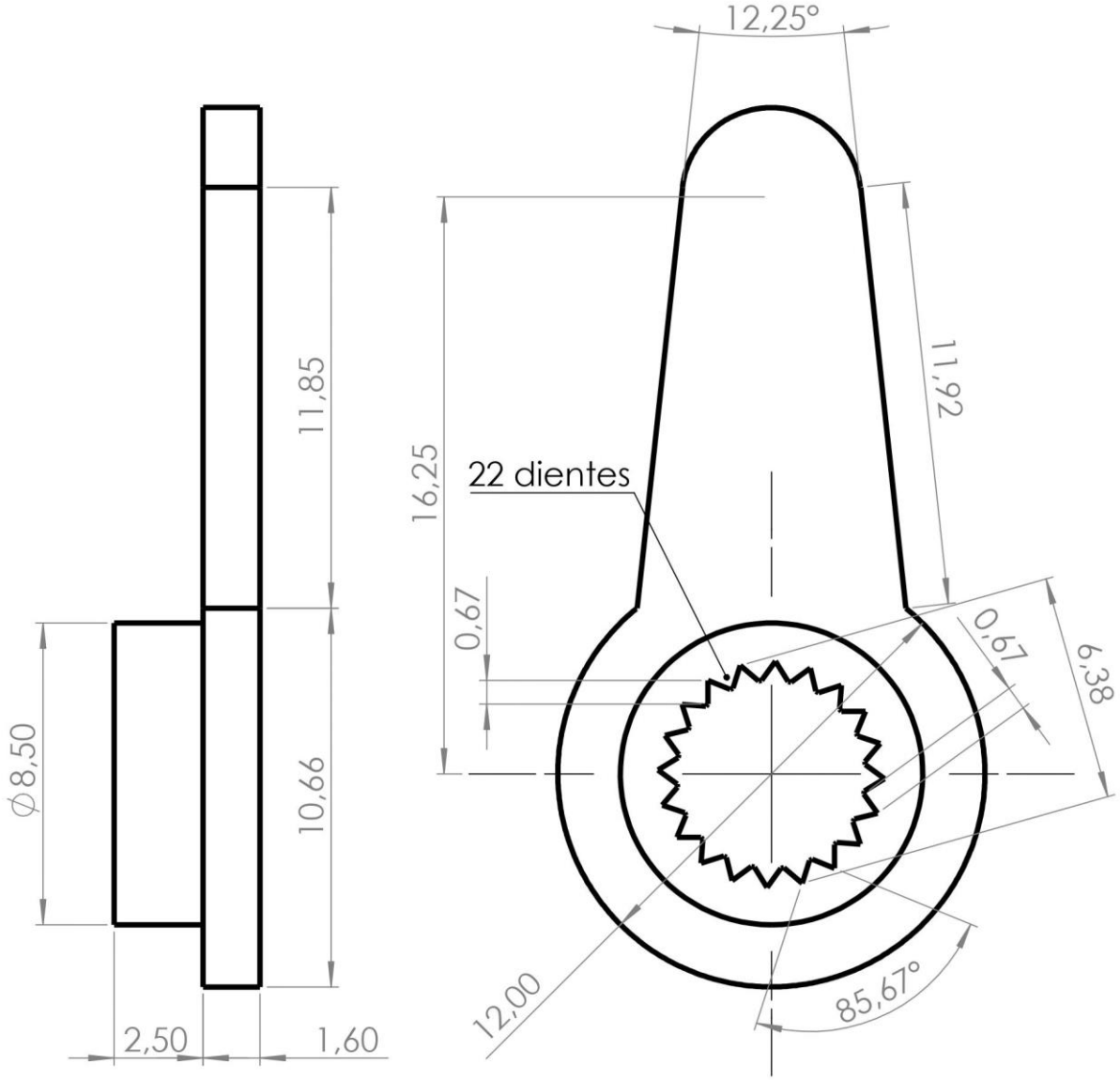




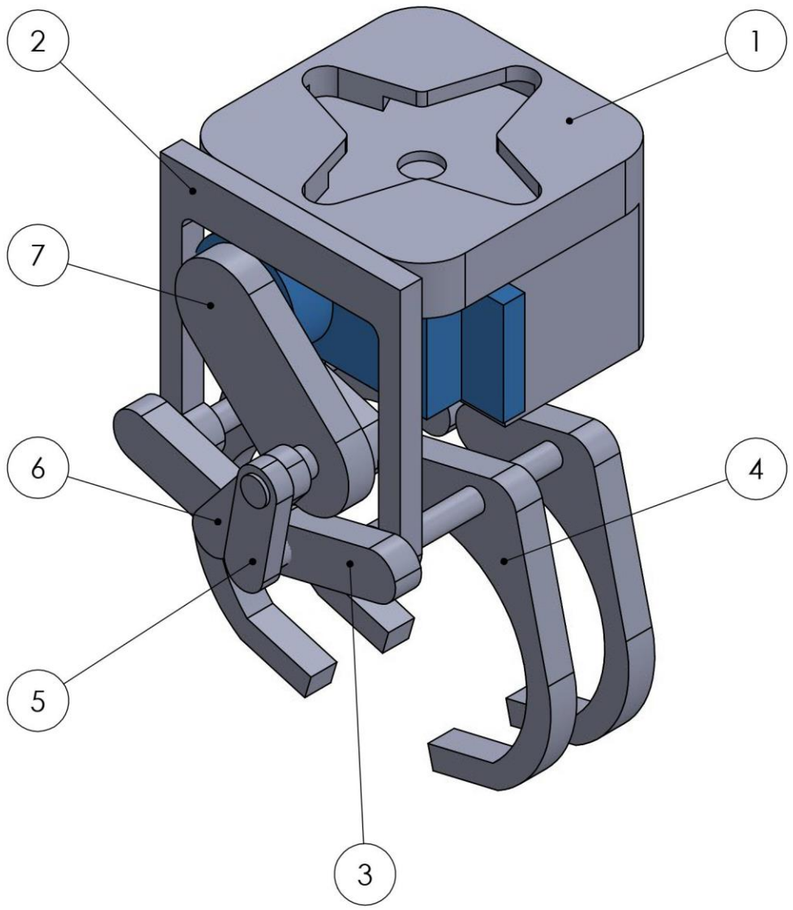


NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.4.2.4	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 4	REFERENCIA: 250320-4	ORIENTACIÓN:
TÍTULO: ESLABÓN 4			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 2:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 25/03/20	HOJA: 1/1 A3





NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.4.2.5	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 5	REFERENCIA: 250320-5	ORIENTACIÓN:
TÍTULO: CONECTOR MOTOR			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 5:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 25/03/20	HOJA: 1/1 A4



Marca	Nº Piezas	Designación	Norma	Material
7	1	Unión	UNE-EN ISO 15015	ABS
6	1	Unión izquierda	UNE-EN ISO 15015	ABS
5	1	Unión derecha	UNE-EN ISO 15015	ABS
4	4	Garra	UNE-EN ISO 15015	ABS
3	2	Eje	UNE-EN ISO 15015	ABS
2	1	Elemento 1	UNE-EN ISO 15015	ABS
1	1	Base pinza	UNE-EN ISO 15015	ABS

NOMBRE DEL PROYECTO:
DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE

Nº PLANO: 1.4.2.4.3

FIRMA: JORGE DOMÉNECH JARA

TIPO DE DOCUMENTO: Dibujo de subconjunto

REFERENCIA: 260320-1

ORIENTACIÓN:

TÍTULO:

PINZA

CREADO POR: AUTOMA

REVISIÓN: A

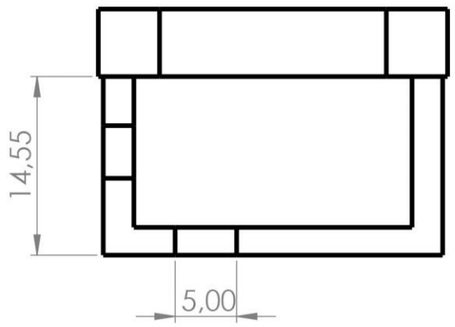
ESCALA: 2:1

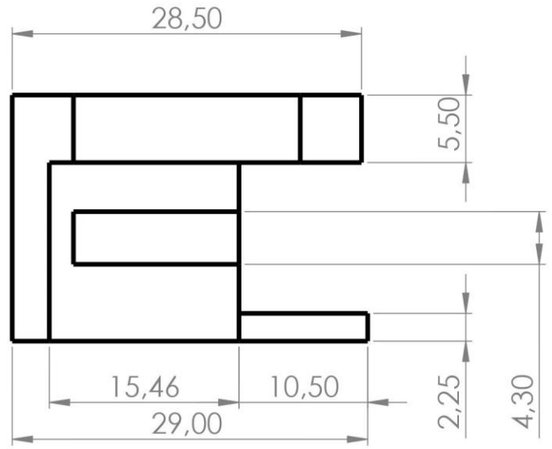
MATERIAL: ABS

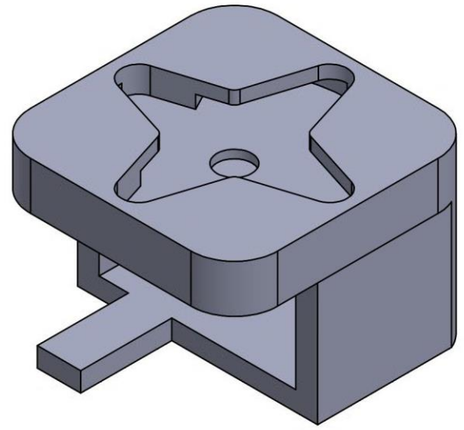
FECHA DE EDICIÓN: 26/03/20

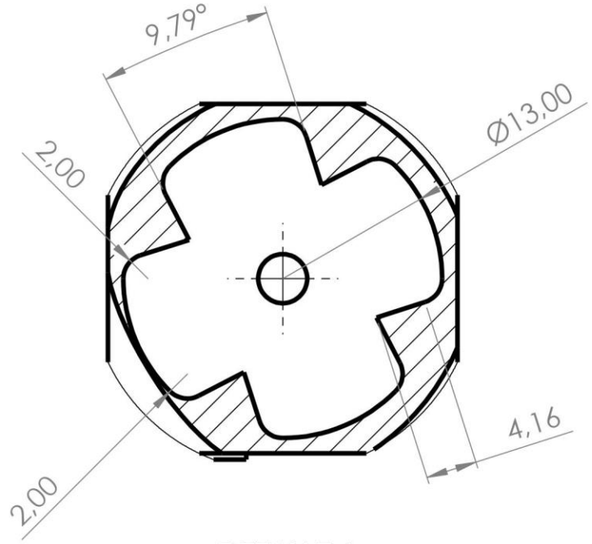
HOJA: 1/1

A3



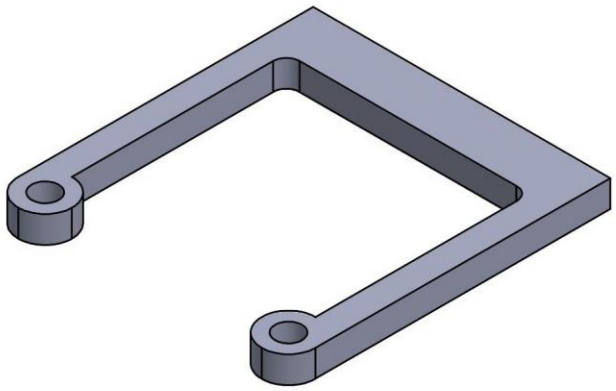


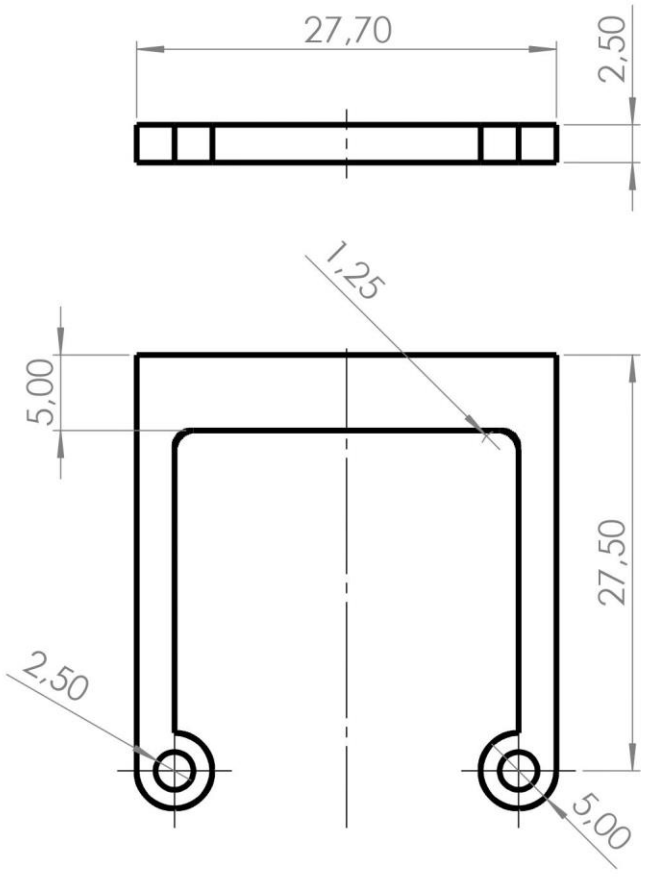




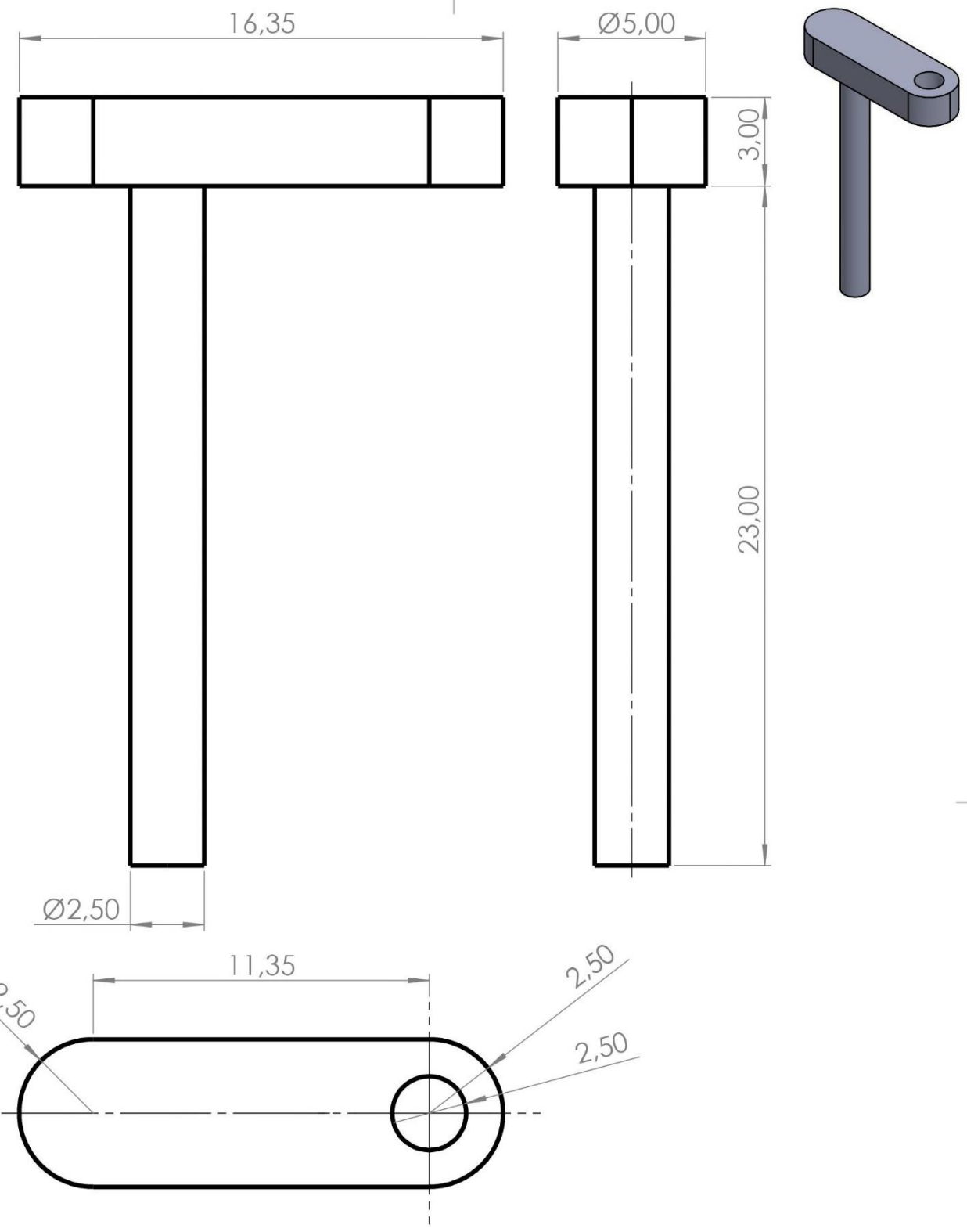
DETALLE A

NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.4.3.1	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 1	REFERENCIA: 250320-6	ORIENTACIÓN:
TÍTULO: BASE PINZA			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 2:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 25/03/20	HOJA: 1/1 A3

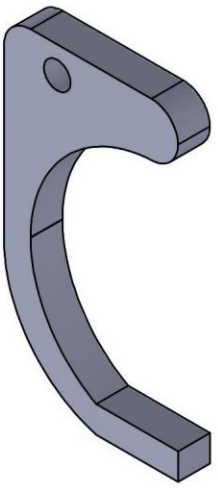


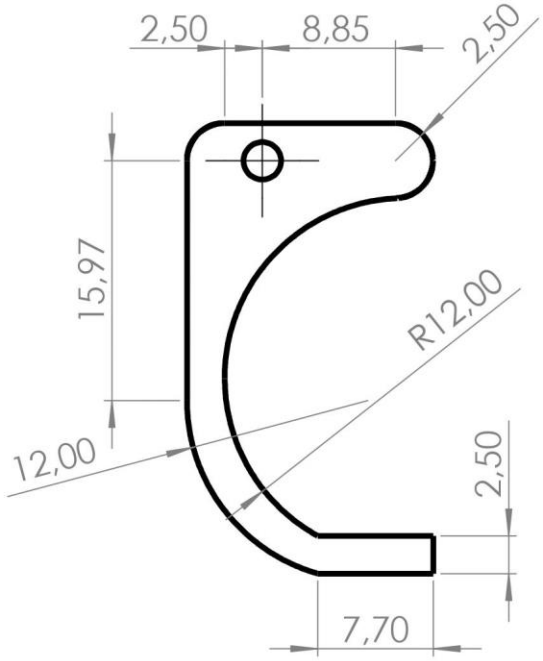


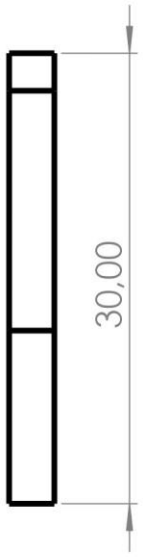
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.4.3.2	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 2	REFERENCIA: 250320-8	ORIENTACIÓN:
TÍTULO: ELEMENTO 1			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 2:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 25/03/20	HOJA: 1/1 A4

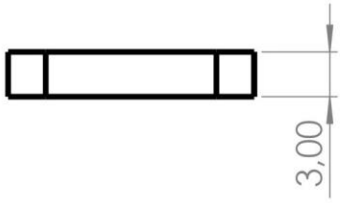


NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.4.3.3	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 3	REFERENCIA: 250320-11	ORIENTACIÓN:
TÍTULO: EJES			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 5:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 25/03/20	HOJA: 1/1 A4

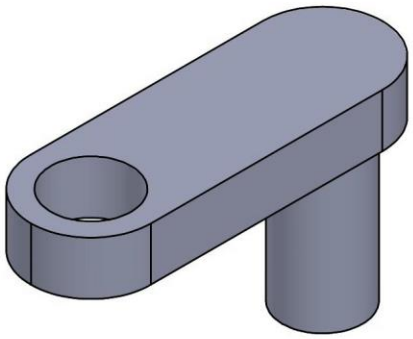


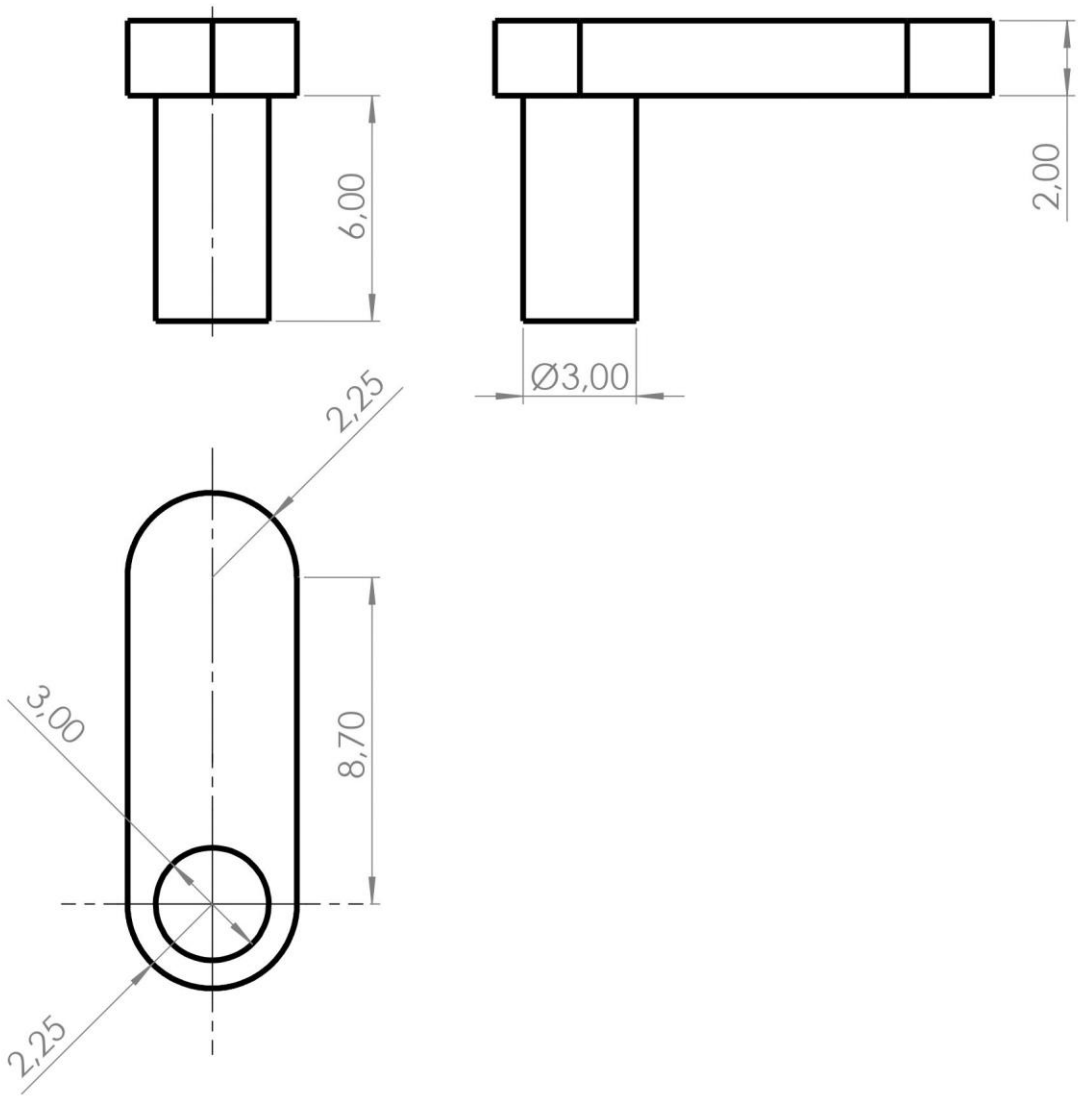




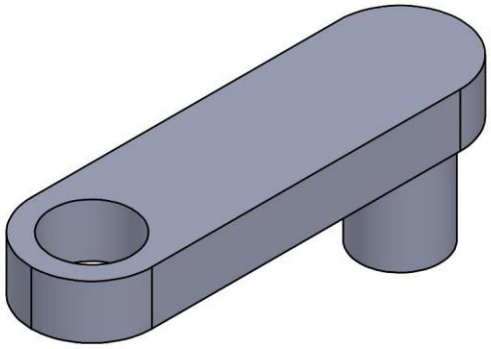


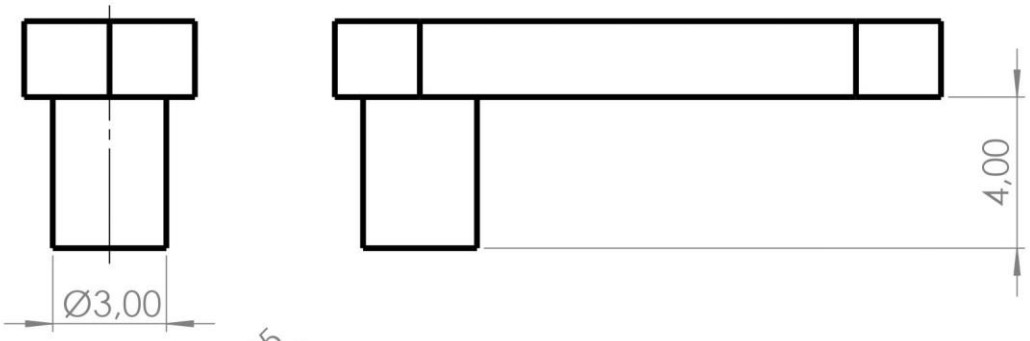
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.4.3.4	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 4	REFERENCIA: 250320-12	ORIENTACIÓN:
TÍTULO: GARRA			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 2:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 25/03/20	HOJA: 1/1 A4

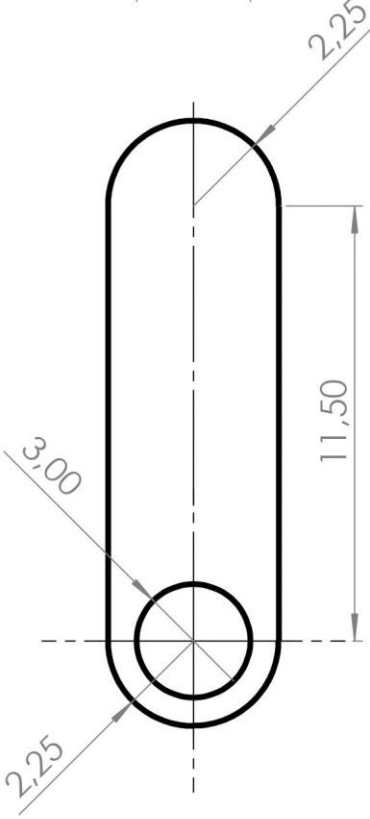




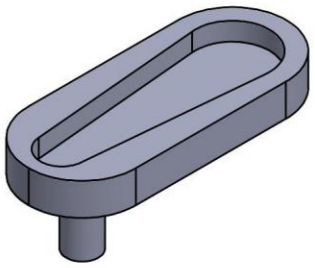
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.4.3.5	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 5	REFERENCIA: 250320-13	ORIENTACIÓN:
TÍTULO: UNIÓN DERECHA			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 5:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 25/03/20	HOJA: 1/1 A4

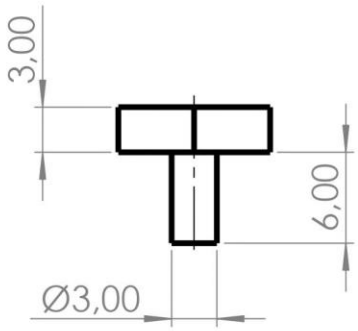


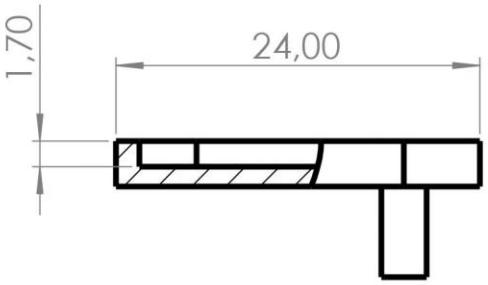


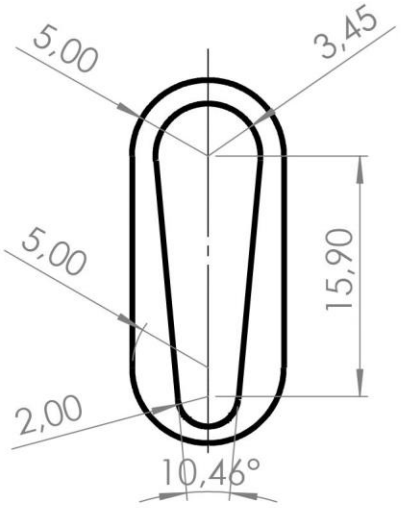


NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.4.3.6	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 6	REFERENCIA: 250320-14	ORIENTACIÓN:
TÍTULO: UNIÓN IZQUIERDA			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 5:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 25/03/20	HOJA: 1/1 A4

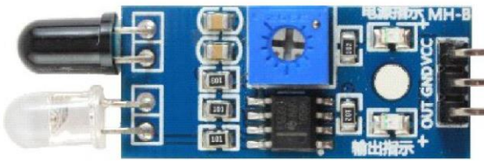


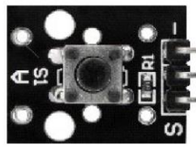


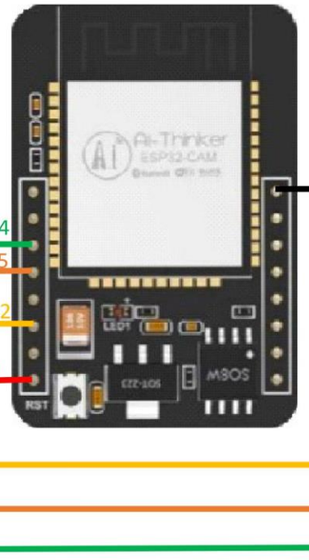




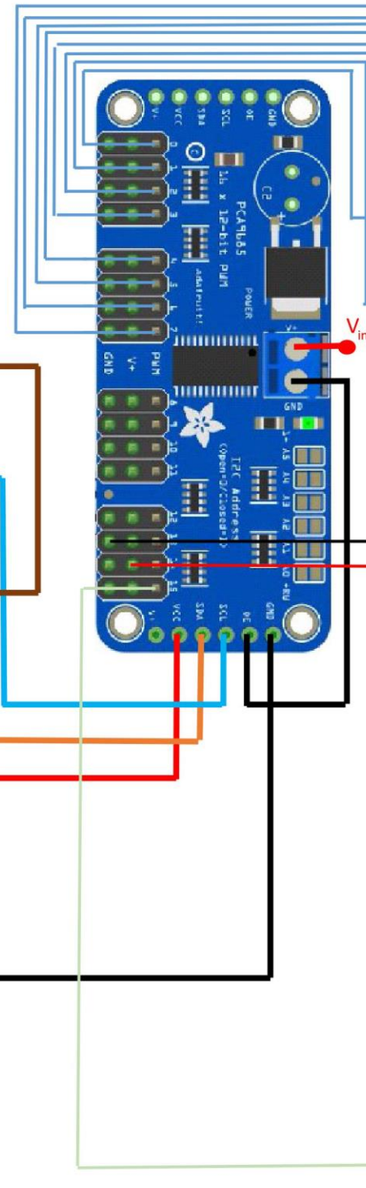
NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE			Nº PLANO: 1.4.2.4.3.7	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	MARCA: 7	REFERENCIA: 250320-15	ORIENTACIÓN:
TÍTULO: UNIÓN			CREADO POR: AUTOMA	REVISIÓN: A
			ESCALA: 2:1	MATERIAL: ABS
			FECHA DE EDICIÓN: 25/03/20	HOJA: 1/1 A4

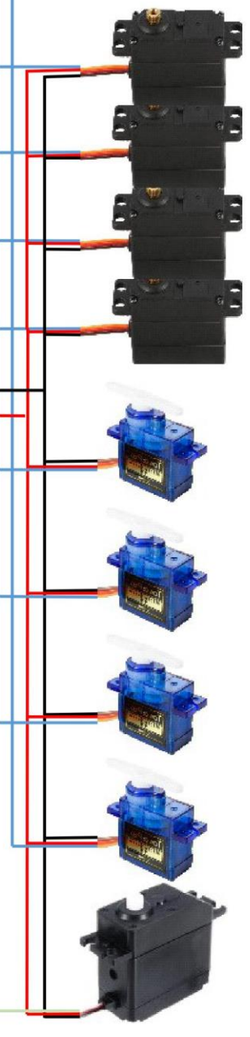






V_{in} = 6 V





NOMBRE DEL PROYECTO: DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK&PLACE		Nº PLANO: N/A	
FIRMA: JORGE DOMÉNECH JARA	TIPO DE DOCUMENTO: Dibujo de subconjunto	REFERENCIA: 140420-1	ORIENTACIÓN: N/A
TÍTULO: ESQUEMA CONEXIONES ELÉCTRICAS		CREADO POR: AUTOMA	REVISIÓN: A
		ESCALA: S/E	MATERIAL: ABS
		FECHA DE EDICIÓN: 14/04/20	HOJA: 1/1 A3

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Documento número 3:

PLIEGO DE CONDICIONES TÉCNICAS

DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK & PLACE.

TRABAJO FINAL DEL

Grado en Ingeniería Electrónica Industrial y Automática

REALIZADO POR

Jorge Doménech Jara

TUTORIZADO POR

Ángel Valera Fernández

Carlos Ricolfe Viala

CURSO ACADÉMICO: 2019/2020

3.1 Objeto

Se procede a diseñar en 3D, imprimir y montar un brazo robótico de 6 grados de libertad que tiene una longitud entre 48 y 52 cm estirado. El brazo utiliza servomotores controlados con un módulo PCA9685.

El diseño de la cinta transportadora dispone de una cámara y de un sensor infrarrojo para detectar la pieza que circula a través de la cinta. Para la cámara se usa un ESP32cam y para el microcontrolador un ESP32. La cámara utilizada para captar la forma y orientación de la pieza tiene una resolución de 640 x 480 píxeles.

Se usa Arduino para el entorno de desarrollo y MATLAB para realizar cálculos y pruebas de funcionamiento. Para el diseño de las piezas en 3D se utiliza SolidWorks.

El proyecto no precisa de mantenimiento y en caso de rotura o desgaste de alguna pieza se facilita su modelo en 3D para sustituirla.

3.2 Condiciones de los Materiales

En este apartado se describen los componentes del proyecto y el control de calidad seguido.

3.2.1 Descripción o características

A continuación, se realiza una descripción de las características de los materiales y componentes:

- Materiales electrónicos:
    - Un microcontrolador ESP32 con un procesador Xtensa Dual-Core 32-bit, una frecuencia del procesador de 160 MHz y una cantidad de entradas y salidas de 10 a 36 GPIOs.
    - Un ESP32cam, con un procesador incorporado Xtensa Dual-Core 32-bit. Dispone de la suficiente potencia computacional para que realice las operaciones en menos de 1 segundo y es capaz de comunicarse con el microcontrolador.
    - Una fuente de alimentación capaz de convertir 220 V y 60 Hz de tensión alterna en 12 V y 10 A de corriente continua, con una eficiencia mayor del 90 %.
    - Un convertidor BUCK capaz de reducir una tensión continua de 12 V a una de 6 V con una eficiencia superior al 95 %.
    - Un encoder digital con una sensibilidad mínima de 10 pulsaciones/s y alimentado dentro de un rango de entre 3,3 V y 5 V.
    - Un sensor infrarrojo digital capaz de detectar un objeto en un rango de entre 15 mm y 100 mm, alimentado con una tensión de entre 3,3 V y 5 V.
    - Una controladora de servomotores PCA9685 para liberar parte de las salidas del microcontrolador y alimentar los motores de forma externa. También se puede alimentar con una tensión de entre 3,3 V y 5 V.

- Materiales eléctricos:
  - En cuanto al cableado, para el conexionado de los módulos se ha usado el de tipo Dupont que es el utilizado habitualmente en Arduino. Para los motores se utiliza un cable de una sección mayor dado el incremento de consumo de corriente, ésta sección es de 1 mm^2.
  - La tira de LEDs blanco frío es utilizada para iluminar el interior de la estructura de la cámara con el fin de controlar la iluminación y así conseguir que sea invariable y casi constante. La tira está compuesta de cuatro partes que forman un rectángulo: dos lados de 100 mm y los otros dos de 120 mm de longitud. La potencia máxima del conjunto de LEDs es como máximo de 10 W y la tensión de alimentación de 12 V. A su vez, la tira tiene una densidad de 1 LED cada 15 mm y una luminosidad de entre 5 y 6 lúmenes.

- Fuerza motriz:
  - Un motor de rotación continua DS04-NFC que ejerce un par mínimo de 3 kg·cm. Su tensión de operación es de 6 V y sus dimensiones de 42 x 22 x 38 mm (largo x ancho x alto).
  - Cuatro servomotores MG996R que ejercen un par de 11 kg·cm a 6 V y tienen unas dimensiones de 42 x 22 x 38 mm
  - Cuatro servomotores SG90 que soportan un par máximo de 1,5 kg·cm a 5 V y tienen unas dimensiones de 23 x 12 x 25 mm.

- Materiales de plástico:
  - Todas las piezas de plástico se realizan con ABS cumpliendo la norma UNE-EN ISO 15015. Disponen de una resistencia ante impacto de 58 kJ/m^2 y una elasticidad de tracción de 2030 MPa.

- Otros:
  - Una correa dentada que tiene una longitud de 240 mm y una anchura de 10 mm. Sus engranajes son de una aleación de aluminio y tienen un diámetro de 15 mm.
  - La goma eva cuya composición es etileto y acetato de vinilo cubre la estructura de la cámara y compone la cinta. La cinta mide 100 x 2 x 1200 mm y la de la estructura mide 54 x 2 x 24 mm.

3.2.2 Control de calidad

Se detallan los requisitos de calidad que deben cumplir todos los componentes del proyecto.

En cuanto a las piezas impresas, ninguna de ellas debe presentar defectos de impresión que afecten a su funcionalidad o estética como cantos afilados, grietas o bordes dentados. El material utilizado, una bobina de plástico ABS, no debe presentar ni burbujas ni deformaciones pues ello provoca imperfecciones en las piezas.

El material de goma eva no debe de tener ni pequeños agujeros ni deformidades que puedan afectar a su estética.

En cuanto a los servomotores, antes de utilizarlos se comprueba su correcto funcionamiento midiendo si se posiciona correctamente el ángulo de giro.

El cableado no debe presentar partes quemadas por cortocircuitos o similar.

3.3 Condiciones de la Ejecución

En este apartado se describen el proceso de ejecución del proyecto y el control de calidad seguido. Se facilita el soporte digital de la geometría conforme a la norma UNE-EN ISO/ASTM 52915.

3.3.1 Descripción del proceso de ejecución

A continuación, se detallan los pasos a seguir para el montaje del proyecto:

1. Montar las piezas tal y como muestran las imágenes de los explosionados siguientes:

   - Explosionado de la pinza:

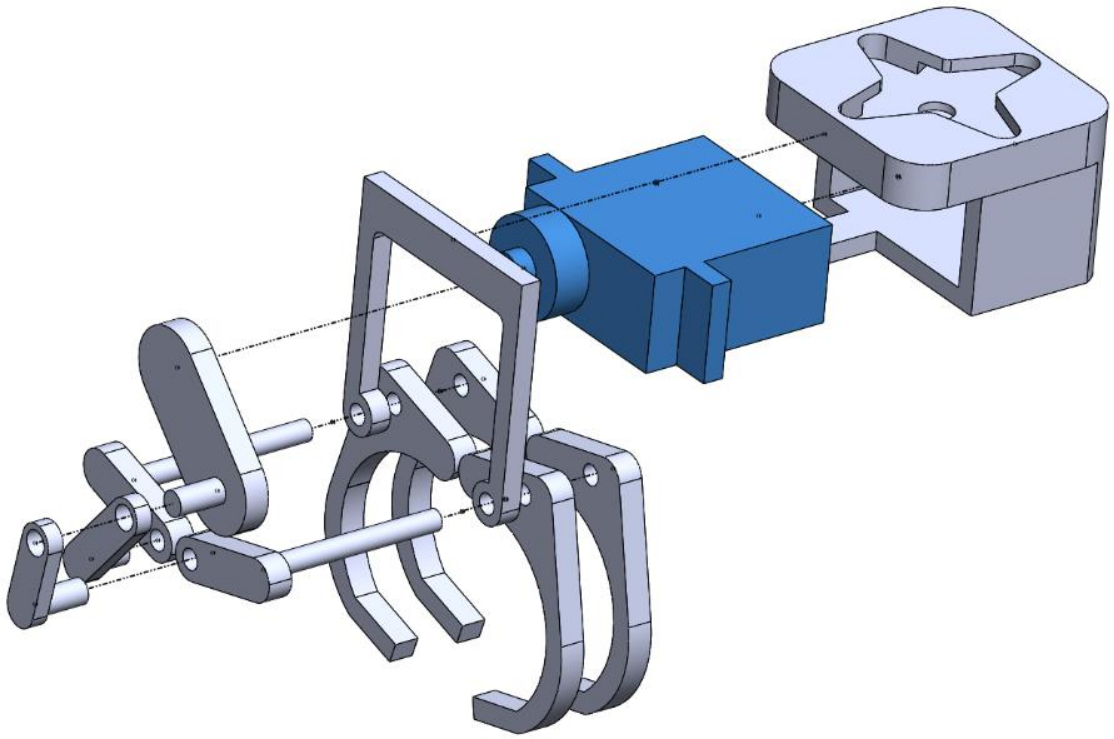


Ilustración 28. Montaje de la pinza.

- Explosionado del planetario:

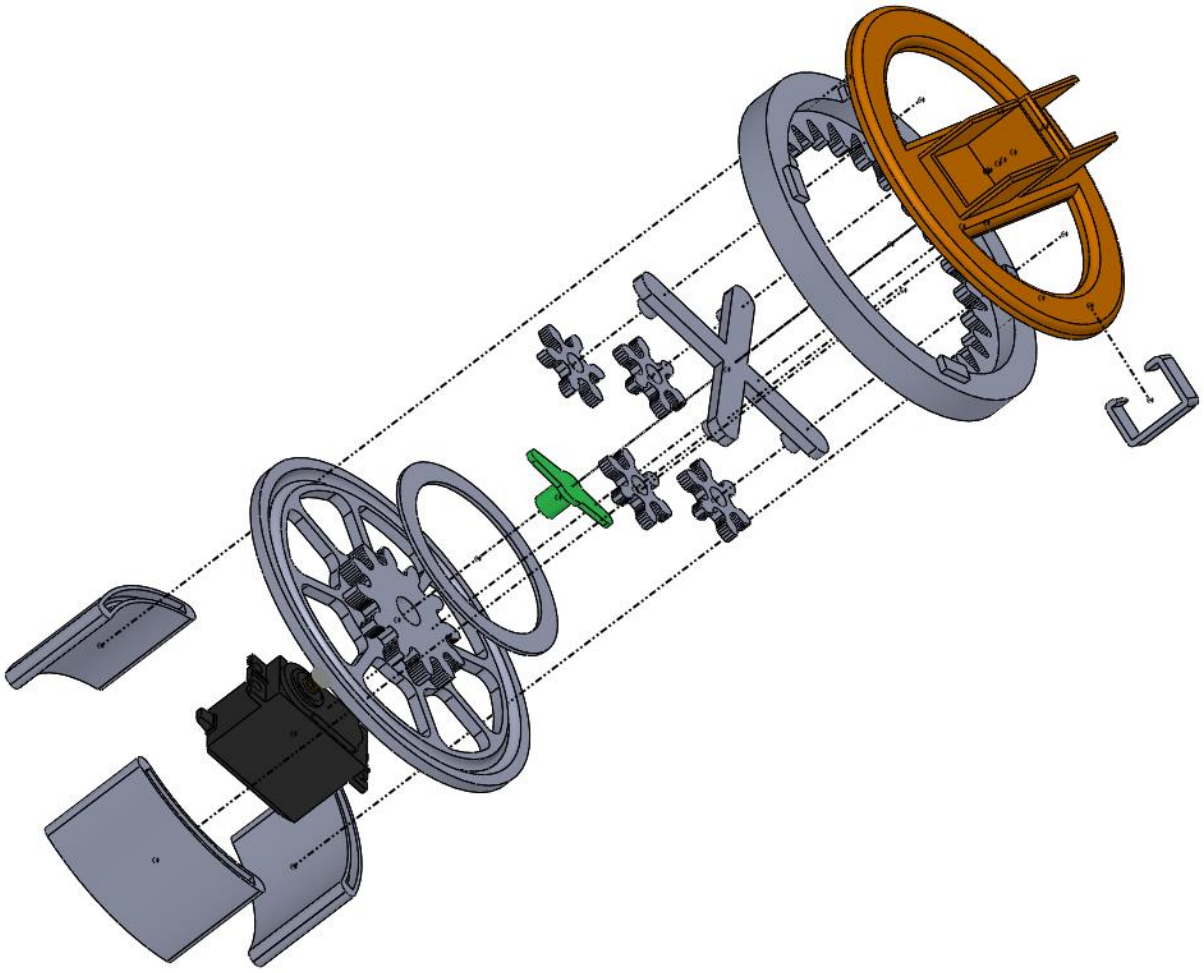


Ilustración 29. Montaje de la base del brazo robótico.

- Explosionado de los eslabones:

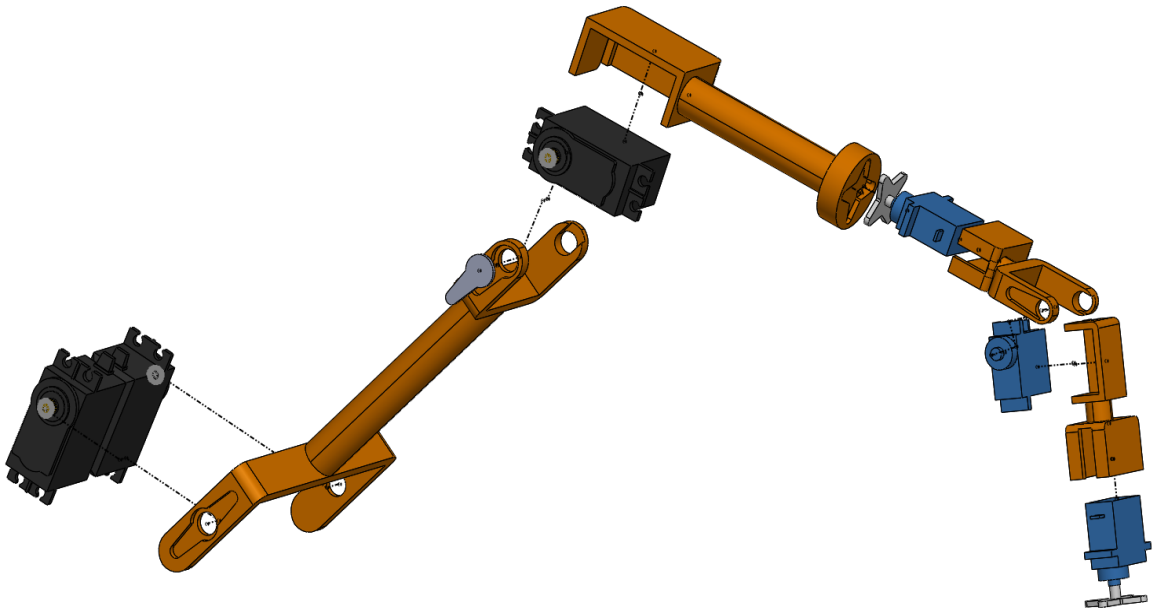


Ilustración 30. Montaje del brazo robótico.

- Explosionado de la estructura de la cámara:

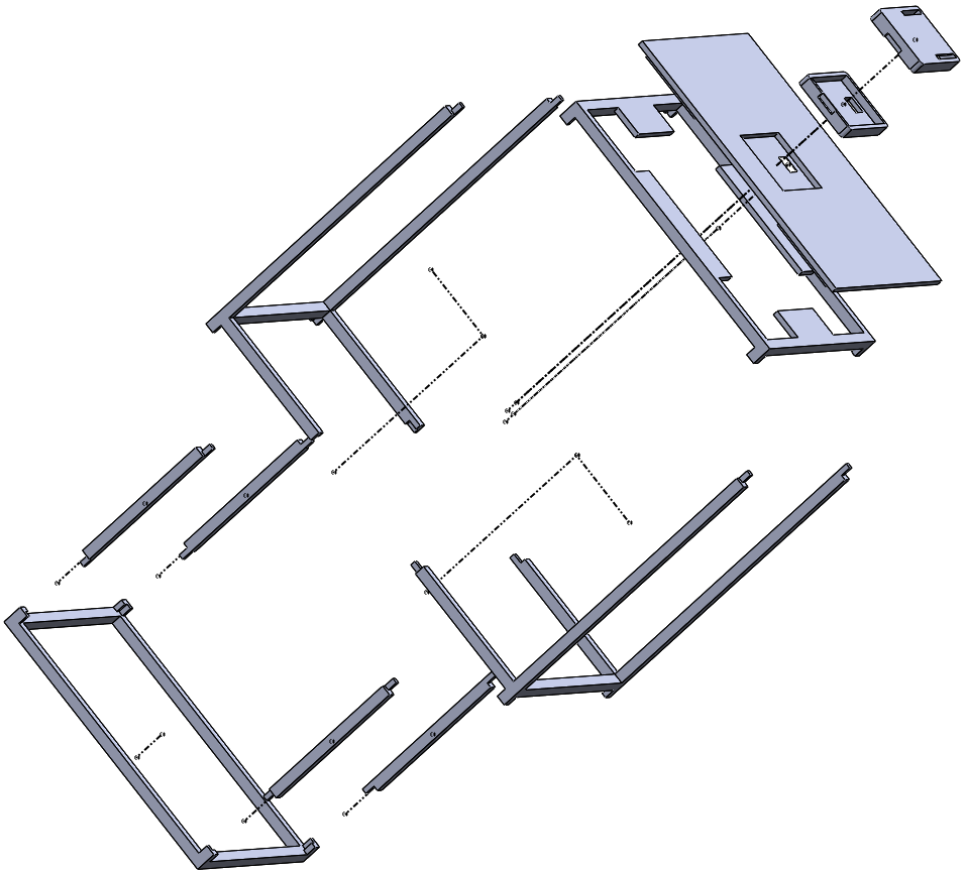


Ilustración 31. Montaje de la estructura de la cámara.

2. Presentar la estructura de la cámara y el brazo robótico en su posición tal y como aparece en la imagen siguiente:

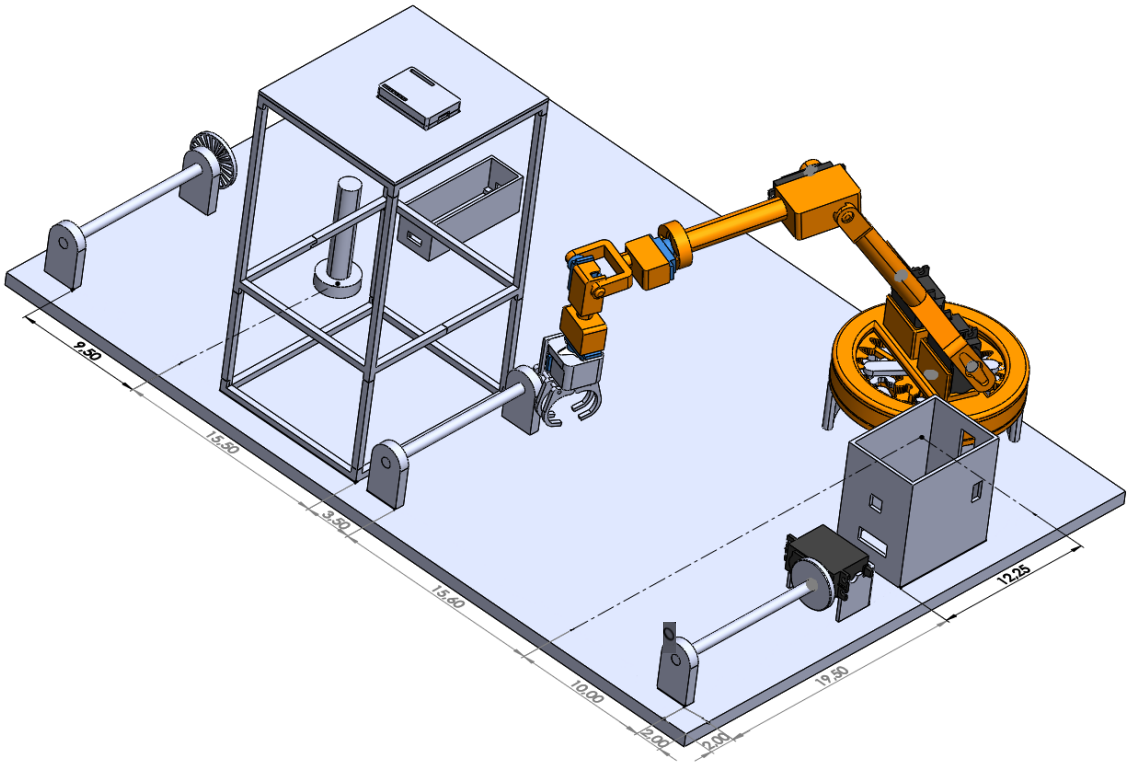


Ilustración 32. Imagen del conjunto de piezas de ABS dispuestas en su lugar.

3. Colocar los soportes de la cinta y añadir los engranajes en sus ejes.

4. Añadir el motor de rotación continua y seguidamente poner y fijar la goma eva que formará la cinta.

5. Cubrir con goma eva la estructura de la cámara y situar la cámara en su posición.

6. Poner el resto de sensores y la rueda del encoder en su lugar correspondiente.

7. Añadir la fuente de alimentación y el BUCK a la base de madera.

8. Conectar todos los sensores y actuadores tal y como muestra el plano de conexiones eléctricas (apartado 2.48).

9. Finalmente calibrar la cámara. Véase el apartado 1.3.4.2.

Tras finalizar el montaje, el proyecto tiene la siguiente apariencia:

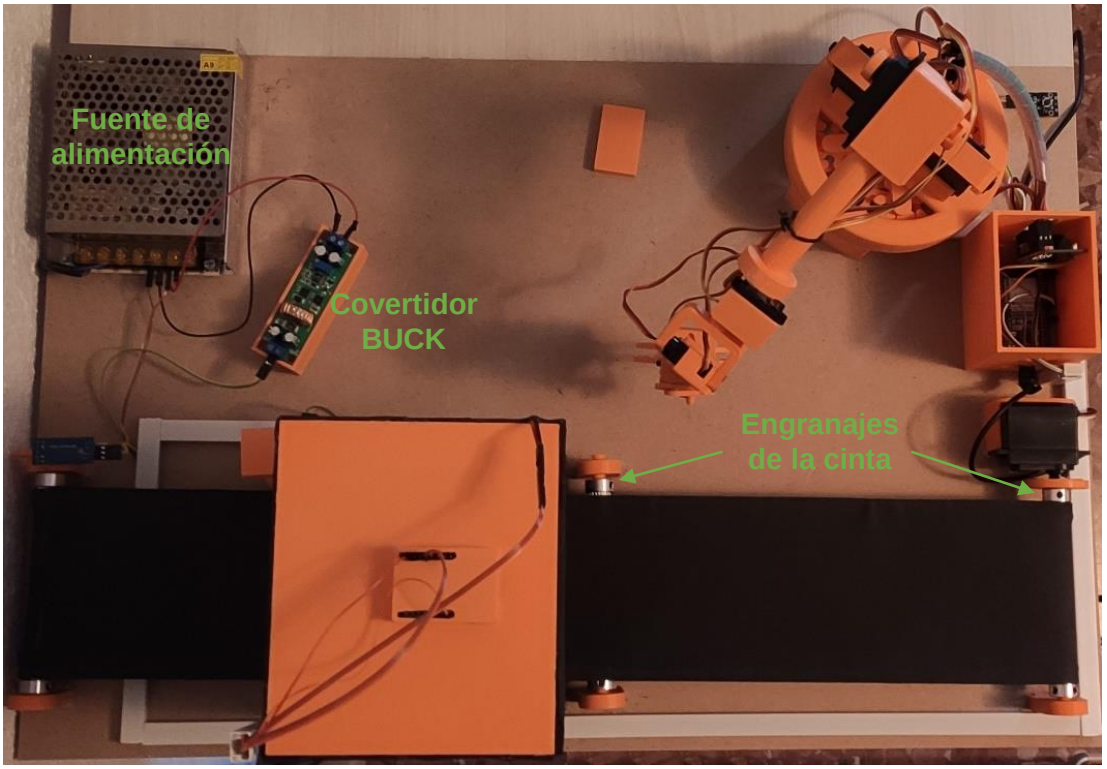


Ilustración 33. Proyecto acabado visto de planta.

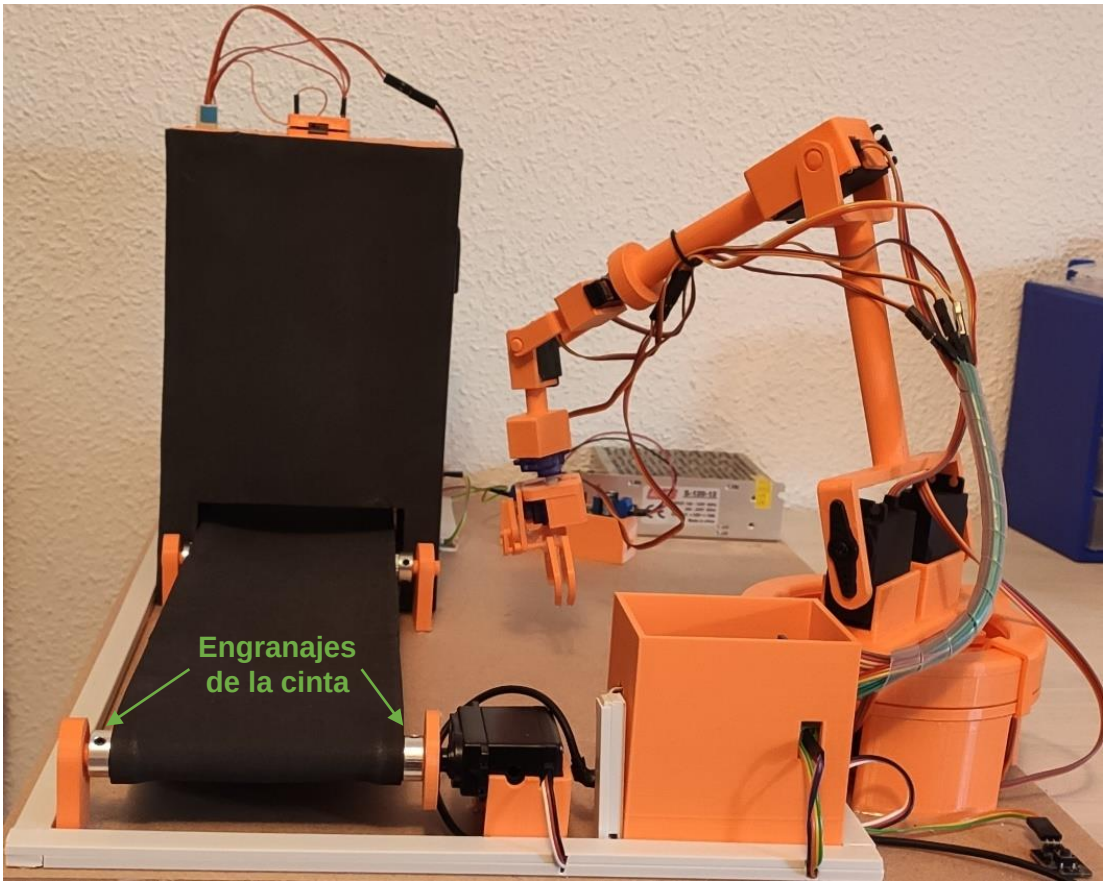


Ilustración 34. Proyecto acabado visto de perfil.

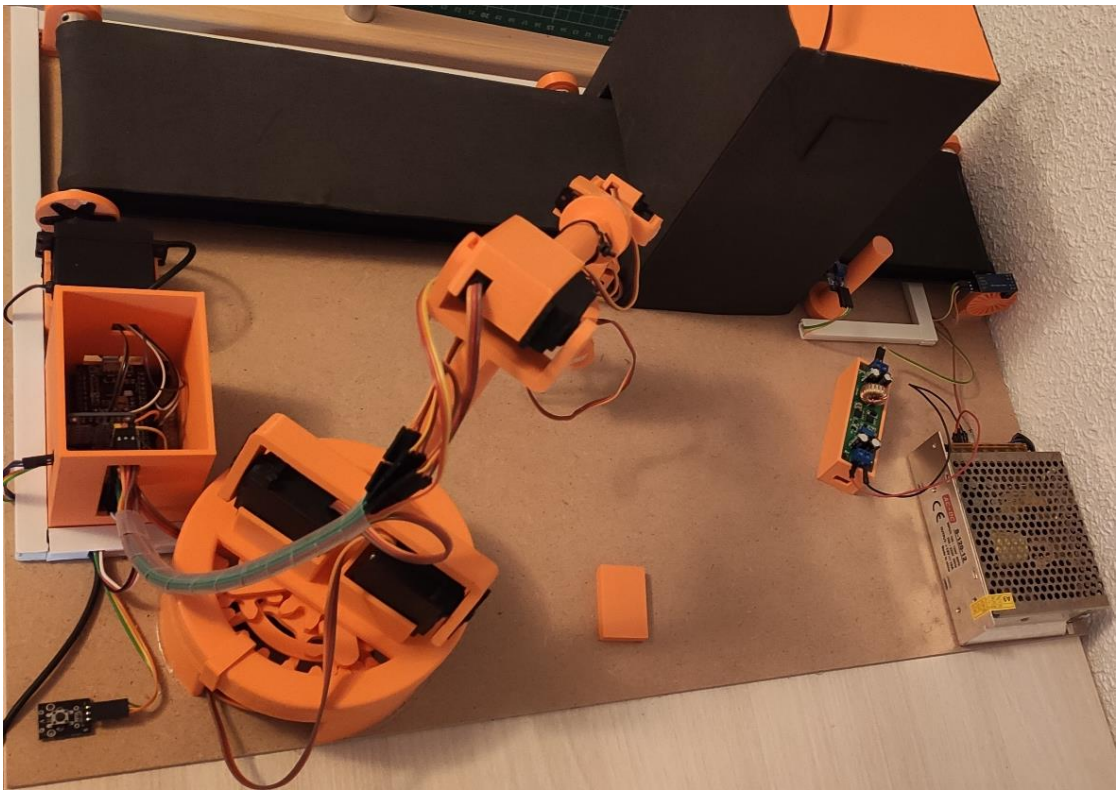


Ilustración 35. Proyecto acabado vista en conjunto.

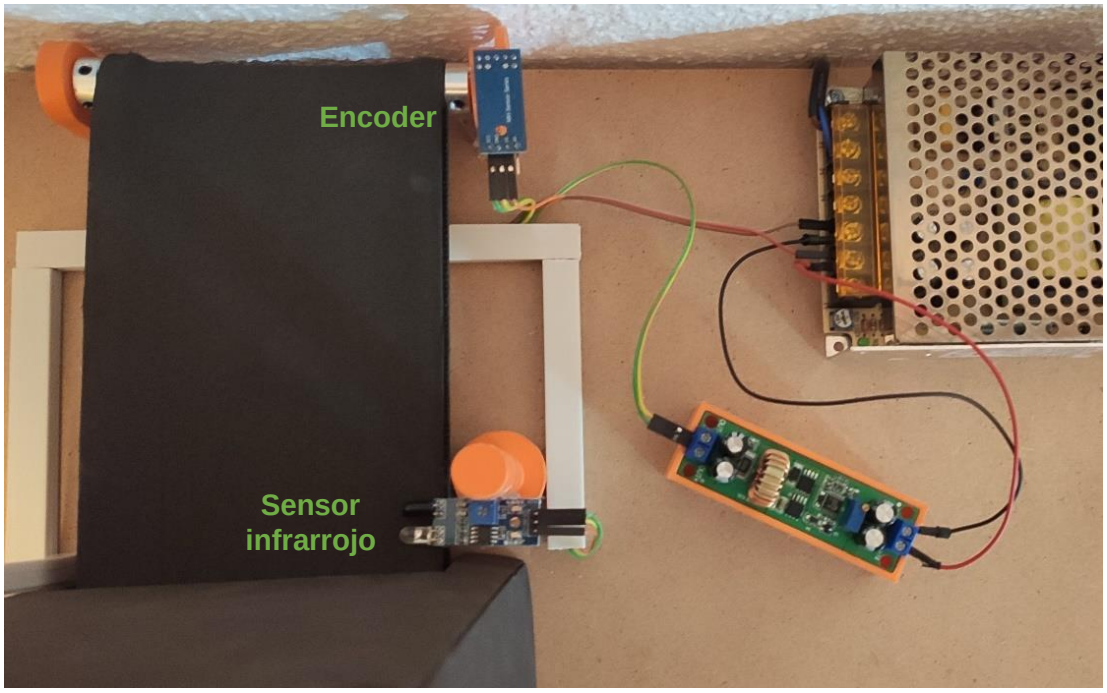


Ilustración 36. Proyecto acabado que muestra el comienzo del tramo de la cinta y la etapa de alimentación.

3.3.2 Control de calidad

En el control de calidad de la ejecución se comprueba que todas las piezas móviles encajan correctamente y se deslizan con poca fricción. Tras montar los diferentes explosionados también se comprueba que su electrónica funciona correctamente. Y finalmente se revisan las uniones que precisan del uso de la pistola termofusible por si el acabado no es consistente o presenta burbujas de aire.

3.4 Pruebas y Ajustes Finales o de Servicio

Antes de comprobar su funcionamiento se han revisado con un téster las conexiones eléctricas y las soldaduras realizadas.

Se han comprobado todas las entradas y salidas de forma individual y a continuación se ha seguido la lógica del programa con el fin de encontrar fallos de programación.

Así mismo, se ha ejecutado repetidamente el proceso entero, desde que se posiciona la pieza en la cinta hasta que el brazo robótico la deposita en su lugar. Esta prueba se ha realizado durante 30 minutos para asegurarse que no se ha sobrecalentado ningún elemento ni descalibrado la cámara.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Documento número 4:

PRESUPUESTO

DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK & PLACE.

TRABAJO FINAL DEL

Grado en Ingeniería Electrónica Industrial y Automática

REALIZADO POR

Jorge Doménech Jara

TUTORIZADO POR

Ángel Valera Fernández

Carlos Ricolfe Viala

CURSO ACADÉMICO: 2019/2020

PRESUPUESTO DE COSTES SEGÚN SU NATURALEZA					
MATERIALES					
Grupo	Código	Descripción	Cantidad (ud)	Precio (€/ud)	Total (€)
1.1. Electrónico					**128,13**
	1.1.1	ESP32cam: cámara con un microprocesador programable en arduino. Cámara OV2640 2MP, CPU 32 bits de doble núcleo de baja potencia, frecuencia de reloj de hasta 240 MHz, soporta interfaces: UART / SPI / I2C / PWM / ADC / DAC.	1,00	23,55	23,55
	1.1.2	ESP32: microprocesador del proyecto. Procesador dual core Xtensa® LX6 de 32 bits, compatible con Arduino, velocidad de reloj 240 MHz, con 520 Kb de RAM y 36 GPIO pins: 16 Analog-to-Digital Converter (ADC) de 12 bits de resolución y 2 Digital to Analog converter DAC de 8 bits. Se pueden definir 16 canales de PWM.	1,00	24,98	24,98
	1.1.3	S120-12: fuente de alimentación de señal de entrada alterna (220 V y 50 Hz) y señal de salida continua (12 V y 10 A).	1,00	37,48	37,48
	1.1.4	Convertidor BUCK: convertidor de corriente continua que reduce la tensión de la señal de entrada de 12 V a 6 V mediante un potenciómetro.	1,00	23,98	23,98
	1.1.5	PCA 9685: controlador de servomotores para potencias mayores de 1 W y menores de 10 W. Compatible con arduino.	1,00	6,63	6,63
	1.1.6	Sensor infrarrojo: detecta objetos a una distancia de entre 2 cm y 8 cm.	1,00	5,65	5,65
	1.1.7	Encoder óptico: Encoder óptico optoacoplador encargado de detectar los pulsos junto la rueda.	1,00	5,88	5,88
1.2. Eléctrico					**63,98**
	1.2.1.1	Cable dupont: cable usado para baja tensión y corriente muy reducida.	4,00	4,63	18,50
	1.2.1.2	Cable de 1 mm^2 de sección: para alimentar los servomotores MG 996R, por su consumo de corriente mayor.	1,00	39,98	39,98
	1.2.2	Tira de LEDs: tira de LEDs recortable, de baja potencia, de color blanco frío y alimentados con 12 V.	1,00	5,50	5,50
1.3. Fuerza motriz					**93,68**
	1.3.1.1	DS04-NFC: motor de rotación continua destinado para mover la cinta. Alimentado con 6 V y compatible con arduino.	1,00	25,98	25,98
	1.3.2.1	SG 90: servomotor con un rango de giro de 180º, alimentado a 5 V, un par de 1,8 kgf·cm y unas dimensiones máximas de 22,2 x 11,8 x 31 mm. Compatible con arduino.	4,00	6,50	26,00
	1.3.2.2	MG 996R: servomotor con un rango de giro de 120º, alimentado a 6 V, con un par máximo de 11 kgf·cm, unas dimensiones máximas de 40,7 x 19,7 x 42,9 mm un pico de corriente máximo de 2,5 A y una consumo de 0,9 A. Compatible con arduino.	4,00	10,43	41,70
1.4. Materiales de plástico					**52,50**
	1.4	Piezas de plástico impresas con ABS, con un relleno del 15% y una altura de capa de 1,5 mm.	1,00	52,50	52,50
1.5. Otros					**43,70**
	1.5.1	Engranajes para la cinta: engranajes de aluminio de 15 mm de diámetro con adaptador y fijador de posición en ejes.	6,00	2,20	13,20
	1.5.2	Goma eva: usado para fabricar la cinta y cubrir la estructura de la cámara.	1,00	4,38	4,38
	1.5.3	Tabla de madera: usado como soporte del proyecto	1,00	5,88	5,88
	1.5.4	Correa dentada: correa con dientes compatible con los engranajes 1.5.1.	1,00	12,38	12,38
	1.5.5	Canaleta de PVC: canaleta destinada a cubrir los cables del proyecto.	1,00	7,63	7,63
	1.5.6	PCB: placa virgen de conexiones.	1,00	0,25	0,25
TOTAL MATERIALES					**381,98**

MANO DE OBRA			
Descripción	Cantidad (h)	Precio (€/h)	Total (€)
Ingeniero junior	300,00	20,00	6.000,00
Ingeniero senior	15,00	50,00	750,00
TOTAL MANO DE OBRA			6.750,00

MEDIOS AUXILIARES			
Descripción	Cantidad (%)	Precio (€)	Total (€)
Medios auxiliares sobre costes directos: uso de impresora 3D, ordenador, software, estaño, heramientas...	10,00	7.131,98	713,20
TOTAL MEDIOS AUXILIARES			713,20
TOTAL PRESUPUESTO DE EJECUCIÓN MATERIAL			7.845,17

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Anejos

DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK & PLACE.

TRABAJO FINAL DEL

Grado en Ingeniería Electrónica Industrial y Automática

REALIZADO POR

Jorge Doménech Jara

TUTORIZADO POR

Ángel Valera Fernández

Carlos Ricolfe Viala

CURSO ACADÉMICO: 2019/2020

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Anejo A-1: Código Arduino de ESP32

DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK & PLACE.

TRABAJO FINAL DEL

Grado en Ingeniería Electrónica Industrial y Automática

REALIZADO POR

Jorge Doménech Jara

TUTORIZADO POR

Ángel Valera Fernández

Carlos Ricolfe Viala

CURSO ACADÉMICO: 2019/2020

```
/*
- Seleccionar la placa "ESP32 Dev Module"
- Seleccionar Partion Scheme "Default 4MB with spiffs
*/

#include <Wire.h>
#include <Adafruit_PWMServoDriver.h>

#define IMPRIMIR_ARRAY(a) {for (int index = 0; index < sizeof(a) / sizeof(a[0]);
index++){Serial.print(a[index]); Serial.print(" ");} Serial.println();};

//usa la dirección por defecto 0x40
Adafruit_PWMServoDriver pwm = Adafruit_PWMServoDriver();

//Definimos los valores de los lados del robot
const float L1=102.25, L2=153, L3=112.4, L4=58.5, L5=120;

//Definimos los ángulos de las articulaciones
float q1=0, q2=0, q3=0, q4=0, q5=0, q6=0;

//PINES
const int EmpezarPIN = 33, HacerFotoPIN = 32, FlashPIN = 17;

//Variables
unsigned int estado = 0, i;
bool repetido = 0;
const unsigned int t=18;    //tiempo de espera entre cambio de ángulos
float distancia = 0.0;

//Comunicación por puerto serial para arrays
String CadenaRecibida="";
const char separador = ',';

//Valores de servos mínimos
unsigned int Nmin[8]={80,80,80,150,150,170,325,200};

//Valores de servos máximos
unsigned int Nmax[8]={360,360,360,555,530,525,380,550};

//Posición inicial de los motores
unsigned int angulosActuales[8]={220,220,220,353,340,348,325,0};

unsigned int angulosDeseados[8]={0,0,0,0,0,0,0,0};

//Del 0-5 servos con angulos, el 6 bool(abrieto/cerrado) y el 7 de continua (0 parado 100 al max de velocidad)
int angulosActualesGrados[8]={135,0,0,0,0,0,1,0};   //Todos en grados
int angulosDeseadosGrados[8]={0,0,0,0,0,0,0,0};     //En grados

//Centroide pieza
int Cx1=0, Cx2=0, Cy1=0, Cy2=0;
float Cx=0.0, Cy=0.0, Cz=0.0;

//Angulo pieza
int alpha=0, alpha1=0, alpha2=0;

unsigned int tipo = 0;    //tipo de pieza al hacer la foto

//Matriz: homografía
const float H[3][3]={{0.0438452499097,  0.1123827205629,  212.4988154199951},
                     {0.2293771673193, -0.0148601381966, -3.0303173196820},
                     {0.0001085366993, -0.0005391815755,  1.0}};
float px=0, py=0, pz=0;
```

```
//Inicializar a 0 los tiempos
long startTimeEncoder = 0, startTimeIR = 0, startTimeCOMUNICATION1 = 0,
startTimeCOMUNICATION2 = 0;

//tiempo en que no se tienen en cuenta las posibles oscilaciones mecánicas: t(ms)=1/nº de detecciones por
vuelta con la condición: w=1rpm
const int timeThresholdEncoder = 25, timeThresholdIR = 100, timeThresholdCOMUNICATION1 = 40,
timeThresholdCOMUNICATION2 = 40;

//Estructuras de las interrupciones:
struct Encoder {
  const uint8_t PIN;
  uint32_t NTicks;
};
struct IR {
  const uint8_t PIN;
  bool Detectado;
};
struct CAMBIODIGITO {
  const uint8_t PIN;
  int CambioDigitoDetectado;
};
struct VALORDIGITO {
  const uint8_t PIN;
  int Valor;
};

//Asignar valores
Encoder Encoder1 = {18, 0};
IR IR1 = {19, false};
CAMBIODIGITO CambioDigito1 = {12, 0};
VALORDIGITO ValorDigito1 = {14, 0};

//Cambiar las coordenadas de la referencia de la cámara a la referencia del brazo robótico
void cambiarRefCamaraARefRobot(){
  pz =  Cx*H[2][0]+Cy*H[2][1]+H[2][2];
  px = (Cx*H[0][0]+Cy*H[0][1]+H[0][2])/pz;
  py = (Cx*H[1][0]+Cy*H[1][1]+H[1][2])/pz;
  pz =  30;
}

//INTERRUPCIONES:
//Interrupción sensor Encoder
void IRAM_ATTR isrEncoder1() {
  if (millis() - startTimeEncoder > timeThresholdEncoder){
    Encoder1.NTicks += 1;
    startTimeEncoder = millis();
  }
}

//Interrupción sensor IR
void IRAM_ATTR isrIR1() {
  if (millis() - startTimeIR > timeThresholdIR){
    IR1.Detectado = true;
    startTimeIR = millis();
  }
}
```

```
//Interrupción para comunicación con ESP32cam: salto de dígito
void IRAM_ATTR isrCOMUNICATION1() {
  if (millis() - startTimeCOMUNICATION1 > timeThresholdCOMUNICATION1){
    CambioDigito1.CambioDigitoDetectado++;

    if(CambioDigito1.CambioDigitoDetectado == 1){
       //es el reinicio de la cámara que produce un pico en todas las salidas pulsando el RESET del disparo de
       la cámara
    }
    else if(CambioDigito1.CambioDigitoDetectado == 2){
      Cx1 = ValorDigito1.Valor;
    }
    else if(CambioDigito1.CambioDigitoDetectado == 3){
      Cx2 = ValorDigito1.Valor;
    }
    else if(CambioDigito1.CambioDigitoDetectado == 4){
      Cx = Cx1*1.0 + Cx2*0.1 + ValorDigito1.Valor*0.01;
    }
    else if(CambioDigito1.CambioDigitoDetectado == 5){
      Cy1 = ValorDigito1.Valor;
    }
    else if(CambioDigito1.CambioDigitoDetectado == 6){
      Cy2 = ValorDigito1.Valor;
    }
    else if(CambioDigito1.CambioDigitoDetectado == 7){
      Cy = Cy1*1.0 + Cy2*0.1 + ValorDigito1.Valor*0.01;
    }
    else if(CambioDigito1.CambioDigitoDetectado == 8){
      alpha1 = ValorDigito1.Valor;
    }
    else if(CambioDigito1.CambioDigitoDetectado == 9){
      alpha2 = ValorDigito1.Valor;
    }
    else if(CambioDigito1.CambioDigitoDetectado == 10){
      alpha = alpha1*100 + alpha2*10 + ValorDigito1.Valor;
    }
    else if(CambioDigito1.CambioDigitoDetectado == 11){
      tipo = ValorDigito1.Valor;
      CambioDigito1.CambioDigitoDetectado = 0;
    }
    ValorDigito1.Valor = 0;
    startTimeCOMUNICATION1 = millis();
  }
}

//Interrupción para comunicación con ESP32cam: valor del dígito
void IRAM_ATTR isrCOMUNICATION2() {
  if (millis() - startTimeCOMUNICATION2 > timeThresholdCOMUNICATION2){
    ValorDigito1.Valor++;
    startTimeCOMUNICATION2 = millis();
  }
}

void IrPosicion(float px, float py, float pz, int alpha, int pinza){

  ciRRR16(px, py, pz, alpha);
  Serial.printf("Cx = %.2f, Cy = %.2f, alpha = %d, tipo = %d \n", Cx, Cy, alpha, tipo);
  Serial.printf("px = %.2f, py = %.2f, pz = %.2f, ángulo = %dº \n", px, py, pz, alpha);
  Serial.printf("Solución c. i. : [%.2f, %.2f, %.2f, %.2f, %.2f, %.2f]\n", q1, q2, q3, q4, q5, q6);

  if(q1<=180){
    angulosDeseadosGrados[0]=q1;
```

```
  }
  if(q2<=60){
    angulosDeseadosGrados[1]=q2;
  }
  if(q3<=60){
    angulosDeseadosGrados[2]=q3;
  }
  if(q4<=90){
    angulosDeseadosGrados[3]=q4;
  }
  if(q5<=90){
    angulosDeseadosGrados[4]=q5;
  }
  if(q6<=90){
    angulosDeseadosGrados[5]=q6;
  }
  angulosDeseadosGrados[6]=pinza;
  angulosDeseadosGrados[7]=0;

  ConversionDeGrados(Nmax, Nmin, angulosDeseadosGrados, angulosActualesGrados, angulosDeseados);
  moverMotores(angulosDeseados, angulosActuales);
}


//Funcion para mover todos los servos
void moverMotores(unsigned int angulosDeseados[8], unsigned int angulosActuales[8]){
  //Servo motor 1:
  if(angulosActuales[0]>angulosDeseados[0]){
    for(i=angulosActuales[0];i>angulosDeseados[0];i--){
      pwm.setPWM(12, 0, i);    //(canal,valor en que empieza on,valor en que acaba on)
      delay(t);
    }
  }
  if(angulosActuales[0]<angulosDeseados[0]){
    for(i=angulosActuales[0];i<angulosDeseados[0];i++){
      pwm.setPWM(12, 0, i);    //(canal,valor en que empieza on,valor en que acaba on)
      delay(t);
    }
  }
  angulosActuales[0]=angulosDeseados[0];

  //Servo motor 2 y 3:
  if(angulosActuales[1]>angulosDeseados[1]){
    for(i=angulosActuales[1];i>angulosDeseados[1];i--){
      pwm.setPWM(14, 0, i);    //(canal,valor en que empieza on,valor en que acaba on)
      delay(t);
    }
  }
  if(angulosActuales[1]<angulosDeseados[1]){
    for(i=angulosActuales[1];i<angulosDeseados[1];i++){
      pwm.setPWM(14, 0, i);    //(canal,valor en que empieza on,valor en que acaba on)
      delay(t);
    }
  }
  angulosActuales[1]=angulosDeseados[1];

  //Servo motor 4:
  if(angulosActuales[2]>angulosDeseados[2]){
    for(i=angulosActuales[2];i>angulosDeseados[2];i--){
      pwm.setPWM(15, 0, i);    //(canal,valor en que empieza on,valor en que acaba on)
      delay(t);
    }
```

```
 }
 if(angulosActuales[2]<angulosDeseados[2]){
   for(i=angulosActuales[2];i<angulosDeseados[2];i++){
     pwm.setPWM(15, 0, i);    //(canal,valor en que empieza on,valor en que acaba on)
     delay(t);
   }
 }
 angulosActuales[2]=angulosDeseados[2];

 //Servo motor 5:
 if(angulosActuales[3]>angulosDeseados[3]){
   for(i=angulosActuales[3];i>angulosDeseados[3];i--){
     pwm.setPWM(8, 0, i);    //(canal,valor en que empieza on,valor en que acaba on)
     delay(t);
   }
 }
 if(angulosActuales[3]<angulosDeseados[3]){
   for(i=angulosActuales[3];i<angulosDeseados[3];i++){
     pwm.setPWM(8, 0, i);    //(canal,valor en que empieza on,valor en que acaba on)
     delay(t);
   }
 }
 angulosActuales[3]=angulosDeseados[3];

 //Servo motor 6:
 if(angulosActuales[4]>angulosDeseados[4]){
   for(i=angulosActuales[4];i>angulosDeseados[4];i--){
     pwm.setPWM(9, 0, i);    //(canal,valor en que empieza on,valor en que acaba on)
     delay(t);
   }
 }
 if(angulosActuales[4]<angulosDeseados[4]){
   for(i=angulosActuales[4];i<angulosDeseados[4];i++){
     pwm.setPWM(9, 0, i);    //(canal,valor en que empieza on,valor en que acaba on)
     delay(t);
   }
 }
 angulosActuales[4]=angulosDeseados[4];

 //Servo motor 7:
 if(angulosActuales[5]>angulosDeseados[5]){
   for(i=angulosActuales[5];i>angulosDeseados[5];i--){
     pwm.setPWM(10, 0, i);    //(canal,valor en que empieza on,valor en que acaba on)
     delay(t);
   }
 }
 if(angulosActuales[5]<angulosDeseados[5]){
   for(i=angulosActuales[5];i<angulosDeseados[5];i++){
     pwm.setPWM(10, 0, i);    //(canal,valor en que empieza on,valor en que acaba on)
     delay(t);
   }
 }
 angulosActuales[5]=angulosDeseados[5];

 //Servo motor 8:
 if(angulosActuales[6]>angulosDeseados[6]){
   for(i=angulosActuales[6];i>angulosDeseados[6];i--){
     pwm.setPWM(11, 0, i);    //(canal,valor en que empieza on,valor en que acaba on)
     delay(t);
   }
 }
  if(angulosActuales[6]<angulosDeseados[6]){
```

```
    for(i=angulosActuales[6];i<angulosDeseados[6];i++){
      pwm.setPWM(11, 0, i);    //(canal,valor en que empieza on,valor en que acaba on)
      delay(t);
    }
   }
   angulosActuales[6]=angulosDeseados[6];

   //Servo motor 9:
  if(angulosActuales[7]!=angulosDeseados[7]){
    if(angulosDeseados[7]==0){
      pwm.setPWM(0, 0, 0);    //(canal,valor en que empieza on,valor en que acaba on)
    }
    if(angulosDeseados[7]==1){
      pwm.setPWM(0, 0, 120);    //(canal,valor en que empieza on,valor en que acaba on)
    }
  }
  angulosActuales[7]=angulosDeseados[7];
}



void ConversionDeGrados(unsigned int Nmax[8], unsigned int Nmin[8], int angulosDeseadosGrados[8], int
angulosActualesGrados[8], unsigned int angulosDeseados[8]){

  //Servo motor 1:
  angulosDeseados[0]=angulosDeseadosGrados[0]*(((Nmax[0]-Nmin[0])/180.0))+Nmin[0];  //conversión de
grados
  angulosActualesGrados[0]=angulosDeseadosGrados[0];

  //Servo motor 2 y 3:
  //anguloDeseado[1]=angulosDeseadosGrados[1]*((-1.0)*(Nmax[1]-
Nmin[1])/120.0)+((Nmax[1]+Nmin[1])/2.0);  //conversión de grados motor 2
  angulosDeseados[1]=angulosDeseadosGrados[1]*((Nmax[1]-
Nmin[1])/120.0)+((Nmax[1]+Nmin[1])/2.0);  //conversión de grados motor 3
  angulosActualesGrados[1]=angulosDeseadosGrados[1];

  //Servo motor 4:
  angulosDeseados[2]=angulosDeseadosGrados[2]*((Nmax[2]-
Nmin[2])/120.0)+((Nmax[2]+Nmin[2])/2.0);  //conversión de grados
  angulosActualesGrados[2]=angulosDeseadosGrados[2];



  //Servo motor 5:
  angulosDeseados[3]=angulosDeseadosGrados[3]*((Nmax[3]-Nmin[3])/180.0)+352.5;  //conversión de
grados
  angulosActualesGrados[3]=angulosDeseadosGrados[3];

  //Servo motor 6:
  angulosDeseados[4]=angulosDeseadosGrados[4]*((Nmax[4]-Nmin[4])/180.0)+340;  //conversión de grados
  angulosActualesGrados[4]=angulosDeseadosGrados[4];

  //Servo motor 7:
  angulosDeseados[5]=angulosDeseadosGrados[5]*((Nmax[5]-Nmin[5])/180.0)+348;  //conversión de grados
  angulosActualesGrados[5]=angulosDeseadosGrados[5];

  //Servo motor 8:
  angulosDeseados[6]=(Nmin[6]*angulosDeseadosGrados[6])-(Nmax[6]*(angulosDeseadosGrados[6]-1));
//conversión abierto-cerrado (pinza)
  angulosActualesGrados[6]=angulosDeseadosGrados[6];

  //Servo motor 9 (cinta):
```

```
  angulosDeseados[7]=angulosDeseadosGrados[7]; //velocidad cinta
  angulosActualesGrados[7]=angulosDeseadosGrados[7];
}


void ciRRR16(float pxx, float pyy, float pzz, int anguloPinza){

  float dx=pxx, dy=pyy, dz=pzz+L5-L1;

  float L=L3+L4;

  //Cálculo de q1 considerando el cuadrante en el que esté
  if((dx>0)&&(dy>0)){
     q1=atan(dy/dx);
  }
  else if((dx>0)&&(dy<0)){
     q1=atan(dy/dx);
  }
  else if((dx<0)&&(dy>0)){
     q1=PI+atan(dy/dx);
  }
  else if((dx<0)&&(dy<0)){
     q1=-PI+atan(dy/dx);
  }
  else if((dx==0)&&(dy>0)){
     q1=PI/2;
  }
  else if((dx==0)&&(dy<0)){
     q1=3*(PI/2);
  }
  else if((dx>0)&&(dy==0)){
     q1=0;
  }
  else if((dx<0)&&(dy==0)){
     q1=PI;
  }

  //MUÑECA POR ARRIBA DEL PUNTO, L2 IZQUIERDA
  //Trigonometría para obtener los valores de los â
  float r=-sqrt(pow(dx,2)+pow(dy,2));

  //Cálculo de q3
  q3=acos((pow(dx,2)+pow(dy,2)+pow(dz,2)-pow(L2,2)-pow(L,2))/(2*L2*L));

  //Cálculo de q2
  q2=atan(dz/r)-atan((L*sin(q3))/((L2+L*cos(q3))));

  q2=q2+(PI/2);
  q3=q3-(PI/2);
  q4=atan(0);
  q5=acos((cos(q2)*sin(q3)+cos(q3)*sin(q2))/(pow(cos(q2),2)*pow(cos(q3),2)+pow(cos(q2),2)*pow(sin(q3),
2) + pow(cos(q3),2)*pow(sin(q2),2) + pow(sin(q2),2)*pow(sin(q3),2)));
  q6=atan(sin(q1)/cos(q1));

  q1=(q1*(180/PI))+135;
  q2=q2*(180/PI)-10.5;
  q3=q3*(180/PI)-9;
  q4=q4*(180/PI);
  q5=q5*(180/PI)-11;
  q6=q1-135-20+anguloPinza;
}
```

```
void setup() {
  Serial.begin(115200);//Comunicación serial a 115200 de vel. de subida

  pwm.begin();       //Llamar a la librería de PWM
  pwm.setPWMFreq(50); //Poner a 50Hz la frecuencia máxima de PWM

  //Pines Interrupciones
  pinMode(Encoder1.PIN, INPUT_PULLUP);
  pinMode(IR1.PIN, INPUT_PULLUP);
  pinMode(CambioDigito1.PIN, INPUT_PULLUP);
  pinMode(ValorDigito1.PIN, INPUT_PULLUP);

  pinMode(EmpezarPIN, INPUT_PULLUP);
  pinMode(HacerFotoPIN, OUTPUT);

  Serial.println("\n\n\nEmpieza el programa:\n");
}

void loop() {

  //ESTADOS:
  switch (estado) {
  case 0: //Esperar a que se inicie el bucle
    if(repetido==0){
       Serial.println("\n\nESTADO 0");
       repetido=1;
      }
    break;

  case 1: //Encender cinta y activar interrupción IR
    if(repetido==0){
       Serial.println("\n\nESTADO 1");
       attachInterrupt(IR1.PIN, isrIR1, FALLING);
       pwm.setPWM(0, 0, 120);
       repetido=1;
      }
    break;

  case 2: //Activar interrupciones encoder y desactivar la del IR
    if(repetido==0){
       Serial.println("\n\nESTADO 2");
       IR1.Detectado = false;
       Encoder1.NTicks=0;
       attachInterrupt(Encoder1.PIN, isrEncoder1, CHANGE);
       detachInterrupt(IR1.PIN);
       repetido=1;
      }
    break;

  case 3: //Parar cinta, desactivar interrupciones del encoder, ordenar que haga foto y activar interrupciones
para transmisión de datos
    if(repetido==0){
       Serial.println("\n\nESTADO 3");

       pwm.setPWM(0, 0, 0);
       detachInterrupt(Encoder1.PIN);

       float d = 0.28274 * 0.5 * Encoder1.NTicks;
       Serial.printf("Ha avanzado %.2f cm", d);

       Encoder1.NTicks=0;
```

```
      attachInterrupt(CambioDigito1.PIN, isrCOMUNICATION1, FALLING);
      attachInterrupt(ValorDigito1.PIN, isrCOMUNICATION2, FALLING);

      digitalWrite(HacerFotoPIN, HIGH);
      delay(110);
      digitalWrite(HacerFotoPIN, LOW);

      delay(1000);
      long trespuesta = millis();
      while(tipo == 0){
       if(millis()-trespuesta>5500){
         digitalWrite(HacerFotoPIN, HIGH);
         delay(120);
         digitalWrite(HacerFotoPIN, LOW);
         trespuesta = millis();
         Serial.println("Hacer foto otra vez");
       }
      }
      repetido=1;
    }
   break;

  case 4: //Tras recibir la información desactivar interrupciones para transmisión de datos y activar el encoder y encender la cinta
   if(repetido==0){
      Serial.println("\n\nESTADO 4");

      Serial.printf("CentroideRefCamara = [%.2f, %.2f, %.2f] píxeles\n", Cx, Cy, Cz);

      if(tipo==2){          //Para coger a pieza cuadrada.
        alpha=alpha-45;     //Convertir Rg  [90,0] -> [45, -45]
      }
      if(tipo==3){          //Para coger a pieza rectangular por el lado estrecho.
        alpha=90-alpha;     //Convertir Rg [180,0] -> [-90, 90]
      }

      detachInterrupt(CambioDigito1.PIN);
      detachInterrupt(ValorDigito1.PIN);

      attachInterrupt(Encoder1.PIN, isrEncoder1, CHANGE);
      pwm.setPWM(0, 0, 120);
      repetido=1;
    }
   break;

  case 5: //Apagar cinta, desactivar interrupción encoder, coger pieza y posicionarla en su sitio
   if(repetido==0){
      Serial.println("\n\nESTADO 5");

      pwm.setPWM(0, 0, 0);
      detachInterrupt(Encoder1.PIN);

      float d = 0.28274 * 0.5 * Encoder1.NTicks;
      Serial.printf("Ha avanzado %.2f cm", d);

      Encoder1.NTicks=0;

      //conversión a pixeles
      Cy=Cy*100;
      Cx=Cx*100;

      cambiarRefCamaraARefRobot();//Matriz homeográfica
```

```
      Serial.printf("Centroide Ref Cámara = [%.2f, %.2f] píxeles\n", Cx, Cy);
      Serial.printf("Centroide Ref Robot = [%.2f, %.2f, %.2f] mm\n", px, py, pz);
      Serial.printf("Orinetación de la pieza = %dº \n",alpha);
      Serial.printf("Tipo de pieza = %d \n\n",tipo);

      //Posicionarse para coger pieza
            //(Coord x,Coord y,Coord z,ángulo orinet,abrir/cerrar(1/0) pinza)

      IrPosicion(px, py, pz+45, alpha, 1);
     delay(3000);
      //coger pieza
      IrPosicion(px, py, pz, alpha, 0);
      delay(3000);
      //Separarse de la cinta
      IrPosicion(px, py, pz+45, alpha, 0);
      delay(3000);
      //Posicionarse en el sitio para dejarla y abrir la pinza
      if(tipo == 2){
        IrPosicion(160, -140, 100, 0, 0);
        IrPosicion(100, -150, 40, 0, 0);
        IrPosicion(100, -150, 10, 0, 1);
        IrPosicion(160, -140, 100, 0, 1);
      }
      else{
        IrPosicion(160, -140, 100, 0, 0);
        IrPosicion(10, -150, 40, 0, 0);
        IrPosicion(10, -150, 10, 0, 1);
        IrPosicion(160, -140, 100, 0, 1);
      }
      delay(3000);

      repetido=1;
    }
   break;
 default:
   Serial.println("ERROR EN ASIGNAR ESTADO");
   break;
}

 //TRANSICIONES DE LOS ESTADOS:

 if((estado==0)&&(digitalRead(EmpezarPIN) == LOW)){
   estado=1;
   repetido=0;
 }
 else if((estado==1)&&(IR1.Detectado == true)){
   estado=2;
   repetido=0;
 }
 else if((estado==2)&&(2.8274*0.5*Encoder1.NTicks>=110)){
 //distancia=2.8274*Encoder1.NTicks;   //En mm
   estado=3;
   repetido=0;
 }
 else if((estado==3)&&(tipo != 0)){ //información recibida
   estado=4;
   repetido=0;
 }
 else if((estado==4)&&(2.8274*0.5*Encoder1.NTicks >= 340)){   //distancia donde el robot cogerá la pieza
   estado=5;
   repetido=0;
```

```
  }
  else if((estado==5)&&(digitalRead(EmpezarPIN)==LOW)){
    estado=1;
    repetido=0;
    Encoder1.NTicks=0;
    Cx1=0;
    Cx2=0;
    Cy1=0;
    Cy2=0;
    Cx=0.0;
    Cy=0.0;
    tipo=0;
  }
}
```

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Anejo A-2: Código Arduino de ESP32-cam

DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK & PLACE.

TRABAJO FINAL DEL

Grado en Ingeniería Electrónica Industrial y Automática

REALIZADO POR

Jorge Doménech Jara

TUTORIZADO POR

Ángel Valera Fernández

Carlos Ricolfe Viala

CURSO ACADÉMICO: 2019/2020

```
/*
- Seleccionar Placa "ESP32 Wrover Module"
- Seleccionar Partion Scheme "Huge APP (3MB No OTA)
- GPIO 0 debe estar conectada con GND para subir el sketch
- Después de conectar GPIO 0 con GND, apretar el botón de RESET de la ESP32-CAM para poner la placa en "flashing mode"
*/
#include <stdint.h>
#include "esp_camera.h"
#include "esp_timer.h"
#include "img_converters.h"
#include "Arduino.h"
#include "fb_gfx.h"
#include "fd_forward.h"
#include "fr_forward.h"
#include "soc/soc.h"             //Desactivar para problemas de brownour
#include "soc/rtc_cntl_reg.h"  //Desactivar para problemas de brownour
#include "driver/rtc_io.h"

//Definición de pines para el modelo de cámara AI_THINKER
#define PWDN_GPIO_NUM     32
#define RESET_GPIO_NUM    -1
#define XCLK_GPIO_NUM      0
#define SIOD_GPIO_NUM     26
#define SIOC_GPIO_NUM     27

#define Y9_GPIO_NUM       35
#define Y8_GPIO_NUM       34
#define Y7_GPIO_NUM       39
#define Y6_GPIO_NUM       36
#define Y5_GPIO_NUM       21
#define Y4_GPIO_NUM       19
#define Y3_GPIO_NUM       18
#define Y2_GPIO_NUM        5
#define VSYNC_GPIO_NUM    25
#define HREF_GPIO_NUM     23
#define PCLK_GPIO_NUM     22

int pictureNumber = 0;
int comp=0, g=0, f=0;

int Comunicacion1Pin = 14, Comunicacion2Pin = 15, HacerFotoPIN = 13;   //PINS
int tiempo=25;

float Cx=9.53, Cy=5.54;     //Centroide pieza: formato X.xx (pixeles/100)

unsigned int tipo = 2;          //tipo de pieza al hacer la foto
int alpha=0;                    //Orientación pieza

int Valor = 0, unidad = 0, decima = 0, centesima = 0;

long k=0, startTimeInterrupcion=0, timeThresholdFoto=3000;
boolean imagenRoja[168][210];
int foto=0;

void EnviarDatos(){
  digitalWrite(Comunicacion1Pin, HIGH);
  delay(tiempo);
  digitalWrite(Comunicacion1Pin, LOW);
  delay(tiempo);
```

```
//Enviar Cx:
Valor = Cx*100;
unidad = Valor/100;
decima = (Valor - unidad*100)/10;
centesima = (Valor - unidad*100 - decima*10);
for(g=0;g<3;g++){
  if(g==0){
    comp=unidad;
  }
  else if(g==1){
    comp=decima;
  }
  else{
    comp=centesima;
  }
  for(f=0;f<comp;f++){
    digitalWrite(Comunicacion2Pin, HIGH);
    delay(tiempo);
    digitalWrite(Comunicacion2Pin, LOW);
    delay(tiempo);
  }
  digitalWrite(Comunicacion1Pin, HIGH);
  delay(tiempo);
  digitalWrite(Comunicacion1Pin, LOW);
  delay(tiempo);
}

//Enviar Cy:
Valor = Cy*100;
unidad = Valor/100;
decima = (Valor - unidad*100)/10;
centesima = (Valor - unidad*100 - decima*10);
for(g=0;g<3;g++){
  if(g==0){
    comp=unidad;
  }
  else if(g==1){
    comp=decima;
  }
  else{
    comp=centesima;
  }
  for(f=0;f<comp;f++){
    digitalWrite(Comunicacion2Pin, HIGH);
    delay(tiempo);
    digitalWrite(Comunicacion2Pin, LOW);
    delay(tiempo);
  }
  digitalWrite(Comunicacion1Pin, HIGH);
  delay(tiempo);
  digitalWrite(Comunicacion1Pin, LOW);
  delay(tiempo);
}
```

```
  //Enviar alpha:
  if(tipo==2){
    Valor = 45+alpha;     //Rg [45, -45]->[90, 0]
  }
  else if(tipo==3){
    Valor = 90+alpha;     //Rg [90, -90]->[180, 0]
  }
  unidad = Valor/100;
  decima = (Valor - unidad*100)/10;
  centesima = (Valor - unidad*100 - decima*10);
  for(g=0;g<3;g++){
    if(g==0){
      comp=unidad;
    }
    else if(g==1){
      comp=decima;
    }
    else{
      comp=centesima;
    }
    for(f=0;f<comp;f++){
      digitalWrite(Comunicacion2Pin, HIGH);
      delay(tiempo);
      digitalWrite(Comunicacion2Pin, LOW);
      delay(tiempo);
    }
    digitalWrite(Comunicacion1Pin, HIGH);
    delay(tiempo);
    digitalWrite(Comunicacion1Pin, LOW);
    delay(tiempo);
  }

  //Enviar tipo de pieza:
  comp=tipo;
  for(f=0;f<comp;f++){
    digitalWrite(Comunicacion2Pin, HIGH);
    delay(tiempo);
    digitalWrite(Comunicacion2Pin, LOW);
    delay(tiempo);
  }
    digitalWrite(Comunicacion1Pin, HIGH);
    delay(tiempo);
    digitalWrite(Comunicacion1Pin, LOW);
    delay(tiempo);
}


void setup() {
  WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0); //disable brownout detector

  Serial.begin(115200);

  camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
  config.pin_d6 = Y8_GPIO_NUM;
```

```
  config.pin_d7 = Y9_GPIO_NUM;
  config.pin_xclk = XCLK_GPIO_NUM;
  config.pin_pclk = PCLK_GPIO_NUM;
  config.pin_vsync = VSYNC_GPIO_NUM;
  config.pin_href = HREF_GPIO_NUM;
  config.pin_sscb_sda = SIOD_GPIO_NUM;
  config.pin_sscb_scl = SIOC_GPIO_NUM;
  config.pin_pwdn = PWDN_GPIO_NUM;
  config.pin_reset = RESET_GPIO_NUM;
  config.xclk_freq_hz = 20000000;
  config.pixel_format = PIXFORMAT_JPEG;

  if(psramFound()){
   //UXGA FRAMESIZE_ + QVGA|CIF|VGA|SVGA|XGA|SXGA|UXGA
   config.frame_size = FRAMESIZE_VGA;
   config.jpeg_quality = 10;          //640 x 480
    config.fb_count = 2;
  } else {
    config.frame_size = FRAMESIZE_SVGA;
    config.jpeg_quality = 12;
    config.fb_count = 1;
  }

  // Init Camera
  esp_err_t err = esp_camera_init(&config);
  if (err != ESP_OK) {
    Serial.printf("Camera init failed with error 0x%x", err);
    return;
  }

  //Interrupciones
  pinMode(HacerFotoPIN, INPUT_PULLUP);
  pinMode(Comunicacion1Pin, OUTPUT);
  pinMode(Comunicacion2Pin, OUTPUT);
}

void loop() {
if(digitalRead(HacerFotoPIN)==1){
    foto=1;
    Serial.println("Haz foto");
}
if(foto==1){
  camera_fb_t * fb = NULL;
  uint8_t * _bmp_buf = NULL;
  size_t _bmp_buf_len = 0;

  //Tomar una imagen con la cámara
  fb = esp_camera_fb_get();
  if(!fb) {
    Serial.println("Camera capture failed");
    return;
  }
  Serial.println("Imagen tomada");

  bool imagen = frame2bmp(fb,&_bmp_buf,&_bmp_buf_len);
  esp_camera_fb_return(fb);

  //Transformar la matriz _bmp_buf a una matriz imagen:
  //Segmentación: ThresHold
  long SumCoordX=0;
  long SumCoordY=0;
  int nPixeles=0;
```

```
 k=153600;
 for(int i=0;i<168;i++){
   k=k+540;
   for(int l=0;l<210;l++){
     if(_bmp_buf[k]>240){
       imagenRoja[i][l]=1;
       SumCoordX=SumCoordX+l;
       SumCoordY=SumCoordY+i;
       nPixeles=nPixeles+1;
     }
     if(_bmp_buf[k]<=140){
       imagenRoja[i][l]=0;
     }

     k=k+6;
   }
   k=k+120;
   k=k+1920;
 }
 Serial.println(" ");

 //Tipo de pieza:
 if(nPixeles>=3000){
   tipo=3;
 }
 else tipo=2;

//Para imprimir por serial la imagen segmentada
//  for(int i=0;i<168;i++){
//    for(int l=0;l<210;l++){
//      if(imagenRoja[i][l]==1){
//        Serial.print("255");
//      }
//      if(imagenRoja[i][l]==0){
//        Serial.print("0");
//      }
//      Serial.print(" ");
//    }
//    Serial.println(";");
//  }
//Para imprimir por serial la imagen completa
//  int j=0;
//  for(int i=0;i<_bmp_buf_len;i++){    //480*640*3 aprox.
//    j++;
//    Serial.print(_bmp_buf[i]);
//    if(j%3==0){
//        Serial.println(";");
//      }
//    else{
//        Serial.print(",");
//      }
//  }

//Obtener centroides:
Cx=SumCoordX/nPixeles;
Cy=SumCoordY/nPixeles;

//Obtener orientación:
float distACentroide=0.0;
float distACentroideMayor=0.0;
float angOrientac=0.0;
```

```
float CoordenadasPtoMasLejano[2]={0.0,0.0};

for(int j=0;j<168;j++){
   for(int i=0;i<210;i++){
      if(imagenRoja[j][i] == 1){
         distACentroide=sqrt(pow(Cx-j,2)+pow(Cy-i,2));
         if(distACentroideMayor < distACentroide){
            distACentroideMayor=distACentroide;
            CoordenadasPtoMasLejano[0]=j;
            CoordenadasPtoMasLejano[1]=i;
         }
      }
   }
}

if(tipo==2){       //Cuadrado diagonal-horizontal -> 45º
  float valorTan=abs((Cy-CoordenadasPtoMasLejano[1])/(Cx-CoordenadasPtoMasLejano[0]));
  angOrientac=abs(atan(valorTan)*180/PI)-45;
  alpha=(int)angOrientac;
}

if(tipo==3){     //Rectángulo diagonal-horizontal -> 32º

  float valorTan=abs((Cy-CoordenadasPtoMasLejano[1])/(Cx-CoordenadasPtoMasLejano[0]));
  angOrientac=abs(atan(valorTan)*180/PI);
  alpha=(int)angOrientac;

  Serial.print("Alpha = ");
  Serial.println(alpha);

  //Obtener vértices del rectángulo:
  int v1[2]={0,0}, v2[2]={0,0}, v3[2]={0,0}, v4[2]={0,0};
  for(int j=0;j<168;j++){
    for(int i=0;i<210;i++){
      if((imagenRoja[j][i] == 1)&&(v1[0] == 0)){
         v1[0]=j;
         v1[1]=i;
      }
    }
  }
  Serial.printf("V1 = [%d, %d]\n",v1[0],v1[1]);
  for(int i=0;i<210;i++){
    for(int j=0;j<168;j++){
      if((imagenRoja[j][i] == 1)&&(v2[0] == 0)){
         v2[0]=j;
         v2[1]=i;
      }
    }
  }
  Serial.printf("V2 = [%d, %d]\n",v2[0],v2[1]);
  for(int j=165;j>2;j--){
    for(int i=0;i<210;i++){
      if((imagenRoja[j][i] == 1)&&(v3[0] == 0)){
         v3[0]=j;
         v3[1]=i;
      }
    }
  }
```

```
  Serial.printf("V3 = [%d, %d]\n",v3[0],v3[1]);
  for(int i=200;i>10;i--){
    for(int j=0;j<168;j++){
       if((imagenRoja[j][i] == 1)&&(v4[0] == 0)){
          v4[0]=j;
          v4[1]=i;
       }
    }
  }
  Serial.printf("V4 = [%d, %d]\n",v4[0],v4[1]);

  //Selecciono V1 y busco el 2º vértice más cercano o lejano:
  float d12=sqrt(pow(v1[0]-v2[0],2)+pow(v1[1]-v2[1],2));
  float d13=sqrt(pow(v1[0]-v3[0],2)+pow(v1[1]-v3[1],2));
  float d14=sqrt(pow(v1[0]-v4[0],2)+pow(v1[1]-v4[1],2));
  float al=0.0, dif1=0.0, dif2=0.0;

  if((d12>d13)&&(d12<d14)){
    dif1=v1[1]-v2[1];
    dif2=v1[0]-v2[0];
    al=atan2(abs(v1[1]-v2[1]),abs(v1[0]-v2[0]))*180/PI;
    Serial.printf("al1 = %f\n",al);
  }
  else if((d12>d14)&&(d12<d13)){
    dif1=v1[1]-v2[1];
    dif2=v1[0]-v2[0];
    al=atan2(abs(v1[1]-v2[1]),abs(v1[0]-v2[0]))*180/PI;
    Serial.printf("al2 = %f\n",al);
  }
  else if((d13>d12)&&(d13<d14)){
    dif1=v1[1]-v3[1];
    dif2=v1[0]-v3[0];
    al=atan2(abs(v1[1]-v3[1]),abs(v1[0]-v3[0]))*180/PI;
    Serial.printf("al3 = %f\n",al);
  }
  else if((d13>d14)&&(d13<d12)){
    dif1=v1[1]-v3[1];
    dif2=v1[0]-v3[0];
    al=atan2(abs(v1[1]-v3[1]),abs(v1[0]-v3[0]))*180/PI;
    Serial.printf("al4 = %f\n",al);
  }
  else if((d14>d12)&&(d14<d13)){
    dif1=v1[1]-v4[1];
    dif2=v1[0]-v4[0];
    al=atan2(abs(v1[1]-v4[1]),abs(v1[0]-v4[0]))*180/PI;
    Serial.printf("al5 = %f\n",al);
  }
  else if((d14>d13)&&(d14<d12)){
    dif1=v1[1]-v4[1];
    dif2=v1[0]-v4[0];
    al=atan2(abs(v1[1]-v4[1]),abs(v1[0]-v4[0]))*180/PI;
    Serial.printf("al6 = %f\n",al);
  }
  if(dif1*dif2>=0){
    alpha=(int)al;
  }
  else if(dif1*dif2<0){
    alpha=(-1)*(int)al;
  }
}

//Serial.println("LISTO PARA ENVIAR INFORMACIÓN");
```

```
  //Borrar imágen
  for(int i=0;i<168;i++){
    for(int l=0;l<210;l++){
        imagenRoja[i][l]=0;
      }
  }
  //Cambiar escala
  Cx=(2*Cx)/100;
  Cy=(2*Cy)/100;

  Serial.println(CoordenadasPtoMasLejano[0]);
  Serial.println(CoordenadasPtoMasLejano[1]);
  Serial.print("Alpha = ");
  Serial.println(alpha);
  Serial.print("Tipo de pieza = ");
  Serial.println(tipo);

  EnviarDatos();
  Serial.println("Datos ENVIADOS");
  foto=0;
  delay(250);
}
delay(100);
}
```

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Anejo A-3: Código MATLAB

DISEÑO, DESARROLLO Y PROGRAMACIÓN DE UN BRAZO ROBOT DE 6 GRADOS DE LIBERTAD. UTILIZACIÓN EN APLICACIONES DE PICK & PLACE.

TRABAJO FINAL DEL

Grado en Ingeniería Electrónica Industrial y Automática

REALIZADO POR

Jorge Doménech Jara

TUTORIZADO POR

Ángel Valera Fernández

Carlos Ricolfe Viala

CURSO ACADÉMICO: 2019/2020

```
%SCRIPT CINEMÁTICA DIRECTA:
q1 = ángulo 1 * (pi/180);
q2 = ángulo 2 * (pi/180);
q3 = ángulo 3 * (pi/180);
q4 = ángulo 4 * (pi/180);
q5 = ángulo 5 * (pi/180);
q6 = q1;

%Datos
L1 = 102.25;
L2 = 153;
L3 = 112.4;
L4 = 58.5;
L5 = 120;

A01=[cos(q1) 0 -sin(q1) 0;
     sin(q1) 0  cos(q1) 0;
           0 -1       0 L1;
           0  0       0  1];
A12=[ sin(q2) cos(q2) 0  L2*sin(q2);
     -cos(q2) sin(q2) 0 -L2*cos(q2);
           0       0 1           0;
           0       0 0           1];
A23=[cos(q3)   0 -sin(q3) 0;
     sin(q3)   0  cos(q3) 0;
           0 -1       0 0;
           0  0       0  1];
A34=[-cos(q4)   0 -sin(q4)      0;
      sin(q4)   0 -cos(q4)      0;
           0 -1       0 L3+L4;
           0  0       0      1];
A45=[ sin(q5)    0 cos(q5) 0;
     -cos(q5)   0  sin(q5) 0;
           0 -1       0  0;
           0  0       0  1];
A56=[cos(q6) -sin(q6) 0   0;
     sin(q6)  cos(q6) 0   0;
           0       0 1 L5;
           0       0 0   1];
round(A01*A12*A23*A34*A45*A56)
```

```
%SCRIPT PARA MOSTRAR LA IMAGEN DE LA CÁMARA:
RGB=[]; %Poner valores obtenidos de la interfaz de comunicación de la ESP32-cam
RGB = uint8(RGB);
image(RGB)
axis equal
```

```
%SCRIPT FUNCIÓN HOMOGRÁFICA:
function [H, A] = homografia(X,U)
% [Hest] = homomografia(X,U)
% Calcula la homografía entre dos imágenes de las cuales se tienen puntos que coinciden X y U
% X - matriz con los puntos en la imagen 1 en coordenadas homogéneas organizados en columnas
% U - matriz de los puntos en la imagen 2 en coordenadas homogéneas organizados en columnas
% H - homografía estimada estimada
% Estimación de la homografía:
nPuntos = size(X,2);

A=[];
for (i=1:1:nPuntos)
   A=[A; X(:,i)' zeros(1,3) -U(1,i)*X(:,i)';
      zeros(1,3) X(:,i)' -U(2,i)*X(:,i)'];
end
% Homografía estimada
b = -A(:,9);
a = A(:,1:8);
C = inv(a'*a);
H = C*a'*b;

H=[H;1];

H=reshape(H,3,3)';
```

```
%SCRIPT PARA LA OBTENCIÓN DE LA MATRIZ HOMOGRÁFICA:
% Grupo de puntos (ejemplo)
X=[ 64 242 1;
   280  54 1;
   194 154 1;
   284 148 1;
    72 138 1;
    64  54 1;
   160  62 1;
   286 230 1;
   184  226 1];
U=[275 10 30;
   230 60 30;
   255 45 30;
   255 60 30;
   250 10 30;
   225 11 30;
   230 35 30;
   275 67 30;
   275 38 30];

x=X(1:9,1:3)';
u=U(1:9,1:3)';

H=homografia(x,u);

error=[];
for i=1:1:9
    n=H*X(i,:)';
    m=n./n(3)+[0 0 29]';
    error(i,:)=(U(i,:)'-m)';
end

abs(error)

plot(X(:,1),X(:,2),'*g');
hold on
axis equal
plot(U(:,1),U(:,2),'*r');
legend('Pntos cámara','Pntos brazo');

zp=X(9,1)*H(3,1)+X(9,2)*H(3,2)+H(3,3);
xp=(X(9,1)*H(1,1)+X(9,2)*H(1,2)+H(1,3))/zp
xp=(X(9,1)*H(2,1)+X(9,2)*H(2,2)+H(2,3))/zp
zp=40

H*X(9,:)'./n(3)+[0 0 29]'
n=H*[280  54 1]'
H*[280  54 1]'./n(3)+[0 0 29]'

format long
format short
```

%SCRIPT PARA LA OBTENCIÓN DE LA FÓRMULA DE LAS 3 ÚLTIMAS ARTICULACIONES:

```
syms q1 q2 q3 L1 L2 x y z
A01=[cos(q1) 0 -sin(q1) 0;
        sin(q1)  0  cos(q1) 0;
        0         -1    0       L1;
        0          0     0       1];
A12=[-sin(q2) -cos(q2) 0 -L2*sin(q2);
         cos(q1) -sin(q2) 0  L2*cos(q2);
         0               0       1  0;
         0               0       0  1];
A23=[cos(q3) 0 -sin(q3) 0;
        sin(q3)  0  cos(q3) 0;
          0       -1      0       0;
          0        0      0       1];

A03=A01*A12*A23;
R03=A03(1:3,1:3);
R30=inv(R03);

A06=[1 0 0 x;
        0 1 0 y;
        0 0 -1 z;
        0 0 0 1];

R=R30*A06(1:3,1:3);
q4=atan(R(2,3)/R(1,3));
q5=acos(R(3,3))-(pi/2);
q6=atan(R(3,2)/-R(3,1));

%Se obtiene las fórmulas siguientes:
%q4=atan((-(sin(q1)*(cos(q1)*cos(q3)sin(q2)*sin(q3))) / (cos(q2)*pow(cos(q1),3)*pow(cos(q3),2) +
cos(q2)*pow(cos(q1),3)*pow(sin(q3),2) + pow(cos(q1),2)*pow(cos(q3),2)*pow(sin(q2),2) +
pow(cos(q1),2)*pow(sin(q2),2)*pow(sin(q3),2) + cos(q2)*cos(q1)*pow(cos(q3),2)*pow(sin(q1),2) +
cos(q2)*cos(q1)*pow(sin(q1),2)*pow(sin(q3),2) + pow(cos(q3),2)*pow(sin(q1),2)*pow(sin(q2),2) +
pow(sin(q1),2)*pow(sin(q2),2)*pow(sin(q3),2))), (-(cos(q1)*(cos(q1)*cos(q3) - sin(q2)*sin(q3))) /
(cos(q2)*pow(cos(q1),3)*pow(cos(q3),2) + cos(q2)*pow(cos(q1),3)*pow(sin(q3),2) +
pow(cos(q1),2)*pow(cos(q3),2)*pow(sin(q2),2) + pow(cos(q1),2)*pow(sin(q2),2)*pow(sin(q3),2) +
cos(q2)*cos(q1)*pow(cos(q3),2)*pow(sin(q1),2) + cos(q2)*cos(q1)*pow(sin(q1),2)*pow(sin(q3),2) +
pow(cos(q3),2)*pow(sin(q1),2)*pow(sin(q2),2) + pow(sin(q1),2)*pow(sin(q2),2)*pow(sin(q3),2))));

%q5=acos((cos(q2)*sin(q3) + cos(q3)*sin(q2)) / (pow(cos(q3), 2)*pow(sin(q2), 2) +
cos(q1)*cos(q2)*pow(cos(q3), 2) + pow(sin(q2),2)*pow(sin(q3),2) + cos(q1)*cos(q2)*pow(sin(q3), 2)));

%q6=atan(0/1);
```