UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

CELDA ROBOTIZADA CON VISIÓN ARTIFICIAL PARA EL EMPAQUETADO DE CÁPSULAS DE CAFÉ

Trabajo Fin de Grado

Grado en Ingeniería Electrónica Industrial y Automática

AUTOR/A: Navarro Escrivá, Francisco Javier

Tutor/a: Ivorra Martínez, Eugenio

CURSO ACADÉMICO: 2021/2022

ÍNDICE

1. Memoria

1.1. Objetivo

1.1.1. Objetivo general del proyecto

El principal objetivo de este proyecto es el diseño e implementación de una celda robotizada con visión artificial destinada a la clasificación y el empaquetado de cápsulas de café.

Con dicho propósito se han establecido los siguientes objetivos:

- Diseñar, fabricar y programar un robot scara de 3 grados de libertad el cual manipule las cápsulas mediante una ventosa
- Implementar y programar un sistema de visión artificial el cual realice la clasificación de las cápsulas en tipos diferenciados.
- Diseñar y fabricar dos cabezales giratorios los cuales trasladen las cápsulas y cajas hasta el robot .
- Integrar los sistemas anteriores para que actúen de forma coordinada en el empaquetado de las cápsulas de café en cajas de 16 unidades del mismo tipo de cápsula.

1.1.2. Objetivo específicos

La solución implementada realiza el empaquetado de las cápsulas según su tipo conformando cajas de 16 unidades de un tipo concreto y descartando las cápsulas de otros tipos considerándose erróneas . Los elementos de la celda que llevan a cabo dicho propósito són:

- Robot tipo scara: su objetivo dentro de la celda robotizada es realizar una aplicación que se conoce como “pick & place”, recogiendo las cápsulas de café y empaquetandolas.

- Sistema de visión artificial: tiene la función de realizar el reconocimiento de las cápsulas para determinar el tipo de cápsula extrayendo las características propias de esta para su identificación y la posterior toma de decisiones.

- Dos cabezales giratorios: tienen la función de colocar las cápsulas y cajas en la posición de empaquetado.

- Software de Python: su objetivo consiste en actuar como el controlador de la celda robotizada sincronizando todos los elementos para llevar a cabo satisfactoriamente el objetivo general del proyecto. Además, la inteligencia del sistema de visión artificial está incluida en este software ya que es el que realiza el procesamiento de las imágenes y extracción de características.

1.2. Alcance

- Se ha diseñado e implementado una celda robotizada capaz de desarrollar satisfactoriamente el objetivo descrito anteriormente.

- Se ha diseñado e implementado un robot scara de 3 grados de libertad.

- Se ha diseñado e implementado un sistema de visión artificial.

- Se han diseñado e implementado dos cabezales giratorios.

- Se han integrado los sistemas mediante una aplicación desarrollada en Python.

1.3. Relación con los ODS

Este proyecto contribuye al objetivo 9, industria, innovación e infraestructura de los objetivos de desarrollo sostenible. La celda robotizada desarrollada pretende ser una solución económica a la demanda de este tipo de unidades productivas de la industria moderna. También integra un grado de innovación tecnológica con elementos como el sistema de visión artificial.

Figura 1. Logo del ODS 9, industria, innovación e infraestructura

La innovación tecnológica es un elemento fundamental en el desarrollo de la sociedad moderna. Sin embargo esta ha ido en decremento los últimos años por lo que es nuestro deber el contribuir a desarrollar nuevas tecnologías que ayuden a la sociedad. En el caso concreto de este proyecto la celda robotizada supone un conjunto de elementos innovadores los cuales han sido desarrollados tomando como referente las últimas tecnologías.

1.4. Antecedentes de la robótica industrial

Según la RAE la palabra robot se define como "Máquina o ingenio electrónico programable capaz de manipular objetos y realizar diversas operaciones.

Ninguna tecnología ha revolucionado tanto la industria en los últimos ochenta años como la robótica industrial. Pese a que en la actualidad estamos acostumbrados a ver los robots en la mayoría de cadenas de producción,la robótica industrial moderna ha sufrido una evolución muy rápida en los siglos XX y XXI.

Esta era ha sido denominada como la era de la Industria 4.0 también llamada cuarta revolución industrial. Esto implica que han existido 3 revoluciones anteriores cada una caracterizada por la introducción de un avance tecnológico ya bien sea una nueva técnica o material. La característica de esta 4a revolución industrial es el uso de robots industriales en celdas de fabricación. Estas celdas son unidades productivas formadas por un grupo o un solo robot con una función específica dentro del proceso productivo.

Actualmente es cada vez más común en la industria la implementación de sistemas de visión artificial para guiar a los robots en sus tareas permitiéndoles realizar funciones como empaquetado y clasificación de objetos.

Debido a la importancia de las celdas robotizadas en la industria moderna se ha decidido implementar un sistema de estas características. Además el tópico de la visión artificial desarrollará un papel clave en el desarrollo de nuevas tecnologías en los próximos años por lo cual se ha decidido incluirlo en el proyecto.

1.5. Estudio de necesidades: condicionantes y limitaciones

La celda robotizada debe realizar el encajado de cápsulas de café según el tipo de cápsula en cajas de 16 unidades. Para ello el sistema debe ser capaz de realizar la clasificación de las cápsulas de café según el tipo de dicha cápsula. A su vez debe ser capaz de descartar cualquier cápsula que no sea del mismo tipo de las que se está envasando. Con dicho objetivo en mente se deben de tener en consideración los condicionantes y limitaciones de cada elemento de nuestra celda.

1.5.1. Condicionantes del robot

El robot dispone de un área de trabajo limitada por lo cual las cápsulas deberán circular por esa área para que este pueda recogerlas. De la misma manera las cajas también deberán situarse dentro de esta área para que el robot sea capaz de alcanzarlas. El hecho de que el robot sólo pueda operar dentro de esta área de trabajo implica una limitación en el espacio físico para ubicar los componentes de la celda.

Figura 2. Vista superior del robot y de su área de trabajo

A su vez, la velocidad de funcionamiento del robot es otra limitación la cual se debe de tomar en consideración. Esta velocidad influye en el tiempo de ciclo del sistema para el empaquetado de las cápsulas. Esto es debido a que el robot se debe de sincronizar con los demás elementos de la celda para trabajar de forma conjunta con el objetivo de reducir al máximo el tiempo de ciclo.

Otra limitación del robot es el peso máximo que este puede cargar. Esto es debido a la construcción del robot la cual se ha desarrollado pensando en cargar elementos de poco peso como las cápsulas de café.

La resolución de los motores de pasos de las articulaciones del robot son una limitación en cuanto a la exactitud del robot para lograr alcanzar un punto concreto. Las configuraciones articulares calculadas mediante la cinemática inversa del robot no se pueden lograr con la precisión de cálculo si no que hay que tener en cuenta el error de ángulo provocado por la resolución de los motores.

1.5.2. Condicionantes del sistema de visión

El principal condicionante para que nuestro sistema de visión funcione con la mayor exactitud posible es la iluminación aplicada a las cápsulas. Esto es debido a que el menor cambio en la iluminación puede alterar los colores y formas percibidos por la cámara, parámetros empleados por nuestro sistema de visión para distinguir las cápsulas. Una iluminación excesiva ocasiona reflejos en la superficie de las cápsulas distorsionando la percepción de las formas, mientras que una iluminación deficiente no genera suficiente contraste entre colores.

Una limitación de nuestro sistema de visión son los fotogramas por segundo que puede captar nuestra cámara, 30fps y que tiene repercusión en el retardo de la imagen captada respecto a la situación real. La consecuencia de esto es que la cápsula en movimiento en el mundo real no ocupa la misma posición que en la imagen. Esto se compensa añadiendo un factor de compensación el cual será el retardo de la cámara.

1.5.3. Condicionantes de los cabezales giratorios

Los cabezales giratorios diseñados permiten el procesamiento de las cápsulas de manera individual lo cual limita el tiempo de ciclo mínimo que podemos lograr comparado con el procesamiento simultáneo de varias cápsulas. El tiempo de ciclo también se ve influenciado por la limitación de la velocidad de giro del cabezal giratorio, el cual ha sido establecido tomando en cuenta la limitación de la velocidad de funcionamiento del robot.

1.5.4. Condicionantes del software de Python e integración

Con el objetivo de que el software fuese simple de utilizar para que cualquier operario pudiera emplearlo sin ninguna dificultad se ha desarrollado una interfaz gráfica de usuario. Esto implica la necesidad de un dispositivo con pantalla en el que ejecutar el código. Además este dispositivo debe tener Python instalado.

En cuanto a la integración de los sistemas, la interconexión de estos se ha realizado mediante el medio físico de cables lo cual implica que la ubicación de los elementos de nuestros sistemas debe ser cercana. Este factor no significa una limitación importante ya que para el funcionamiento óptimo de nuestra celda, la ubicación de los elementos debe ser la diseñada y no se modificará.

1.6. Soluciones alternativas

1.6.1. Alternativas de empaquetado

En esta sección se discutirán posibles alternativas en lo que concierne al empaquetado de cápsulas de café, las cuales podrían emplearse en sustitución de nuestro conjunto de robot scara con dos cabezales giratorios. La técnica empleada por nuestro robot se conoce como “pick & place” sin embargo cabe destacar que hay distintas técnicas de empaquetado entre las cuales están la carga superior o top loading y el empaquetado por peso.

En primer lugar existe la técnica de empaquetado mediante un brazo robótico realizando operaciones de "pick & place". En operaciones de empaquetado se necesita una velocidad de trabajo alto por lo cual se emplean robots scara los cuales destacan en este campo. En esta técnica no es necesario hacer control de calidad previo de los productos a empaquetar si no que se puede realizar en la propia cadena de empaquetado. En el caso específico de este proyecto se podría emplear cualquier brazo robótico tipo scara de cualquier fabricante como Kuka, Yaskawa, Omron o Fanuc entre otros. La principal desventaja respecto al robot actual es el precio el cual puede llegar a ser hasta 20 veces superior al actual. Sin embargo la ventaja de estos robots es su velocidad de funcionamiento la cual es muy superior al robot scara actual de la celda. Si comparamos el robot scara SR-3iA de Fanuc (se muestra en la figura 3) con nuestro robot, vemos que la velocidad de giro de sus articulaciones es 20 veces superior (velocidad de giro de 720º/segundo en el robot de Fanuc mientras que en nuestro robot es de 35º/segundo).

Figura 3. Robot scara Fanuc realizando pick and place de pilas(Fuente:(3) FANUC SCARA Robot)

En segundo lugar se podría emplear la técnica de top loading. Esta técnica se caracteriza por la carga superior de los productos agrupados en lotes dentro de las cajas. Esta operación es realizada por un robot cartesiano equipado con una herramienta específica según el producto que se empaqueta(en el caso específico de este proyecto se emplearía una ventosa para coger las cápsulas). Este método es más lento que el pick and place e implica un control de calidad previo de los productos antes de entrar en la cadena de empaquetado. Se podría emplear cualquier robot cartesiano con el alcance suficiente para esta aplicación específica.

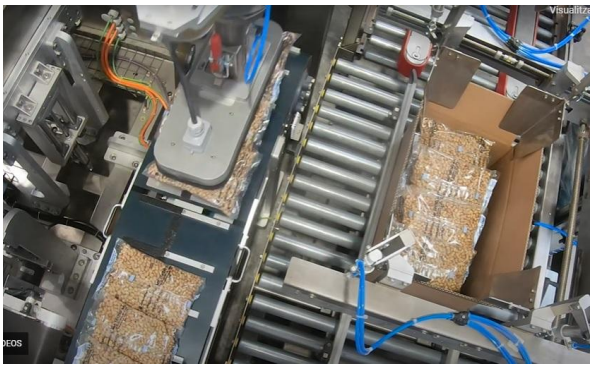


Figura 4. Máquina de empaquetado por top loading de bolsas de cacahuetes.(Fuente:Automatic case packer

En tercer lugar se podría implementar un sistema de empaquetado por peso. Este está formado por tolvas en las que las cápsulas van cayendo hasta lograr un peso concreto. Una vez se logra dicho peso se activa una válvula que abre las compuertas de la tolva y tal cápsulas caen en la caja. Esta técnica no coloca los objetos de manera ordenada dentro de la caja si no que simplemente los deja caer. Al igual que en la técnica anterior, las cápsulas deben pasar un control de calidad previo antes de entrar en la cadena de empaquetado.

Figura 5. Máquina de empaquetado por peso Bosch empaquetando café.(Fuente:soluciones para envasado de café (imco.es))

Extrayendo conclusiones de la discusión de alternativas anterior, el método seleccionado, pick and place, es el idóneo debido a la velocidad de empaquetado que se puede lograr y al hecho de que permite clasificar en la propia cadena de empaquetado los productos en defectuosos y no defectuosos, funcionalidad que no se puede lograr con ninguna de las otras dos técnicas planteadas.

1.6.2. Alternativas al sistema de visión implementado

En esta sección se discutirán las posibles alternativas al sistema de visión artificial implementado para realizar la clasificación de las cápsulas. El sistema de visión actual emplea el color y forma como características para procesar la imagen y extraer el número de cápsulas de cada color presente(cada color corresponde a un tipo de cápsula como se verá posteriormente en el apartado 1.7.5 Programación del sistema de visión) .

La primera alternativa sería emplear un sensor de color para realizar el reconocimiento de las cápsulas. Este es un sensor fotoeléctrico que emite luz desde un transmisor y luego con un receptor detecta la luz reflejada por el objeto. Cada color corresponde con una intensidad de luz recibida. Esto tiene el inconveniente de que cualquier cambio en la iluminación afecta a los valores de la intensidad recibida requiriendo un reajuste del sensor para que vuelva a funcionar correctamente la detección de colores.

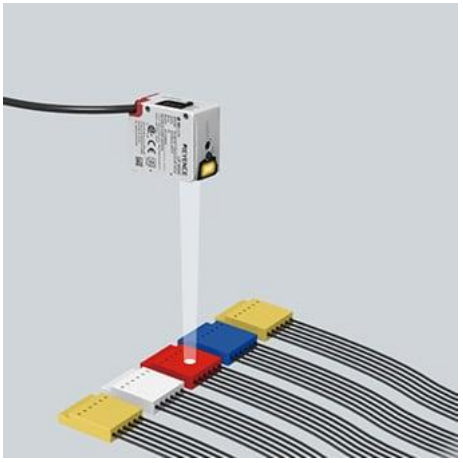


Figura 6. Sensor de color detectando los colores mediante un haz de luz.(Fuente: Sensores de color| KEYENCE)

La segunda alternativa sería emplear una cámara de visión artificial con inteligencia integrada. Esto significa que la propia cámara es capaz de realizar el reconocimiento de objetos en imágenes sin la necesidad de un dispositivo externo. Esta funcionalidad permite ahorrar espacio ya que todos los elementos que necesitamos para realizar la visión artificial están integrados en un solo dispositivo. Sin embargo estas cámaras pueden llegar a costar miles de euros, precio muy superior al de nuestro sistema implementado.

Figura 7. Cámara de visión artificial con inteligencia integrada realizando el reconocimiento de la colocación de las tuercas en una pieza.(Fuente:Visión Artificial Industrial (edsrobotics.com))

Extrayendo conclusiones de las alternativas al sistema de visión artificial implementado, el sensor de color sería una alternativa ya que nos permite distinguir las cápsulas empleando su característica de color. Sin embargo se debería emplear una iluminación muy estática para evitar reajustar el sensor. Además si se quisiera implementar un control de calidad más complejo como la detección de desperfectos en las cápsulas, este sensor no podría emplearse.
Emplear una cámara de visión artificial con inteligencia integrada sería una buena alternativa por su capacidad de procesamiento y el poco espacio físico que ocupan, sin embargo como se ha resaltado, el precio de estas es muy superior al del sistema actual (el precio del hardware del sistema de visión implementado es de 735 euros, mientras que una cámara de potencia de procesamiento similar puede costar 3000 euros). El sistema de visión artificial implementado nos permite una potencia de procesamiento de imágenes aceptable manteniendo un precio económico para una aplicación de este estilo.

1.6.3. Alternativas al robot scara

Se ha señalado anteriormente que el robot idóneo para aplicaciones pick and place de empaquetado de objetos es el robot scara por su gran velocidad y flexibilidad. Sin embargo si fuese necesario se podría emplear cualquier otro tipo de robot como un robot cartesiano, un brazo robótico de 6 grados de libertad o un robot delta.

Un robot cartesiano dispone de tres ejes prismáticos que le permiten moverse en un espacio tridimensional. Estos robots se emplean en aplicaciones CNC y en impresión 3D entre otras. La principal ventaja de estos robots es el hecho de que pueden cargar con objetos muy pesados. En la aplicación concreta de este proyecto se cargan cápsulas de café por lo que este aspecto es irrelevante. Otra característica de estos robots es la velocidad a la que se desplazan, la cual es lenta en comparación con un robot scara.

Figura 8. Robot cartesiano paletizado cajas.
(Fuente:Robot cartesiano(directindustry.es))

Los brazos robóticos de 6 grados de libertad son los más empleados en la industria debido a la gran variedad de tareas que pueden realizar como soldado, paletizado o manipulación de objetos. La principal ventaja de estos es el hecho de que debido a su construcción pueden alcanzar un punto concreto en el espacio con una orientación específica en su efector final, función que ningún otro robot puede realizar. La velocidad a la que estos robots se desplazan es media comparada con un robot scara.

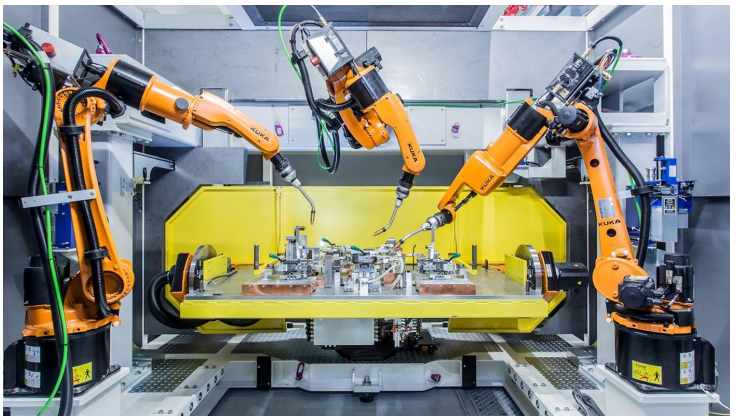


Figura 9. Robots Kuka de 6 grados de libertad realizando el soldado por puntos del chasis de un vehículo.(Fuente:Soldadura por puntos /KUKA)

Los robots delta son robots paralelos de hasta 5 grados de libertad. El término paralelo significa que la herramienta siempre se desplaza paralelamente a la base fija del robot. Debido a la construcción de este tipo de robots las ecuaciones que rigen los movimientos del robot son más complejas de calcular que las de los demás tipos nombrados. La velocidad a la que operan estos robots es semejante a la de un robot scara.

Figura 10. Robot delta manipulando azulejos.
(Fuente: Robot delta (edsrobotics.com))

Extrayendo conclusiones de alternativas al robot scara, la principal alternativa sería emplear un robot delta. Esto es debido al hecho que este tipo de robots opera a una velocidad semejante a la de un robot scara. El robot cartesiano no podría emplearse por su baja velocidad de funcionamiento. El robot de 6 grados de libertad sí que podría ser utilizado aunque esto supondría un tiempo de ciclo de empaquetado de cápsulas mayor comparado con el de un robot scara o delta.

1.6.4. Alternativas a los cabezales giratorios

Los cabezales giratorios implementados nos permiten desplazar las cápsulas por la cadena de empaquetado con rapidez y precisión. Además están optimizados en lo que respecta al espacio que ocupan dentro de la celda robotizada de esta aplicación concreta.

Las cintas transportadoras son la principal alternativa a los cabezales giratorios. Estas cuentan con el gran inconveniente del espacio físico que ocupan comparado con la solución adoptada. Otro inconveniente es el hecho de que para desplazar una cápsula requiere el funcionamiento continuo de un motor, mientras que el cabezal giratorio simplemente tiene que girar unos grados para alimentar la siguiente cápsula. Esto permite lograr un tiempo de ciclo mucho menor e incrementar la producción. Sin embargo se podría adaptar una cinta transportadora convencional para esta aplicación pese a los inconvenientes planteados.

1.6.5. Toma de decisiones

El sistema implementado cuenta con los elementos de un brazo robótico tipo scara para la manipulación de las cápsulas, un sistema de visión artificial compuesto por cámara, dispositivo de procesamiento e iluminación y un par de cabezales giratorios para trasladar los objetos.

Se ha escogido el proceso de empaquetado mediante un brazo robótico scara realizar operaciones pick & place ya que esta técnica permite combinar en la propia cadena de empaqueta el clasificado y control de calidad de las cápsulas con el empaquetado de las mismas. Como se ha analizado, ninguna otra técnica de empaquetado cuenta con dicha ventaja.

Se ha optado por un robot tipo scara por su velocidad de funcionamiento y su facilidad de implementación comprado con la principal alternativa, el robot delta el cual cuenta con una velocidad de funcionamiento similar sin embargo su implementación es más compleja. Los demás tipos de robots discutidos podrían emplearse como alternativa aunque esto significa obtener tiempos de ciclo de empaquetado de cápsulas mucho menores que los logrados con la alternativa escogida.

En lo que concierne al sistema de visión artificial implementado, la principal alternativa es emplear una cámara de visión artificial con inteligencia integrada. Sin embargo, el precio de esta opción es mucho mayor que el del sistema actual. Aunque el sistema actual no tiene todos los elementos (cámara, dispositivo de procesamiento de imágenes e iluminación) integrados en un solo dispositivo, la potencia de procesamiento es similar a la de una cámara con inteligencia integrada por lo que el sobrecoste de esta se puede evitar.

Los cabezales giratorios implementados están optimizados en lo que respecta al espacio que ocupan para la aplicación concreta, aspecto con el que no cuentan su principal alternativa, una cinta transportadora comercial. Además nos permiten lograr una velocidad de funcionamiento mucho mayor.

Se puede concluir que los elementos escogidos son los idóneos para la implementación de la celda robotizada con visión artificial para el empaquetado de cápsulas de café.

1.7. Descripción detallada de la solución adoptada

1.7.1. Esquema general de la solución adoptada

Este apartado pretende dar una primera visión generalizada de la solución implementada la cual se tratará con más detalle en los próximos apartados. La solución implementada para la celda robotizada con visión artificial para el empaquetado de cápsulas de café cuenta con un robot scara el cual manipula las cápsulas mediante una ventosa, el sistema de visión artificial que realiza el reconocimiento de las cápsulas y dos cabezales giratorios los cuales alimentan las cápsulas y cajas al robot.

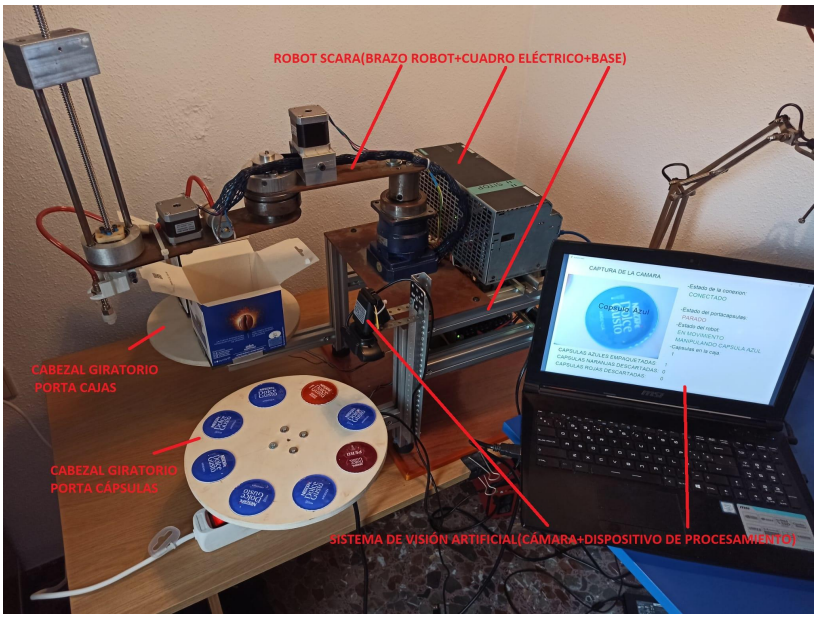


Figura 11. Implementación física de la celda robotizada

Todos los elementos de la celda están fijados a la base del robot scara. Esta es una estructura de perfil de aluminio CNC a la cual se ha fijado el brazo robótico scara con su cuadro eléctrico, el soporte de la cámara y los soportes de los cabezales giratorios. Esto nos permite integrar nuestra celda en un único conjunto el cual mantiene siempre las mismas distancias entre elementos.

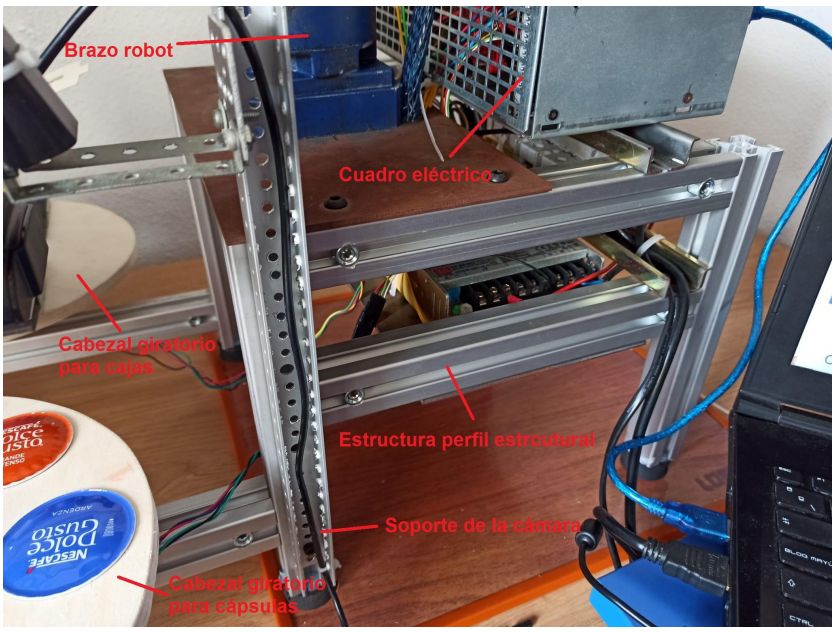


Figura 12. Estructura base del robot scara

El robot scara implementado está sujeto a la base mediante la reductora planetaria de su primer eje. Esta, junto con el motor nema 23, nos permite lograr el par requerido para operar el brazo robot. El segundo eje cuenta con un motor NEMA 17 desmultiplicado mediante poleas dentadas con correa. Se ha empleado un rodamiento axial y radial con el objetivo de que este soporte los pares generados por el peso del tercer eje sobre el segundo. En el tercer eje se ha implementado un conjunto de husillo y tuerca para desplazar el efector final verticalmente empleando un motor NEMA 17. Se manipulan las cápsulas mediante una ventosa la cual va conectada a una bomba de vacío que proporciona un flujo de succión de 2 L de aire por minuto. El controlador del robot es un Arduino Mega 2560 con una Ramps 1.4 a la cual se conectan módulos DRV8825 para controlar motores de pasos. Se ha programado el arduino con la cinemática inversa del robot, un generador de trayectorias polinomial y un protocolo de comunicación. La bomba de vacío es controlada mediante las salidas de 12Vdc de la Ramps 1.4. Debido al carácter educativo de este robot y a su baja velocidad de funcionamiento no se ha incluido ningún dispositivo de seguridad como una seta de emergencia o barreras físicas. Si se adaptara la celda implementada para trabajar en un entorno industrial se deberían incluir estos elementos de seguridad.

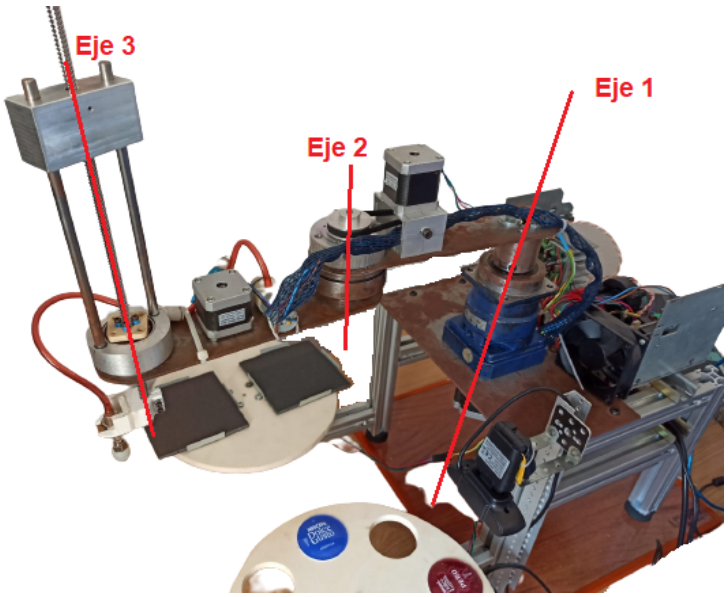


Figura 13 . Brazo robot scara implementado

El robot scara implementado presenta tres articulaciones. La articulación 1 gira en torno al eje de rotación 1 y tiene un rango articular de -35º a 215º. Esta hace rotar el brazo robot a una velocidad de 32º/segundo. La articulación 2 gira en torno al eje de rotación 2 y presenta un rango articular de -55º a 235º. Esta hace rotar el efector final en torno al eje 2 a una velocidad de 37.5º/segundo. La articulación 3 se desplaza verticalmente en el eje Z con un rango articular de 0mm a 156 mm a una velocidad de 25 mm/s.(cálculo de las velocidades del eje 1 y 2 en el apartado 1.7.3.3 Generador de pulsos y cálculo de la velocidad del eje 3 en el apartado 1.7.3.4 Función recoger cápsula)

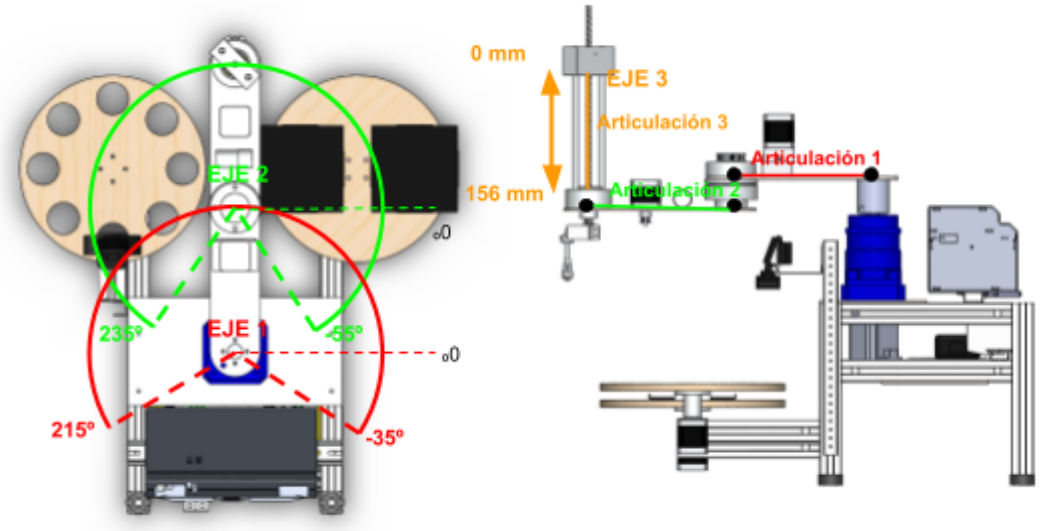


Figura 14 . Rango articular del brazo robot scara implementado

Se han implementado dos cabezales giratorios. Ambos cabezales van fijados a la estructura del robot scara mediante sus soportes. Se han empleado dos motores NEMA 17 controlados por el mismo arduino y ramps 1.4 que el robot scara. El cabezal derecho(izquierda en la imagen) realiza rotaciones de 180º cambiando la caja una vez esta está llena. El cabezal izquierdo (derecha en la imagen) puede alojar hasta 8 cápsulas las cuales va situando delante de la cámara para que las identifique. Este cabezal además sitúa las cápsulas en la posición de recogida donde el robot las coje.

Figura 15. Cabezales giratorios implementados

La cámara empleada en el sistema de visión artificial va fijada a la estructura base del robot scara mediante un soporte lo que le permite mantener una posición fija en el conjunto. Esta realiza la adquisición de las imágenes que serán procesadas posteriormente por el algoritmo de visión artificial programado.

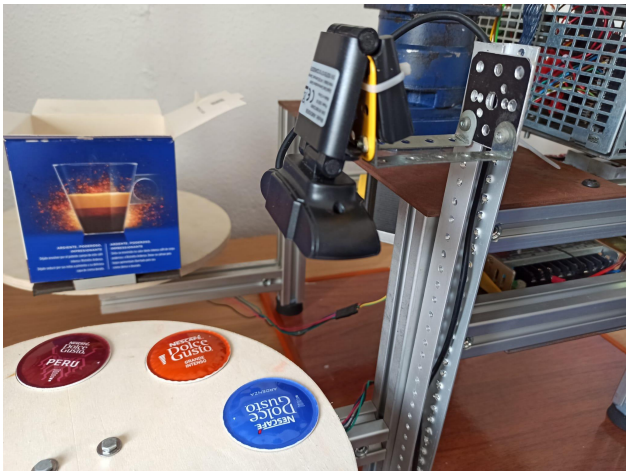


Figura 16. Cámara en su soporte

La sincronización de los elementos anteriores se ha realizado mediante un software programado en Python el cual se ejecuta en un ordenador. Este software además incluye el algoritmo de visión artificial el cual realiza la identificación de las cápsulas. El programa principal recopila los datos resultantes del análisis del sistema de visión, realiza la toma de decisiones en base a esos datos y le comunica al arduino las órdenes que tiene que realizar por el puerto serial. Además se ha programado un monitor estilo HMI para conocer el estado del proceso en todo momento.

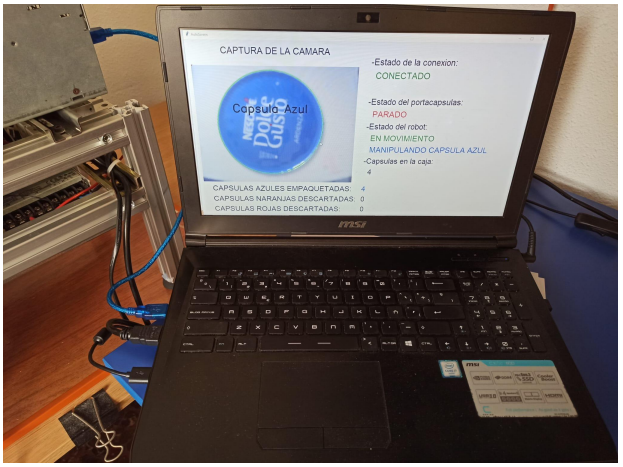


Figura 17. Ordenador ejecutando el software que integra los sistemas

El funcionamiento del programa principal es el siguiente:

- Se establece la comunicación serial entre arduino y ordenador.
- Se inicia la captura y procesado de imágenes la cual se ejecuta continuamente en un hilo del programa actualizando las variables de la cantidad de cápsulas de cada tipo presentes en la imagen.
- En otro hilo se consultan dichas variables y se toman las decisiones según su valor.
    - Si la cápsula es azul el programa manda un mensaje al arduino para que mande al robot scara a empaquetar la cápsula. Este recibe el mensaje, rota el cabezal giratorio hasta colocar la cápsula en la posición de recogida, el robot recoge la cápsula, la coloca en la caja y se queda esperando nuevas órdenes.
    - Si la cápsula es naranja o roja el programa manda un mensaje al arduino para que mande al robot scara a descartar la cápsula. Este recibe el mensaje, rota el cabezal giratorio hasta colocar la cápsula en la posición de recogida, el robot recoge la cápsula, la descarta y se queda esperando nuevas órdenes.
    - Si no hay cápsula el programa manda un mensaje al arduino para que rote el cabezal giratorio de las cápsulas en intervalos de 45º hasta que encuentre una la cámara.

- Una vez se ha llenado una caja con 16 cápsulas el programa manda un mensaje al arduino para que rote el cabezal giratorio de las cajas.

- El número de cápsulas procesadas de cada tipo, las cápsulas actuales dentro de la caja y el estado de robot y cabezales giratorios se muestra en el HMI.

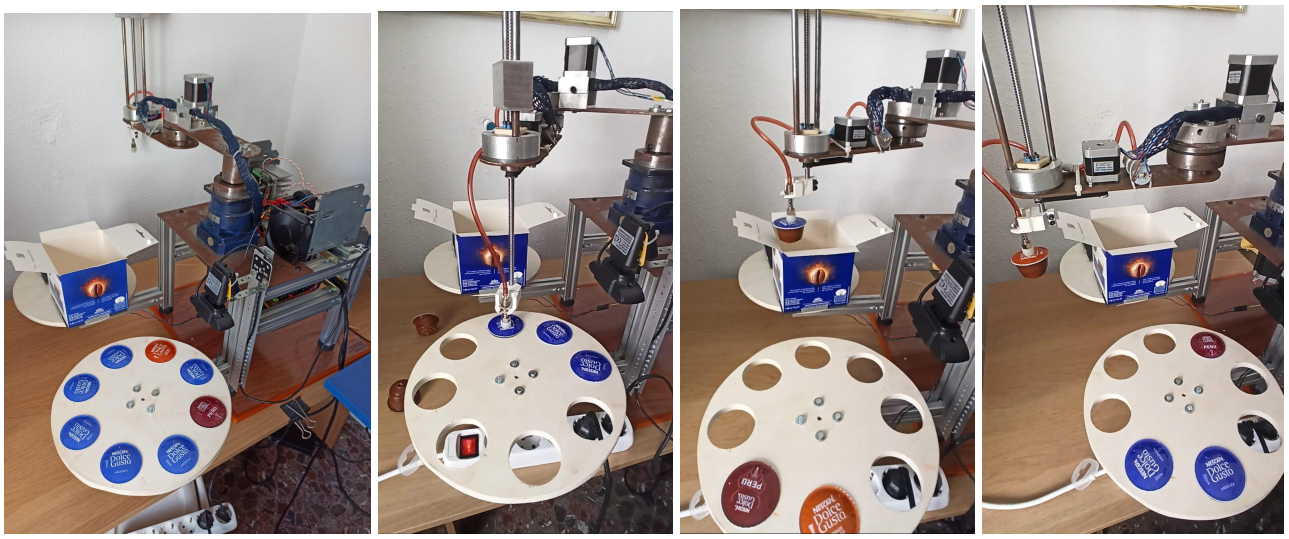


Figura 18. Posiciones a las que se desplaza el robot, posición inicial(izquierda), posición de recogida de cápsula(centro izquierda), posición de encajado(centro derecha) y posición de descarte(derecha)

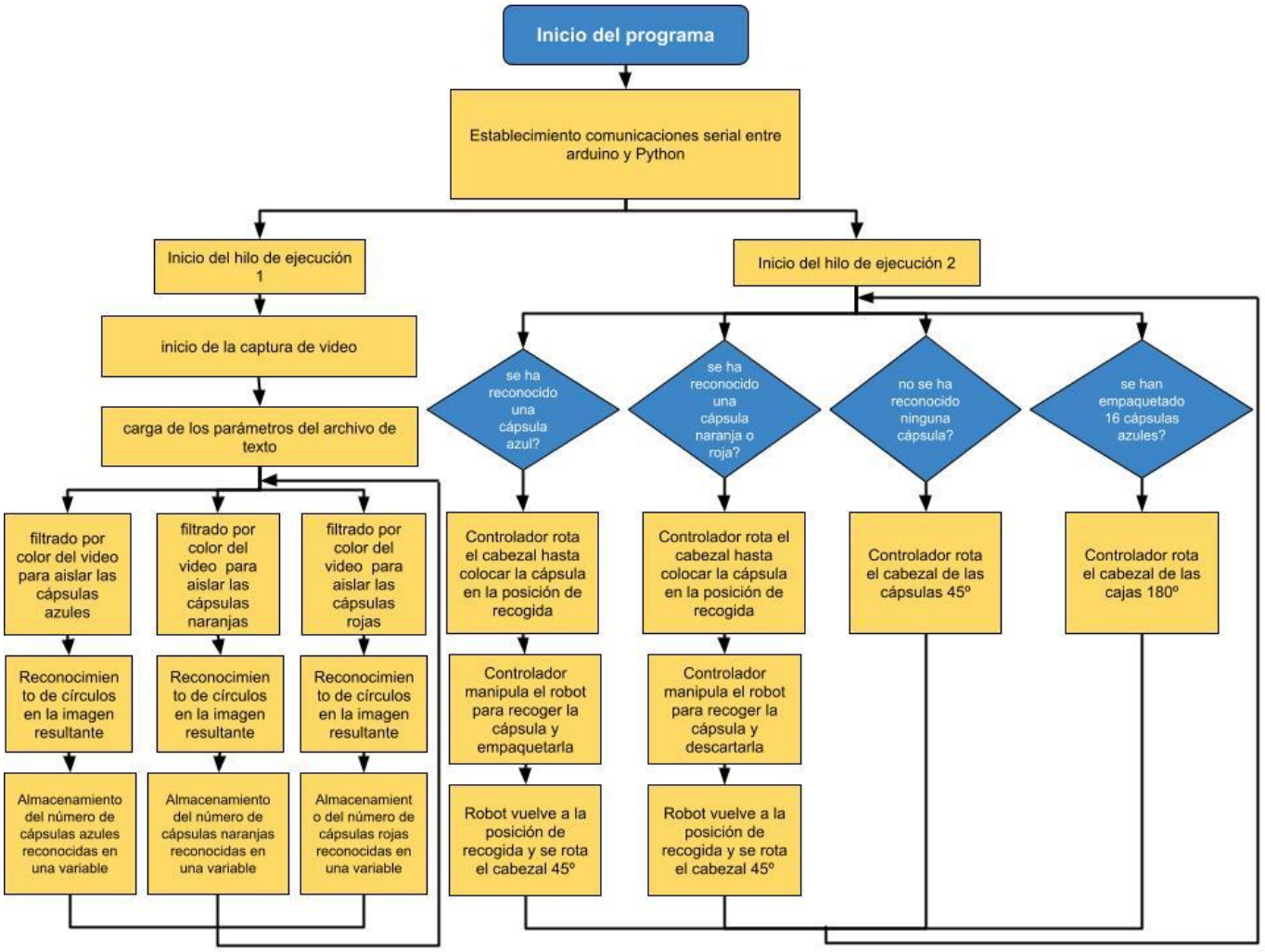


Figura 19. Diagrama de flujo del funcionamiento de la celda robotizada

1.7.2. Hardware del robot scara

Este apartado pretende mostrar las partes de las de que se compone el robot scara dividiéndolas en tres grupos diferenciados: los componentes electrónicos los cuales nos permiten controlar el robot, los elementos mecánicos que lo articulan y los elementos estructurales que actúan de soporte para los elementos anteriores.

1.7.2.1. Elementos electrónicos

El controlador del robot es una placa arduino mega 2560. Esta se basa en el microcontrolador ATmega2560 el cual es el más potente de la familia de arduino. Dispone de 54 entradas y salidas digitales, 15 de las cuales son PWM. En cuanto a las comunicaciones cuenta con 4 puertos serial por hardware(UART). Esta placa es compatible con gran cantidad de placas de extensión.

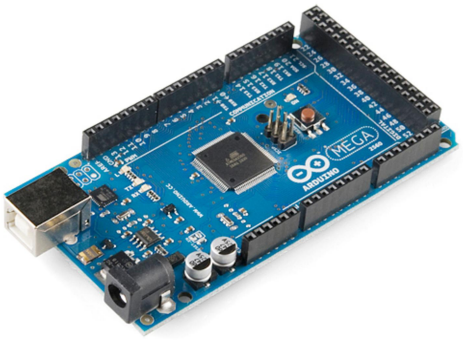


Figura 20. Arduino Mega 2560

Para este proyecto se ha empleado la placa de extensión Ramps 1.4. Esta placa se conecta directamente empleando los pones macho a los pines hembra de la placa de arduino. La Ramps 1.4 cuenta con 6 huecos para drivers de motores de pasos pololu los cuales nos permiten controlar 6 motores de pasos. Además podemos controlar 4 servos y activar o desactivar 3 salidas de 12-35V empleando tres salidas digitales conectadas a tres transistores..

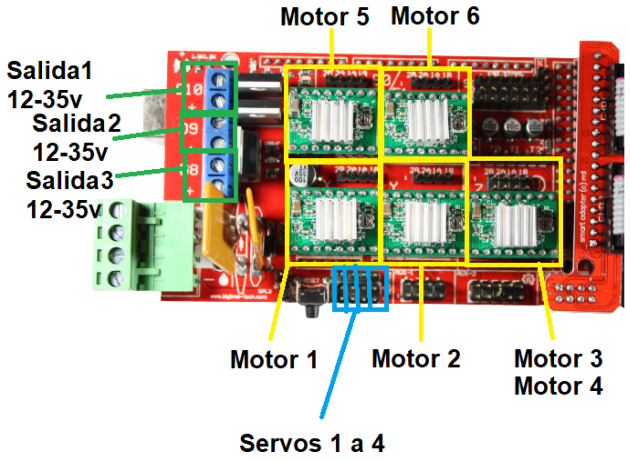


Figura 21. Ramps 1.4 con los módulos pololu

Para controlar los motores de pasos es necesario emplear los drivers Pololu DRV8825. Estos drivers nos permiten controlar un motor de pasos empleando únicamente dos salidas digitales, una para indicar la dirección de rotación y otra para dar los pulsos que corresponden a los pasos del motor acorde a la relación 1.8º/pulso. Sin embargo esta relación se modifica ajustando el valor lógico de tres pines en el driver pololu pudiendo lograr 1 / 2, 1 / 4, 1 / 8 y 1/16 de desmultiplicación digital logrando así una mayor resolución el en ángulo. La corriente nominal máxima que puede proporcionar el DRV8825 a los motores de pasos es de 2.2A.

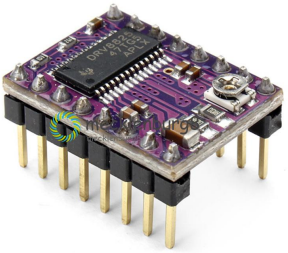


Figura 22. Driver DRV8825

Los drivers Pololu necesitan un ventilador para disipar el calor que generan en funcionamiento. Se ha empleado el ventilador centrífugo EC5015M12S.

Figura 23. Ventilador centrífugo EC5015M12S.

El motor NEMA 17 17HS5412-3 ha sido empleado en el eje 2 del robot conjuntamente con una desmultiplicación 1/3 por poleas dentadas. Este motor tiene una corriente nominal de 1.2A y un par en reposo de 48 Ncm.

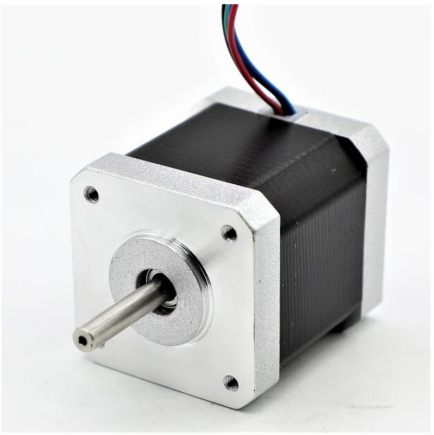


Figura 24. Motor NEMA 17 17HS5412-3

El motor NEMA 17 42BYGHW609 ha sido empleado en el eje 3 del robot conectado mediante poleas dentadas iguales a la varilla roscada del efector final. Este motor tiene una corriente nominal de 1.7A y un par en reposo de 40 Ncm.

Figura 25. Motor NEMA 17 42BYGHW609

El driver bifase para motores de pasos SANMOTION F2 BS1D200P10 nos permite controlar un motor de pasos de hasta 3A mediante dos entradas digitales. Este se ha empleado para controlar el motor del eje 1 del robot el cual requiere una corriente nominal superior a los motores de los ejes 1 y 2.

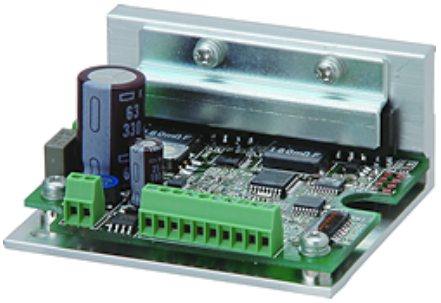


Figura 26. Driver de motor de pasos SANMOTION F2 BS1D200P10

El motor empleado en el eje 1 es el motor stepper NEMA 23 103H7823-5740 de 3A corriente nominal y con un par en reposo de 2.7Nm . Este motor, junto con un reductor planetario nos permite lograr la fuerza necesaria en esta articulación la cual ha de soportar el peso de todo el brazo y los esfuerzos generados por el movimiento del mismo.

Figura 27. Motor NEMA 23 103H7823-5740

La unidad de succión del efector final cuenta con la micro bomba de aire para crear vacío TMJ-370CA-15330 alimentada a 12Vdc.

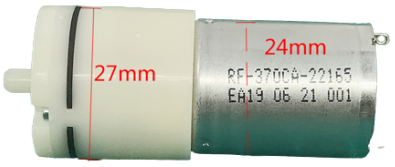


Figura 28. Bomba de vacio TMJ-370CA-15330

Se han empleado dos disyuntores Siemens Circuit Breaker 5SX2 MAX 277 AC C15 para cortar la alimentación de las fuentes de 24V y 12V en caso de cortocircuito.

Figura 29. Disyuntores Siemens Circuit Breaker 5SX2 MAX 277 AC C15

La fuente de alimentación MeanWell RSP-320-24 de 24V y 13.4A se ha utilizado para alimentar el driver SANMOTION.

Figura 30. Fuente de alimentación de 24V MeanWell RSP-320-24

El resto de componentes están alimentados por en transformador de Channel Well Technology PAA060F de 12V y 5.0A

Figura 31. Transformador CWT PAA060F

La conexiones de los elementos descritos anteriormente son las siguientes:

- Salida de 24V de la fuente de alimentación de 24Vdc MeanWell RSP-320-24 conectada a la entrada del disyuntor Siemens 5SX2 MAX 277 AC C15 (disyuntor 1). (cable rojo)

- Salida del disyuntor (disyuntor 1) conectado a la entrada de voltaje del driver de motor de pasos SANMOTION F2 BS1D200P10. (cable rojo)

- Entrada de GND de la fuente de alimentación de 24V conectada a la salida de GND del driver de motor SANMOTION. (cable negro)

- Motor de pasos NEMA 23 103H7823-5740 conectado al conector de motor del driver SANMOTION. (cables amarillo ,azul, rojo y naranja)

- Entrada digital de pulsos del driver SANMOTION conectada a la salida digital de pulsos correspondiente al motor E de la Ramps 1.4. (cable amarillo)

- Entrada digital de dirección del driver SANMOTION conectada a la salida digital de posición correspondiente al motor E0 de la Ramps 1.4. (cable naranja)

- Salida de GND de pulsos del driver SANMOTION conectada a un terminal de carril DIN conectado a GND de la fuente de 12V. (cable negro)

- Salida de GND de dirección del driver SANMOTION conectada a un terminal de carril DIN conectado a GND de la fuente de 12V. (cable negro)

- Terminal positivo de la bomba de vacío TMJ-370CA-15330 conectado al terminal positivo de la salida controlada de 12V de la Ramps 1.4. (cabe marrón)

- Terminal negativo de la bomba de vacío TMJ-370CA-15330 conectado al terminal negativo de la salida controlada de 12V de la Ramps 1.4. (cable azul)

- Salida de 12V del transformador CWT PAA060F conectada a la entrada del disyuntor Siemens 5SX2 MAX 277 AC C15 (disyuntor 2). (cable rojo)

- Salida del disyuntor Siemens (disyuntor 2) al embarrado de terminales DIN rojos. (cabe rojo)

- Terminal GND del transformador de 12Vdc al embarrado de terminales de carril DIN negros. (cabe negro)

- Terminal positivo del ventilador centrífugo EC5015M12S al embarrado de terminales DIN rojos. (cabe rojo)

- Terminal negativo del ventilador centrífugo EC5015M12S al embarrado de terminales DIN negros. (cabe negro)

- Entrada de 12Vdc de la Ramps 1.4 conectada al embarrado de terminales de carril DIN rojos. (cable rojo)

- Terminal de GND de la Ramps 1.4 conectada al embarrado de terminales de carril DIN negros. (cable negro)

- Pines macho de la placa Arduino Mega 2560 conectados a los pines hembra de la Ramps 1.4. (conexión directa)

- Pines macho de 4 módulos pololu DRV8825 conectados a los pines hembra de las posiciones de los motores X, Y, Z y E1. (conexión directa)

- Conector hembra del motor NEMA 17 17HS5412-3 al conector macho de motor del motor de la posición X.

- Conector hembra del motor NEMA 17 42BYGHW609 al conector macho de motor del motor de la posición Y.

Las conexiones descritas anteriormente pueden observarse en la siguiente figura.

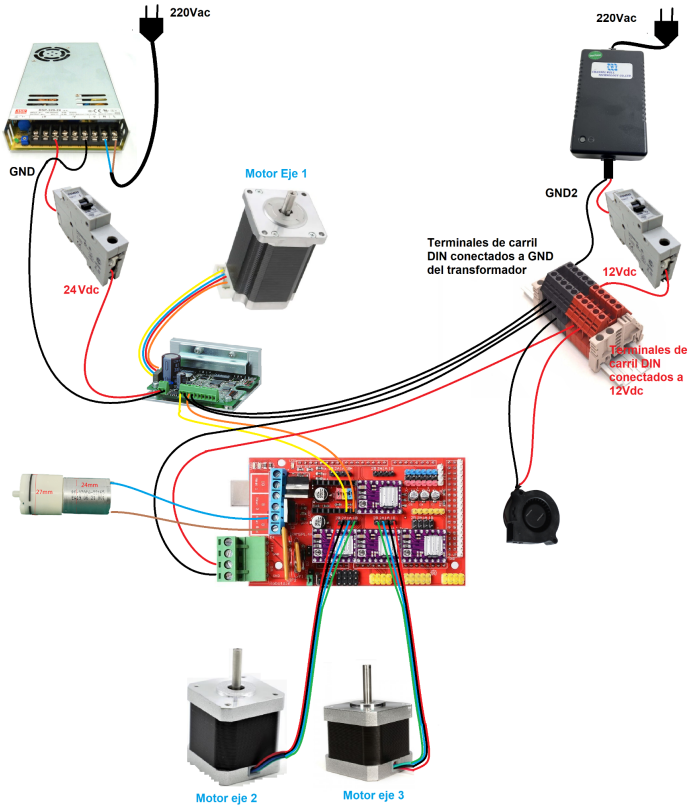


Figura 32. Vista general de las conexiones electrónicas realizadas en robot scara

Los elementos electrónicos del robot se han ubicado en su base empleando contenedores con carril DIN. Se han separado en dos zonas del robot los componentes de potencia y los de señal para evitar cualquier interferencia que las fuentes de alimentación puedan generar sobre los cables de señal. En la parte baja del chasis del robot se han ubicado las fuentes de alimentación con sus disyuntores.

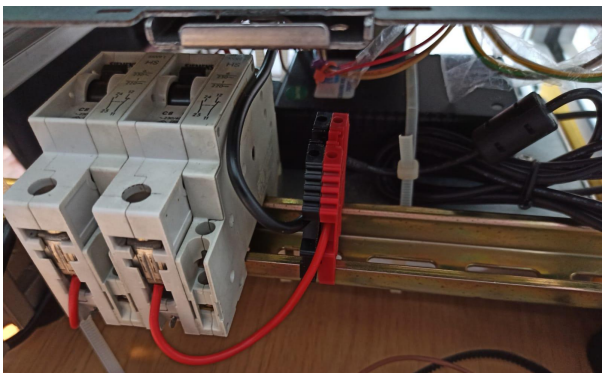


Figura 33. Elementos electrónicos de potencia del robot scara

Los elementos de señal se han ubicado dentro de un cuadro eléctrico con un embarrado de terminales DIN de donde toman el voltaje. Este cuadro está fijado al carril DIN superior del robot.

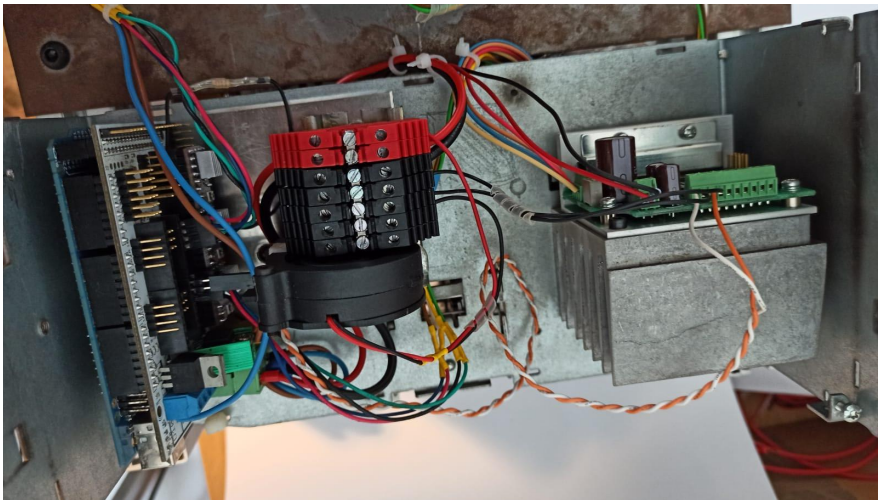


Figura 34. Elementos electrónicos de señal del robot scara

1.7.2.2. Elementos mecánicos

Los motores de pasos empleados, por sí solos no son capaces de desarrollar el par requerido para operar las articulaciones del robot scara diseñado, por lo que elementos desmultiplicadores se han empleado con el objetivo de lograr un par mayor con los mismos motores.

En el eje 1 del robot se ha utilizado la reductora planetaria Alpha SP 075-MF1-7-131-000 la cual tiene un ratio de desmultiplicación 7:1. El motor de este eje, el NEMA 23 103H7823-5740 desarrolla un par en reposo de 2.7Nm por sí solo. Grácias a la reductora planetaria logramos un par de 7 veces este valor, 18.9Nm. Este par es más que suficiente para mover el eje 1 del robot. El motor está atornillado a la cara inferior de la placa de la base del robot mientras que la reductora planetaria está atornillada en la cara superior. El eje del motor va fijado al eje hueco inferior de la reductora mediante un tornillo prisionero de forma que al rotar el motor este también rota. La reductora ejerce la desmultiplicación y se transmite el movimiento al eje superior donde se ha fijado la placa que une los ejes 1 y 2 mediante 4 tornillos.

Figura 35. Conjunto de Reductor planetario 075-MF1-7-131-000 y motor de pasos NEMA 23 103H7823-5740

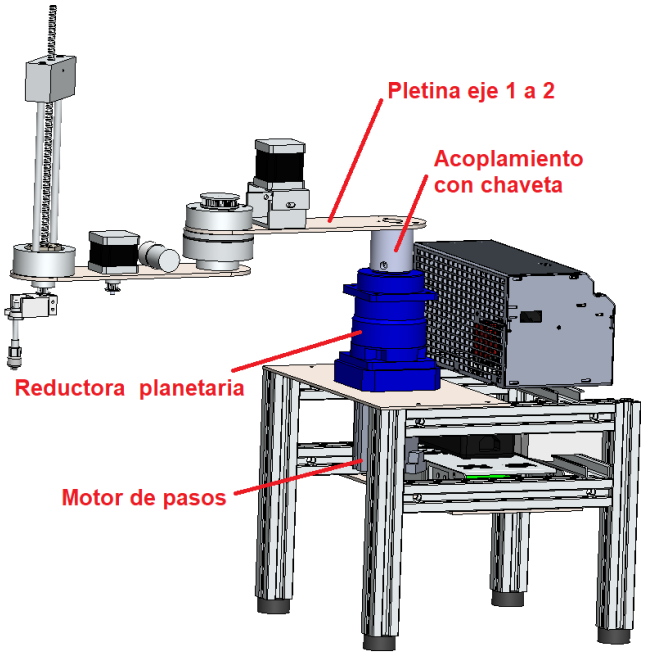


Figura 36. Montaje en solidworks del conjunto mecánicos del eje 1

En el eje 2 del robot se han empleado dos poleas dentadas con correa las cuales tienen una desmultiplicación de 3:1. El motor empleado, el NEMA 17 42BYGHW609 tiene un par en reposo de 48 Ncm, el cual no es suficiente para mover la articulación 2. Esto se soluciona con la desmultiplicación usando una polea de 20 dientes con correa en la entrada y una de 60 en la salida.

$$Relación\ de\ desmultiplicación = dientes\ polea\ salida/dientes\ polea\ entrada$$
$$Relación\ de\ desmultiplicación = 3:1$$

Se ha utilizado un rodamiento axial y radial para soportar el par generado sobre el eje 2 por el peso del brazo del robot. Este rodamiento requiere la compresión superior e inferior de las pistas centrales para el correcto funcionamiento de este por lo que se han empleado dos piezas de aluminio torneado, el cierre superior e inferior, unidas por un tornillo central. La parte exterior del rodamiento está sujeto a la placa que une el eje 1 y 2 mediante tornillos allen M6. Un eje de 5mm de acero se ha fijado a la polea de 60 dientes mediante un tornillo prisionero y al cierre superior con otro tornillo prisionero para que ambos giren al accionar el motor.

Figura 37. Conjunto mecánico del eje 2

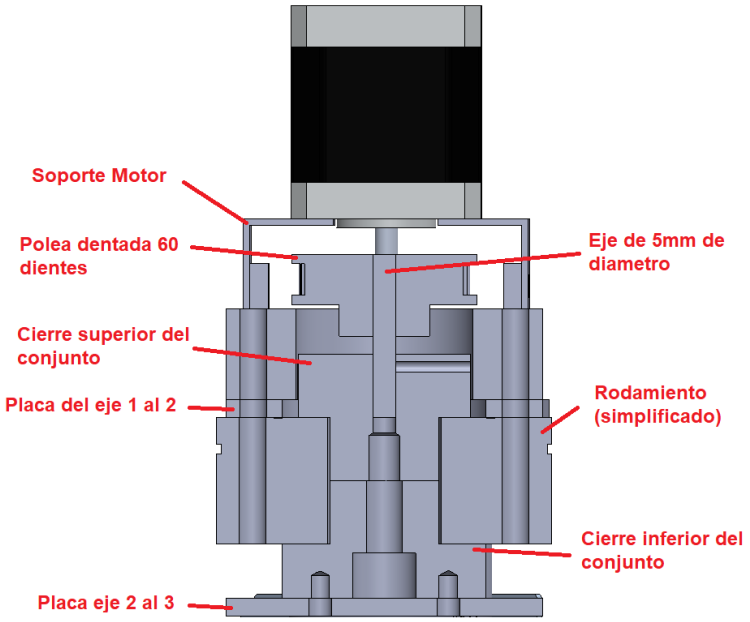


Figura 38. Vista de sección del eje 2 realizada en solidworks

El eje 3 del robot está formado por un conjunto mecánico de Husillo guiado y tuerca. El motor de pasos NEMA 17 42BYGHW609 hace girar una polea de 20 dientes fijada a su eje lo que hace girar otra polea dentada idéntica mediante una correa. Esta segunda polea está fijada a la parte inferior de un casquillo el cual está a su vez atornillado a la tuerca igus DST-LS-8X10-R-ES. Al rotar el motor, este conjunto de polea, casquillo y tuerca rotan, desplazando el husillo igus DST-JFRM-C-01-DS8X10 verticalmente. Para que esto ocurra el husillo debe estar fijado ya que si estuviera libre rotaría junto con la tuerca, por lo que el bloque guía está unido al husillo mediante un tornillo prisionero. Este bloque guía mantiene el husillo alineado verticalmente grácias a las dos guías verticales. El conjunto de husillo y rosca tiene un paso de 10mm lo que significa que si queremos desplazar el husillo 10 mm deberemos realizar una vuelta completa de motor.

$$Vueltas\ de\ motor\ = \frac{Desplazamiento\ deseado\ del\ Husillo(mm)}{10mm}$$

El portaherramientas está fijado por su parte superior al husillo mediante una abrazadera integrada en la misma pieza. En su parte inferior lleva la herramienta, en este caso concreto una ventosa.

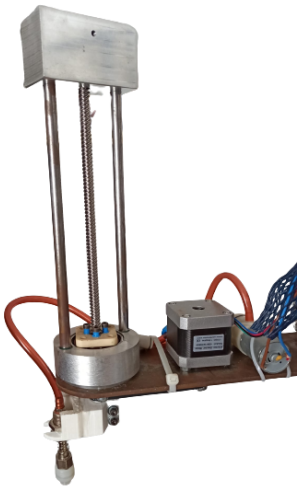


Figura 39 .Conjunto mecánico Eje 3

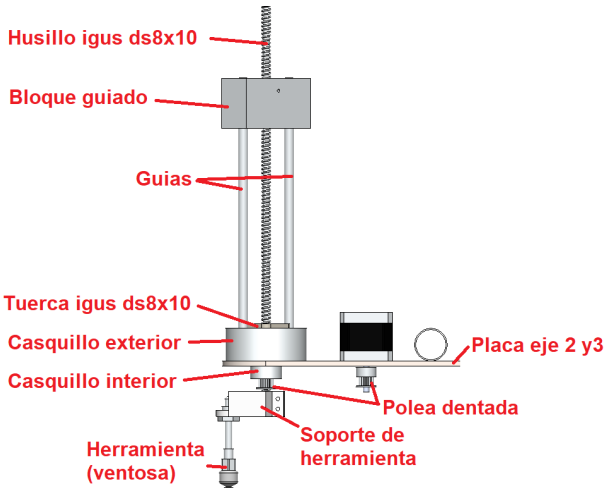


Figura 40. Ensamblaje del eje 3 realizada en solidworks

1.7.2.3. Elementos estructurales

Los elementos estructurales sirven de soporte y mantienen unidos los elementos electrónicos y mecánicos del robot. Los elementos estructurales del robot son la estructura de la base del robot, el contenedor del cuadro eléctrico, la pletina que une el eje 1 y 2 y la pletina que une los ejes 2 y 3.

La estructura de la base del robot está formada por dos estructuras rectangulares de perfil estructural de 30x30mm, dos placas de acero, dos carriles DIN y 4 patas de goma. La placa superior actúa como soporte del motor del eje 1 y la reductora planetaria. Esta placa además une las dos estructuras rectangulares fabricadas con perfil estructural de 30x30mm a las que está fijada mediante tornillos y tuercas con ranura en T. La placa inferior aloja la fuente de alimentación de 24V y el transformador de 12V los cuales están fijados a esta mediante bridas. Esta placa, al igual que la superior, mantiene unidas las dos estructuras rectangulares a las que está fijada. Las dos estructuras rectangulares están fabricadas con perfil estructural de 30x30mm. Los largueros horizontales y verticales de esta estructura están unidos mediante conectores perpendiculares. Los demás elementos estructurales están fijados a estas mediante conjunto de tornillo y tuerca con ranura en T. Se han utilizado dos carriles DIN, un carril superior al que se ha fijado el contenedor del cuadro eléctrico y un carril inferior donde se ubican los disyuntores de la fuente de 24V y 12V. Las 4 patas de goma mantienen la estructura elevada y otorgan estabilidad al conjunto.

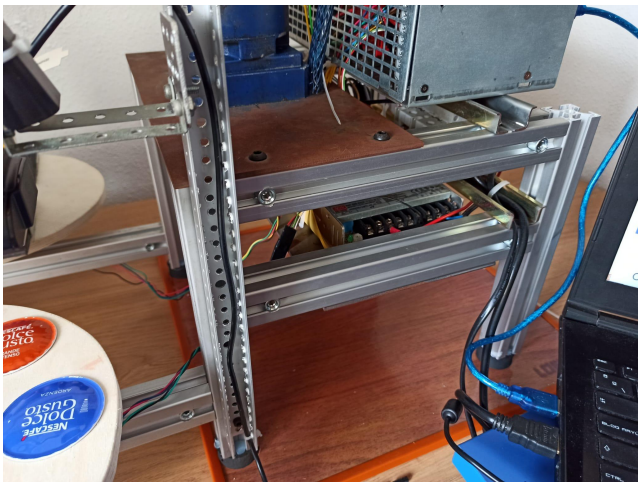


Figura 41. Conjunto de la base del robot

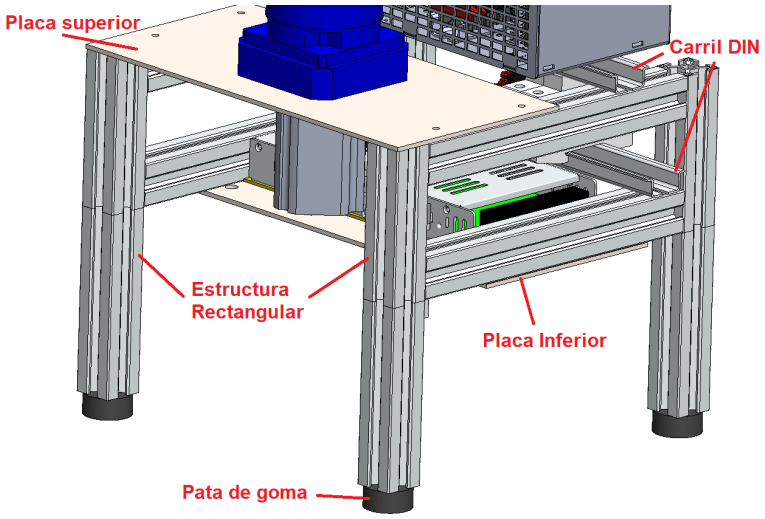


Figura 42. Ensamblaje de la estructura base del robot en solidworks

1.7.3. Software del robot scara

La programación del robot scara se ha realizado mediante el entorno Arduino IDE. Se ha implementado la cinemática inversa del robot la cual nos permite obtener las configuraciones articulares del robot correspondientes a un punto en el espacio tridimensional con respecto al origen de coordenadas del robot. Con tal de desplazar las articulaciones del robot a una configuración articular determinada mediante un movimiento coordinado se ha desarrollado un generador de trayectorias polinomial. Este emplea la función de generación de pulsos para mover los motores. El robot está en todo momento conectado con el sistema de visión mediante comunicación serial por lo que se ha programado un controlador para dichas comunicaciones. Se han programado dos funciones para gestionar el efector final, una para recoger una cápsula y otra para depositarla.

1.7.3.1. Cinemática inversa

Debido a la propia construcción del robot este tiene una serie de parámetros los cuales serán empleados en el cómputo de la cinemática inversa. Para obtener dichos parámetros planteamos la matriz denavit-hartenberg del robot obtenida a partir de las transformaciones entre los sistemas de referencia de cada articulación del robot.

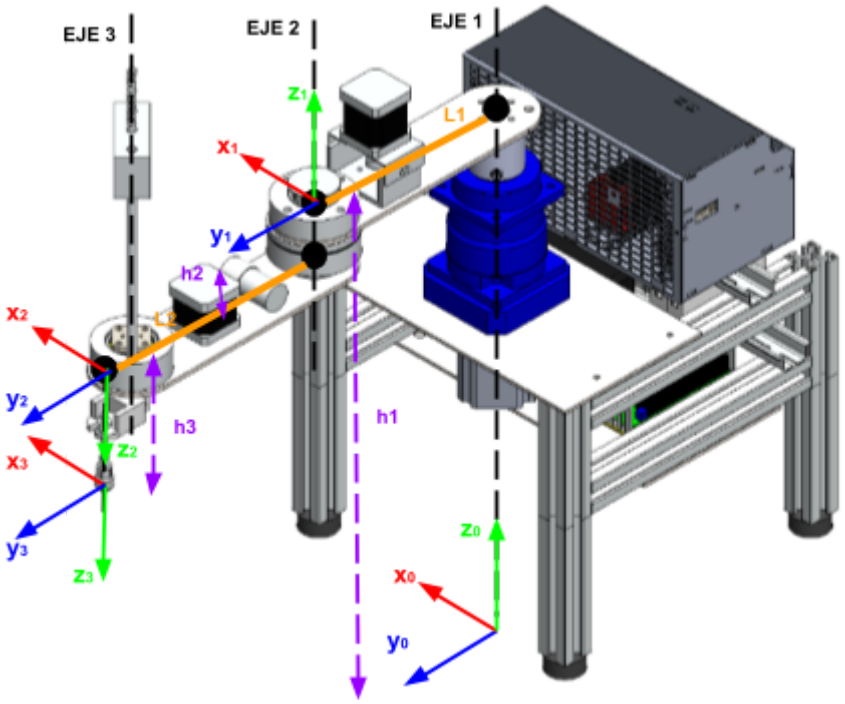


Figura 43. Planteamiento de los sistemas de coordenadas de las articulaciones del robot.

Articulación	Tipo	θ	d.z	d.x	α
1	Revolución	q_1	h_1	L_1	0
2	Revolución	q_2	$-h_2$	L_2	π
3	Prismática	0	$-q_3 - h_3$	0	0

Figura 44. Tabla de parámetros Denavit-Hartenberg del robot scara

Los valores de dichos parámetros se obtienen directamente del robot implementado:

Articulación	Tipo	$\theta(rad)$	d.z(mm)	d.x(mm)	α(rad)
1	Revolución	q_1	431	200	0
2	Revolución	q_2	-44	230	π
3	Prismática	0	$-q_3 - 101$	0	0

Figura 45. Tabla de parámetros Denavit-Hartenberg del robot scara con valores numéricos

Debemos resolver el problema cinemático inverso del robot para obtener las ecuaciones a implementar en el controlador del robot y que le permitirán convertir las coordenadas de un punto en el espacio tridimensional respecto al sistema de referencia de la base del robot, en la configuración articular correspondiente a dicho punto. Al tratarse de un robot scara de 3 grados de libertad, podemos utilizar el método geométrico por su simplicidad.

Los valores articulares de las dos primeras articulaciones, q1 (ángulo respecto el eje x) y q2 (ángulo respecto L1) son obtenidos mediante la simplificación teórica de la representación trigonométrica del brazo del robot. Escogemos la configuración articular de codo derecho pese a que también podríamos seleccionar la de codo izquierdo siendo indiferente emplear u otra en esta aplicación en concreto. Este criterio si que es determinante en otras aplicaciones donde el espacio físico en el que se ubica el robot es limitado por lo que este no tiene libertad total de movimiento, sin embargo este no es nuestro caso.

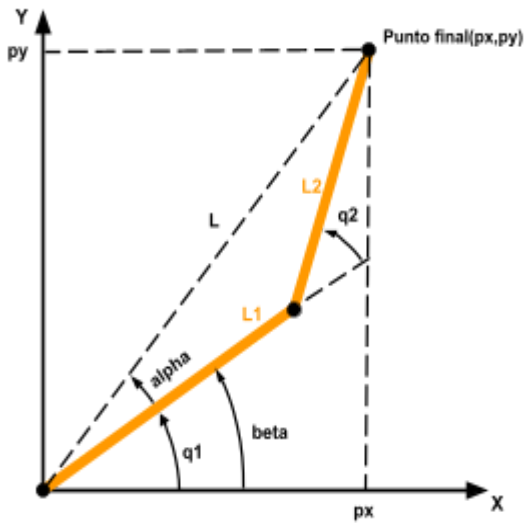


Figura 46. Representación trigonométrica simplificada de la articulación 1 y 2 con configuración de brazo derecho.

Planteamos un sistema de ecuaciones el cual nos dará como solución los valores de q1 y q2 en función de la coordenada del punto empleado:

Definimos la ecuación de L con el teorema de pitágoras:

(1) $$L^2 = p_x^{\ 2} + p_y^{\ 2}$$

Definimos el ángulo beta como el arco tangente de las coordenadas del punto:

$$(2) \qquad \beta = arctg(\frac{p_y}{p_x})$$

Aplicamos el teorema del coseno para obtener el valor de q2:

$$(3) \qquad L^2 = L_1^{\ 2} + L_2^{\ 2} + 2 \cdot cos(q_2)$$

Despejamos q2 de la ecuación (3):

$$(4) \qquad q_2 = cos^{-1}(\frac{L^2 - L_1^{\ 2} - L_2^{\ 2}}{2 \cdot L_1 \cdot L_2})$$

Sustituyendo la ecuación (1) en la (4) obtenemos la configuración articular q2 en función de las coordenadas del punto:

$$(5) \qquad q_2 = cos^{-1}(\frac{p_x^{\ 2} + p_y^{\ 2} - L_1^{\ 2} - L_2^{\ 2}}{2 \cdot L_1 \cdot L_2})$$

Definimos el ángulo alpha:

$$(6) \qquad \alpha = tan^{-1}(\frac{L_2 \cdot sen(q_2)}{L_1 + L_2 \cdot cos(q_2)})$$

Observamos que el parámetro articular q1 es la sustracción de los dos ángulo definidos previamente:

$$(7) \qquad q_1 = (\beta - \alpha)$$

Sustituyendo las ecuaciones (2) y (6) en la (7) obtenemos la configuración articular q1 en función de las coordenadas del punto:

$$(8) \qquad q_1 = (arctg(\frac{p_y}{p_x}) - tan^{-1}(\frac{L_2 \cdot sen(q_2)}{L_1 + L_2 \cdot cos(q_2)}))$$

La tercera articulación es prismática y se encarga de regular la altura de el punto del efector final en el eje z respecto el sistemas de coordenadas de la base del robot. El valor articular de esta q3, se obtiene mediante el sumatorio de las alturas de los sistemas de referencia de cada articulación del robot con respecto al sistema de referencia zero .

$$(9) \qquad q_3 = \sum_{i=1}^{3} z_i = z_{0-1} - z_{1-2} - z_{2-3} - Pz = h1 - h2 - h3 - Pz$$

Con tal de implementar el sistema de ecuaciones definido creamos una función en nuestro código de arduino, inverseKin, la cual tendrá como parámetros de entrada un punto en el espacio tridimensional,los parámetros de nuestro robot scara y la variable en la que almacenará la configuración articular calculada. Realizará la cinemática inversa empleando las ecuaciones anteriores y retornará la configuración articular correspondiente a dicho punto. (Encontramos dicha función en el apartado 5.1.2.1 Cinemática inversa del Anexo.)

Figura 47. Bloque funcional de la cinemática inversa

1.7.3.2. Generador de trayectorias

Una vez hemos realizado la cinemática inversa de nuestro robot el siguiente objetivo es realizar un generador de trayectorias que logre un movimiento coordinado de nuestro robot al desplazarse de un punto a otro. Para ello se ha programado un generador de trayectoria polinomial.

El generador de trayectoria polinomial permite al robot rotar sus ejes de una configuración inicial a otra final siguiendo una curva suave. Se controla el robot eje a eje, de manera que este se mueve a una configuración articular determinada por los valores de referencia que establecemos en un tiempo concreto. Los ejes funcionan de manera coordinada para alcanzar la posición de destino al mismo tiempo.Se define una trayectoria en el espacio articular mediante una expresión polinomial. La velocidad de esta trayectoria se obtiene a partir de la derivada de la misma y se toman las condiciones de contorno. Estas consisten en el punto inicial y final de la trayectoria y sus derivadas las cuales serán 0.

(10) $q_i = a_i \cdot t^3 + b_i \cdot t^2 + c_i \cdot t + d_i$ posición en la trayectoria

(11) $\frac{dq_i}{dt} = a_i \cdot t^3 + b_i \cdot t^2 + c_i \cdot t + d_i$ velocidad en la trayectoria

(12) $q_i(0) = q_i^0 \qquad q_i(T) = q_i^T$ condiciones de contorno

$\frac{dq_i(0)}{dt} = 0 \qquad \frac{dq_i(T)}{dt} = 0$

Aplicando las condiciones de contorno obtenemos los valores de los parámetros de la trayectoria polinomial:

(13) $a_i = -\frac{2 \cdot (q_i^T - q_i^0)}{T^3} \qquad b_i = -\frac{2 \cdot (q_i^T - q_i^0)}{T^2} \qquad c_i = 0 \qquad d_i = q_i^0$

Implementamos el generador de trayectoria polinomial en la función moveAbsJ del código de arduino (incluida en el apartado 5.1.2.2 Generador de trayectorias del Anexo). Esta función calcula los parámetros de la trayectoria polinomial utilizando las configuraciones articulares inicial y final. Posteriormente con estos parámetros se genera la trayectoria polinomial para mover el robot de la configuración inicial a la final en el tiempo establecido. Se toma como intervalo de tiempo entre los puntos de la trayectoria el tiempo mínimo que admite el microprocesador el cual es 20ms. Cada 20 ms se calcula un punto de la trayectoria y se manda el robot a dicha configuración.

La dirección de rotación del eje se determina mediante el ángulo inicial y final. Si el ángulo final es mayor al inicial el pin de dirección se desactiva permitiendo la rotación en sentido horario del motor. Sin embargo si el ángulo final es menor al inicial se activa el pin de dirección y la rotación se realiza en sentido antihorario.

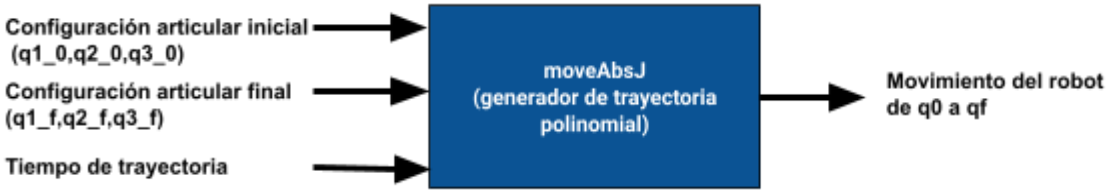


Figura 48. Bloque funcional de la función moveAbsJ

1.7.3.3. Generador de pulsos

La función de generación de pulsos es empleada por el generador de trayectorias para mover los motores a la configuración articulada calculada. Con dicho propósito, esta función es la encargada de mandar pulsos a los drivers de motor los cuales hacen girar los motores. Para poder mandar estos pulsos debemos conocer las salidas digitales a las que están conectados los drivers, las cuales encontramos en la siguiente figura.

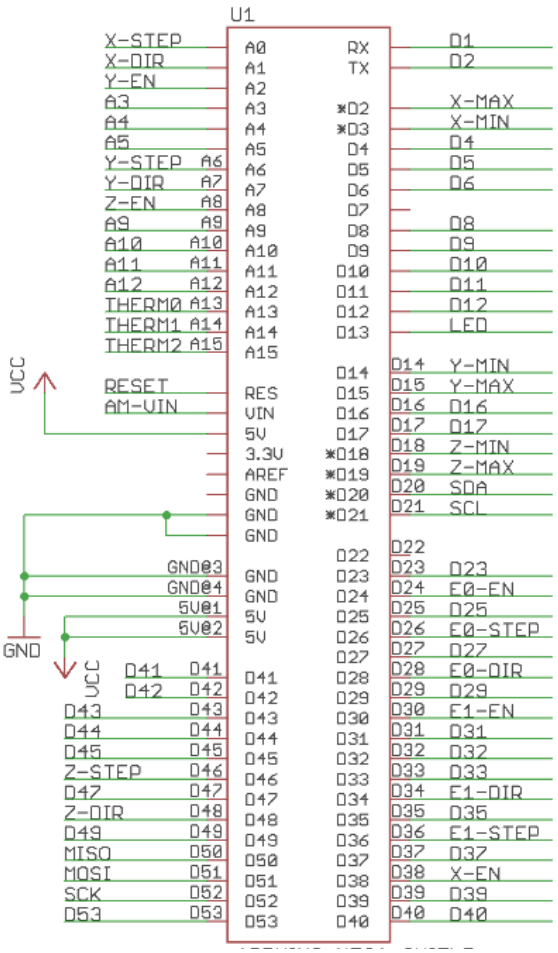


Figura 49. Mapeado sobre el Arduino Mega 2560 de las conexiones con la Ramps 1.4

El driver encargado de controlar el motor del eje 1 tiene su entrada digital de pulso conectado a la salida digital del pin D26 (E0-Step en la figura 47) del arduino y su entrada digital de dirección conectado a la salida digital del pin D28 (E0-Dir en la figura 47) del arduino. El número de pulsos a aplicar se obtiene mediante la división del incremento de ángulo

entre el ángulo actual y el objetivo, dividido la desmultiplicación angular total del eje 1. Este es el ángulo que rota el motor por paso dividido entre la multiplicación de la desmultiplicación mecánica por la desmultiplicación digital.

Así pues el cálculo de la desmultiplicación angular es el siguiente:

$$Desmultiplicación\ angular\ total = \frac{1.8º}{1\ paso} / (Desmultiplicación\ mecánica \cdot Desmultiplicación\ digital)$$

$$Desmultiplicación\ total = \frac{1.8º}{1\ paso} / (7{:}\ 1 \cdot 4{:}\ 1) = \frac{1.8º}{28}$$

El número de pulsos a aplicar para mover el motor a un nuevo ángulo es el siguiente:

$$Número\ de\ pulsos = \frac{\Delta q}{\frac{1.8º}{28}} = \frac{q_{1\ actual} - q_{1\ final}}{\frac{1.8º}{28}}$$

El arduino aplica un tren de impulsos de 2ms de periodo y N impulsos siendo N los impulsos calculados a la entrada digital de impulsos del driver de motor del eje 1. La velocidad de rotación de la articulación del eje 1 se calcula como la desmultiplicación total dividido el periodo del tren de impulsos:

$$V_{rotación\ eje\ 1} = Desmultiplicación\ total / periodo\ tren\ impulsos$$

$$V_{rotación\ eje\ 1} = \frac{0.064º/impulso}{0.002s/impulso} = 32º/s$$

El driver encargado de controlar el motor del eje 2 tiene su entrada digital de pulso conectado a la salida digital del pin D54 (X-Step en la figura 47) del arduino y su entrada digital de dirección conectado a la salida digital del pin D55 (X-Dir en la figura 47) del arduino. El número de pulsos a aplicar se obtiene mediante la división del incremento de ángulo entre el ángulo actual y el objetivo, dividido la desmultiplicación angular total del eje 2. Este es el ángulo que rota el motor por paso dividido entre la multiplicación de la desmultiplicación mecánica por la desmultiplicación digital.

Así pues el cálculo de la desmultiplicación angular es el siguiente:

$$Desmultiplicación\ angular\ total = \frac{1.8º}{1\ paso} / (Desmultiplicación\ mecánica \cdot Desmultiplicación\ digital)$$

$$Desmultiplicación\ total = \frac{1.8º}{1\ paso}/(3{:}\ 1{\cdot}8{:}\ 1) = \frac{1.8º}{24}$$

El número de pulsos a aplicar para mover el motor a un nuevo ángulo es el siguiente:

$$Número\ de\ pulsos = \frac{\Delta q}{\frac{1.8º}{24}} = \frac{q_{2actual} - q_{2\ final}}{\frac{1.8º}{24}}$$

El arduino aplica un tren de impulsos de 2ms de periodo y N impulsos siendo N los impulsos calculados a la entrada digital de impulsos del driver de motor del eje 2. La velocidad de rotación de la articulación del eje 1 se calcula como la desmultiplicación total dividido el periodo del tren de impulsos:

$$V_{rotación\ eje\ 1} = Desmultiplicación\ total/periodo\ tren\ impulsos$$

$$V_{rotación\ eje\ 1} = \frac{0.075º/impulso}{0.002s/impulso} = 37.5º/s$$

1.7.3.4. Función recogerCápsula

Esta función es empleada por el arduino para recoger una cápsula tras desplazarse a la posición de recogida mediante la función moveAbsJ. Se manda un tren de impulsos a través de la salida digital del pin 60 del arduino el cual hace rotar el motor del eje 3 que a su vez hace rotar el husillo bajando el efector final hasta la altura de recogida. El número de impulsos de dicho tren se determina mediante la siguiente relación:

$$N\ impulsos = 200\ pulsos/cm{\cdot}\ \mathrm{h}$$

A su vez la velocidad de desplazamiento del efector final situado en dicho husillo es la siguiente:

$$V_{effector} = \frac{1}{200\ pulsos/cm}{\cdot}2ms/pulso = 2.5\ cm/s$$

La altura del punto de recogida es de 16.5cm por lo que el número de impulsos será de 3300. Cuando se alcanza dicha altura se activa la bomba de vacío mediante el pin digital 10. Esta succiona la cápsula y se vuelve a subir el efector final hasta su posición inicial.

1.7.3.5. Función depositarCápsula

Esta función es empleada por el arduino para depositar una cápsula tras desplazarse a la posición de empaquetado mediante la función moveAbsJ. Se sigue el mismo procedimiento que en la función recogerCapsula con la diferencia de que se desactiva la salida digital del pin 10 para que la bomba deje de succionar y la cápsula caiga.

1.7.3.6. Comunicaciones serial

El arduino Mega está en todo momento conectado mediante su puerto serial con el sistema de visión artificial. Ambos sistemas actúan de manera coordinada para lograr empaquetar las cápsulas. Esta coordinación se logra mediante el intercambio de mensajes entre ambos sistemas.

El inicio de la comunicación se produce al iniciar el modo automático de funcionamiento. Se inicia el puerto serial y el sistema de visión manda una cadena de inicio al robot. Este la recibe y contesta como confirmación de recepción del mensaje. Posteriormente tras esta secuencia inicial, el sistema de visión avisará al robot cada vez que este tenga que recoger una cápsula. Cuando esto sucede el robot realiza un movimiento coordinado mediante moveAbsJ hasta la posición de recogida, coge la cápsula mediante recogerCapsula y se la lleva hasta la posición de descarga mediante otro movimiento coordinado (la caja si es una cápsula azul o la posición de descarte si es roja o naranja). Posteriormente emplea depositarCapsula para descargarla y vuelve a la posición de recogida mediante un movimiento coordinado donde espera la siguiente orden.

1.7.4. Hardware del sistema de visión

El sistema de visión integra los elementos propios de un sistema de estas características los cuales son una cámara para la adquisición de imágenes, un dispositivo de procesamiento el cual aplica el algoritmo de visión programado y una iluminación.

1.7.4.1. Cámara

La cámara adecuada a la aplicación de este proyecto se ha escogido conforme los criterios de la resolución, precisión del color y precio del dispositivo. La cámara escogida es la webcam AMDIS de Conceptronics.

Figura 50. Webcam Amdis de Conceptronics

La resolución es el nivel de detalle con el que se pueden capturar imágenes. Este dispositivo cuenta con una resolución de 1280x720 píxeles. Emplear una cámara con mayor resolución no es determinante en el correcto funcionamiento de la aplicación de visión ya que con el objetivo de acelerar el procesamiento de las imágenes y consumir menos recursos del dispositivo de procesamiento se ha realizado un reescalado de la imagen a una resolución de 640x480 antes de su procesamiento.

La precisión del color es la capacidad de nuestra cámara de reproducir los colores que observamos. Está determinada tanto por los sensores de la cámara como por la iluminación aplicada. Los elementos luminosos tienen un precio menor que las cámaras por lo cual se ha optado por abaratar la cámara y aplicar una mejor iluminación para lograr una buena precisión de color.

El precio de la cámara ha sido el factor determinante para su elección. Este es de tan solo 19,99 euros. La relación calidad precio de esta cámara es inmejorable considerando los aspectos discutidos.

1.7.4.2. Dispositivo de procesamiento

El dispositivo de procesamiento empleado es el ordenador portátil MSI CX6QD. Se ha empleado este ordenador por la capacidad de procesamiento de su microprocesador intel core i7 de octava generación. Sin embargo tiene el inconveniente de el espacio que ocupa comparado con una Raspberry Pi la cual sería una alternativa si este fuese un condicionante.

El ordenador emplea la webcam conectada a un puerto USB para adquirir las imágenes. Estas imágenes son procesadas mediante el algoritmo de visión artificial programado en Python el cual se ejecuta en dicho dispositivo.

El ordenador también actúa como pasarela de comunicación entre el robot scara y el sistema de visión artificial. Estos dos sistemas intercambian información mediante los puertos seriales del ordenador.

1.7.4.3. Iluminación

La iluminación es una de las partes más importantes de un sistema de visión ya que esta determinará el correcto funcionamiento del mismo. Las cámaras, al igual que el ojo humano, capturan la luz reflejada en los objetos por lo que un cambio en la iluminación puede alterar características de dichos objetos como el color o la forma percibida.

Con tal de evitar estos inconvenientes se ha dotado de flexibilidad al algoritmo de visión de forma que al iniciar se realiza un ajuste de los colores manual. Esto permite asegurarse de que la iluminación de la ubicación del sistema implementado no influye negativamente en el correcto funcionamiento del mismo.

Se ha empleado luz lateral generada por una fuente de luz puntual combinada con la luz ambiente que hay en el entorno en el que se ubica el sistema implementado.

1.7.5. Programación del sistema de visión

La programación del sistema de visión artificial para el reconocimiento de las cápsulas se ha realizado en Python empleando la librería Open CV la cual integra un paquete de funciones para aplicaciones de visión artificial.

El sistema de visión artificial implementado realiza una captura constante de imágenes (video) a razón de 30 imágenes por segundo (30fps) por medio de la cámara. Las imágenes adquiridas son preprocesadas con el objetivo de preparar la imagen para su procesado. Se realiza la segmentación de la imagen la cual separa el fondo de los objetos de interés (las cápsulas). Posteriormente se analiza la imagen resultante extrayendo las características de los objetos que nos permitirán identificarlos. Finalmente se clasifican los objetos reconocidos creando un clasificador y se realizan decisiones en base a los resultados obtenidos.

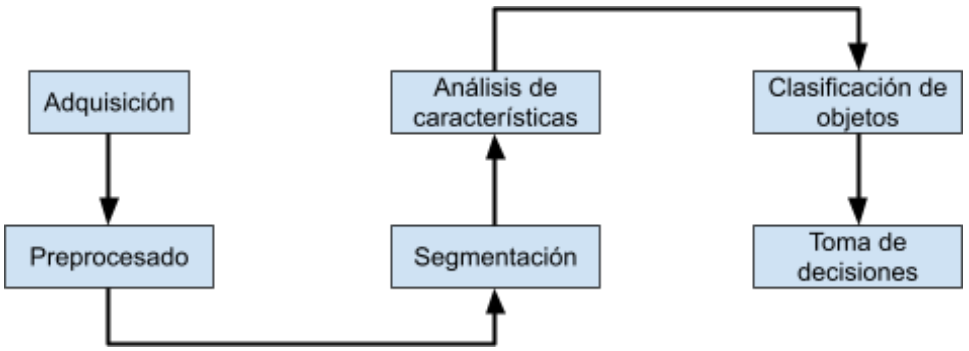


Figura 51. Etapas del diseño del sistema de visión artificial

Los objetos a identificar son cápsulas de café. Se han establecido tres tipos de cápsulas las cuales circularán por la cadena de empaquetado, cápsulas nescafé dolce gusto ardenza caracterizadas por su color azul, cápsulas nescafé dolce gusto perú caracterizadas por su color rojo y cápsulas nescafé dolce gusto grande intenso caracterizadas por su color naranja. El sistema de visión artificial se ha programado para identificar estos tres tipos.

Figura 52. Tipos de cápsula a identificar

Se ha programado la función filterColor la cual realiza el preprocesamiento y segmentación de la imagen basándose en criterios de filtrado por color. A su vez se ha programado la función findCircles que realiza la extracción de características de los objetos en la imagen procesada empleando el criterio de la forma geométrica. Se ha implementado la función Iniciar, la cual inicia la captura de video y carga los parámetros iniciales de las funciones anteriores. Finalmente se ha programado la función Visualizar la cual emplea las funciones filterColor y findCircles para obtener las características de los objetos en el video que permitan realizar su clasificación.

1.7.5.1. Función filterColor

Esta función es la encargada de preprocesar y segmentar la imagen. El preprocesamiento de la imagen consiste en el filtrado empleando un filtro bilateral para suavizar la imagen y en la conversión de la imagen del espacio de color RGB a HSV. La segmentación consiste en el uso de una máscara creada mediante umbrales de color para realizar una operación aritmética de adición con la imagen original y extraer los objetos deseados. El funcionamiento del procesamiento completo realizado por la función filterColor se va a explicar empleando la siguiente imagen de ejemplo en la cual se van a segmentar las cápsulas azules.

Figura 53. Imagen de ejemplo

El filtrado bilateral de la imagen nos permite reducir el ruido y suavizar la imagen . Este filtro reemplaza el valor de intensidad de cada píxel de la imagen por una media ponderada de los valores de la intensidad de los píxeles cercanos. El resultado de dicho proceso de filtrado se puede observar en la figura siguiente.

Figura 54. Imagen filtrada bilateralmente

La imagen capturada por la cámara emplea el espacio de color RGB mediante el cual se define el color de cada pixel de la imagen con la adición de los tres colores primarios rojo, verde y azul. La intensidad de cada uno de estos colores se mide de 0 a 255. Este espacio de color puede ser dificil de interpretar por lo que se ha optado por el espacio de colores HSV el cual define los colores basándose en el matiz (Hue), la saturación (Saturation) y el valor (Value), aproximación mas semejante a como las personas experimentamos con los colores.

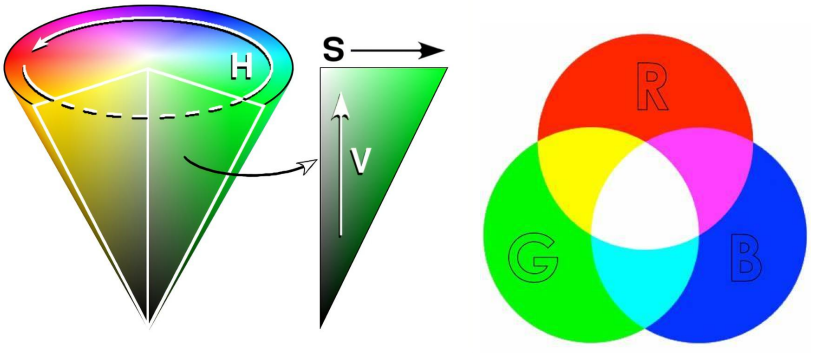


Figura 55. Espacio de color HSV(Izquierda) y RGB(derecha)

Realizamos la conversión de la imagen filtrada bilateralmente del espacio de colores RGB al espacio de colores HSV y obtenemos el siguiente resultado:

Figura 56. Imagen convertida de RGB a HSV

Empleando la imagen convertida a espacio HSV creamos una máscara binarizando dicha imagen asignando a los píxeles de los objetos de interés y a los del fondo valores diferentes (255 y 0 respectivamente). Para binarizar se emplea un rango de valores para el matiz, saturación y valor el cual definirá el color del objeto a aislar. Los píxeles que no estén en dicho rango (píxeles del fondo) se convierten a negro (0) mientras que los que esten en el rango se convierten a blanco (255). En el caso concreto de este ejemplo, el rango de valores para las cápsulas azules de la imagen original es matiz de 29 a 255, saturación de 83 a 255 y valor de 97 a 255. La máscara obtenida se muestra en la siguiente figura:

Figura 57. Máscara creada con los rangos de valores HSV

La máscara obtenida contiene ruido el cual se puede eliminar erosionando la imagen. Se ha empleado un kernel de 7x7 en el proceso de erosión obteniendo la máscara de la siguiente figura:

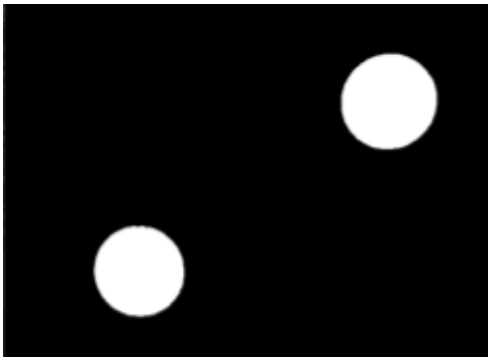


Figura 58. Mascara erosionada

Empleamos la máscara erosionada obtenida y la imagen original para realizar una operación aritmética de adición. Esta operación nos permite aislar las cápsulas azules de la imagen, las cuales son los objetos de interés eliminando todos los demás objetos y el fondo de la imagen original. El resultado final se muestra en la siguiente figura.

Figura 59. Resultado de la operación and entre máscara e imagen original

La función filterColor recibe como parámetros de entrada los valores del rango de matiz, saturación y valor los cuales emplea para calcular la máscara y la imagen a procesar. Realiza las operaciones descritas anteriormente y retorna la imagen procesada.

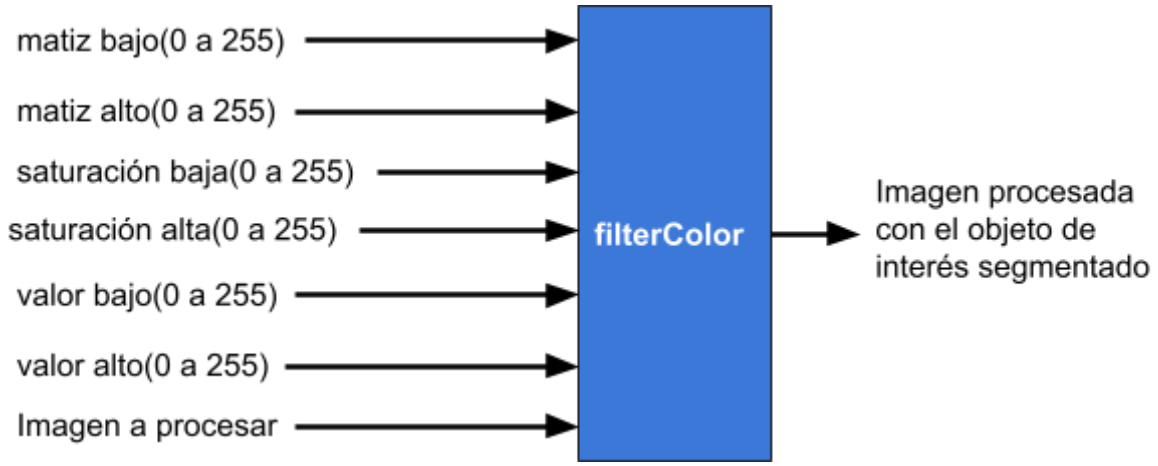


Figura 60. Bloque funcional de la función filterColor

1.7.5.2.Función findCircles

La función find circles implementa el algoritmo de la transformada de círculo de Hogh para la extracción de características de los objetos en la imagen procesada. Este algoritmo emplea una simplificación de otro más complejo, el gradiente de Hough el cual extrae los bordes de los objetos de la imagen tras aplicar un filtro medio. El funcionamiento de la función findCircles va a explicarse mediante la continuación del ejemplo anterior

donde habíamos realizado el preprocesamiento y segmentación de la imagen de ejemplo.

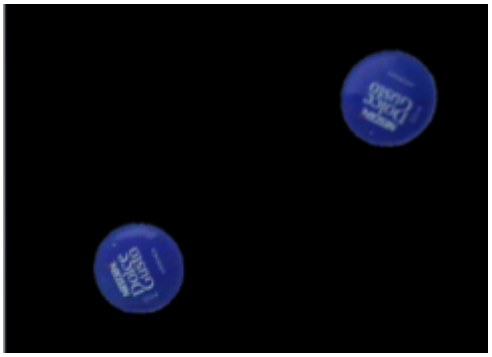


Figura 61. Imagen de ejemplo preprocesada y segmentada

El propio algoritmo de los círculos de Hough requiere un pequeño preprocesamiento adicional de la imagen con el objetivo de obtener unos bordes más definidos que caracterizan nuestro objeto. Por lo tanto convertimos la imagen a escala de grises y aplicamos un filtro medio con un kernel de 19x19, obteniendo la siguiente imagen.

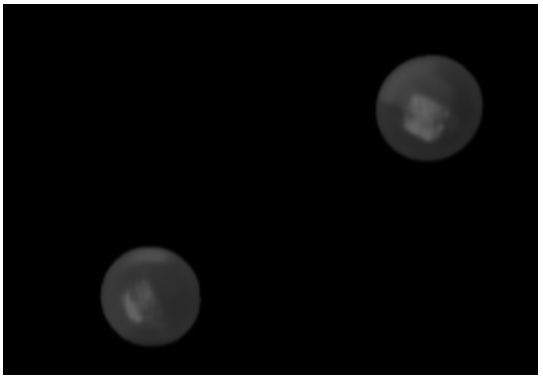


Figura 62. Conversión a escala de grises y aplicación del filtro medio

Posteriormente aplicamos a esta imagen el algoritmo de los círculos de Hough el cual recibe como parámetros de entrada para identificar los círculos en la imagen:

- La imagen a procesar.
- El método del modelo de detección el cual es el gradiente de Hough.

- El valor inverso del acumulador de la resolución de la imagen. Este valor va de 0.1 a 2. A mayor valor más círculos falsos serán detectados mientras que a menor valor más exacto deberá ser el círculo en la imagen para que sea detectado. En el caso del ejemplo se establece en 1.5.

- La distancia mínima entre objetos de la imagen. En el caso del ejemplo se establece en 80 pixeles. Esto evita que falsos círculos sean detectados dentro de otros círculos.

- El parámetro 1, el valor alto del rango de valores del filtro Canny para la detección de bordes en la imagen. Debe ser alto para evitar falsos positivos. En este ejemplo se ha establecido en 100.

- El parámetro 2, el acumulador de valores que determina la exactitud de un círculo empleando su centro. A menor valor más círculos falsos serán detectados. En este ejemplo se ha fijado a 30.

- El rango de radio del círculo a detectar.

El resultado de aplicar este algoritmo sobre una imagen es una tabla donde en cada fila se almacenan las propiedades de un círculo detectado las cuales són las coordenadas del centro del círculo en la imagen (x, y en píxeles) y su radio (en píxeles). Empleando los valores de dicha tabla dibujamos sobre la imagen un círculo verde sobre cada círculo detectado en la imagen y marcamos su centro con un círculo pequeño rojo. Aplicando el algoritmo a la imagen de ejemplo obtenemos el siguiente resultado.

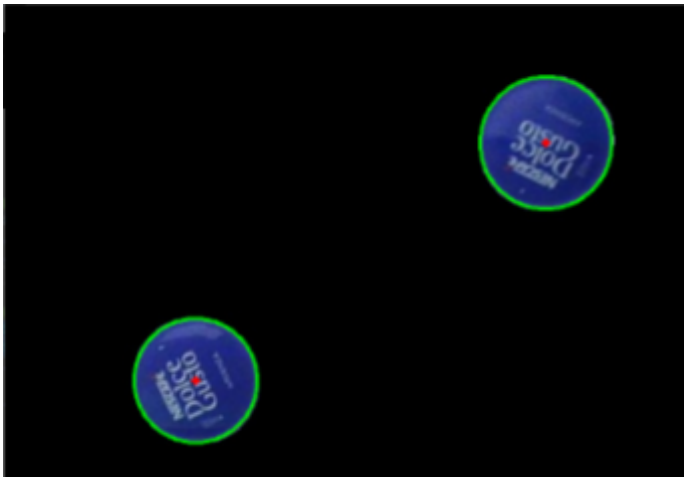


Figura 63. Cápsulas caracterizadas en la imagen

La función findCircles devuelve la imagen con los círculos caracterizados mediante un círculo verde para el perímetro y uno rojo para el centro, el número de círculos en la imagen y una lista de python en la cual se ha almacenado la tabla de las propiedades de cada círculo identificado en la imagen.

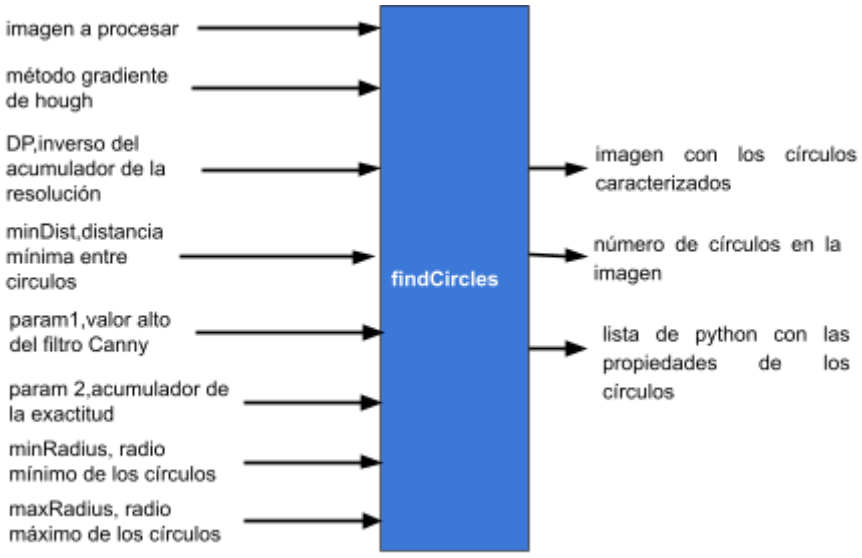


Figura 64. Bloque funcional de la función findCircles

1.7.5.3.Función Iniciar

Esta función se emplea para iniciar la captura de video de la cámara. A su vez carga los valores de los parámetros de entrada de la función filterColor para los tres colores de cápsula analizados y los parámetros de la función findCircles para la detección de círculos. Estos valores están almacenados en un archivo de texto el cual se encuentra en la misma carpeta que el código. La función abre dicho archivo, lee los valores escritos en él y los almacena en variables dentro del programa. Una vez se ha iniciado la cámara y los valores han sido cargados se inicia la función Visualizar.

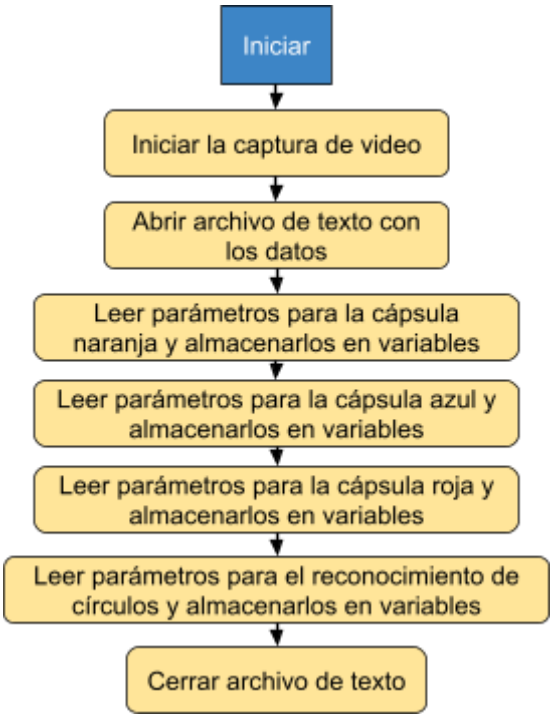


Figura 65. Diagrama de flujo función Iniciar

1.7.5.4.Función Visualizar

La función Visualizar es iniciada por la función iniciar tras la inicialización de la captura de video y de las variables que serán empleadas como parámetros de entrada de findColor y findCircles.

Esta función aplica la función findColor con los tres conjuntos de valores correspondientes a cada cápsula de manera que obtiene tres imágenes, una imagen preprocesada y segmentada con las cápsulas naranjas, otra con las azules y otra con las rojas. Posteriormente emplea la función findCircles en cada una de las tres imágenes para identificar las características de cada tipo de cápsula de forma que obtenemos la cantidad de cápsulas de cada tipo que hay en el video y una lista de python con sus propiedades. Con los datos recopilados se sobrepone en el video una etiqueta con el tipo de cápsula presente y dos círculos, uno verde para el perímetro y otro rojo para el centro.

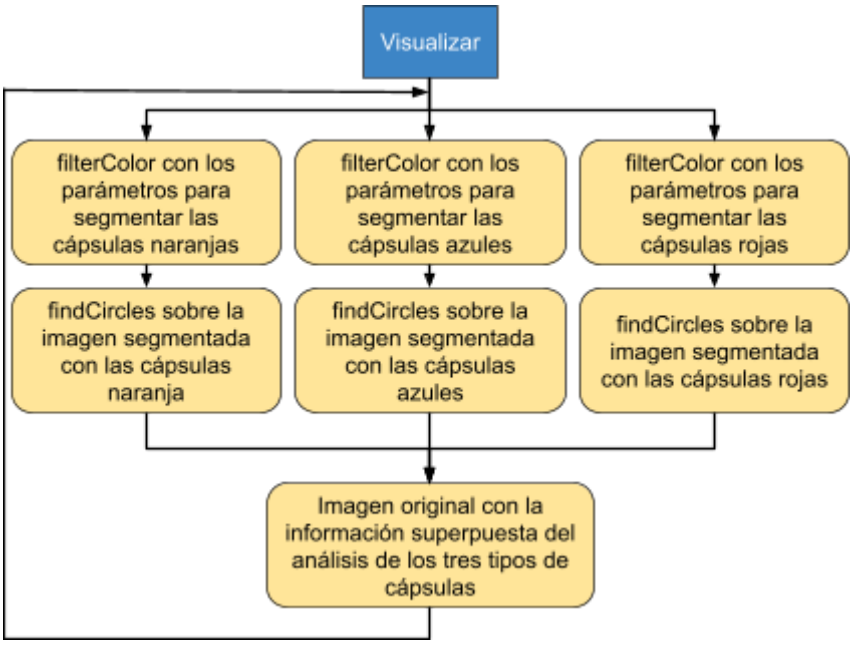


Figura 66. Diagrama de flujo función Visualizar

Aplicando la función Visualizar sobre el sistema de visión artificial montado en su posición final obtenemos el siguiente resultado de procesamiento en el que observamos que se han reconocido los tres tipos de cápsula empleados.

Figura 67. Reconocimiento de las cápsulas sobre el cabezal giratorio.(cápsula azul, roja y naranja etiquetadas)

1.7.6. Hardware de los cabezales giratorios

Se han implementado dos cabezales giratorios. Ambos cabezales van fijados a la estructura del robot scara mediante dos perfiles CNC de 30x30 de 200 milímetros de longitud. Una pletina fijada al perfil actúa de soporte del motor. Se han empleado dos motores NEMA 17 los cuales están conectados a la Ramps 1.4 del robot scara. Los platos están hechos con madera contrachapada.

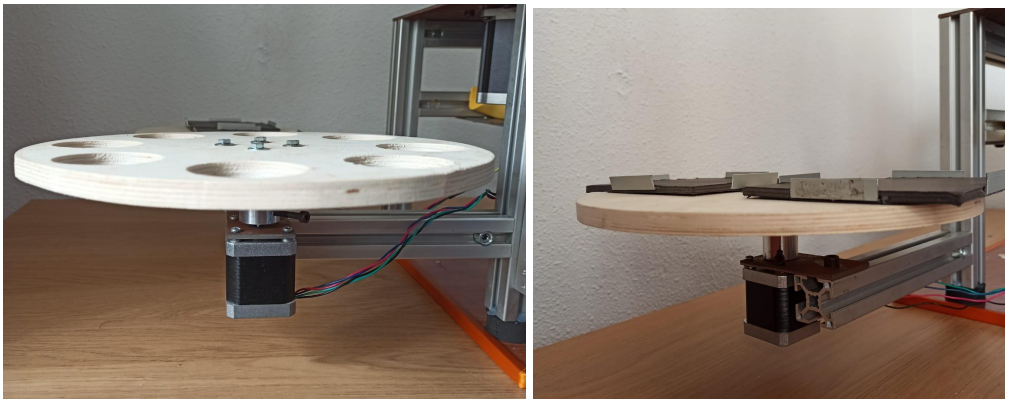


Figura 68. Cabezal giratorio porta cápsulas(izquierda) y portacajas(derecha)

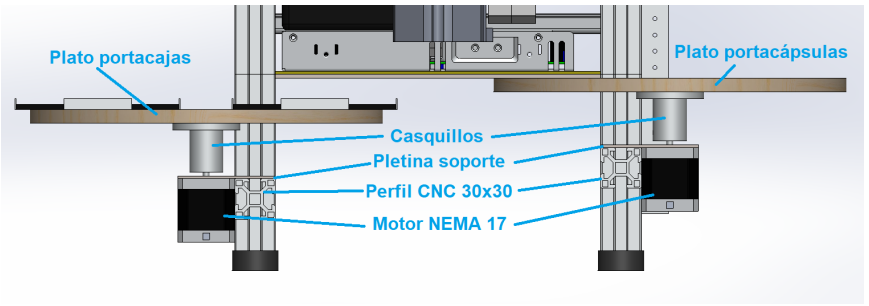


Figura 69. Cabezales giratorios diseñados en solidworks

1.7.7. Software de los cabezales giratorios

El arduino ha sido programado con dos funciones para manipular los cabezales, rotarPlatoCapsula y rotarPlatoCaja.

1.7.7.1. Función rotarPlatoCápsula

La primera función rota el cabezal giratorio porta cápsulas 45º respecto su ángulo actual. Esta se emplea durante el ciclo principal de programa para ir colocando las cápsulas delante de la cámara y en la posición de recogida para que el robot las coja.

1.7.7.2. Función rotarPlatoCaja

La segunda función rota el cabezal giratorio porta cajas 180º. Se emplea en el programa principal para cambiar de caja una vez ésta haya sido llenada con 16 cápsulas.

1.7.8. Integración de los sistemas

1.7.8.1. Modo de ajuste de cámara

El sistema de visión artificial cuenta con un menú para ajustar los parámetros del filtrado por color y de algoritmo de la transformada de círculo de Hough. Esto nos permite ajustar rápidamente nuestro sistema de visión ante cualquier cambio en la iluminación del entorno donde la celda esté instalada. Se selecciona un tipo de cápsula y posteriormente se ajusta el matiz(Hue), saturación(saturation) y valor(value) hasta que el fondo se elimine y se muestre únicamente la cápsula de dicho tipo. Posteriormente se ajustan los parámetros del algoritmo de la transformada de los círculos de Hough hasta que el número de círculos sea uno. En este ajuste hay que tener en cuenta que sobrepasar el valor idóneo puede causar la detección de círculos falsos. Los valores ajustados se pueden guardar presionando el botón save settings de forma que la próxima vez que iniciemos nuestro programa no tendremos que reajustar de nuevo si no que estos valores se cargarán de forma automática.

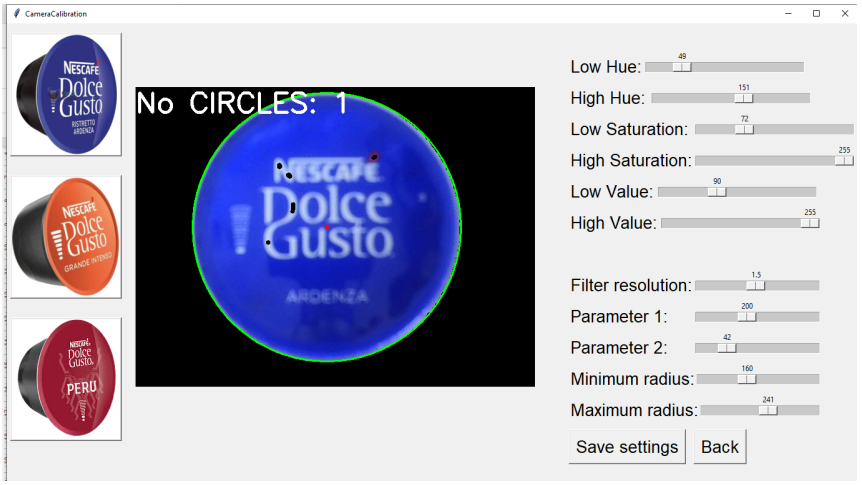


Figura 70. Pantalla de ajuste de parámetros

Tras ajustar el sistema de visión artificial procedemos a iniciar el modo automático de funcionamiento.

1.7.8.2.Modo de ciclo automático

Los elementos descritos anteriormente se han sincronizado para llevar a cabo el objetivo de este proyecto, mediante un intercambio de mensajes a través de comunicaciones seriales entre el arduino mega y el programa de Python. Este manda al arduino mega tres mensajes tras cuya recepción se realizan tres acciones distintas:

- Mensaje "PICK": el arduino mega desplaza el robot de la posición actual en la que se encuentre a la posición de recogida de cápsulas mediante la función moveAbsJ. Posteriormente emplea la función recogerCapsula para coger la cápsula. Vuelve a emplear la función moveAbsJ para desplazarse a la posición de depositado de cápsulas(la de la caja). Esta posición tiene cuatro valores que se van alternando de manera que cada vez se coloca la cápsula en una diferente. Se emplea la función depositarCapsula la cual coloca la cápsula en la caja. El robot vuelve a la posición de recogida y se queda esperando órdenes. El arduino rota el cabezal de las cápsulas mediante rotarPlatoCapsula. Se incrementa un contador de cápsulas empaquetadas. Si las cápsulas empaquetadas son 16 se rota el cabezal de las cajas mediante rotarPlatoCaja. Finalmente se manda el mensaje "Done" al programa de Python indicando que se ha terminado la acción.

- Mensaje “ROTATE”: el arduino mega rota el plato de las cápsulas mediante rotarPlatoCapsulas. Manda el mensaje “Done” al programa de python tras realizar esta acción.

- Mensaje “REM”: el arduino mega desplaza el robot de la posición actual en la que se encuentre a la posición de recogida de cápsulas mediante la función moveAbsJ. Posteriormente emplea la función recogerCapsula para coger la cápsula. Vuelve a emplear la función moveAbsJ para desplazarse a la posición de descarte de cápsulas. Se emplea la función depositarCapsula la cual suelta la cápsula. El robot vuelve a la posición de recogida y se queda esperando órdenes. El arduino rota el cabezal de las cápsulas mediante rotarPlatoCapsula. Finalmente se manda el mensaje “Done” al programa de Python indicando que se ha terminado la acción.

Las decisiones son tomadas por el programa principal de Python. Este tiene dos hilos de ejecución, uno para gestionar el sistemas de visión artificial y otro para la toma de decisiones y la comunicación con el arduino mega:

- Hilo de ejecución 1: se ejecuta la función Iniciar la cual inicia la captura de video y carga los valores de los parámetros de filtrado por color y del algoritmo de la transformada de círculo de Hough almacenados en un archivo de texto. Posteriormente se inicia la función visualizar la cual aplica las funciones de filterColor y findCircle con el conjunto de parámetros de cada color para identificar el número de cápsulas de cada color presente en la imagen. Almacena este valor en tres variables, uno por tipo de cápsula. Si aparece alguna cápsula en la imagen muestra los resultados del análisis superponiendo sobre esta un círculo verde en el perímetro, uno rojo en el centro y una etiqueta con el tipo de cápsula.

- Hilo de ejecución 2: se inician las comunicaciones seriales entre el arduino y el programa. El programa consulta los valores de las tres variables del hilo 1 que indican el número de cápsulas de cada tipo en la imagen. Si no hay ninguna cápsula manda el mensaje “ROTAR” al arduino hasta que se detecta alguna. Tras esta detección, si se trata de una cápsula azul manda el mensaje “PICK” para que la encaje. Si es roja o naranja manda el mensaje “REM” para que la descarte. Este hilo también se encarga de actualizar el HMI el cual se emplea para controlar el proceso.

Figura 71. HMI del modo automático de funcionamiento.

La subdivisión del programa principal en dos hilos de ejecución, el hilo 1 encargado de ejecutar el algoritmo de visión artificial y el hilo 2 encargado de tomar decisiones y comunicarse con el robot, permite realizar un control contínuo de las cápsulas y del robot. Sin embargo otra alternativa sería emplear únicamente un hilo de ejecución en el cual se inspeccionarán todas las posiciones del cabezal giratorio de las cápsulas de una sola vez y posteriormente se indicara al robot la operación a realizar con cada cápsula detectada.

1.8. Resultados

Tras realizar la integración de los sistemas descritos anteriormente se han realizado 10 ciclos ininterrumpidos en los que no se ha cometido ningún fallo de funcionamiento y se han empaquetado exitosamente 10 cajas de cápsulas. Se han reproducido las mismas condiciones en estos 10 ensayos, en los que se han introducido 8 cápsulas azules se ha dejado vaciar el porta cápsulas y se han introducido 8 más. A su vez se han introducido una cápsula roja y una naranja las cuales han sido descartadas exitosamente en los 10 ensayos.

El sistema de visión artificial ha sido evaluado mediante la matriz de confusión del clasificador la cual ha sido calculada con los datos obtenidos de los 10 ensayos realizados.

		Tipo de cápsula real			
		Ardenza(Azul)	Grande Intenso(naranja)	Perú(rojo)	Sin capsula
Tipo de cápsula predecida	Ardenza(Azul)	**160** 100%	**0** 0%	**0** 0%	**0** 0%
	Grande Intenso(naranja)	**0** 0%	**10** 100%	**0** 0%	**0** 0%
	Perú(rojo)	**0** 0%	**0** 0%	**10** 100%	**0** 0%
	Sin capsula	**0** 0%	**0** 0%	**0** 0%	**80** 100%

Figura 72. Matriz de confusión del sistema de visión artificial con los datos de los ensayos.

A partir de la matriz obtenida calculamos la precisión, exactitud, sensibilidad y especificidad, los cuales son indicadores para evaluar nuestro clasificador de cápsulas. Estos valores se calculan para cada tipo de cápsula predecida.

$$Precisión = \frac{Predicciones\ Correctas}{Nº\ total\ de\ predicciones} \quad Exactitud = \frac{Verdaderos\ positivos}{Verdaderos\ positivos + Falsos\ positivos}$$

$$Sensibilidad = \frac{Verdaderos\ positivos}{Verdaderos\ positivos + Falsos\ negativos} \quad Especificidad = \frac{Verdaderos\ negativos}{Verdaderos\ negativos + Falsos\ positivos}$$

Para las cápsulas azules:

$$Precisión = \frac{160}{160} = 1 \quad Exactitud = \frac{160}{160+0} = 1$$

$$Sensibilidad = \frac{160}{160+0} = 1 \quad Especificidad = \frac{20}{20+0} = 1$$

Los mismos resultados se repiten para los otros dos tipos de cápsulas. Observamos que la precisión, exactitud, sensibilidad y especificidad de nuestro sistema es del 100% por lo que este es capaz de clasificar exitosamente el 100% de las cápsulas analizadas. Esto es en parte grácias al ajuste dinámico de la cámara mediante el panel de configuración programado que nos permite reajustar los parámetros ante cualquier cambio de iluminación. Otro factor que contribuye a dicha precisión es el método de filtrado de color empleado el cual se ve mucho menos afectado por cambios bruscos de iluminación que los convencionales evitando inconvenientes.

El robot scara implementado tiene una precisión alcanzando las posiciones deseadas y una repetibilidad en el cambio de posición para volver a la posición inicial muy altas por lo que pese a la gran cantidad de movimientos que realiza cambiando de posiciones, siempre vuelve a los mismos puntos. Esto se ha logrado gracias a la desmultiplicación mediante elementos mecánicos de los motores que mueven sus articulaciones. El eje 1 tiene una desmultiplicación total de 0.064º/paso mientras que en la 2 es de 0.075º/paso. Mediante esta desmultiplicación se ha logrado disminuir el error de ángulo generado por el cómputo de la cinemática inversa de las posiciones a las que se desplaza el robot. Al tener una desmultiplicación tan pequeña se logra llegar con mayor exactitud a los valores angulares calculados. A su vez el eje Z logra una velocidad de desplazamiento de 2.5 cm/segundo grácias al sistema de husillo y tuerca implementados. Los ensayos realizados son prueba de la repetibilidad de nuestro robot, el cual ha encajado correctamente 160 cápsulas y descartado 20 realizando un total de 1080 movimientos exitosamente.

Los cabezales giratorios muestran una gran repetibilidad en su labor de transportar las cápsulas. Ambos cabezales rotan 1.8º/paso ya que no se ha empleado ninguna desmultiplicación. Esta no era requerida porque se manejan objetos de poco peso, las cápsulas. El cabezal de las cápsulas, durante los ensayos realizados, ha realizado más de 180 giros de 45º sin descontar ni un solo grado al igual que el de las cajas el cual ha realizado 10 giros de 180º.

La integración de estos tres sistemas se ha llevado a cabo exitosamente como verifican los resultados de los ensayos en los cuales se han empaquetado correctamente un total de 160 cápsulas del mismo tipo distribuidas en 10 cajas. El tiempo medio que ha tardado la celda en llenar una caja desde que se pone la primera cápsula en el porta cápsulas a cuando da la vuelta al porta cajas para entregarnos la caja es de 6 min y 40. Este tiempo es aceptable teniendo en cuenta las limitaciones y el precio de nuestra celda robotizada. Sin embargo se debería adaptar la celda si se quisiera incorporar en una instalación industrial para reducir este tiempo lo máximo posible.

1.9. Conclusiones

El principal objetivo de este proyecto de diseñar e implementar una celda robotizada con visión artificial para el empaquetado de cápsulas de café se ha llevado a cabo de forma exitosa. Este proyecto engloba todos los campos que me apasionan los cuales són la visión artificial combinada con la robótica y la programación. Pese a esto ha supuesto un reto personal ya que nunca había llevado a cabo un proyecto de dicha magnitud.
Se ha cumplido el objetivo de diseñar, fabricar y programar un robot scara de 3 grados de libertad el cual ha sido dotado de una ventosa para poder manipular las cápsulas. Se ha optado por un arduino mega como controlador del robot ya que debido a su popularidad hay multitud de placas de extensión en el mercado entre ellas la Ramps 1.4 para controlar motores de pasos. Otro factor ha sido la facilidad con la que se programa permitiendo implementar la cinemática inversa y otras funciones requeridas en esta aplicación. Esta facilidad de usar viene con un inconveniente y es que no puede realizar más de un proceso a la vez. Esto habría sido útil para reducir el tiempo de ciclo ya que habría permitido, por ejemplo, mover el robot y girar los cabezales giratorios al mismo tiempo.

Se ha implementado y programado un sistema de visión artificial el cual nos ha permitido clasificar las cápsulas que circulaban por los cabezales giratorios. Se ha utilizado un método de filtrado por color el cual nos permite paliar con el problema típico de los sistemas de visión artificial el cual es una mala iluminación. Mediante el empleo del espacio de color HSV y la creación de una interfaz gráfica para ajustar los parámetros del filtro por color y el algoritmo de la transformada del círculo de hough se ha permitido el ajuste dinámico del sistema. Este punto es realmente importante ya que permite que alguien con conocimiento casi nulo de programación pueda usar el sistema implementado.

Se han diseñado y fabricado dos cabezales giratorios los cuales trasladan las cápsulas hasta el robot y realizan el cambio de caja cuando esta está llena. Estos cabezales nos permiten una alta velocidad de procesamiento ya que simplemente rotando el motor unos grados alimentamos una cápsula mientras que un cinta tiene que rotar varias veces el motor. El tiempo de ciclo se ha visto reducido grácias a esta mejora.

Los sistemas anteriores han sido integrados de forma que actúan coordinados en el empaquetado de las cápsulas de café en cajas del mismo tipo de cápsula. Se ha optado por las comunicaciones seriales entre arduino y programa de Python por su fiabilidad aunque también se podrían haber

empleado comunicaciones wifi o bluetooth. En el caso concreto del sistema diseñado emplear una comunicación inalámbrica no es de vital importancia ya que el dispositivo que ejecuta el programa está ubicado junto al robot. El tiempo de ciclo logrado está ligeramente por encima de los 6 minutos lo cual es admisible.

Personalmente este proyecto ha supuesto un enriquecimiento personal por los conocimientos que he adquirido en su desarrollo en los campos que más me apasionan.

1.10. Bibliografía

Documentación Python:

- (3) OpenCV Course - Full Tutorial with Python - YouTube
- GUI con Tkinter y OpenCV en Python | Videos ? » omes-va.com
- PySerial Python Arduino comunicación serial - HETPRO/TUTORIALES (hetpro-store.com)
- Python and Arduino Serial, decoding issue - Stack Overflow
- tkinter — Python interface to Tcl/Tk — Python 3.10.5 documentation
- python - Tkinter Scale slider with float values doesn't work with locale of language that uses comma for floats - Stack Overflow
- can't send string from python to arduino - Stack Overflow

Documentación arduino:

- ▷ Arduino Mega 2560 Características, Especificaciones | Proyecto Arduino
- Python TO Serial TO Arduino. Send a string? - Using Arduino / Interfacing w/ Software on the Computer - Arduino Forum
- Stepper Motor with DRV8825 and Arduino Tutorial (4 Examples) (makerguides.com)
- PySerial Python Arduino comunicación serial - HETPRO/TUTORIALES (hetpro-store.com)

Métodos de empaquetado:

- Encajadora pick and place - AP Series - SYNERLINK - wraparound / automática / para botellas (directindustry.es)
- Empacadora de cajas | MF TECNO
- Equipos y soluciones para envasado de café en grano, molido y en cápsulas (imco.es)

Fundamentos de robótica:

- PoliformaT : SR-GIEA : Recursos (upv.es)
- Robot Scara. Aplicaciones, fabricantes y ejemplos de procesos (revistaderobots.com)
- What is SCARA Robot? | ATO.com
- Cinemática del robot SCARA (RRP) - Kinematics & Dynamics (kinematicsanddynamics.com)

2. Planos

Los planos de la celda robotizada con visión artificial se incluyen en el apartado 5.5 Planos del Anexo y siguen la siguiente estructura:

- **Plano Nº 1.** Ensamblaje celda robotizada (Hoja 1/39)
- **Plano Nº 2.** Ensamblaje robot scara (Hoja 2/39)
- **Plano Nº 3.** Ensamblaje cámara(Hoja 3/39)
- **Plano Nº 4.** Ensamblaje cabezal giratorio derecho (Hoja 4/39)
- **Plano Nº 5.** Ensamblaje cabezal giratorio izquierdo(Hoja 5/39)
- **Plano Nº 6.** Ensamblaje base (Hoja 6/39)
- **Plano Nº 7.** Ensamblaje electrónica (Hoja 7/39)
- **Plano Nº 8.** Ensamblaje eje 1 (Hoja 8/39)
- **Plano Nº 9.** Ensamblaje eje 2 (Hoja 9/39)
- **Plano Nº 10.** Ensamblaje eje 3 (Hoja 10/39)
- **Plano Nº 11.** Perfil estructural aluminio 30x30 L-240mm taladrado (Hoja 11/39)
- **Plano Nº 12.** Perfil estructural aluminio 30x30 L-240mm (Hoja 12/39)
- **Plano Nº 13.** Pletina inferior base (Hoja 13/39)
- **Plano Nº 14.** Pletina superior base (Hoja 14/39)
- **Plano Nº 15.** Carril DIN L-300mm (Hoja 15/39)
- **Plano Nº 16.** Carril DIN L-64mm (Hoja 16/39)
- **Plano Nº 17.** Acoplamiento chaveta (Hoja 17/39)
- **Plano Nº 18.** Pletina eje 1 a 2 (Hoja 18/39)
- **Plano Nº 19.** Soporte motor en U (Hoja 19/39)
- **Plano Nº 20.** Soporte motor en L (Hoja 20/39)
- **Plano Nº 21.** Eje corto eje 2 (Hoja 21/39)
- **Plano Nº 22.** Suplemento eje 2 (Hoja 22/39)
- **Plano Nº 23.** Cierre superior eje 2 (Hoja 23/39)
- **Plano Nº 24.** Cierre inferior eje 2 (Hoja 24/39)
- **Plano Nº 25.** Pletina eje 2 a 3 (Hoja 25/39)

- **Plano Nº 26.** Guia eje Z (Hoja 26/39)
- **Plano Nº 27.** Casquillo eje Z (Hoja 27/39)
- **Plano Nº 28.** Sujección husillo eje Z (Hoja 28/39)
- **Plano Nº 29.** Portaherramientas (Hoja 29/39)
- **Plano Nº 30.** Perfil en U taladrado (Hoja 30/39)
- **Plano Nº 31.** Escuadra (Hoja 31/39)
- **Plano Nº 32.** Placa cámara (Hoja 32/39)
- **Plano Nº 33.** Perfil estructural 30x30 L-200mm (Hoja 33/39)
- **Plano Nº 34.** Plato porta cápsulas (Hoja 34/39)
- **Plano Nº 35.** Plato porta cajas (Hoja 35/39)
- **Plano Nº 36.** Pletina soporte motor (Hoja 36/39)
- **Plano Nº 37.** Ensamblaje soporte caja (Hoja 37/39)
- **Plano Nº 38.** Acoplamiento plato(Hoja 38/39)
- **Plano Nº 39.** Suplemento eje Z (Hoja 39/39)

3. Pliego de condiciones

3.1. Objeto

El pliego de condiciones de este proyecto tiene la finalidad de describir el ensamblaje de los distintos componentes que conforman la celda robotizada con visión artificial. Para ello se va a subdividir la celda en sus tres subensamblajes, el robot scara, los cabezales rotatorios y el sistema de visión. Por último se pretende indicar las condiciones de la integración de dichos componentes.

3.2. Condiciones de los materiales

3.2.1. Robot scara

3.2.1.1. Estructura base

La estructura que conforma la base del robot y sirve de soporte para los demás subensamblajes debe estar fabricada con perfil estructural de 30x30. Las dos pletinas, la superior y la inferior, las cuales mantienen las dos estructuras rectangulares unidas deben estar fabricadas con acero de al menos 4 milímetros de espesor. Los carriles DIN deben estar normalizados conforme a la norma que le otorga su nombre DIN. La caja metálica que actúa como cuadro eléctrico debe tener un soporte para colocarse sobre el carril DIN. Las patas de goma deben tener al menos 20 mm de altura.

3.2.1.2. Componentes electrónicos

El arduino mega 2560 debe adquirirse en un distribuidor oficial. Este debe ser alimentado a 12Vdc a través de la Ramps 1.4. La Ramps 1.4 se debe adquirir del fabricante Reprap. Esta se debe alimentar con 12Vdc. Los drivers DRV8825 deben emplearse con el disipador de calor que incluyen. El ventilador centrífugo EC5015M12 debe alimentarse a 12 Vdc.

El motor NEMA 17 17HS5412-3 debe tener un torque de 48 Ncm. El motor NEMA 42BYGHW609 debe tener un torque de 40 Ncm. El driver SANMOTION F2 BS1D200P10 debe alimentarse a 24Vdc. El motor NEMA 23 103H7823-5740 debe tener un torque de 2.7 Nm. La bomba de vacío TMJ-370CA-15330 debe alimentarse a 12Vdc desde la salida de la Ramps 1.4. Los disyuntores Siemens 5SX2 MAX 277 AC C15 deben ubicarse en la salida de las fuentes de alimentación. La fuente de alimentación de 24V MeanWell RSP-320-24 se debe alimentar con 230Vac. Su salida se debe conectar al disyuntor de 24Vdc. El transformador CWT PAA060F debe alimentarse con 230 Vac. Su salida debe conectarse al disyuntor de 12Vdc.

3.2.1.3. Componentes mecánicos

En el eje 1 del robot debe emplearse una reductora planetaria de relación 7:1. Debe emplearse el modelo 075-MF1-7-131-000 o en su defecto una de especificaciones similares. Debe fijarse a esta el acoplamiento con chaveta. La pletina del eje 1 al 2 debe tener una longitud entre centros de 200mm. El soporte del motor del eje 2 debe permitir tensar la correa de este eje. El conjunto del eje 2 debe permitir transferir el movimiento del motor con el menor rozamiento posible. El cierre superior e inferior del eje 2 deben comprimir las pistas interiores del rodamiento. Debe emplearse un rodamiento axial radial NSK. El motor del tercer eje debe emplear una correa para transmitir el movimiento. El husillo del tercer eje debe tener un paso de 10 mm. El soporte porta herramienta debe tener un peso inferior a 150 gramos. La herramienta debe ser una ventosa.

3.2.2. Sistema de visión artificial

El sistema de visión artificial debe incluir la cámara y el dispositivo de procesamiento siendo la iluminación opcional. La cámara debe ser la webcam Amdis de conceptronics o en su defecto una cámara con especificaciones similares. El dispositivo de procesamiento debe tener suficiente potencia de procesamiento para realizar análisis en tiempo real de video. La cámara debe fijarse en el soporte de cámara diseñado. El soporte debe estar fabricado en aluminio. Este soporte debe fijarse al soporte base del robot scara.

3.2.3. Cabezales giratorios

Los cabezales rotatorios deben estar fijados a la estructura del robot scara. Los soportes de estos deben estar fabricados con perfil estructural de 30x30. La pletina que soporta el motor debe estar hecha de acero. Los acoplamientos entre motor y plato deben estar hechos de aluminio. Los platos porta cápsulas y porta cajas deben estar hechos de madera contrachapada. Los soportes de las cajas deben estar hechos de madera contrachapada de 4 mm.

3.2.4. Control de calidad

Las medidas de la estructura base deben ser comprobadas tras su fabricación conforme a las establecidas en el plano. Las piezas de fabricación propia deben ser comprobadas tras su fabricación. Se debe comprobar que las medidas de la pieza se corresponden con las del plano con una exactitud de 0.05mm.

El correcto funcionamiento de los elementos electrónicos se debe comprobar de forma individual. Los motores deben conectarse a sus respectivos drivers y se debe comprobar el par generado para cumplir con las especificaciones. La capacidad de succión de la bomba de vacío debe comprobarse. El cumplimiento de las medidas de las piezas de los conjuntos mecánicos debe ser corroborada. Estas deben de complir con las características descritas.

Las piezas del soporte de la cámara se deben comprobar garantizando el cumplimiento de las medidas. La cámara se debe conectar para verificar su correcto funcionamiento. Se debe de comprobar que el dispositivo de procesamiento es capaz de procesar video.

Las piezas de los cabezales rotatorios deben cumplir con las medidas especificadas. Las piezas de madera no deben presentar ningún desperfecto.

3.3. Condiciones de ejecución

3.3.1. Ensamblaje del robot scara

El proceso de ensamblaje del robot scara debe realizarse siguiendo las siguientes operaciones:

- Se deben ensamblar las dos estructuras rectangulares formadas por cuatro perfiles estructurales de aluminio de 30x30 de longitud 240 mm (Plano Nº12) y cuatro perfiles estructurales de aluminio de 30x30 de longitud 240 mm los cuales tienen 2 agujeros para colocar conectores perpendiculares (Plano Nº11). Se deben apretar estos hasta tener una estructura sólida.

- Se deben conectar ambas estructuras rectangulares mediante las pletinas superior (Plano Nº14) e inferior (Plano Nº13). Estas se deben atornillar con tornillos M5 y tuercas T M5.

- Se deben fijar los dos carriles DIN de longitud 300 mm(Plano Nº15) con un tornillo M4 y tuerca T M4 en cada extremo del carril.

- Se debe instalar la fuente de 24V en la parte superior de la pletina inferior y se debe fijar con bridas a través de los agujeros de la pletina inferior. Entre la fuente y la pletina se debe ubicar una capa fina de goma eva.

- Se debe instalar el transformador de 12Vdc el cual se fijará con bridas.

- Se deben colocar los dos disyuntores en el carril DIN inferior.

- Se debe conectar el terminal positivo de la fuente de 24Vdc a un disyuntor.

- Se debe conectar el terminal positivo de la fuente de 12Vdc a el otro disyuntor.

- Se debe instalar la caja metálica en el carril DIN superior. (Emplear el plano Nº6 Ensamblaje base como referencia)

- Se debe fijar con un tornillo M4 y tuerca M4 el carril DIN de 64 mm (Plano Nº16) en el centro de la caja metálica.

- Se deben colocar dos terminales de carril DIN rojos y 5 negros en el carril DIN corto instalado.

- Se debe conectar la salida del disyuntor de 12 Vdc al terminal rojo de carril DIN.

- Se debe conectar el negativo del transformador de 12Vdc al terminal negro de carril DIN instalado en el carril corto.

- Se debe conectar el arduino mega 2560 con la Ramps 1.4 y los módulos DRV8825. Este conjunto se debe instalar en el lateral interior de la caja metálica mediante los pilones hexagonales de M2.5 y las tuercas M 2.5.

- El ventilador centrífugo se debe colocar en el carril DIN corto orientado hacia el arduino mega 2560.

- Se debe instalar en driver SANMOTION dentro de la caja metálica en el lateral opuesto al arduino.

- Se debe instalar el motor NEMA 23 103H7823-5740 en la cara inferior de la pletina superior de la base y se fijará con tornillos allen de M5 y tuerca M5. (Hasta este punto se corresponde con el plano Nº 7, ensamblaje electrónica)

- Se debe instalar la reductora planetaria mediante 4 tornillos de M6.

- Se debe apretar el tornillo prisionero que une el eje del motor NEMA 23 con el rotor de la reductora. (Plano Nº 8, ensamblaje eje 1)

- Se debe colocar el acoplamiento con chaveta (Plano Nº17) sobre la reductora planetaria y se debe fijar apretando el tornillo prisionero de M8.

- Se debe fijar la pletina de los ejes 1 y 2 (Plano Nº18) al acoplamiento con chaveta mediante 4 tornillos hexagonales M5.

- Se debe instalar el soporte en U (Plano Nº19) sobre la pletina de los ejes 1 a 2 mediante 2 tornillos de M4 y tuercas de M4.

- Se deben colocar en el motor NEMA 17 17HS5412-3 dos soportes de motor en L (Plano Nº20) con tornillos de M3.

- Se debe fijar una polea dentada de 20 dientes en el motor.

- Se deben fijar los dos soportes en L al soporte en U.

- Se debe fijar el cierre superior (Plano Nº23) del eje 2 y el inferior (Plano Nº24) comprimiendo el rodamiento radial axial NSK mediante un tornillo de M8.

- El conjunto de rodamiento comprimido se debe fijar a la parte inferior de la pletina del eje 2 al 3 (Plano Nº25) mediante los tornillos de M6x70 que atraviesan hasta el suplemento del eje 2 (Plano Nº22) colocado en la parte superior de la pletina del eje 2 al 3.

- Se debe colocar el eje corto en el eje 2 (Plano Nº21) y se debe fijar al cierre superior mediante un tornillo prisionero.

- Se debe instalar una polea dentada fija de 60 dientes en este eje. (Hasta este punto se corresponde con el plano Nº 9, ensamblaje del eje 2)

- Se debe colocar la correa 200-2GT entre las poleas de 60 dientes y la de 20 y se debe tensar mediante los agujeros rasgados del soporte motor.

- Se debe fijar la pletina del eje 2 al 3 (Plano Nº25) al cierre inferior del eje 2 mediante 4 tornillos hexagonales de M5.

- Sobre esta pletina se debe fijar la bomba de vacío con dos bridas.

- Se debe fijar el motor NEMA 17 42BYGHW609 mediante cuatro tornillos de M3

- Se debe colocar una polea dentada de 20 dientes fijada a su eje con un tornillo prisionero.

- Se debe pegar la tuerca igus ds8x10 al carril interior de los rodamientos SKF 6004 .

- Se debe pegar este conjunto en el suplemento del eje Z (Plano Nº39) el cual se debe fijar a la pletina del eje 2 a 3 mediante 2 tornillos de M4.

- Se debe atornillar el casquillo del eje Z (Plano Nº27) a la tuerca igus ds8x10. Se debe colocar una polea dentada de 20 dientes en el casquillo del eje Z.

- Se deben instalar las dos guías del eje Z (Plano Nº26) en este suplemento.

- Se debe colocar el husillo el cual se debe fijar a la sujeción (Plano Nº28) mediante un tornillo prisionero.

- En el extremo del husillo se debe fijar el portaherramientas (Plano Nº29) el cual se rosca sobre el husillo.

- Se debe instalar la ventosa en el portaherramientas.(Hasta este punto se corresponde con el plano Nº 10, ensamblaje eje 3)

- Se debe conectar la ventosa a la bomba de vacío.

- Se deben realizar las conexiones eléctricas especificadas en el apartado de componentes eléctricos del robot scara.

3.3.2. Ensamblaje del sistema de visión

El proceso de ensamblaje del sistema de visión debe realizarse siguiendo las siguientes operaciones:

- Se debe ensamblar el soporte de la cámara.

- Para ello se deben colocar las dos placas (Plano Nº32) atornilladas a los extremos de las escuadras (Plano Nº31) mediante tornillo allen M4 y tuerca M4.

- Este conjunto se debe fijar al perfil en U taladrado (Plano Nº30) mediante tornillo y tuerca de M4. (Plano Nº 3, ensamblaje cámara)

- El soporte de cámara se debe fijar al lateral izquierdo del soporte base del robot scara mediante tornillo allen y tuerca T M4. (Emplear como referencia el plano Nº1 , ensamblaje celda robotizada)

- Se debe colocar la cámara en el soporte y se debe fijar mediante una brida.

- Se debe conectar la cámara al ordenador

3.3.3. Ensamblaje de los cabezales giratorios

El proceso de ensamblaje de los cabezales giratorios debe realizarse siguiendo las siguientes operaciones:

- Deben fijarse los motores NEMA 17 17HS5412-3 a las pletinas soporte del motor (Plano Nº36) mediante 4 tornillos M3 .

- Estas pletinas deben fijarse al perfil estructural DIN 30x30 de longitud 200 mm (Plano Nº32) mediante tornillo y tuerca T M5.

- Se deben fijar los acoplamientos de los platos (Plano Nº39) a los motores mediante un tornillo prisionero M3.

- Se deben fijar los platos de madera (Plano Nº34 y 35) a los acoplamientos mediante 4 tornillos hexagonales y tuercas de M4. (Hasta este punto de corresponde con los planos Nº 4, ensamblaje cabezal derecho y Nº5, ensamblaje cabezal izquierdo)

- Se debe fijar el cabezal con el plato porta cápsulas en la parte izquierda de la estructura base del robot scara mediante un conector angular. (Emplear como referencia el plano Nº1 , ensamblaje celda robotizada)

- Se debe fijar el cabezal con el plato porta cajas en la parte derecha de la estructura base del robot scara mediante un conector angular. (Emplear como referencia el plano Nº1 , ensamblaje celda robotizada)

3.3.4. Control de calidad

Tras realizar el montaje de toda la instalación se debe realizar un chequeo del correcto ensamblaje de los componentes. Los tornillos y tuercas se revisar para asegurar que estén correctamente apretadas. De no ser así se procederá a apretarlos. Se debe comprobar que los subensamblajes ocupan las posiciones especificadas en los planos. Estos subensamblajes se revisarán para comprobar que no haya ningún desperfecto ocasionado por el montaje de los mismos.

Una vez realizadas las comprobaciones anteriores se procederá a comprobar el correcto funcionamiento del brazo robot scara cargando el el arduino el software para su control. Posteriormente se hará funcionar los cabezales rotatorios comprobando que los motores roten sin problema. Finalmente se chequeará el funcionamiento del sistema de visión artificial.

3.4. Pruebas y ajustes finales

La puesta en marcha del sistema debe ser realizada con el objetivo de sincronizar a la perfección la instalación ensamblada. Esta puesta en marcha consiste en el ajuste de las velocidades de funcionamiento del robot y los cabezales rotatorios con la intención de lograr el menor tiempo de ciclo posible. Se debe tomar como referencia las pruebas realizadas en el apartado de resultados de la memória los cuales establecen un tiempo de ciclo de 6min y 40 segundos. Con este tiempo en mente se debe realizar el ajuste de la celda.

El sistema de visión se debe ajustar mediante el panel de ajuste de cámara el cual permite modificar los filtros de color y parámetros del algortimo de los circulos de hough. Este ajuste debe realizarse con la mayor exactitud posible para lograr evitar la detección de falsos positivos.

Finalmente tras el ajuste del sistema de visión artificial y de las velocidades de robot y cabezales se deben realizar 5 ciclos de empaquetado continuo en los cuales se debe de lograr un 100% de eficacia. De no ser así se solucionará el problema existente hasta lograr dicha cifra.

4. Presupuesto

4.1. Precios unitarios

1. Precios Unitarios			
1.1 Materiales			
Referencia	**Unidades**	**Descripción**	**Coste (€)**
m1	u.	Perfil estructural aluminio 30x30 L-240mm	4.00
m2	u.	Perfil estructural aluminio 30x30 L-240mm taladrado	6.00
m3	u.	Pletina superior base	5.20
m4	u.	Pletina inferior base	5.40
m5	u.	Pata de goma	0.50
m6	u.	Carril DIN L-300mm	5.00
m7	u.	Carril DIN L-64mm	1.50
m8	u.	Caja de aluminio	45.00
m9	u.	Arduino Mega 2560	25.00
m10	m.	Ramps 1.4	11.00
m11	m.	Driver DRV8825	2.80
m12	u.	Ventilador centrífugo EC5015M12S	8.50
m13	u.	Driver SANMOTION F2 BS1D200P10	218.00
m14	u.	Siemens circuit breaker 5SX2 MAX 277 AC C15	18.50
m15	u.	MeanWell RSP-320-24 fuente de alimentación 24V	52.00
m16	u.	Transformador CWT PAA060F	17.30
m17	u.	Cable de alimentación	10.00
m18	u.	Terminal de carril DIN rojo	0.85
m19	u.	Terminal de carril DIN negro	0.85
m20	u.	Motor NEMA 23 103H7823-5740	104.00

m21	u.	Reductor Planetario 075-MF1-7-131-000	472.00
m22	u.	Acoplamiento con chaveta	25.00
m23	u.	Pletina eje 1 a 2	4.30
m24	u.	Soporte motor en U	3.60
m25	u.	Soporte motor en L	1.60
m26	u.	Motor NEMA 17 17HS5412-3	14.60
m27	u.	Polea dentada 20 dientes	3.00
m28	u.	Polea dentada 60 dientes	6.00
m29	u.	Eje corto eje 2	2.47
m30	u.	Suplemento eje 2	12.00
m31	u.	Cierre superior eje 2	11.50
m32	u.	Cierre inferior eje 2	10.80
m33	u.	Rodamiento axial radial NSK	280.00
m34	u.	Pletina eje 2 a 3	4.20
m35	u.	Bomba de vacio TMJ-370CA-15330	2.26
m36	u.	Motor NEMA 17 42BYGHW609	29.00
m37	u.	Ventosa	7.50
m38	u.	Guia eje Z	6.50
m39	u.	Sujección husillo Eje Z	8.20
m40	u.	Husillo igus DS8x10	40.00
m41	u.	Tuerca igus DS8x10	15.30
m42	u.	Rodamiento SKF 6004	5.00
m43	u.	Casquillo eje Z	11.80
m44	u.	Portaherramientas	7.00
m45	u.	Suplemento eje Z	12.3
m46	u.	Tornillo allen M4	0.20
m47	u.	Tuerca T M4	0.35
m48	u.	Tornillo allen M5	0.47
m49	u.	Tuerca T M5	0.53
m50	u.	Tornillo M2.5	0.20
m51	u.	Pilon hexagonal M2.5	0.40

m52	u.	Tuerca M2.5	0.06
m53	u.	Tornillo allen M6	0.56
m54	u.	Tornillo avellanado M5	0.55
m55	u.	Tornillo sin cabeza M8	0.42
m56	u.	Tornillo hexagonal M5	0.48
m57	u.	Tuerca M4	0.12
m58	u.	Tuerca M5	0.16
m59	u.	Tornillo allen M6x70	1.50
m60	u.	Tuerca M6	0.20
m61	u.	Tornillo allen M3	0.34
m62	u.	Correa 200-2GT	0.36
m63	u.	Rollo de Cables	20.00
m64	u.	Tubo de goma	15.00
m65	u.	Pack de bridas	6.65
m66	u.	Webcam Amdis Conceptronics	15.00
m67	u.	Portatil MSI CX6QD	700.00
m68	u.	Perfil en U taladrado	5.00
m69	u.	Placa cámara	2.50
m70	u.	Escuadra	2.00
m71	u.	Tornillo cabeza redonda M4	0.42
m72	u.	Perfil estructural aluminio 30x30 L-200 mm taladrado	5.30
m73	u.	Pletina soporte motor	1.20
m74	u.	Acoplamiento plato	7.50
m75	u.	Plato porta cápsulas	9.00
m76	u.	Plato porta cajas	8.00
m77	u.	Soporte Caja	3.50
m78	u.	Escuadra aluminio	2.50
m79	u.	Arandelas M4	0.03
m80	u.	Tornillo Hexagonal M4	0.39
m81	u.	Conector perpendicular	2.40

1.2 Mano de obra			
Referencia	**Unidades**	**Descripción**	**Coste (€)**
h1	h	Ingeniero Junior	20.00
h2	h	Técnico de programación de robots y visión	15.00

Tabla 1. Precios unitarios del proyecto

4.2. Desglose de precios unitarios

2. Desglose de precios unitarios					
Referencia	**Unidades**	**Descripción**	**Coste (€)**	**Cantidad**	**Total**
d1	**u.**	**Robot Scara**			
Materiales					
m1	u.	Perfil estructural aluminio 30x30 L-240mm	4.00	4	16
m2	u.	Perfil estructural aluminio 30x30 L-240mm taladrado	6.00	4	24
m3	u.	Pletina superior base	5.20	1	5.2
m4	u.	Pletina inferior base	5.40	1	5.4
m5	u.	Pata de goma	0.50	4	2
m6	u.	Carril DIN L-300mm	5.00	2	10
m7	u.	Carril DIN L-64mm	1.50	1	1.5
m8	u.	Caja de aluminio	45.00	1	45
m9	u.	Arduino Mega 2560	25.00	1	25
m10	m.	Ramps 1.4	11.00	1	11
m11	m.	Driver DRV8825	2.80	1	2.8
m12	u.	Ventilador centrífugo EC5015M12S	8.50	1	8.5
m13	u.	Driver SANMOTION F2 BS1D200P10	218.00	1	218

m14	u.	Siemens circuit breaker 5SX2 MAX 277 AC C15	18.50	2	37
m15	u.	MeanWell RSP-320-24 fuente de alimentación 24V	52.00	1	52
m16	u.	Transformador CWT PAA060F	17.30	1	17.3
m17	u.	Cable de alimentación	10.00	1	10
m18	u.	Terminal de carril DIN rojo	0.85	3	2.55
m19	u.	Terminal de carril DIN negro	0.85	5	4.25
m20	u.	Motor NEMA 23 103H7823-5740	104.00	1	104
m21	u.	Reductor Planetario 075-MF1-7-131-000	472.00	1	472
m22	u.	Acoplamiento con chaveta	25.00	1	25
m23	u.	Pletina eje 1 a 2	4.30	1	4.3
m24	u.	Soporte motor en U	3.60	1	3.6
m25	u.	Soporte motor en L	1.60	2	3.2
m26	u.	Motor NEMA 17 17HS5412-3	14.60	1	14.6
m27	u.	Polea dentada 20 dientes	3.00	3	9
m28	u.	Polea dentada 60 dientes	6.00	1	6
m29	u.	Eje corto eje 2	2.47	1	2.47
m30	u.	Suplemento eje 2	12.00	1	12
m31	u.	Cierre superior eje 2	11.50	1	11.5
m32	u.	Cierre inferior eje 2	10.80	1	10.8
m33	u.	Rodamiento axial radial NSK	280.00	1	280
m34	u.	Pletina eje 2 a 3	4.20	1	4.2
m35	u.	Bomba de vacio TMJ-370CA-15330	2.26	1	2.26
m36	u.	Motor NEMA 17 42BYGHW609	29.00	1	29
m37	u.	Ventosa	7.50	1	7.5
m38	u.	Guia eje Z	6.50	2	13
m39	u.	Sujección husillo Eje Z	8.20	1	8.2
m40	u.	Husillo igus DS8x10	40.00	1	40

m41	u.	Tuerca igus DS8x10	15.30	1	15.3
m45	u.	Suplemento eje Z	12.3	1	12.3
m42	u.	Rodamiento SKF 6004	5.00	2	10
m43	u.	Casquillo eje Z	11.80	1	11.8
m44	u.	Portaherramientas	7.00	1	7
m81	u.	Conector perpendicular	2.40	8	19.2
m46	u.	Tornillo allen M4	0.20	6	1.2
m47	u.	Tuerca T M4	0.35	4	1.4
m48	u.	Tornillo allen M5	0.47	8	3.76
m49	u.	Tuerca T M5	0.53	8	4.24
m50	u.	Tornillo M2.5	0.20	2	0.4
m51	u.	Pilon hexagonal M2.5	0.40	2	0.8
m52	u.	Tuerca M2.5	0.06	2	0.12
m53	u.	Tornillo allen M6	0.56	4	2.24
m54	u.	Tornillo avellanado M5	0.55	4	2.2
m55	u.	Tornillo sin cabeza M8	0.42	1	0.42
m56	u.	Tornillo hexagonal M5	0.48	8	3.84
m57	u.	Tuerca M4	0.12	2	0.24
m58	u.	Tuerca M5	0.16	2	0.32
m59	u.	Tornillo allen M6x70	1.50	4	6
m60	u.	Tuerca M6	0.20	4	0.8
m61	u.	Tornillo allen M3	0.34	2	0.68
m62	u.	Correa 200-2GT	0.36	2	0.72
m63	u.	Rollo de cables	20.00	1	20
m64	u.	Tubo de goma	15.00	1	15
m65	u.	Pack de bridas	6.65	1	6.65
Mano de obra					
h1	h	Ingeniero Junior	20.00	120	2400
h2	h	Técnico de programación de robots	15.00	80	1200
			Total		**5306.76**

Gastos generales de fabricación					
	%	Costes de producción sobre el coste directo	10.00	5306.76	530.68
Beneficios industriales (5-10%)					
	%	Beneficio industrial	7.00	5306.76	371.47
IVA (21%)					
	%	IVA sobre el coste directo	21.00	5306.76	1114.42
			Coste de ejecución total		7323.33

Tabla 2. Desglose de precios unitarios del robot scara

Referencia	Unidades	Descripción	Coste (€)	Cantidad	Total
d2	**u.**	**Sistema de visión**			
Materiales					
m66	u.	Webcam Amdis Conceptronics	15.00	1	15
m67	u.	Portatil MSI CX6QD	700.00	1	700
m68	u.	Perfil en U taladrado	5.00	1	5
m69	u.	Placa cámara	2.50	2	5
m70	u.	Escuadra	2.00	2	4
m46	u.	Tornillo allen M4	0.20	1	0.2
m47	u.	Tuerca T M4	0.35	1	0.35
m71	u.	Tornillo cabeza redonda M4	0.42	4	1.68
m57	u.	Tuerca M4	0.12	4	0.48
Mano de obra					
h1	h	Ingeniero Junior	20.00	24	480
h2	h	Técnico de programación de visión	15.00	40	600
			Total		**1811.71**
Gastos generales de fabricación					
	%	Costes de producción sobre el coste directo	10.00	1811.71	181.17
Beneficios industriales (5-10%)					
	%	Beneficio industrial	7.00	1811.71	126.82

IVA (21%)					
	%	IVA sobre el coste directo	21.00	1811.71	380.46
			Coste total de ejecución		2500.16

Tabla 3. Desglose de precios unitarios del sistema de visión

Referencia	Unidades	Descripción	Coste (€)	Cantidad	Total
d3	**u.**	**Cabezales giratorios**			
Materiales					
m72	u.	Perfil estructural 30x30 L-200mm	5.30	2	10.6
m73	u.	Pletina soporte motor	1.20	2	2.4
m74	u.	Acoplamiento plato	7.50	2	15
m75	u.	Plato porta cápsulas	9.00	1	9
m76	u.	Plato porta cajas	8.00	1	8
m77	u.	Soporte Caja	3.50	2	7
m78	u.	Pletina aluminio	2.50	8	20
m61	u.	Tornillo allen M3	0.34	4	1.36
m79	u.	Arandelas M4	0.03	20	0.62
m26	u.	Motor NEMA 17 17HS5412-3	14.60	2	29.2
m48	u.	Tornillo allen M5	0.47	2	0.94
m49	u.	Tuerca T M5	0.53	2	1.06
m81	u.	Conector perpendicular	2.40	2	4.8
m80	u.	Tornillo Hexagonal M4	0.39	8	3.12
m57	u.	Tuerca M4	0.12	8	0.96
Mano de obra					
h1	h	Ingeniero Junior	20.00	16	320
			Total		**434.06**
Gastos generales de fabricación					
	%	Costes de producción sobre el coste directo	10.00	434.06	43.41

Beneficios industriales (5-10%)					
	%	Beneficio industrial	7.00	434.06	30.38
IVA (21%)					
	%	IVA sobre el coste directo	21.00	434.06	91.15
			Coste total de ejecución		599.00

Tabla 4. Desglose de precios unitarios de los cabezales rotatorios

4.3. Cantidades

3. Cantidades			
Reference	**Unit**	**Description**	**Quantity**
d1	u.	Robot Scara	1.00
d2	u.	Sistema de visión	1.00
d3	u.	Cabezales giratorios	1.00

Tabla 5. Tabla de cantidades de cada elemento

4.4. Coste total

4. Coste total					
Reference	**Unit**	**Description**	**Cost (€)**	**Quantity**	**Total**
d1	u.	Robot Scara	7323.33	1	7323.33
d2	u.	Sistema de visión	2500.16	1	2500.16
d3	u.	Cabezales giratorios	607.28	1	607.28
			Coste total de proyecto		10422.49

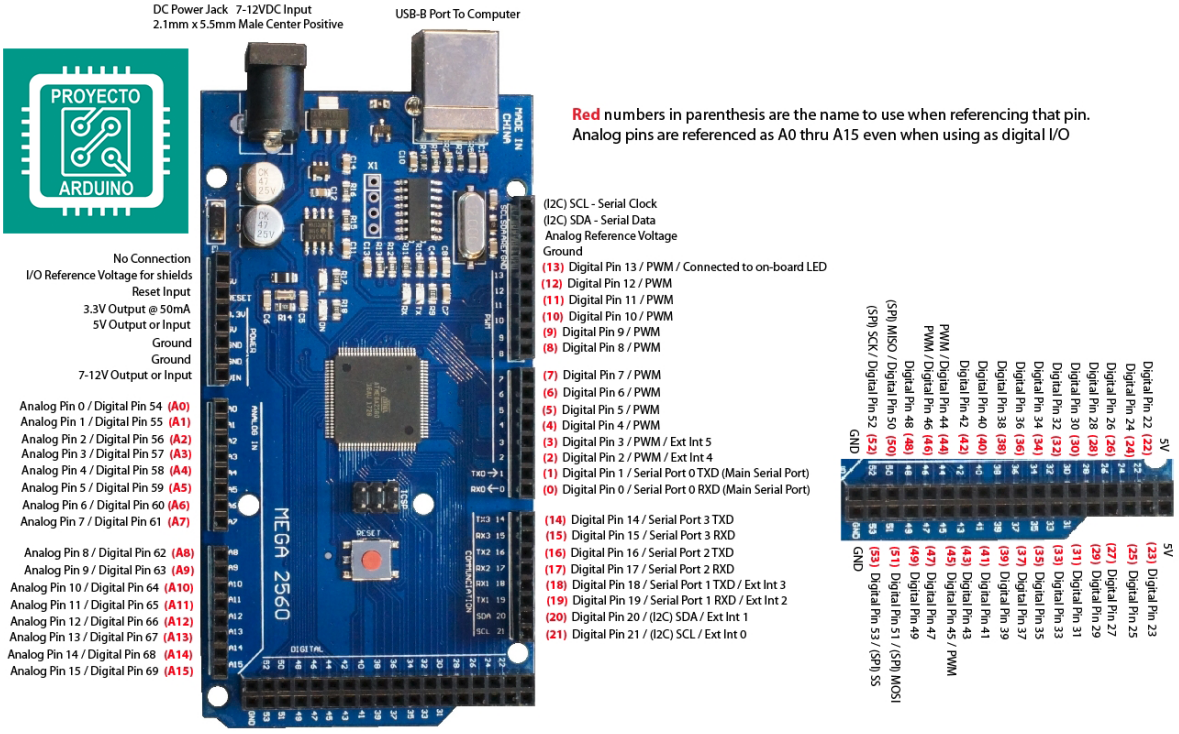
Tabla 6. Coste total del proyecto realizado

5. Anexos

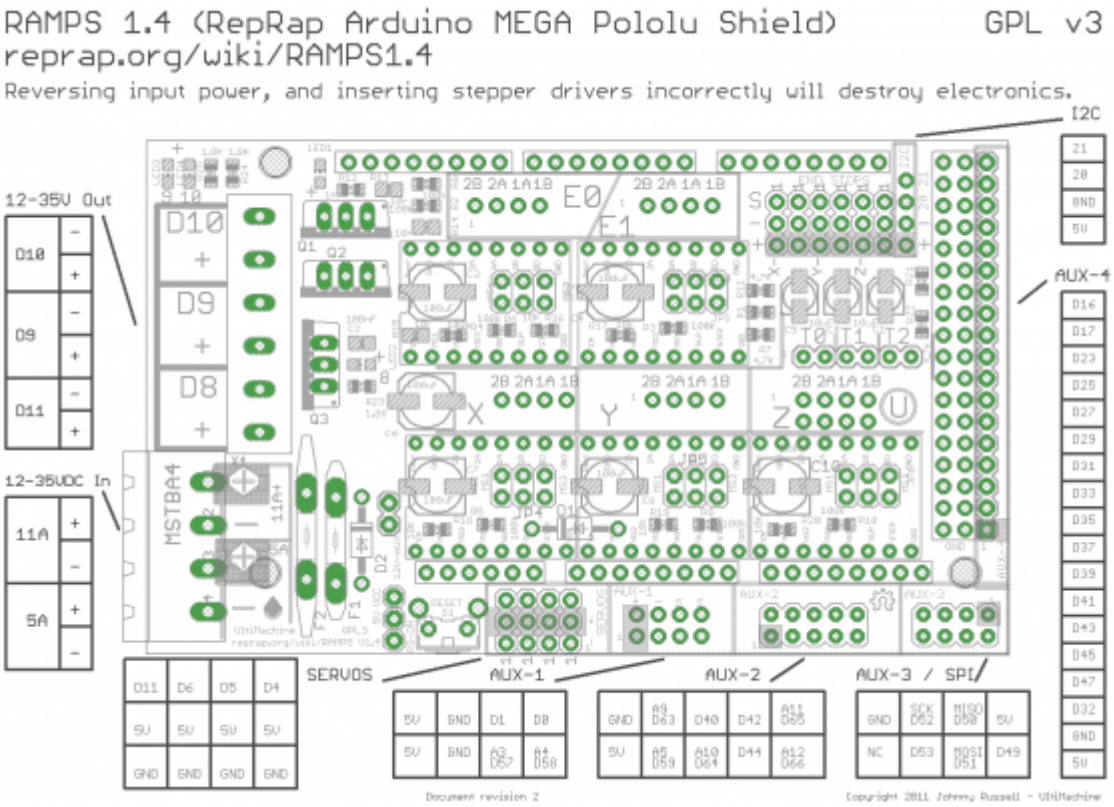
5.1. Robot Scara

5.1.1. Datasheets

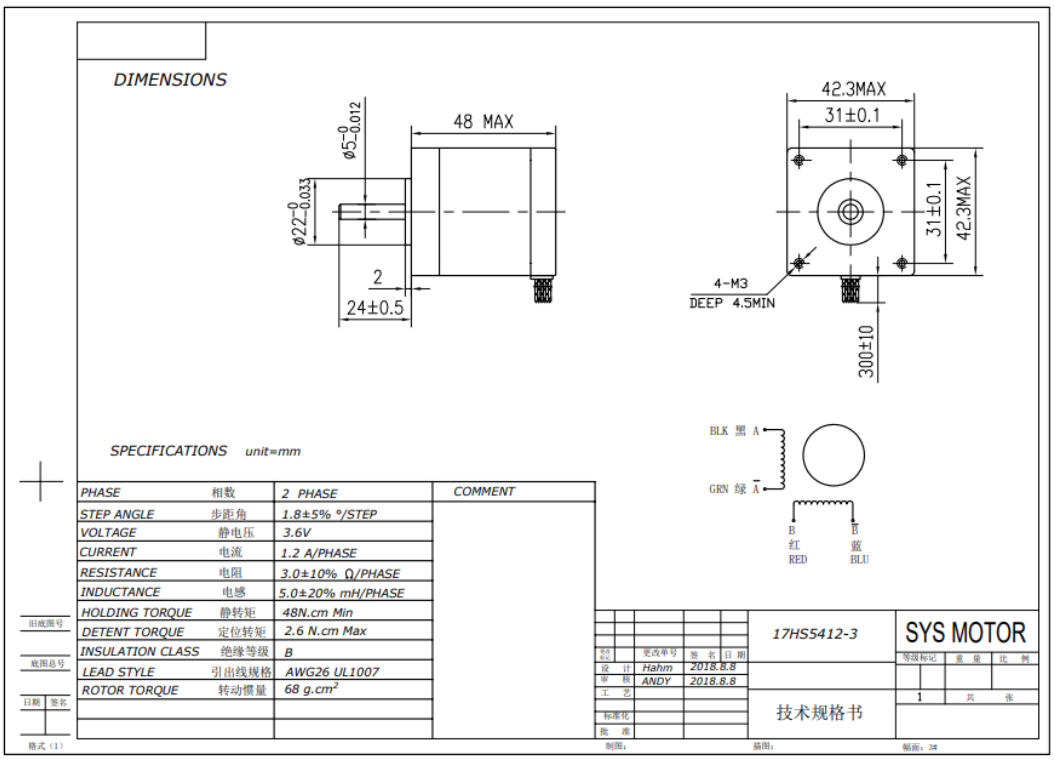
5.1.1.1. Arduino Mega 2560



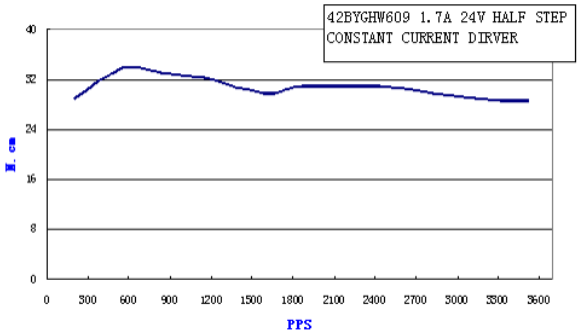
5.1.1.2. Ramps 1.4



5.1.1.3. NEMA 17 17HS5412-3

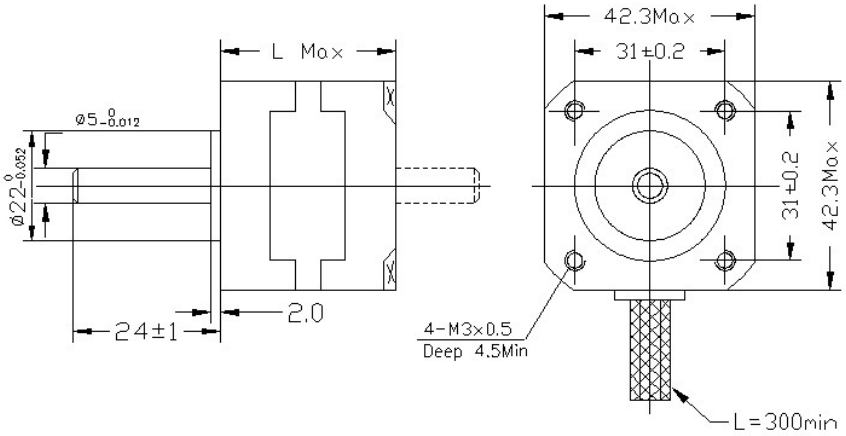


5.1.1.4.NEMA 17 42BYGHW609

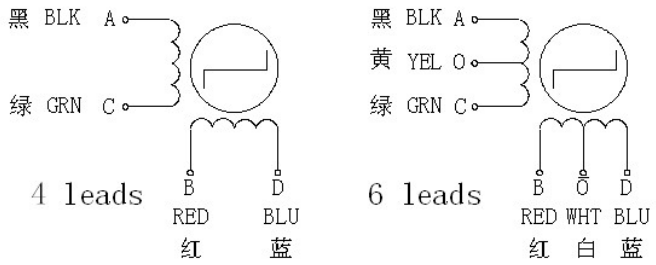


Model	Step Angle (°)	Motor Length L(mm)	Rate Voltage (V)	Rate Current (A)	Phase Resistance (Ω)	Phase Inductance (mH)	Holding Torque (g.cm)	Lead Wire (NO.)	Rotor Inertia (g. cm2)	Detent Torque (g.cm)	Motor Weight (kg)
42BYGHW609	1.8	40	3.4	1.7	2	3	4000	4	54	220	0.24

Mechanical Dimensions(外型图)

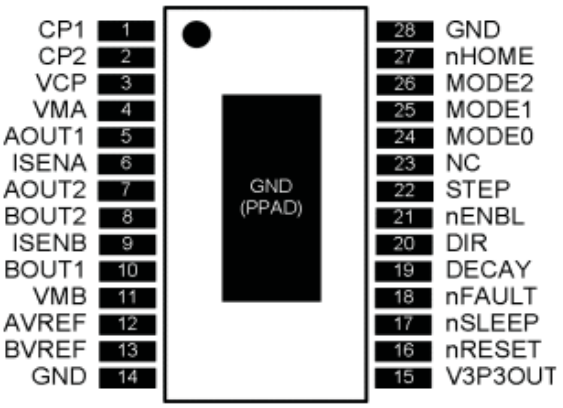


Wiring diagram (接线图):



5.1.1.5.DRV8825

Pin Configuration and Functions



Pin Functions

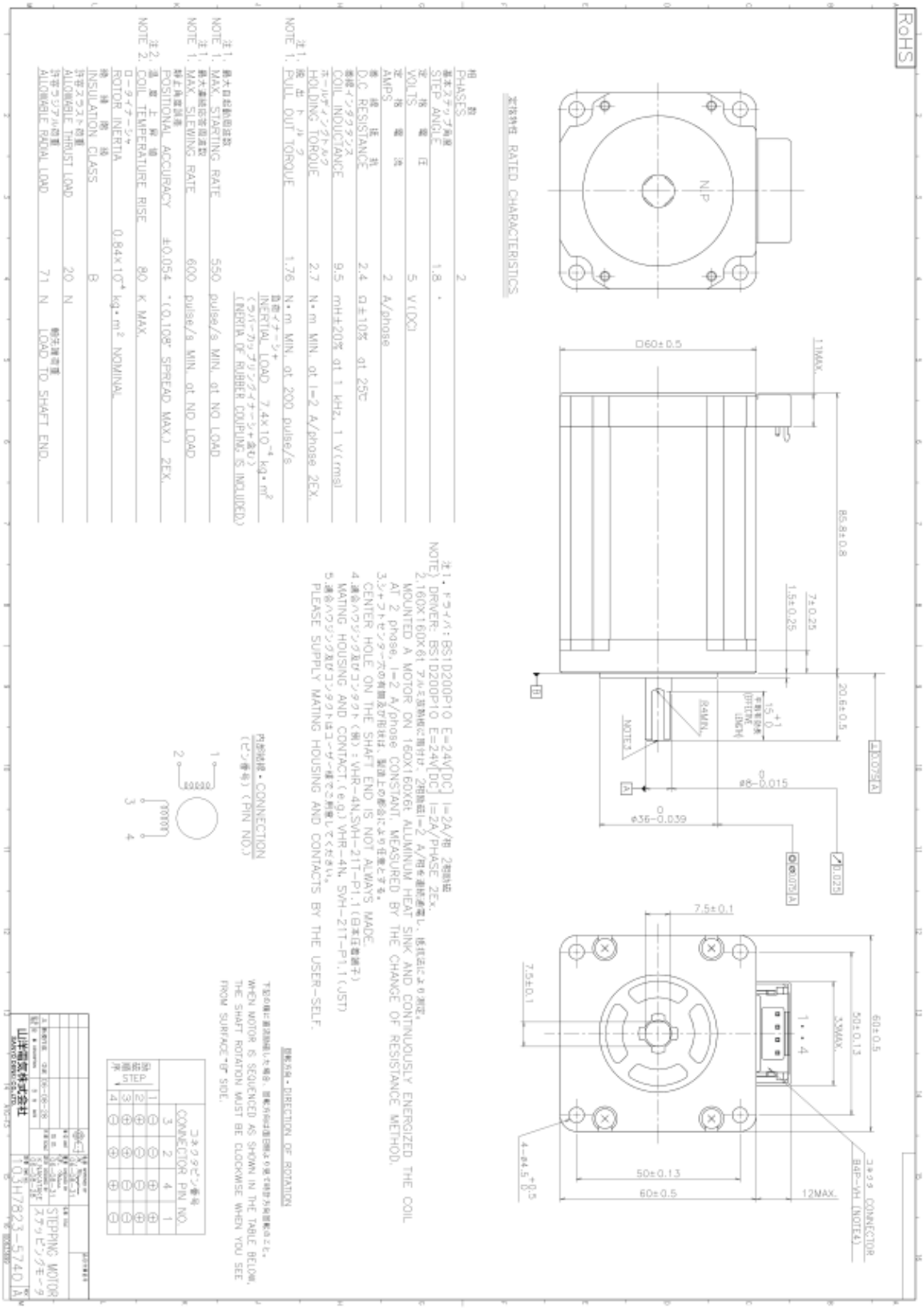
PIN NAME	PIN NO.	I/O(1)	DESCRIPTION	EXTERNAL COMPONENTS OR CONNECTIONS
POWER AND GROUND				
CP1	1	I/O	Charge pump flying capacitor	Connect a 0.01-µF 50-V capacitor between CP1 and CP2.
CP2	2	I/O	Charge pump flying capacitor	
GND	14, 28	—	Device ground	
VCP	3	I/O	High-side gate drive voltage	Connect a 0.1-µF 16-V ceramic capacitor and a 1-MΩ resistor to VM.
VMA	4	—	Bridge A power supply	Connect to motor supply (8.2 to 45 V). Both pins must be connected to the same supply, bypassed with a 0.1-µF capacitor to GND, and connected to appropriate bulk capacitance.
VMB	11	—	Bridge B power supply	
V3P3OUT	15	O	3.3-V regulator output	Bypass to GND with a 0.47-µF 6.3-V ceramic capacitor. Can be used to supply VREF.
CONTROL				
AVREF	12	I	Bridge A current set reference input	Reference voltage for winding current set. Normally AVREF and BVREF are connected to the same voltage. Can be connected to V3P3OUT.
BVREF	13	I	Bridge B current set reference input	
DECAY	19	I	Decay mode	Low = slow decay, open = mixed decay, high = fast decay. Internal pulldown and pullup.
DIR	20	I	Direction input	Level sets the direction of stepping. Internal pulldown.
MODE0	24	I	Microstep mode 0	MODE0 through MODE2 set the step mode - full, 1/2, 1/4, 1/8/ 1/16, or 1/32 step. Internal pulldown.
MODE1	25	I	Microstep mode 1	
MODE2	26	I	Microstep mode 2	
NC	23	—	No connect	Leave this pin unconnected.
nENBL	21	I	Enable input	Logic high to disable device outputs and indexer operation, logic low to enable. Internal pulldown.
nRESET	16	I	Reset input	Active-low reset input initializes the indexer logic and disables the H-bridge outputs. Internal pulldown.
nSLEEP	17	I	Sleep mode input	Logic high to enable device, logic low to enter low-power sleep mode. Internal pulldown.
STEP	22	I	Step input	Rising edge causes the indexer to move one step. Internal pulldown.
STATUS				
nFAULT	18	OD	Fault	Logic low when in fault condition (overtemp, overcurrent)

(1) Directions: I = input, O = output, OD = open-drain output, IO = input/output

5.1.1.6. Ventilador centrífugo EC5015M12S

Tipo De Parte	VENTILADOR
Modelo	EC5015M12S
Marca	
Tamaño	50 mm x 50 mm x 15 mm
Longitud	50 mm
Anchura	50 mm
altura	15 mm
voltaje	12V
Corriente	0.15A
Cable	2

5.1.1.7.NEMA23 103H7823-5740



5.1.1.8.CWT PAA060F

CHANNEL WELL TECHNOLOGY

PAA Series

AC-DC Adaptor, 30W-60W

Features

- Single, 30W - 60W, Output
- AC Universal Input, 90Vac - 264Vac
- IEC 320 C6 or C14 Input Receptacle
- Build in CV, CC, CP mode for charger
- Power Saving meet Blue Angle Specs
- High MTBF and High Efficiency design
- OVP, OPP (Auto Recovery), and Short
- Portable, Compact Size with Low Profile
- Multiple Choice of Output Plug and Cable
- Operating Temperature Range 0°C to 40°C
- UL/cUL, TUV, CCC, CB, and more certified

Electrical Specifications

AC Input

- Input Voltage 90 – 264 VAC
- Input Current 2.5A max.
- Input Frequency 47 – 63 Hz
- Efficiency 70/80% @ full load, nominal line
- Inrush Current 80A max @ cold start
- Leakage Current 0.75mA max. @ 240Vac input

DC Output – 30W-60W Max

Model Number	Power	Voltage	Current	Ripple/Noise	Regulation
PAA030B	30W	5V	6.00A	1%	< 3%
PAA030C	30W	6V	5.00A	1%	< 3%
PAA045E	45W	9V	5.00A	1%	< 3%
PAA040F	40W	12V	3.33A	1%	< 3%
PAA050F	50W	12V	4.16A	1%	< 3%
PAA060F	60W	12V	5.00A	1%	< 3%
PAA060H	60W	15V	4.00A	1%	< 3%
PAA060I	60W	16V	3.75A	1%	< 3%
PAA060J	60W	18V	3.33A	1%	< 3%
PAA060K	60W	19V	3.16A	1%	< 3%
PAA060L	60W	2-V	3.00A	1%	< 3%
PAA060M	60W	24V	2.50A	1%	< 3%
PAA060P	60W	30V	2.00A	1%	< 3%

Mechanical Dimension

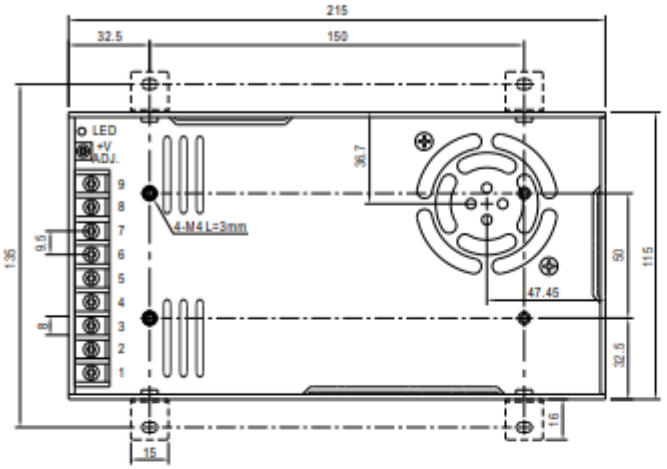
- W x L x H = 65 x 108 x 31 mm / 2.56 x 4.25 x 1.22 inch

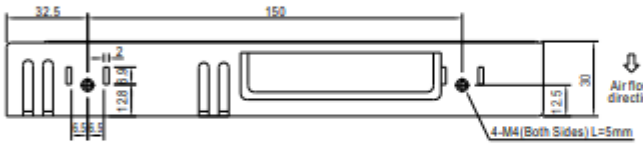
5.1.1.9.MeanWell RSP-320-24

MEAN WELL 320W Single Output with PFC Function **RSP-320 series**

■ Mechanical Specification

Case No.207A Unit:mm



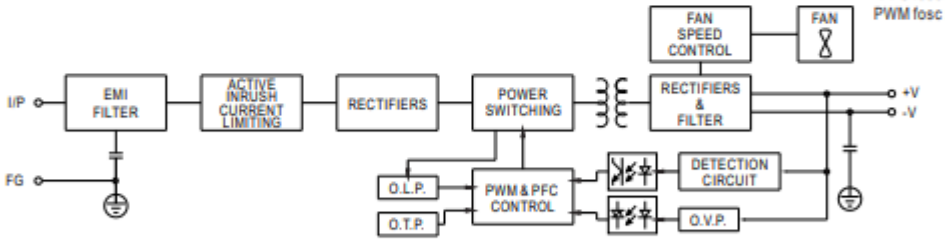


Terminal Pin No. Assignment :

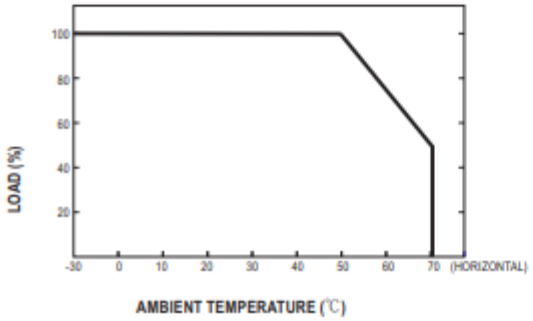
Pin No.	Assignment	Pin No.	Assignment
1	AC/L	4~6	DC OUTPUT -V
2	AC/N	7~9	DC OUTPUT +V
3	FG ⏚		

■ Block Diagram

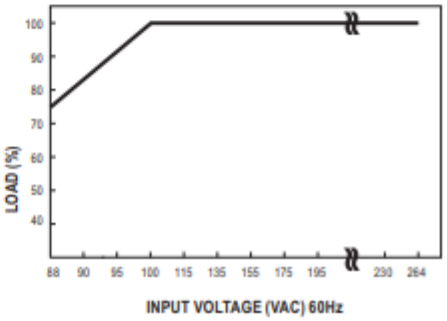
PFC fosc : 100KHz
PWM fosc : 100KHz



■ Derating Curve



■ Static Characteristics



MODEL		RSP-320-13.5	RSP-320-15	RSP-320-24	RSP-320-27	RSP-320-36	RSP-320-48
OUTPUT	DC VOLTAGE	13.5V	15V	24V	27V	36V	48V
	RATED CURRENT	23.8A	21.4A	13.4A	11.9A	8.9A	6.7A
	CURRENT RANGE	0 ~ 23.8A	0 ~ 21.4A	0 ~ 13.4A	0 ~ 11.9A	0 ~ 8.9A	0 ~ 6.7A
	RATED POWER	321.3W	321W	321.6W	321.3W	320.4W	321.6W
	RIPPLE & NOISE (max.) Note.2	150mVp-p	150mVp-p	150mVp-p	200mVp-p	220mVp-p	240mVp-p
	VOLTAGE ADJ. RANGE	12 ~ 15V	13.5 ~ 18V	20 ~ 26.4V	26 ~ 31.5V	32.4 ~ 39.6V	41 ~ 56V
	VOLTAGE TOLERANCE Note.3	±1.0%	±1.0%	±1.0%	±1.0%	±1.0%	±1.0%
	LINE REGULATION	±0.3%	±0.3%	±0.2%	±0.2%	±0.2%	±0.2%
	LOAD REGULATION	±0.5%	±0.5%	±0.5%	±0.5%	±0.5%	±0.5%
	SETUP, RISE TIME	1500ms, 50ms/230VAC 3000ms, 50ms/115VAC at full load					
	HOLD UP TIME (Typ.)	8ms at full load 230VAC /115VAC					
INPUT	VOLTAGE RANGE Note.4	88 ~ 264VAC 124 ~ 370VDC					
	FREQUENCY RANGE	47 ~ 63Hz					
	POWER FACTOR (Typ.)	PF>0.95/230VAC PF>0.98/115VAC at full load					
	EFFICIENCY (Typ.)	88%	88.5%	89%	89%	89.5%	90%
	AC CURRENT (Typ.)	4A/115VAC 2A/230VAC					
	INRUSH CURRENT (Typ.)	20A/115VAC 40A/230VAC					
	LEAKAGE CURRENT	<1mA / 240VAC					
PROTECTION	OVERLOAD	105 ~ 135% rated output power					
		Protection type : Hiccup mode, recovers automatically after fault condition is removed					
	OVER VOLTAGE	15.7 ~ 18.4V	18.8 ~ 21.8V	27.6 ~ 32.4V	32.9 ~ 38.3V	41.4 ~ 48.6V	58.4 ~ 68V
		Protection type : Shut down o/p voltage, re-power on to recover					
	OVER TEMPERATURE	Shut down o/p voltage, recovers automatically after temperature goes down					
ENVIRONMENT	WORKING TEMP.	-30 ~ +70℃ (Refer to "Derating Curve")					
	WORKING HUMIDITY	20 ~ 90% RH non-condensing					
	STORAGE TEMP., HUMIDITY	-40 ~ +85℃, 10 ~ 95% RH					
	TEMP. COEFFICIENT	±0.03%/℃ (0 ~ 50℃)					
	VIBRATION	10 ~ 500Hz, 2G 10min./1cycle, 60min. each along X, Y, Z axes					
SAFETY & EMC (Note 5)	SAFETY STANDARDS	UL62368-1,TUV BS EN/EN62368-1,EAC TP TC 004, CCC GB4943.1,BSMI CNS14336-1, AS/NZS 60950.1, IS13252(Part1)/IEC60950-1(except for 2.5V,48V) approved					
	WITHSTAND VOLTAGE	I/P-O/P:3KVAC I/P-FG:2KVAC O/P-FG:0.5KVAC					
	ISOLATION RESISTANCE	I/P-O/P, I/P-FG, O/P-FG:100M Ohms / 500VDC / 25℃/ 70% RH					
	EMC EMISSION	Compliance to BS EN/EN55032 (CISPR32) Class B, BS EN/EN61000-3-2,-3, EAC TP TC 020, CNS13438, GB9254 Class B, GB17625.1					
	EMC IMMUNITY	Compliance to BS EN/EN61000-4-2,3,4,5,6,8,11, BS EN/EN55024, light industry level, EAC TP TC 020					
OTHERS	MTBF	1826.4K hrs min. Telcordia SR-332 (Bellcore) ; 192.9K hrs min. MIL-HDBK-217F (25℃)					
	DIMENSION	215*115*30mm (L*W*H)					
	PACKING	0.9Kg; 15pcs/14.5Kg/0.67CUFT					
NOTE	1. All parameters NOT specially mentioned are measured at 230VAC input, rated load and 25℃ of ambient temperature. 2. Ripple & noise are measured at 20MHz of bandwidth by using a 12" twisted pair-wire terminated with a 0.1uf & 47uf parallel capacitor. 3. Tolerance : includes set up tolerance, line regulation and load regulation. 4. Derating may be needed under low input voltages. Please check the derating curve for more details. 5. The power supply is considered a component which will be installed into a final equipment. The final equipment must be re-confirmed that it still meets EMC directives. For guidance on how to perform these EMC tests, please refer to "EMI testing of component power supplies." (as available on http://www.meanwell.com) 6. For charging related applications, please consult Mean Well for details. 7. Strongly recommended that external output capacitance should not exceed 5000uF. (Only for: RSP-320-2.5/-3.3/-4/-5/-7.5/-12/-13.5/-15) 8. The ambient temperature derating of 3.5℃/1000m with fanless models and of 5℃/1000m with fan models for operating altitude higher than 2000m(6500ft). ※ Product Liability Disclaimer : For detailed information, please refer to https://www.meanwell.com/serviceDisclaimer.aspx						

5.1.2. Código del Robot Scara

5.1.2.1.Cinemática inversa

```
void inverseKin(const RobotPosition_t &target,const RobotParams_t &params,double *q){

    //Calculamos L
    double l=sqrt(pow(target.x,2)+pow(target.y,2));
    //Calculamos q2
    q[1]=acos((pow(l,2)-pow(params.l2,2)-pow(params.l1,2))/(2*params.l2*params.l1));
    //Calculamos beta
    double beta=atan2(target.y,target.x);
    //Calculamos alpha
    double alpha=atan2(params.l2*sin(q[1]),(params.l1+params.l2*cos(q[1])));

    //Calculamos q1
    if((target.y>=0)&&(target.x<=0)){
      if(beta>=alpha){
        q[0]=beta-alpha;
      }
      else if(alpha>beta){
        q[0]=alpha-beta;
      }
     }
    else if((target.y>=0)&&(target.x>=0)){
        q[0]=beta-alpha;
    }
    else if(target.y<0){
       q[0]=2*PI+beta-alpha;
    }

    //Pasamos q1 y q2 a ángulos
    q[0]=q[0]*180/PI;
    q[1]=q[1]*180/PI;
}
```

5.1.2.2.Generador de trayectoria polinomial

```
//Buffer con la dirección del motor
int dirRM1=0;
int dirRM2=0;

//Buffer con la configuración articular actual
double qactual[JOINTS]={0,0,0};

void moveAbsJ(const double q0[JOINTS], const double qT[JOINTS],
const double T)
{
  double a[JOINTS],b[JOINTS],c[JOINTS],d[JOINTS],q[JOINTS],t,t0;
  //Parámetros de la trayectoria
  for(int i=0; i<JOINTS;i++)
  {
    a[i]=-((2*(qT[i]-q0[i]))/(pow(T,3)));
    b[i]=((3*(qT[i]-q0[i]))/(pow(T,2)));
    c[i]=0.0;
    d[i]=q0[i];
  }

   //Configuramos la dirección de rotación del motor del eje 1
   if(qT[0]>q0[0]){
    digitalWrite(EJE1_DIR_PIN , LOW);
    dirRM1=1;
  }
  else if(qT[0]<q0[0]){
    digitalWrite(EJE1_DIR_PIN , HIGH);
    dirRM1=0;
  }

   //Configuramos la dirección de rotación del motor del eje 2
  if(qT[1]>q0[1]){
    digitalWrite(EJE2_DIR_PIN , HIGH);
    dirRM2=1;
  }
  else if(qT[1]<q0[1]){
    digitalWrite(EJE2_DIR_PIN , LOW);
    dirRM2=0;
  }
```

```
    t0=millis()/1000.0;//tiempo inicial de la trayectoria
    t=0.0;//inicializamos a 0 el tiempo inicial
    while(t<T)//bucle temporizado
    {
      for (int i=0;i<JOINTS;i++)
      {
        //los ángulos de la trayectoria son calculados
        q[i]=a[i]*pow(t,3)+b[i]*pow(t,2)+c[i]*t+d[i];
        //se calcula el incremento de ángulo
        for (int i=0;i<JOINTS;i++){
           if(qT[i]>=q0[i]) qsend[i]=qT[i]-q0[i];
           else if(qT[i]<q0[i]) qsend[i]=q0[i]-qT[i];

           //se almacena la última configuración
           qactual[i]=qT[i];
         }


      }
      //se mandan los ángulos
      moveRobot((int)q);
      delay(20);
      t=millis()/1000.0-t0;
    }
  }
```

5.1.2.3.Generador de pulsos

```
void moveRobot(double q[JOINTS]){
  double Npulsos[JOINTS]={0,0,0};

  //Calculamos el número de impulsos de cada eje
  Npulsos[0]=q[0]/(1.8/28);
  Npulsos[1]=q[1]/(1.8/24);

  //Mandamos los pulsos a los drivers de motor 1 y 2
  if(Npulsos[0]>=Npulsos[1]){
    for(int i=0 ; i<=(int)Npulsos[0] ; i++){
      digitalWrite(EJE1_STEP_PIN , HIGH);
      if(i<=(int)Npulsos[1]) digitalWrite(EJE2_STEP_PIN , HIGH);
      delay(1);

      digitalWrite(EJE1_STEP_PIN , LOW);
      if(i<=(int)Npulsos[1]) digitalWrite(EJE2_STEP_PIN , LOW);
      delay(1);
    }
  }
  else if(Npulsos[1]>Npulsos[0]){
    for(int i=0 ; i<=(int)Npulsos[1] ; i++){
      digitalWrite(EJE2_STEP_PIN , HIGH);
      if(i<=(int)Npulsos[0]) digitalWrite(EJE1_STEP_PIN , HIGH);
      delay(1);

      digitalWrite(EJE2_STEP_PIN , LOW);
      if(i<=(int)Npulsos[0]) digitalWrite(EJE1_STEP_PIN , LOW);
      delay(1);
    }
  }
}
```

5.1.2.4.Función recogerCápsula

```
void recogerCapsula(){
  //Dirección de giro antihoraria
  digitalWrite(EJE3_DIR_PIN , LOW);
  delay(1);

  for (int i = 0; i <= 3300; i++)
  {
  digitalWrite(EJE3_STEP_PIN , HIGH);
  delay(1);
  digitalWrite(EJE3_STEP_PIN , LOW);
  delay(1);
  }

  digitalWrite(BOMBA_VACIO , HIGH);
  delay(100);

  digitalWrite(EJE3_DIR_PIN , HIGH);
  delay(1);

  for (int i = 0; i <= 3300; i++)
  {
  digitalWrite(EJE3_STEP_PIN , HIGH);
  delay(1);
  digitalWrite(EJE3_STEP_PIN , LOW);
  delay(1);
  }
}
```

5.1.2.5.Función depositarCápsula

```
void depositarCapsula(int pasos){
  //Dirección de giro antihoraria
  digitalWrite(EJE3_DIR_PIN , LOW);
  delay(1);

  for (int i = 0; i <= pasos; i++)
  {
  digitalWrite(EJE3_STEP_PIN , HIGH);
  delay(1);
  digitalWrite(EJE3_STEP_PIN , LOW);
  delay(1);
  }

  digitalWrite(BOMBA_VACIO , LOW);
  delay(100);

  digitalWrite(EJE3_DIR_PIN , HIGH);
  delay(1);

  for (int i = 0; i <= pasos; i++)
  {
  digitalWrite(EJE3_STEP_PIN , HIGH);
  delay(1);
  digitalWrite(EJE3_STEP_PIN , LOW);
  delay(1);
  }
}
```

5.1.2.6.Programa principal

```
//DEFINICIONES DE LOS PINES DEL ARDUINO A LOS QUE SE CONECTA CADA MOTOR
#define EJE1_STEP_PIN 26
#define EJE1_DIR_PIN 28
#define EJE1_ENABLE_PIN 24

#define EJE2_STEP_PIN 54
#define EJE2_DIR_PIN 55
#define EJE2_ENABLE_PIN 38

#define EJE3_STEP_PIN 60
#define EJE3_DIR_PIN 61
#define EJE3_ENABLE_PIN 56

#define PLATO1_STEP_PIN 46
#define PLATO1_DIR_PIN 48
#define PLATO1_ENABLE_PIN 62

#define PLATO2_STEP_PIN 36
#define PLATO2_DIR_PIN 34
#define PLATO2_ENABLE_PIN 30

#define BOMBA_VACIO 10

#define PI 3.14159265

#define JOINTS 3

//PARAMETROS DEL ROBOT
typedef struct
{
  double l1;
  double l2;
  double h1;
  double h2;
  double h4;
} RobotParams_t;

RobotParams_t params={0.2,0.2,0.165,0.03,0.04};//en m

typedef struct
```

```
{
  double x;
  double y;
  double z;
} RobotPosition_t;

int q1,q2,q3,q4 =0;


//PUNTOS OBJETIVO DEL EFECTOR FINAL
RobotPosition_t pThome={0.4,0,0};
RobotPosition_t pTpick={-0.06,0.23,0};

RobotPosition_t pTC1={0.08,0.24,0};
RobotPosition_t pTC2={0.08,0.19,0};
RobotPosition_t pTC3={0.12,0.19,0};
RobotPosition_t pTC4={0.12,0.24,0};

RobotPosition_t pTirar={0,0.4,0};

//VARIABLES PARA ALMACENAR CONFIGURACIONES ARTICULARES
double qhome[JOINTS]={0,0,0};
double qpick[JOINTS]={0,0,0};

double qcaja1[JOINTS]={0,0,0};
double qcaja2[JOINTS]={0,0,0};
double qcaja3[JOINTS]={0,0,0};
double qcaja4[JOINTS]={0,0,0};

double qtirar[JOINTS]={0,0,0};

//VARIABLES EN LAS QUE SE GUARDAN LAS CONFIGURACIONES ACTUALES
double qactual[JOINTS]={0,0,0};
RobotPosition_t pTactual={0,0,0};

int dirRM1=0;
int dirRM2=0;

String option="none";

int reverse=0;
```

```
int reverse2=0;

int capsulasProcesadas=0;
int pilas=0;

//BUCLE PRINCIPAL
void setup() {
  //CONFIGURAMOS LOS PINES DEL MOTOR 1
  pinMode(EJE1_STEP_PIN , OUTPUT);
  pinMode(EJE1_DIR_PIN , OUTPUT);
  pinMode(EJE1_ENABLE_PIN , OUTPUT);
  digitalWrite(EJE1_ENABLE_PIN , HIGH);

  //CONFIGURAMOS LOS PINES DEL MOTOR 2
  pinMode(EJE2_STEP_PIN , OUTPUT);
  pinMode(EJE2_DIR_PIN , OUTPUT);
  pinMode(EJE2_ENABLE_PIN , OUTPUT);
  digitalWrite(EJE2_ENABLE_PIN , LOW);

  //CONFIGURAMOS LOS PINES DEL MOTOR 3
  pinMode(EJE3_STEP_PIN , OUTPUT);
  pinMode(EJE3_DIR_PIN , OUTPUT);
  pinMode(EJE3_ENABLE_PIN , OUTPUT);
  digitalWrite(EJE3_ENABLE_PIN , LOW);

  //CONFIGURAMOS LOS PINES DEL MOTOR 4
  pinMode(PLATO1_STEP_PIN , OUTPUT);
  pinMode(PLATO1_DIR_PIN , OUTPUT);
  pinMode(PLATO1_ENABLE_PIN , OUTPUT);
  digitalWrite(PLATO1_ENABLE_PIN , LOW);
  digitalWrite(PLATO1_DIR_PIN , HIGH);

  //CONFIGURAMOS LOS PINES DEL MOTOR 5
  pinMode(PLATO2_STEP_PIN , OUTPUT);
  pinMode(PLATO2_DIR_PIN , OUTPUT);
  pinMode(PLATO2_ENABLE_PIN , OUTPUT);
  digitalWrite(PLATO2_ENABLE_PIN , LOW);
  digitalWrite(PLATO2_DIR_PIN , HIGH);

  //CONFIGURAMOS LOS PINES DE LA BOMBA DE VACIO
  pinMode(BOMBA_VACIO , OUTPUT);
```

```
  digitalWrite(BOMBA_VACIO , LOW);

  //INICIAMOS LA COMUNICACIÓN SERIAL
  Serial.begin(9600);

  //CALCULAMOS LA CINEMATICA INVERSA DE LOS PUNTOS OBJETIVO
  inverseKin(pThome,params,qhome);
  inverseKin(pTpick,params,qpick);
  inverseKin(pTC1,params,qcaja1);
  inverseKin(pTC2,params,qcaja2);
  inverseKin(pTC3,params,qcaja3);
  inverseKin(pTC4,params,qcaja4);
  inverseKin(pTirar,params,qtirar);
}


void loop() {
  //CHECKEAMOS EL PUERTO SERIAL
  if(Serial.available()>0)
  {
    option = Serial.readStringUntil('\n');
  }
  //RECIBIMOS PICK EN EL SERIAL
  if(option=="PICK"){
    capsulasProcesadas++;
    //VAMOS A LA POSICIÓN DE RECOJIDA
    moveAbsJ(qactual,qpick,10);
    recogerCapsula();

    switch(capsulasProcesadas){
      case 1:
          //VAMOS A LA POSICIÓN DE DESCARGA 1
          moveAbsJ(qactual,qcaja1,10);
        break;
      case 2:
          //VAMOS A LA POSICIÓN DE DESCARGA 2
          moveAbsJ(qactual,qcaja2,10);
        break;
      case 3:
          //VAMOS A LA POSICIÓN DE DESCARGA 3
          moveAbsJ(qactual,qcaja3,10);
```

```
    break;
  case 4:
      //VAMOS A LA POSICIÓN DE DESCARGA 4
      moveAbsJ(qactual,qcaja4,10);

      capsulasProcesadas=0;
      pilas++;
    break;
}
switch(pilas){
  case 0:
     //DEPOSITAMOS LA CAPSULA
     depositarCapsula(2000);
    break;
  case 1:
    //DEPOSITAMOS LA CAPSULA
     depositarCapsula(1750);
    break;
  case 2:
     //DEPOSITAMOS LA CAPSULA
     depositarCapsula(1500);
    break;
  case 3:
      //DEPOSITAMOS LA CAPSULA
      depositarCapsula(1250);
      break;
  case 4:
      //DEPOSITAMOS LA CAPSULA
      depositarCapsula(1250);
    break;
}
//VOLVEMOS A LA POSICION DE RECOJIDA
moveAbsJ(qactual,qpick,10);

if(pilas<4){
    //SI AUN NO ESTA LLENA
    //ROTAMOS EL PLATO DE LAS CAPSULAS
    rotarPlatoCapsula();
}
else if(pilas>=4){
  //SI LA CAJA ESTA LLENA
```

```
      //ROTAMOS EL PLATO DE LAS CAPSULAS
      rotarPlatoCapsula();
      //ROTAMOS EL PLATO DE LAS CAJAS
      rotarPlatoCaja();
      pilas=0;
    }
    option="none";
    Serial.println("Done,10");
  }

  //RECIBIMOS ROTATE EN EL SERIAL
  else if(option=="ROTATE"){
    //ROTAMOS EL PLATO DE LAS CAPSULAS
    rotarPlatoCapsula();
    option="none";
    Serial.println("Done,10");
  }

  //RECIBIMOS REM EN EL SERIAL
  else if(option=="REM"){
      //VAMOS A LA POSICION DE RECOJIDA
      moveAbsJ(qactual,qpick,10);
      //COGEMOS LA CAPSULA
      recogerCapsula();
      //VAMOS A LA POSCION DE DESCARTE
      moveAbsJ(qactual,qtirar,10);
      //SOLTAMOS LA CAPSULA
      depositarCapsula(500);
      delay(2000);
      //VOLVEMOS A LA POSICION DE RECOJIDA
      moveAbsJ(qactual,qpick,10);
      //ROTAMOS EL PLATO DE LAS CAPSULAS
      rotarPlatoCapsula();

      option="none";
      Serial.println("Done,10");
  }

}
```

5.2. Cabezales giratorios

5.2.1. Código del los cabezales giratorios

5.2.2.Función rotarPlatoCápsula

```
void rotarPlatoCapsula(){
  for (int i = 0; i <= 24; i++)
  {
  digitalWrite(PLATO1_STEP_PIN , HIGH);
  delay(10);
  digitalWrite(PLATO1_STEP_PIN , LOW);
  delay(10);
  }
}
```

5.2.3.Función rotarPlatoCajas

```
void rotarPlatoCaja(){
  for (int i = 0; i <= 99; i++)
  {
  digitalWrite(PLATO2_STEP_PIN , HIGH);
  delay(10);
  digitalWrite(PLATO2_STEP_PIN , LOW);
  delay(10);
  }
}
```

5.3. Sistema de visión artificial

5.3.1. Código del sistema de visión artificial

5.3.1.1. Filtrado por color

```
from __future__ import division
import cv2
import numpy as np

def filterColor(LH,HH,LS,HS,LV,HV,img):
    frame = img
    lowHue=LH
    highHue=HH
    lowSat=LS
    highSat=HS
    lowVal=LV
    highVal=HV

    #Realizamos un filtrado bilateral
    frameBGR = cv2.bilateralFilter(frame,10 ,1, 1)

    #Convertimos la imagen a espacio de color HSV
    hsv = cv2.cvtColor(frameBGR, cv2.COLOR_BGR2HSV)

    # Rango de parametros que deininen un color en el espacio HSV
    colorLow = np.array([lowHue,lowSat,lowVal])
    colorHigh = np.array([highHue,highSat,highVal])
    mask = cv2.inRange(hsv, colorLow, colorHigh)

    #Creamos una mascara de filtrado
    kernal = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (7, 7))
    mask = cv2.morphologyEx(mask, cv2.MORPH_CLOSE, kernal)
    mask = cv2.morphologyEx(mask, cv2.MORPH_OPEN, kernal)

    # Aplicamos la mascara a la imagen
    result = cv2.bitwise_and(frame, frame, mask = mask)

    return result
```

5.3.1.2.Reconocimiento de círculos

```
from tkinter.constants import NONE
import cv2 as cv
import numpy as np

dataImg=[0,0]

def findCircle(DP,P1,P2,minR,maxR,img):
    cimg=img.copy()

    # convert image to grayscale
    img = cv.cvtColor(img, cv.COLOR_BGR2GRAY)

    # apply a blur using the median filter
    img = cv.medianBlur(img, 19)

    #cv.imshow('blur',img)

    circles = cv.HoughCircles(image=img, method=cv.HOUGH_GRADIENT, dp=DP,
                              minDist=80, param1=P1, param2=P2,minRadius=minR,maxRadius=maxR)

    try:
        for co, i in enumerate(circles[0, :], start=1):
            # draw the outer circle in green
            cv.circle(cimg,( int(i[0]), int(i[1])),int(i[2]),(0,255,0),2)
            # draw the center of the circle in red
            cv.circle(cimg,(int(i[0]),int(i[1])),2,(0,0,255),3)

            centerx=int(i[0])
            centery=int(i[1])
            radius=int(i[2])
            # print the number of circles detected
            ncircles="No CIRCLES: "+str(co)
        cv.putText(cimg,ncircles,(0,40),cv.FONT_ITALIC,1.5,(255,255,255),thickness=4)
        ncircles=str(co)
    except:
        cv.putText(cimg,"Error",(0,40),cv.FONT_ITALIC,1.5,(255,255,255),thickness=4)
        ncircles=0
        centerx=0
        centery=0
        radius=0

    dataImg=[cimg,ncircles,centerx,centery,radius]
```

5.3.1.3.Función Iniciar

```
def iniciar():
    #INICAMOS LA CAPTURA DEL VIDEO
    global cap
    cap = cv.VideoCapture(0)
    visualizar()

    global DP
    global P1
    global P2
    global minR
    global maxR

    #ABRIMOS EL ARCHIVO DE TEXTO
    file1= open("cameraConfig.txt", "+r")

    #LEEMOS EL ARCHIVO DE TEXTO
    data=file1.read()
    datasplit=data.split(",")

    #ALMACENAMOS LAS VARIABLES PARA filterColor
    for i in range (0,3):
        LH[i]=int(datasplit[i+i*5])
        HH[i]=int(datasplit[(i+1)+i*5])
        LS[i]=int(datasplit[(i+2)+i*5])
        HS[i]=int(datasplit[(i+3)+i*5])
        LV[i]=int(datasplit[(i+4)+i*5])
        HV[i]=int(datasplit[(i+5)+i*5])

    #ALMACENAMOS LAS VARIABLES PARA findCircle
    DP=float(datasplit[18])
    P1=int(datasplit[19])
    P2=int(datasplit[20])
    minR=int(datasplit[21])
    maxR=int(datasplit[22])

    #CERRAMOS EL ARCHIVO DE TEXTO
    file1.close()
```

5.3.1.4.Función Visualizar

```
def visualizar():
    global cap
    global lblVideo
    global current
    global DP
    global P1
    global P2
    global minR
    global maxR
    global Nnaranja
    global Nazul
    global Nrojo

    if cap is not None:
        ret, frame = cap.read()
        if ret == True:

            #FILTRAMOS POR COLOR LA IMAGEN
            #IMAGEN CON CAPSULAS NARANJA
            naranja=filterColor(LH[0],HH[0],LS[0],HS[0],LV[0],HV[0],frame)
            #IMAGEN CON CAPSULAS AZULES
            azul=filterColor(LH[1],HH[1],LS[1],HS[1],LV[1],HV[1],frame)
            #IMAGEN CON CAPSULAS ROJAS
            rojo=filterColor(LH[2],HH[2],LS[2],HS[2],LV[2],HV[2],frame)

            #VEMOS LOS CIRCULOS DE CADA COLOR EN LA IMAGEN
            #EXTRAEMOS LOS CIRCULOS DE LA IMAGEN CON CAPSULAS NARANJA
            Dnaranja=findCircle(DP,P1,P2,minR,maxR,naranja)
            #EXTRAEMOS LOS CIRCULOS DE LA IMAGEN CON CAPSULAS AZULES
            Dazul=findCircle(DP,P1,P2,minR,maxR,azul)
            #EXTRAEMOS LOS CIRCULOS DE LA IMAGEN CON CAPSULAS ROJAS
            Drojo=findCircle(DP,P1,P2,minR,maxR,rojo)

            #NUMERO DE CAPSULAS NARANJAS
            Nnaranja = Dnaranja[1]
            #NUMERO DE CAPSULAS AZULES
            Nazul = Dazul[1]
            #NUMERO DE CAPSULAS ROJAS
            Nrojo = Drojo[1]

            #PLOTEAMOS LOS IDENTIFICADORES DE LA CAPSULA NARANJA
            cv.circle(frame,(Dnaranja[2],Dnaranja[3]),Dnaranja[4],(0,255,0),2)
            cv.circle(frame,(Dnaranja[2],Dnaranja[3]),2,(0,0,255),3)
            cv.putText(frame,"Capsula Naranja",((Dnaranja[2]-50),(Dnaranja[3]-50)),cv.FONT_ITALIC,0.5,(0,0,0),

            #PLOTEAMOS LOS IDENTIFICADORES DE LA CAPSULA AZUL
            cv.circle(frame,(Dazul[2],Dazul[3]),Dazul[4],(0,255,0),2)
            cv.circle(frame,(Dazul[2],Dazul[3]),2,(0,0,255),3)
            cv.putText(frame,"Capsula Azul",((Dazul[2]-50),(Dazul[3]-50)),cv.FONT_ITALIC,0.5,(0,0,0),thickness

            #PLOTEAMOS LOS IDENTIFICADORES DE LA CAPSULA ROJA
            cv.circle(frame,(Drojo[2],Drojo[3]),Drojo[4],(0,255,0),2)
            cv.circle(frame,(Drojo[2],Drojo[3]),2,(0,0,255),3)
            cv.putText(frame,"Capsula Roja",((Drojo[2]-50),(Drojo[3]-50)),cv.FONT_ITALIC,0.5,(0,0,0),thickness


            #PLOTEAMOS EL NUMERO DE CIRCULOS DE CADA COLOR
            lblNN.configure(text=str(Dnaranja[1]))
            lblNA.configure(text=str(Dazul[1]))
            lblNR.configure(text=str(Drojo[1]))

            #MOSTRAMOS EL VIDEO EN PANTALLA
            frame = imutils.resize(frame, width=640)
            frame = cv.cvtColor(frame, cv.COLOR_BGR2RGB)
            im = Image.fromarray(frame)
            img = ImageTk.PhotoImage(image=im)
            lblVideo.configure(image=img)
            lblVideo.image = img
            lblVideo.after(10, visualizar)
        else:
            lblVideo.image = ""
            cap.release()
```

5.4. Integración en Python

5.4.1. Integración en Python

5.4.1.1. Inicialización del HMI

```
arduino =None
positions=["none","none","none","none","none","none","none","none"]
Nprocesado=0
Ncaja=0
NdescardN=0
NdescardR=0

def autoScreen():
    global lblVideo

    auto = Tk()
    auto.title('AutoScreen')

    # get the screen dimension
    screen_width = auto.winfo_screenwidth()
    screen_height =auto.winfo_screenheight()
    auto.geometry(f'{screen_width}x{screen_height}')
    #UPVlogo=ImageTk.PhotoImage(Image.open("UPVlogo.jpg"))
    #ETSIDlogo=ImageTk.PhotoImage(Image.open("ETSIDlogo.png"))

    #upvImg=Label(login,image=UPVlogo).place(x=200,y=200)
    #etsidImg=Label(login,image=ETSIDlogo).place(x=400,y=200)
    labelConect1 = Label(auto,text="-Estado de la conexion:",font=("Arial",25))
    labelConect1.place(x=700,y=50)
    labelConect = Label(auto,text="DESCONECTADO",font=("Arial",25),fg="red")
    labelConect.place(x=720,y=100)

    labelPlato1 = Label(auto,text="-Estado del portacapsulas:",font=("Arial",25))
    labelPlato1.place(x=700,y=200)
    labelPlato = Label(auto,text="PARADO",font=("Arial",25),fg="red")
    labelPlato.place(x=720,y=250)

    labelRobot1 = Label(auto,text="-Estado del robot:",font=("Arial",25))
    labelRobot1.place(x=700,y=300)
    labelRobot2 = Label(auto,text="PARADO",font=("Arial",25),fg="red")
    labelRobot2.place(x=720,y=350)
    labelRobot = Label(auto,text="SIN CAPSULA",font=("Arial",25))
    labelRobot.place(x=720,y=400)
```

```
labelRobot = Label(auto,text="SIN CAPSULA",font=("Arial",25))
labelRobot.place(x=720,y=400)

labelCaja1 = Label(auto,text="-Capsulas en la caja:",font=("Arial",25))
labelCaja1.place(x=700,y=450)
labelCaja = Label(auto,text="VACIA",font=("Arial",25))
labelCaja.place(x=720,y=500)

labelEmp1 = Label(auto,text="CAPSULAS AZULES EMPAQUETADAS:",font=("Arial",25))
labelEmp1.place(x=50,y=580)
labelEmp = Label(auto,text="0",font=("Arial",25))
labelEmp.place(x=700,y=580)

labelDN1 = Label(auto,text="CAPSULAS NARANJAS DESCARTADAS:",font=("Arial",25))
labelDN1.place(x=50,y=630)
labelDN = Label(auto,text="0",font=("Arial",25))
labelDN.place(x=700,y=630)

labelDR1 = Label(auto,text="CAPSULAS ROJAS DESCARTADAS:",font=("Arial",25))
labelDR1.place(x=50,y=680)
labelDR = Label(auto,text="0",font=("Arial",25))
labelDR.place(x=700,y=680)

labelVideoT = Label(auto,text="CAPTURA DE LA CAMARA",font=("Arial",25))
labelVideoT.grid(column=1,row=0,padx=15, pady=15)
lblVideo = Label(auto)
lblVideo.grid(column=1,row=1,padx=15, pady=15,rowspan=3)
```

5.4.1.2. Comunicación Python y Arduino

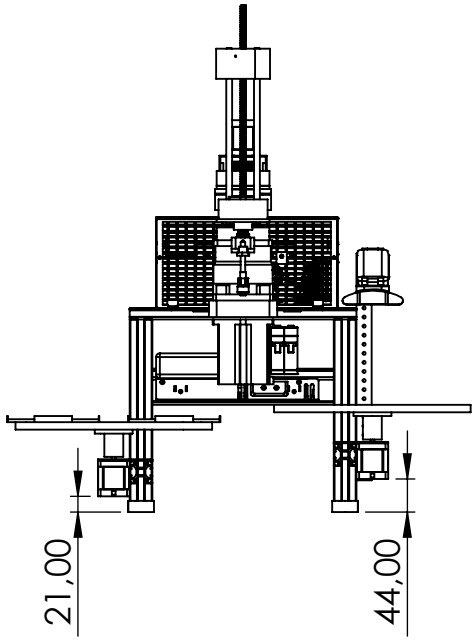
```
def RobotControl():
    global Nnaranja
    global Nazul
    global Nrojo
    global message
    global arduino
    global positions
    global Nprocesado
    global NdescardN
    global NdescardR
    global Ncaja

    time.sleep(1)


    i=0
    clase="azul"
    PICK="PICK"+"\n"
    ROTATE="ROTATE"+"\n"
    REMOVE="REM"+"\n"

    if(Nnaranja != 0):
        positions[i]="naranja"
    elif(Nazul != 0):
        positions[i]="azul"
    elif(Nrojo != 0):
        positions[i]="rojo"

    print(str(i+1))
    print(positions[i])

    print("Starting serial on COM13")
    arduino=serial.Serial("COM13",9600)

    labelConect.config(text="CONECTADO",fg="green")

    time.sleep(1)
```

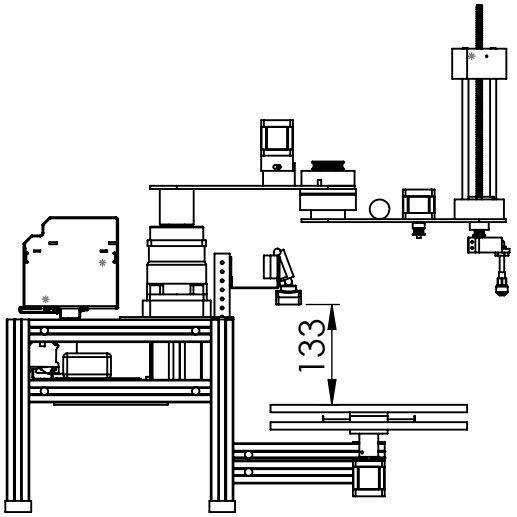
```
arduino.write(ROTATE.encode("utf-8"))

labelPlato.config(text="ROTANDO",fg="green")

reading = (arduino.readline()).decode()
readSplit=reading.split(',')

print(readSplit[0])

while True:
    if readSplit[0]=="Done":
        labelRobot2.config(text="PARADO",fg="red")
        labelRobot.config(text="SIN CAPSULA",fg="red")
        labelPlato.config(text="PARADO",fg="red")
        if(Ncaja==16):
            labelCaja.config(text="CAJA VACIA")
            Ncaja=0

        time.sleep(1)

        if i<7:
            i=i+1
        elif i>=7:
            i=0

        if(Nnaranja != 0):
            positions[i]="naranja"
        elif(Nazul != 0):
            positions[i]="azul"
        elif(Nrojo != 0):
            positions[i]="rojo"
        else:
            positions[i]="none"


        if i<2:
            if positions[6+i]==clase:
                arduino.write(PICK.encode("utf-8"))
                labelRobot2.config(text="EN MOVIMIENTO",fg="green")
                labelRobot.config(text="MANIPULANDO CAPSULA AZUL",fg="blue")
```

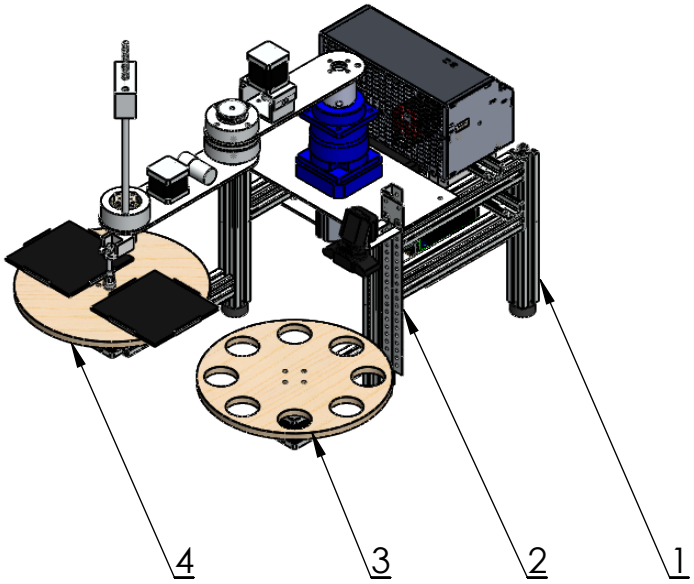
```
                Nprocesado=Nprocesado+1
                Ncaja=Ncaja+1
                print("Capsulas procesadas:")
                print(str(Nprocesado))
                labelEmp.config(text=str(Nprocesado),fg="blue")

                labelCaja.config(text=str(Ncaja))
                if(Ncaja==16):
                    labelCaja.config(text="CAJA LLENA")


            elif (positions[6+i]=="naranja"):
                arduino.write(REMOVE.encode("utf-8"))
                labelRobot2.config(text="EN MOVIMIENTO",fg="green")
                labelRobot.config(text="MANIPULANDO CAPSULA NARANJA",fg="orange")

                NdescardN=NdescardN+1
                labelDN.config(text=str(NdescardN),fg="orange")

            elif (positions[6+i]=="rojo"):
                arduino.write(REMOVE.encode("utf-8"))
                labelRobot2.config(text="EN MOVIMIENTO",fg="green")
                labelRobot.config(text="MANIPULANDO CAPSULA ROJA",fg="red")
                NdescardR=NdescardR+1
                labelDR.config(text=str(NdescardR),fg="red")
            else:
                arduino.write(ROTATE.encode("utf-8"))
                labelPlato.config(text="ROTANDO",fg="green")

        elif i>=2:
            if positions[i-2]==clase:
                arduino.write(PICK.encode("utf-8"))
                labelRobot2.config(text="EN MOVIMIENTO",fg="green")
                labelRobot.config(text="MANIPULANDO CAPSULA AZUL",fg="blue")
                Nprocesado=Nprocesado+1
                Ncaja=Ncaja+1
                print("Capsulas procesadas:")
                print(str(Nprocesado))
                labelEmp.config(text=str(Nprocesado),fg="blue")
```

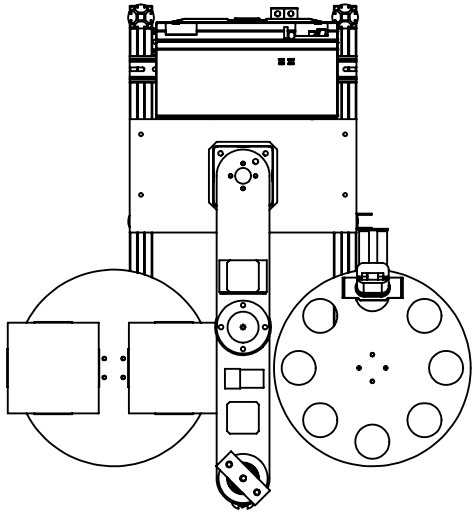
```
                    labelCaja.config(text=str(Ncaja))
                    if(Ncaja==16):
                        labelCaja.config(text="CAJA LLENA")


                elif (positions[i-2]=="naranja"):
                    arduino.write(REMOVE.encode("utf-8"))
                    labelRobot2.config(text="EN MOVIMIENTO",fg="green")
                    labelRobot.config(text="MANIPULANDO CAPSULA NARANJA",fg="orange")
                    NdescardN=NdescardN+1

                    labelDN.config(text=str(NdescardN),fg="orange")

                elif (positions[i-2]=="rojo"):
                    arduino.write(REMOVE.encode("utf-8"))
                    labelRobot2.config(text="EN MOVIMIENTO",fg="green")
                    labelRobot.config(text="MANIPULANDO CAPSULA ROJA",fg="red")
                    NdescardR=NdescardR+1
                    labelDR.config(text=str(NdescardR),fg="red")

                else:
                    arduino.write(ROTATE.encode("utf-8"))
                    labelPlato.config(text="ROTANDO",fg="green")

            readSplit[0]="none"
            reading = (arduino.readline()).decode()
            readSplit=reading.split(',')

#INICIAMOS LA VISION ARTIFICIAL
iniciar()

#INICIAMOS LA COMUNICACIÓN CON EL ROBOT EN OTRO HILO DE EJECUCIÓN
t1=threading.Thread(target=RobotControl)
t1.start()


auto.mainloop()
```

5.5. Planos



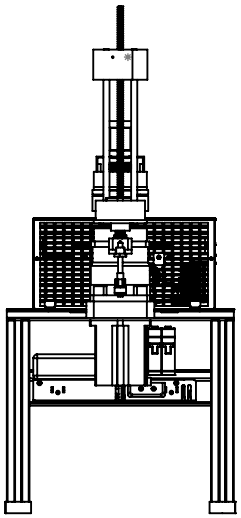


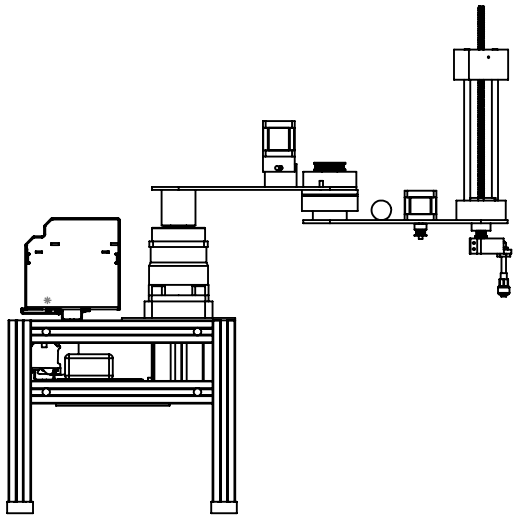


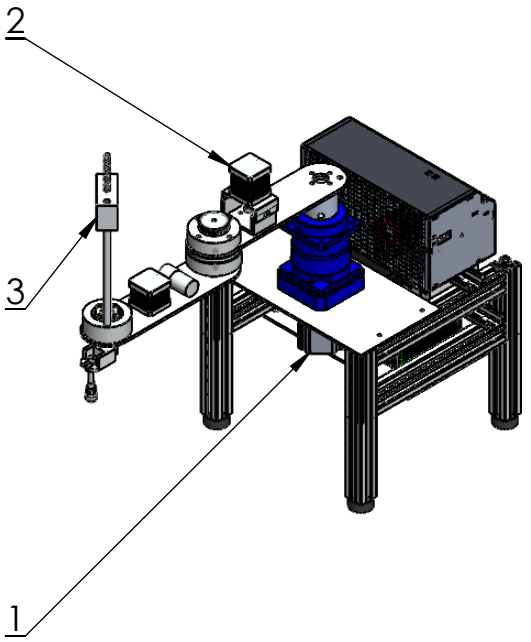


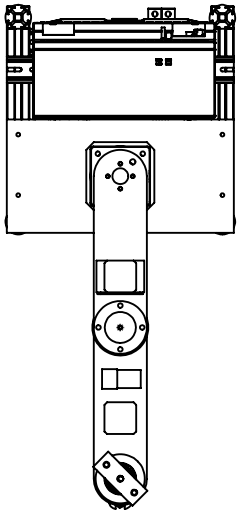
4	1	Ensamblaje cabezal rotatorio derecho	Plano 4,frabricación propia	Varios
3	1	Ensamblaje cabezal rotatorio izquierdo	Plano 5,frabricación propia	Varios
2	1	Ensamblaje cámara	Plano 3,frabricación propia	Varios
1	1	Ensamblaje robot scara	Plano 2,frabricación propia	Varios
Marca	Cantidad	Denominación	Plano y fabricante	Material

ESCALA: 1:10		FECHA	NOMBRE	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
	DIBUJÓ	06 22	Francisco Javier Navarro Escrivá		
ISO E		ENSAMBLAJE CELDA ROBOTIZADA			PLANO Nº 1
					HOJA 1/39

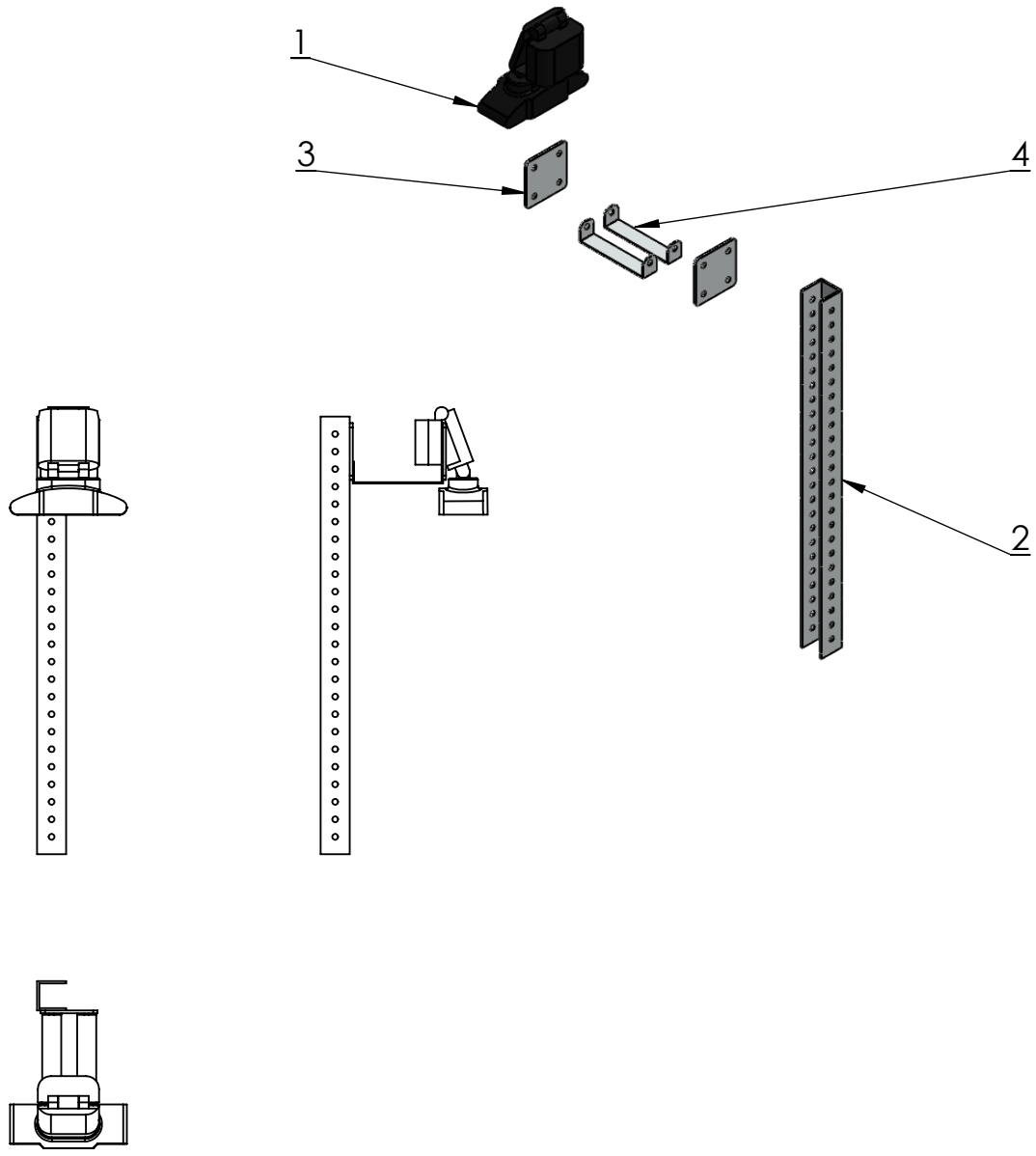




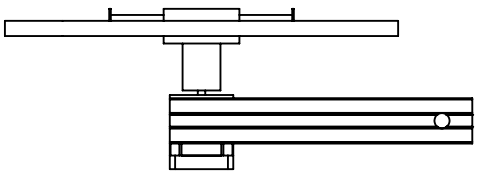


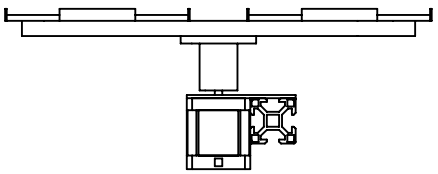


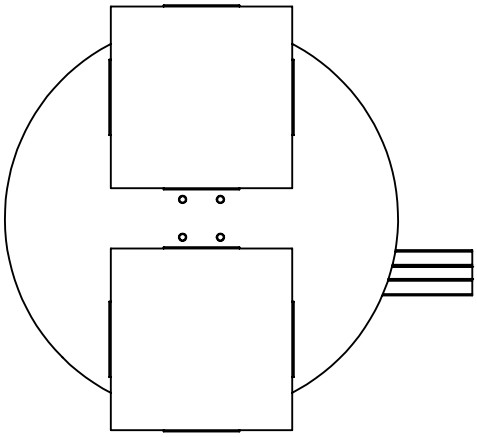
3	1	ENSAMBLAJE EJE 3		Plano 10,frabricación propia	Varios
2	1	ENAMBLAJE EJE 2		Plano 9,frabricación propia	Varios
1	1	ENSAMBLAJE EJE 1		Plano 8,frabricación propia	Varios
Marca	Cantidad	Denominación		Plano y fabricante	Material
ESCALA: 1:10	DIBUJÓ	FECHA: 06 22	NOMBRE: Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
ISO E		ENSAMBLAJE ROBOT SCARA			PLANO Nº 2
					HOJA 2/39

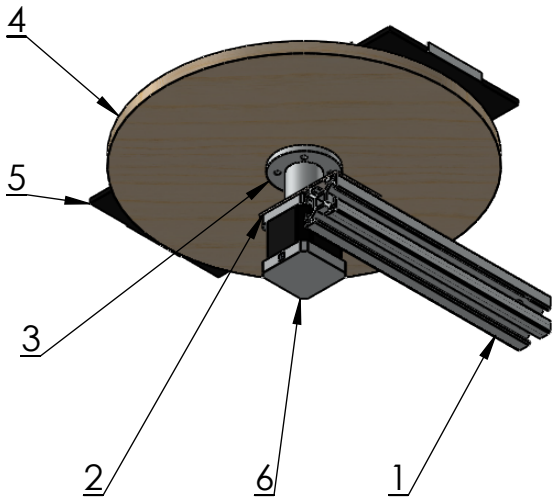


4	2	Escuadra		Plano 31,frabricación propia	Aluminio
3	2	Placa cámara		Plano 32,frabricación propia	Aluminio
2	1	Perfil en U taladrado		Plano 30,frabricación propia	Aluminio
1	1	Webcam Amdis Conceptronics		Amazon store	Varios
Marca	Cantidad	Denominación		Plano y fabricante	Material
ESCALA: 1:5	DIBUJÓ	FECHA: 06 22	NOMBRE: Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
ISO E		ENSAMBLAJE CÁMARA			PLANO Nº 3
					HOJA 3/39



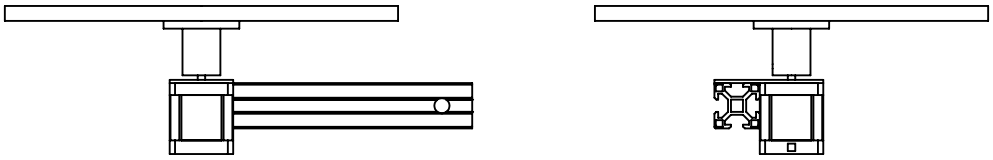


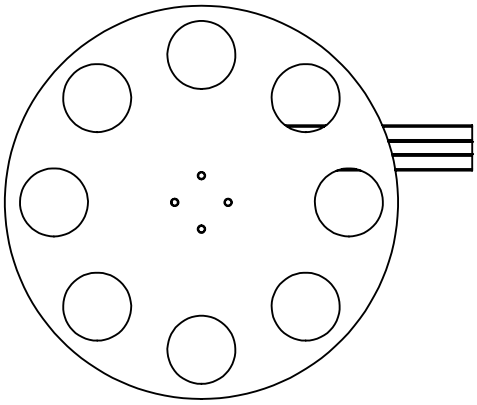


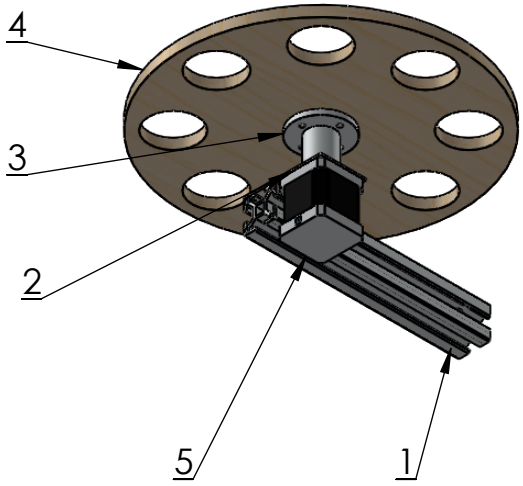


6	1	Motor NEMA 17 17HS5412-3	RS componentes	Varios
5	2	Ensamblaje soporte caja	Plano 37,fabricación propia	Aluminio
4	1	Plato portacajas	Plano 35,frabricación propia	Aluminio
3	1	Acoplamiento plato	Plano 38,frabricación propia	Aluminio
2	1	Pletina soporte motor	Plano 36,frabricación propia	Aluminio
1	1	Perfil Estructural 30x30 L-200mm	Plano 33,frabricación propia	Aluminio
Marca	Cantidad	Denominación	Plano y fabricante	Material

ESCALA:		FECHA	NOMBRE	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
1:5	DIBUJÓ	06 22	Francisco Javier Navarro Escrivá		
		ENSAMBLAJE CABEZAL GIRATORIO DERECHO			PLANO Nº 4
ISO E					HOJA 4/39

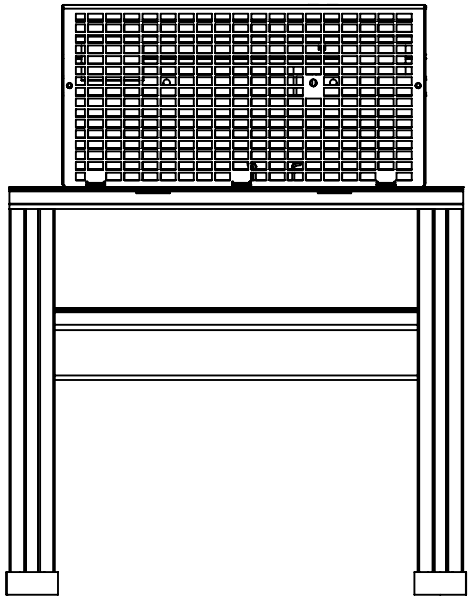


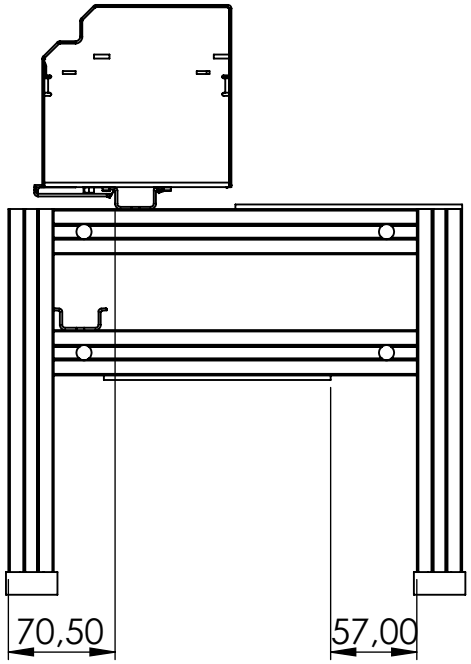


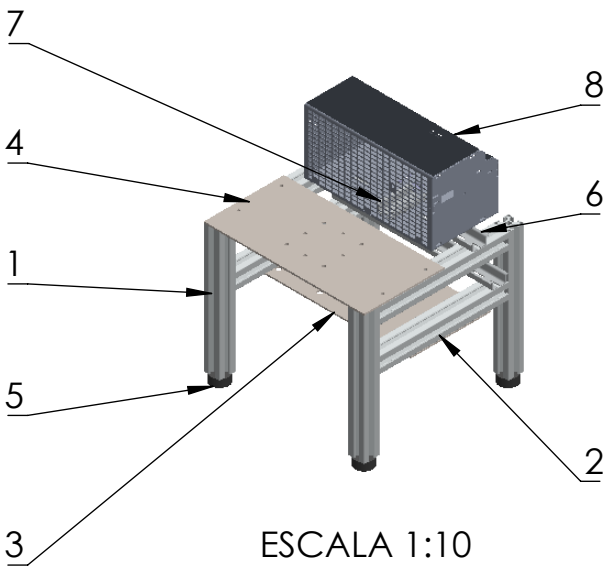


5	1	Motor NEMA 17 17HS5412-3	RS componentes	Varios
4	1	Plato portacapsulas	Plano 34,frabricación propia	Aluminio
3	1	Acoplamiento plato	Plano 38,frabricación propia	Aluminio
2	1	Pletina soporte motor	Plano 36,frabricación propia	Aluminio
1	1	Perfil Estructural 30x30 L-200mm	Plano 33,frabricación propia	Aluminio
Marca	Cantidad	Denominación	Plano y fabricante	Material

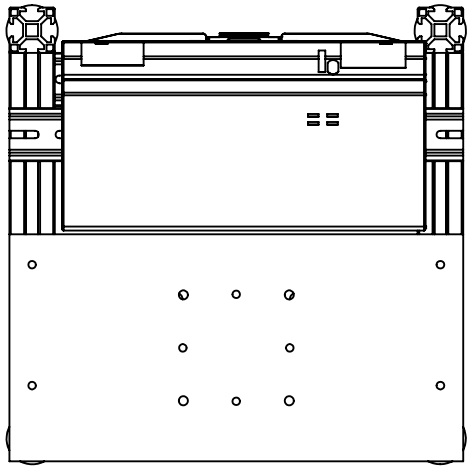
ESCALA:		FECHA	NOMBRE	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
1:5	DIBUJÓ	06 22	Francisco Javier Navarro Escrivá		
ISO E		ENSAMBLAJE CABEZAL GIRATORIO IZQUIERDO			PLANO Nº 5
					HOJA 5/39





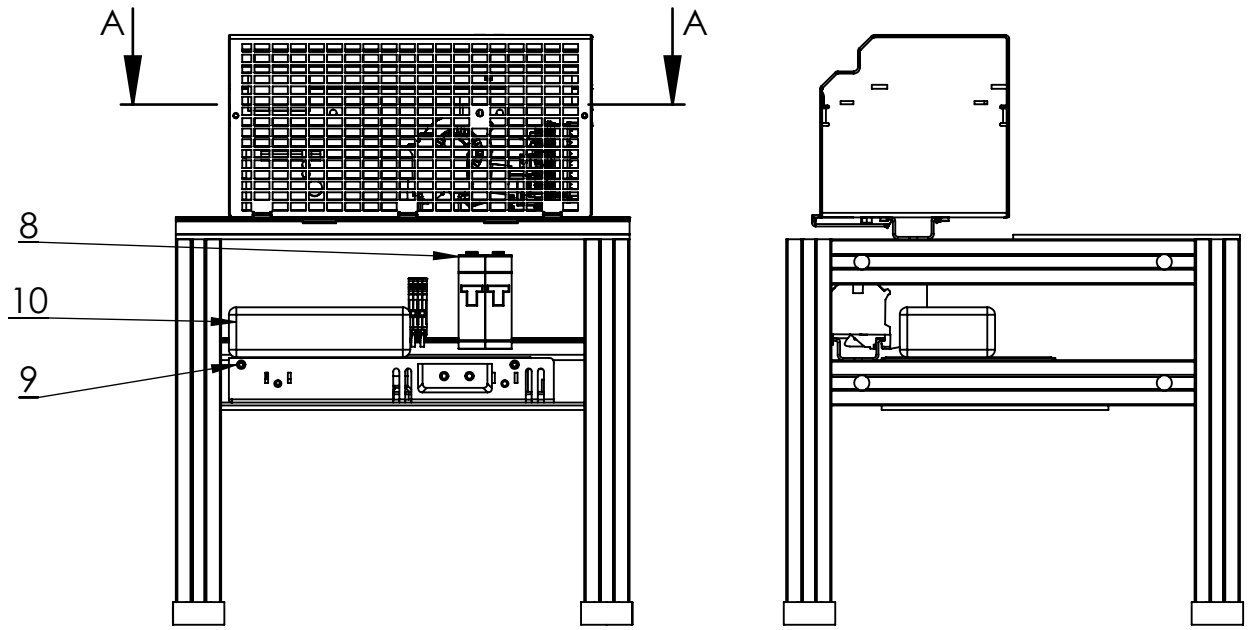


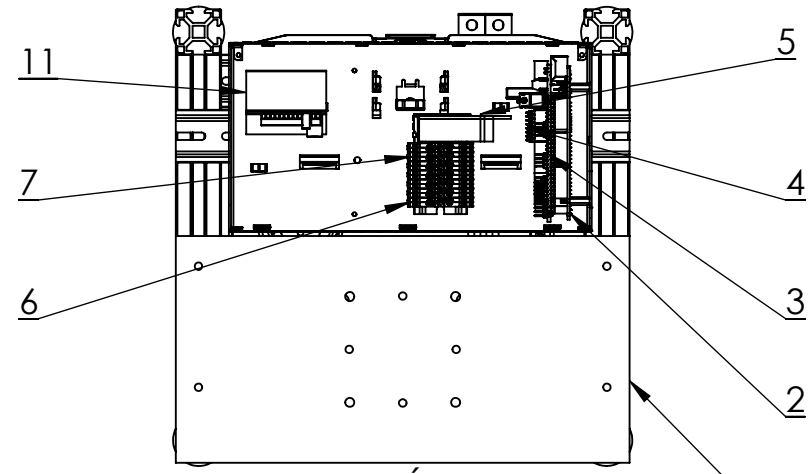
ESCALA 1:10



8	1	Caja de aluminio	Amazon store	Aluminio
7	1	Carril DIN -64mm	Plano 16,frabricación propia	Carril DIN
6	2	Carril DIN L-300mm	Plano 15,frabricación propia	Carril DIN
5	4	Pata de goma	Amazon store	Goma
4	1	Pletina superior base	Plano 14,frabricación propia	Acero
3	1	Pletina inferior base	Plano 13,frabricación propia	Acero
2	4	Perfil Estructural 30x30 L-240mm taladrado	Plano 11,frabricación propia	Perfil Estructural 30x30
1	4	Perfil Estructural 30x30 L-240mm	Plano 12,frabricación propia	Perfil Estructural 30x30
Marca	Cantidad	Denominación	Plano y fabricante	Material

ESCALA: 1:5		FECHA	NOMBRE	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
	DIBUJÓ	06 22	Francisco Javier Navarro Escrivá		
ISO E	ENSAMBLAJE BASE				PLANO Nº 6
					HOJA 6/39

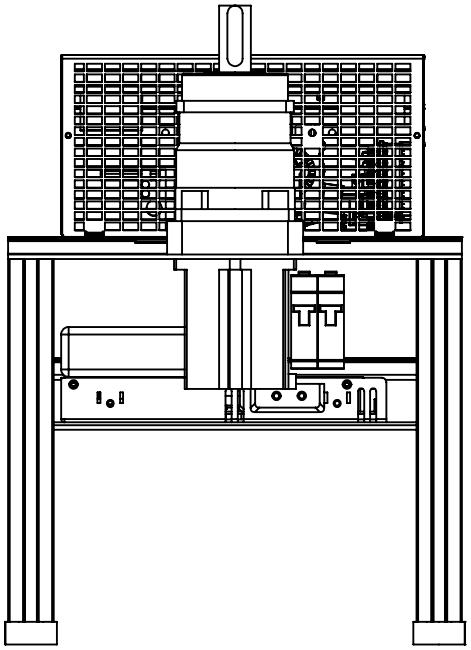


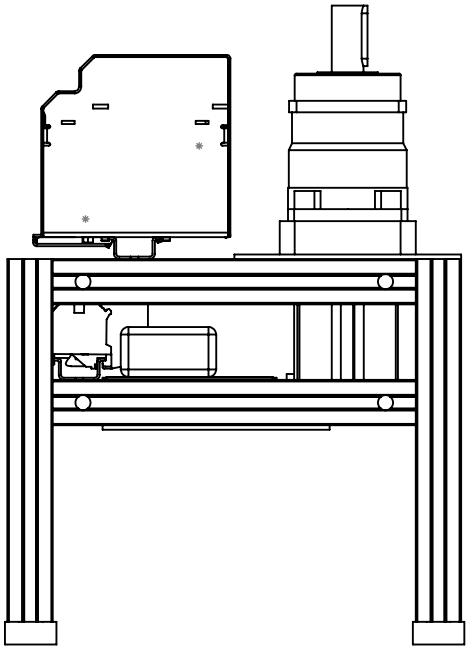


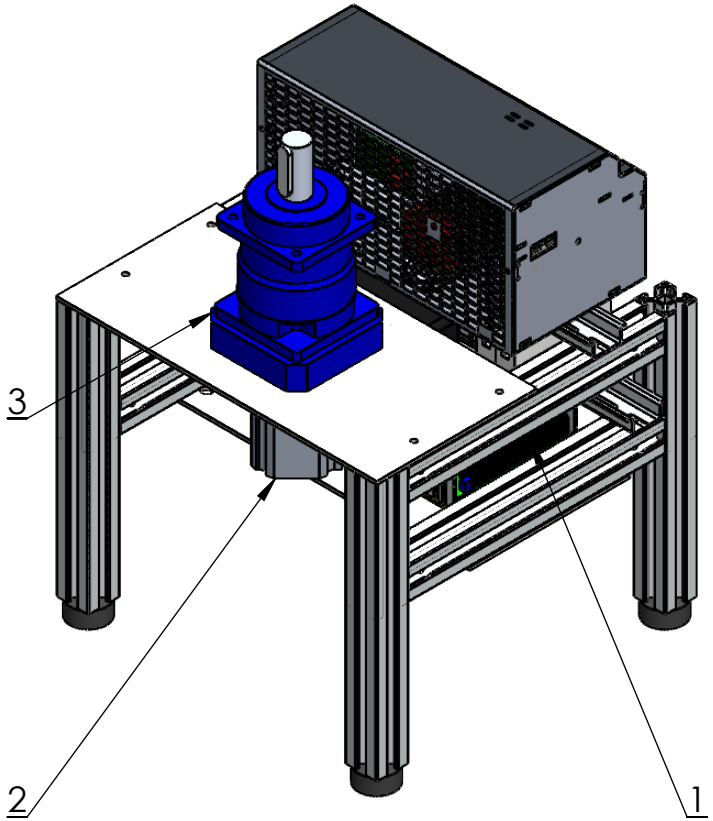
SECCIÓN A-A
ESCALA 1 : 5

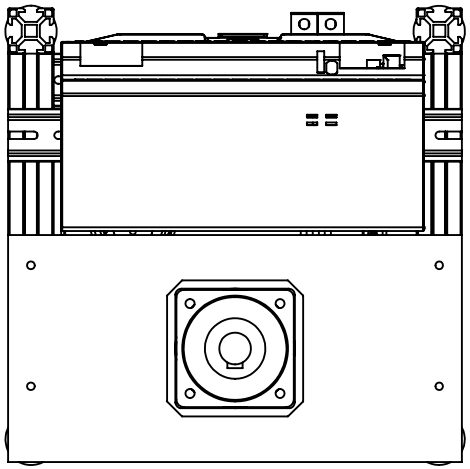
11	1	Driver SANMOTION	SANMOTION store	Varios
10	1	Transformador CWT PAA060F	Ebay store	Varios
9	1	MeanWell RSP-320-24 24Vdc	MeanWell store	Varios
8	2	Siemens 5SX2 MAX 277 AC C15	RS componentes	Varios
7	5	Terminal de carril DIN negro	RS componentes	Varios
6	2	Terminal de carril DIN rojo	RS componentes	Varios
5	1	Ventilador centrífugo EC5015M12S	Amazon store	Varios
4	4	Driver DRV8825	Amazon store	Varios
3	1	Ramps 1.4	Amazon store	Varios
2	1	Arduino Mega 2560	Arduino store	Varios
1	1	ENSAMBLAJE BASE	Plano 6,frabricación propia	Varios
Marca	Cantidad	Denominación	Plano y fabricante	Material

ESCALA:		FECHA	NOMBRE	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
1:5	DIBUJÓ	06 22	Francisco Javier Navarro Escrivá		
		ENSAMBLAJE ELECTRÓNICA			PLANO Nº 7
ISO E					HOJA 7/39









Marca	Cantidad	Denominación	Plano y fabricante	Material
3	1	Reductor Planetario 075-MF1-7-131-000	RS componentes	Varios
2	1	Motor NEMA 23 103H7823-5740	RS componentes	Varios
1	1	ENSAMBLAJE ELECTRÓNICA	Plano 7,frabricación propia	Varios

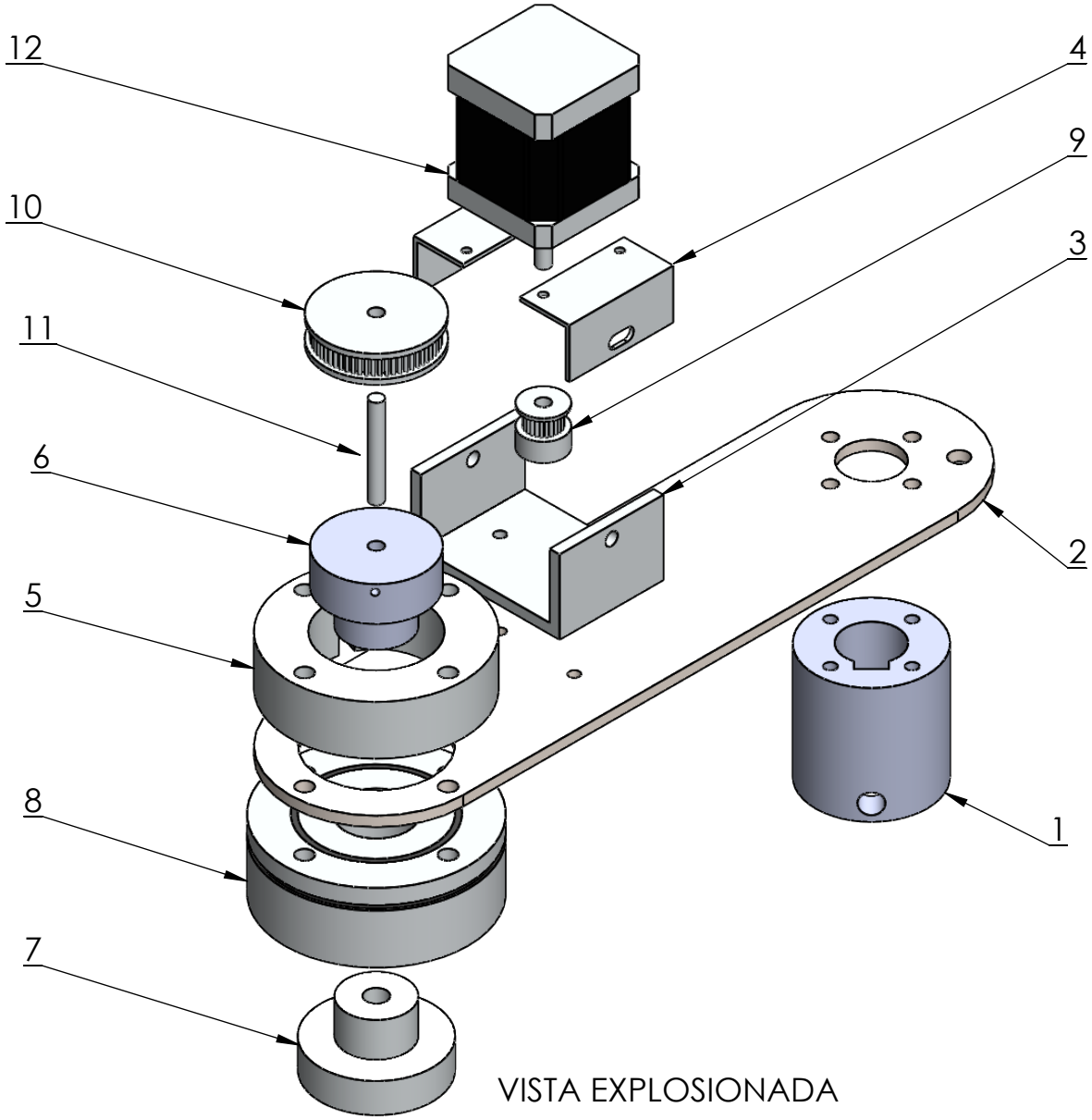
ESCALA:		FECHA	NOMBRE	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
1:5	DIBUJÓ	06 22	Francisco Javier Navarro Escrivá		

ISO E

ENSAMBLAJE EJE 1

PLANO Nº 8

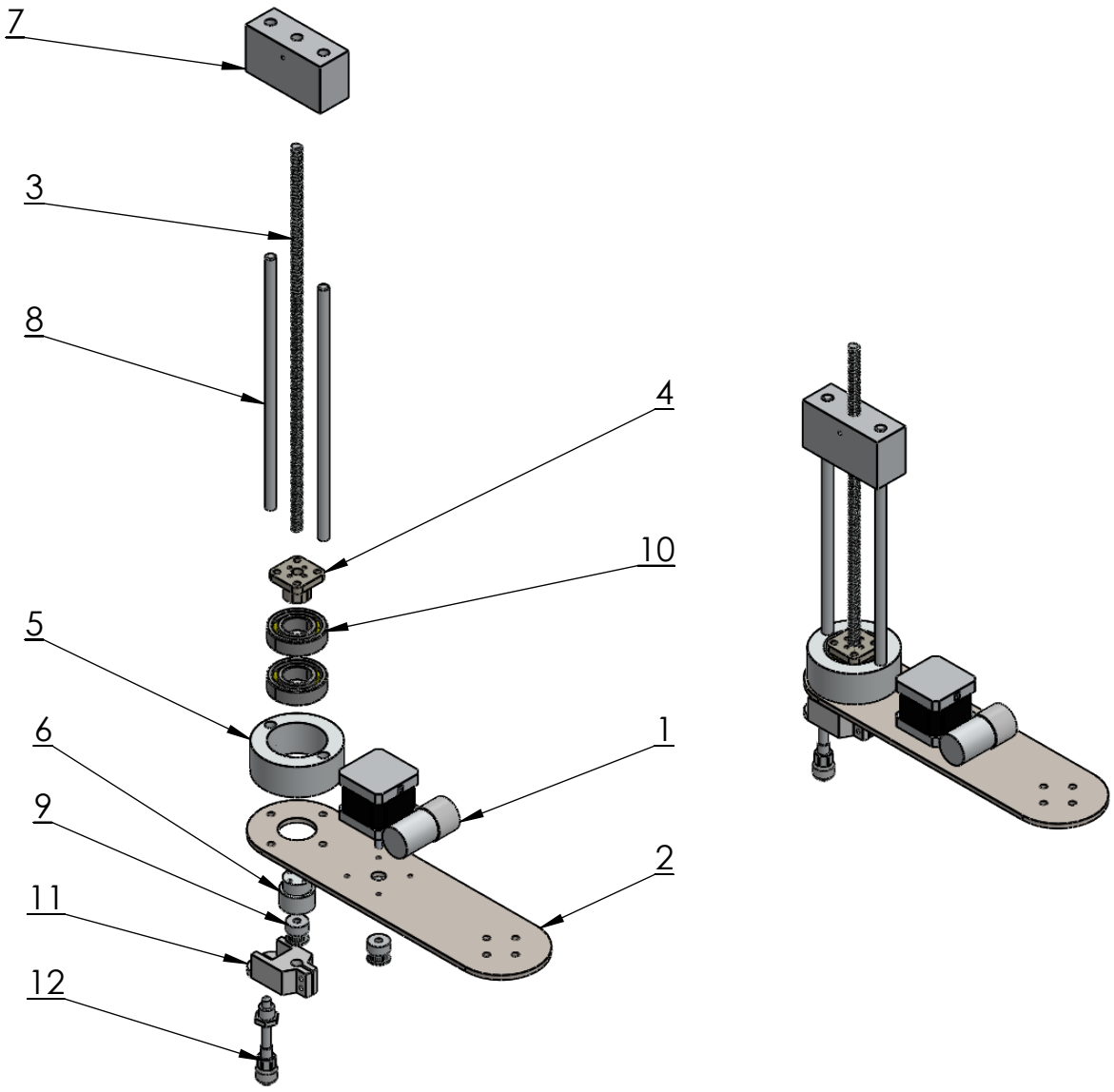
HOJA 8/39



VISTA EXPLOSIONADA

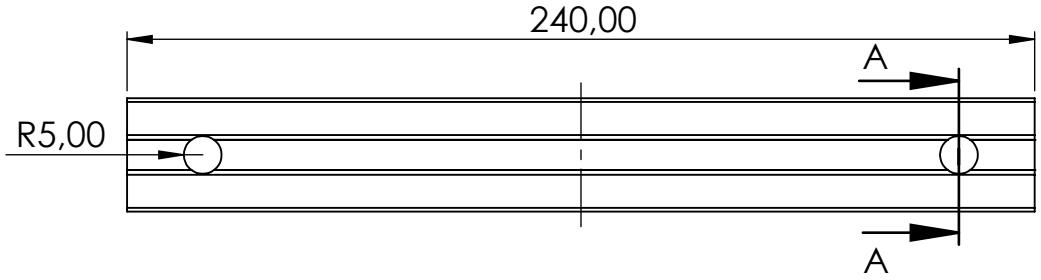
12	1	Motor NEMA 17 17HS5412-3	RS Componentes	Varios
11	1	Eje corto eje 2	Plano 21,frabricación propia	Aluminio
10	1	Polea dentada 60 dientes	Amazon store	Aluminio
9	1	Polea dentada 20 dientes	Amazon store	Aluminio
8	1	Rodamiento axial radial NSK	RS componentes	Varios
7	1	Cierre inferior eje 2	Plano 24,frabricación propia	Aluminio
6	1	Cierre superior eje 2	Plano 23,frabricación propia	Aluminio
5	1	Suplemento eje 2	Plano 22,fabricación propia	Aluminio
4	2	Soporte motor en L	Plano 20,frabricación propia	Aluminio
3	1	Soporte motor en U	Plano 19,frabricación propia	Aluminio
2	1	Pletina eje 1 a 2	Plano 18,frabricación propia	Aluminio
1	1	Acoplamiento con chaveta	Plano 17,frabricación propia	Aluminio
Marca	Cantidad	Denominación	Plano y fabricante	Material

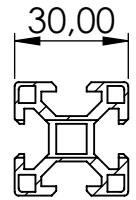
ESCALA:		FECHA	NOMBRE	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
1:2	DIBUJÓ	06 22	Francisco Javier Navarro Escrivá		
		ENSAMBLAJE EJE 2			PLANO Nº 9
ISO E					HOJA 9/39



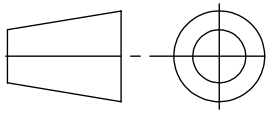
12	1	Ventosa	Amazon store	Varios
11	1	Portaherramientas	Plano 29,frabricación propia	PLA
10	2	Rodamiento SKF 6004	RS componentes	Varios
9	2	Polea dentada 20 dientes	Amazon store	Aluminio
8	2	Guia eje Z	Plano 26,fabricación propia	Aluminio
7	1	Sujección husillo eje Z	Plano 28,frabricación propia	Aluminio
6	1	Casquillo eje Z	Plano 27,frabricación propia	Aluminio
5	1	Suplemento eje Z	Plano 39,fabricación propia	Aluminio
4	1	Tuerca igus DS8x10	Igus online store	Plástico
3	1	Husillo igus DS8x10	Igus online store	Aluminio
2	1	Pletina eje 2 a 3	Plano 25,frabricación propia	Acero
1	1	Bomba de vacio	Amazon store	Varios
Marca	Cantidad	Denominación	Plano y fabricante	Material

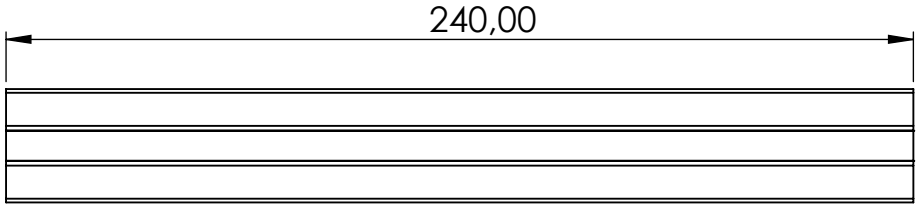
ESCALA:		FECHA	NOMBRE	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
1:5	DIBUJÓ	06 22	Francisco Javier Navarro Escrivá		
	ENSAMBLAJE EJE 3				PLANO Nº 10
ISO E					HOJA 10/39

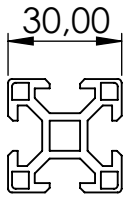


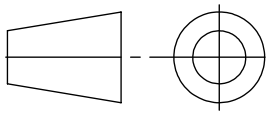


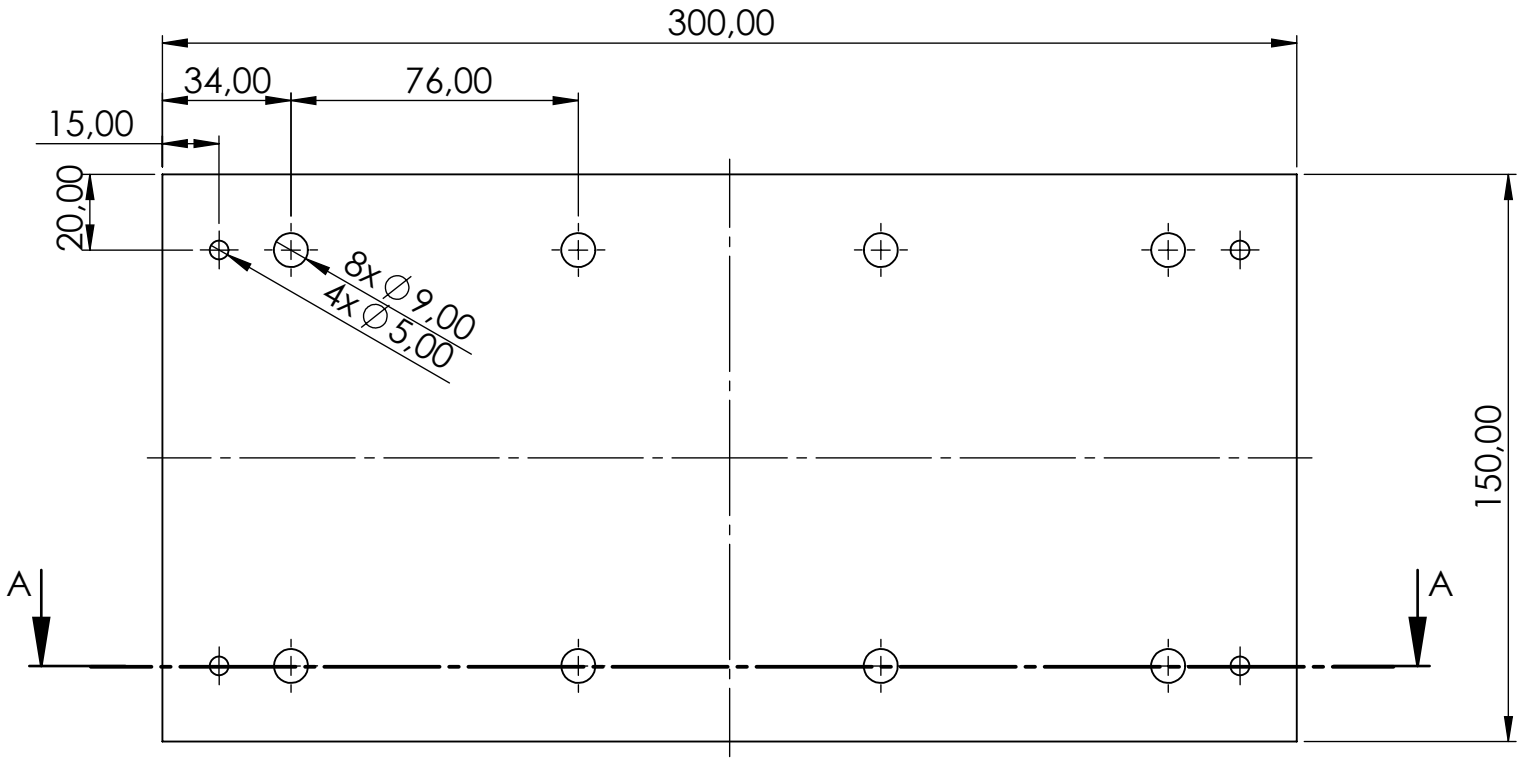
SECCIÓN A-A
ESCALA 1 : 2

ESCALA: 1:2	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
 ISO E	PERFIL ESTRUCTURAL ALUMINIO 30x30 L-240mm TALADRADO				PLANO Nº 11
					HOJA 11/39

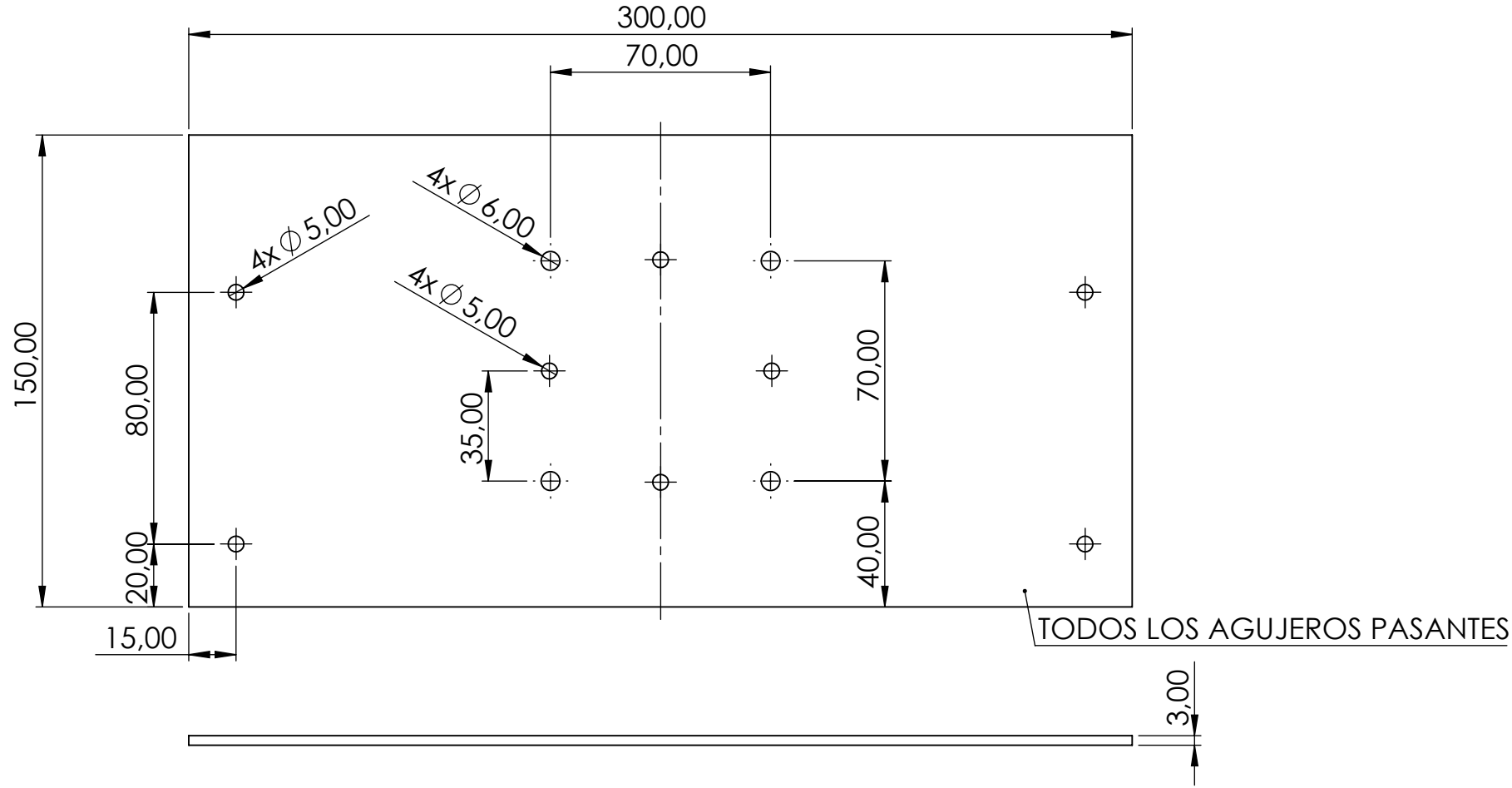




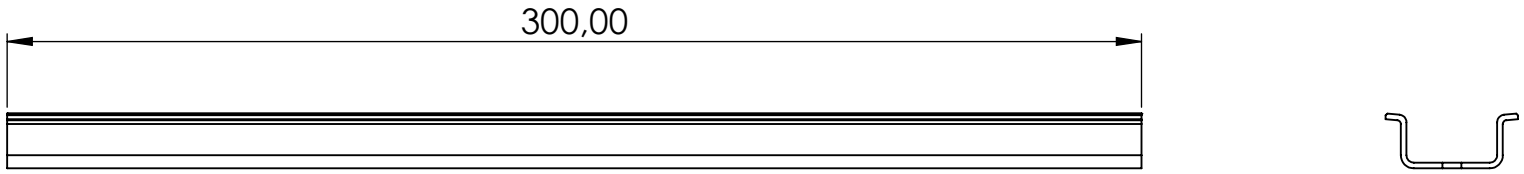
ESCALA: 1:2	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
 ISO E	PERFIL ESTRUCTURAL ALUMINIO 30x30 L-240mm				PLANO Nº 12 / HOJA 12/39



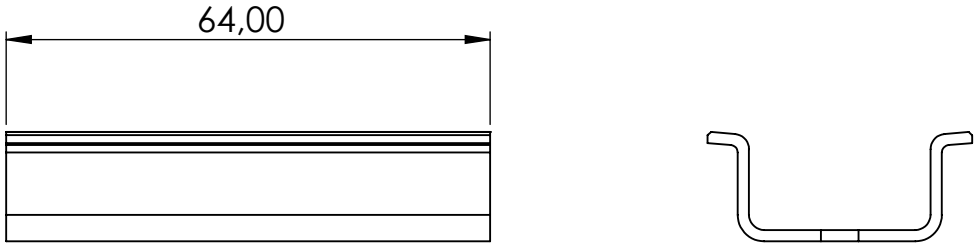
ESCALA: 1:2	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
ISO E	PLETINA INFERIOR BASE				PLANO Nº 13
					HOJA 13/38



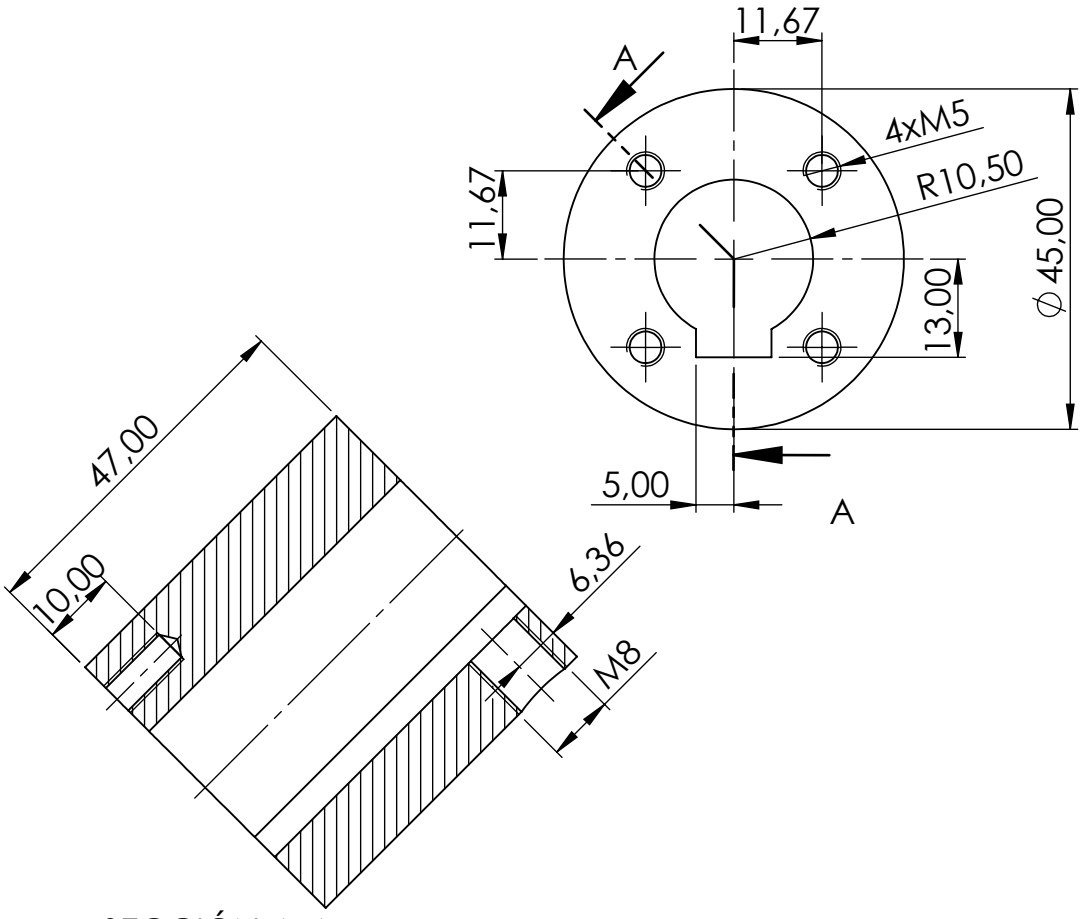
ESCALA: 1:2	DIBUJÓ	FECHA: 06 22	NOMBRE: Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
	PLETINA SUPERIOR BASE				PLANO Nº 14
ISO E					HOJA 14/39



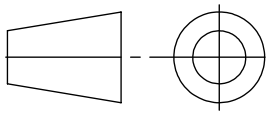
ESCALA: 1:2	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
ISO E	CARRIL DIN L-300mm				PLANO Nº 15
					HOJA 15/39

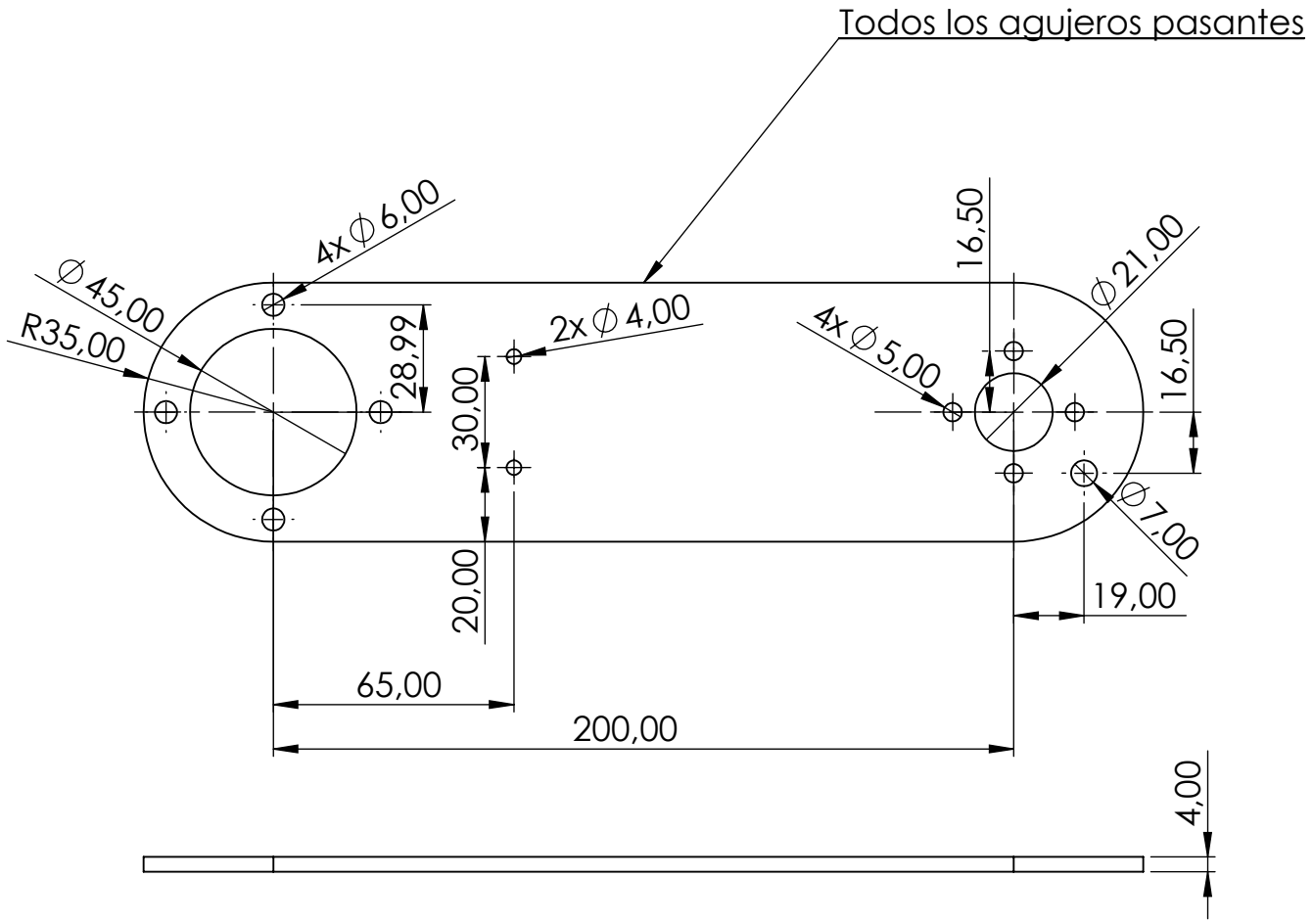


ESCALA: 1:1	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
ISO E	CARRIL DIN L-64mm				PLANO Nº 16
					HOJA 16/39

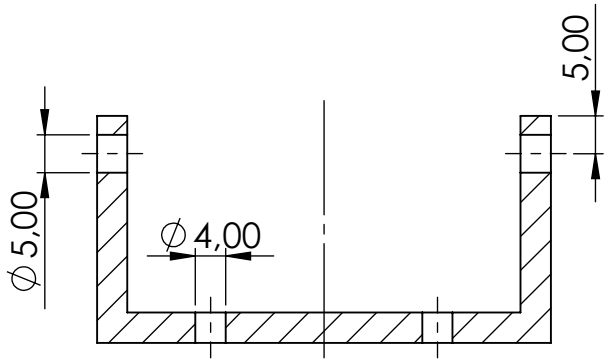


SECCIÓN A-A

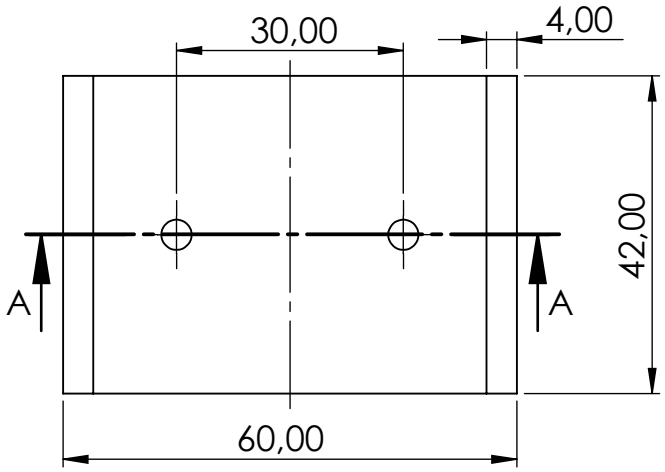
ESCALA: 1:1	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
		ACOPLAMIENTO CON CHAVETA			PLANO Nº 17
ISO E					HOJA 17/39

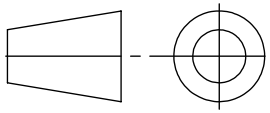


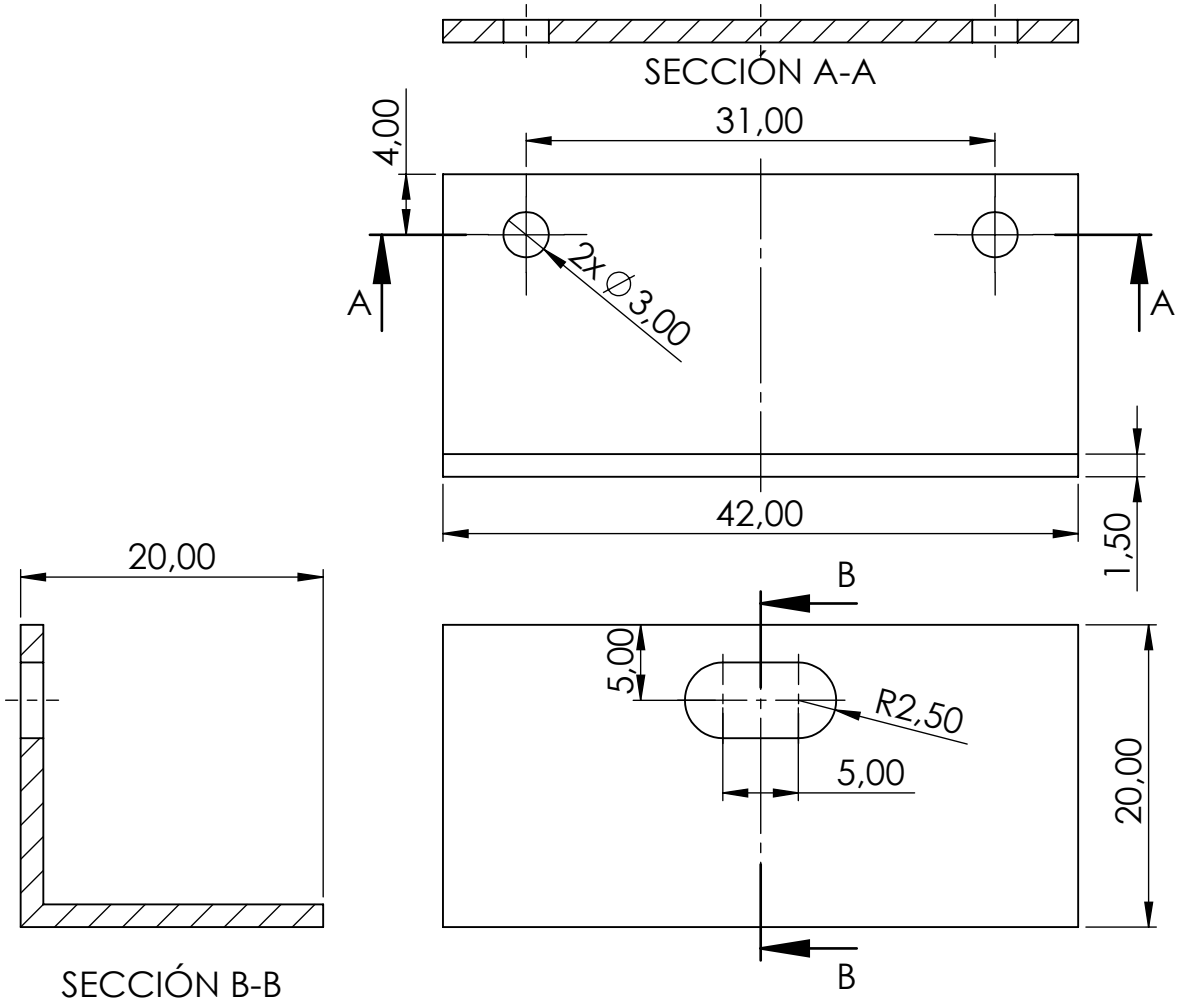
ESCALA: 1: 2	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
		PLETINA EJE 1 A 2			PLANO Nº 18
ISO E					HOJA 18/39



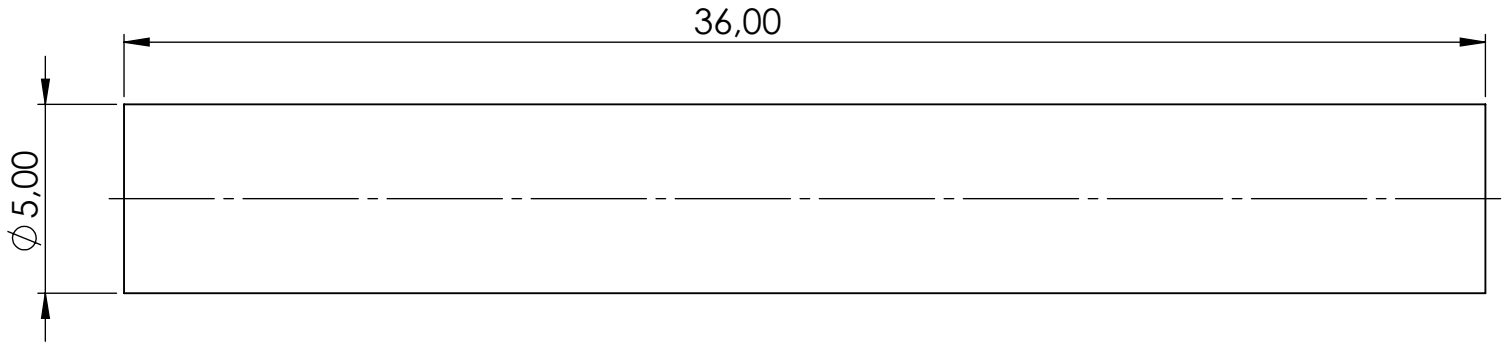
SECCIÓN A-A

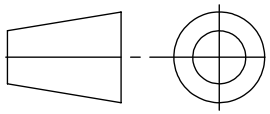


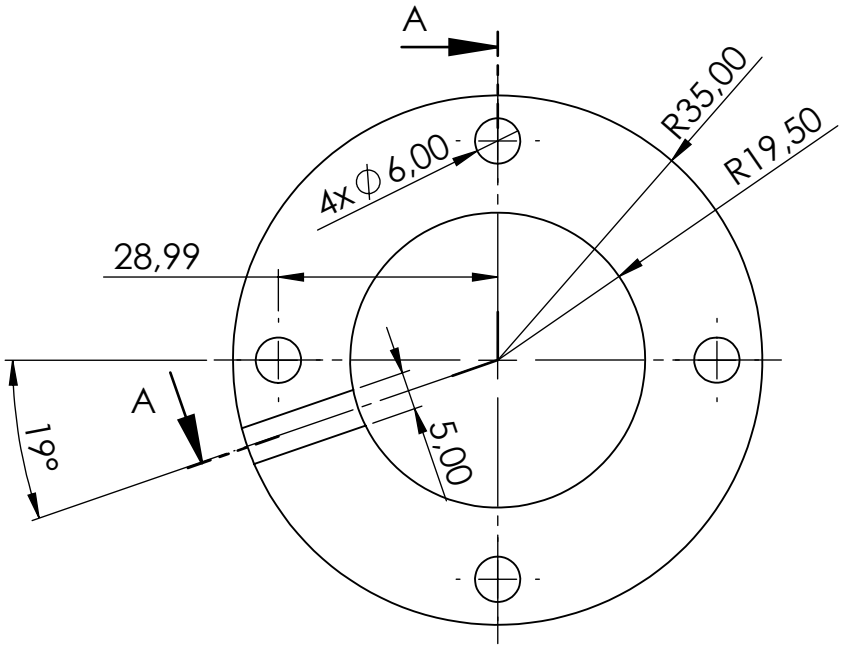
ESCALA: 1:1	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
		SOPORTE MOTOR EN U			PLANO Nº 19
ISO E					HOJA 19/39

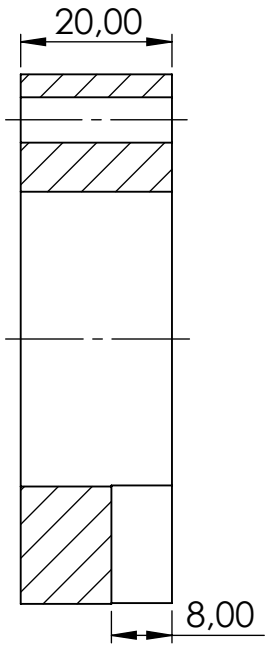


ESCALA: 2:1	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
		SOPORTE MOTOR EN L			PLANO Nº 20
ISO E					HOJA 20/39

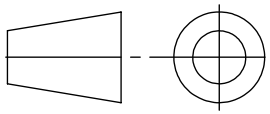


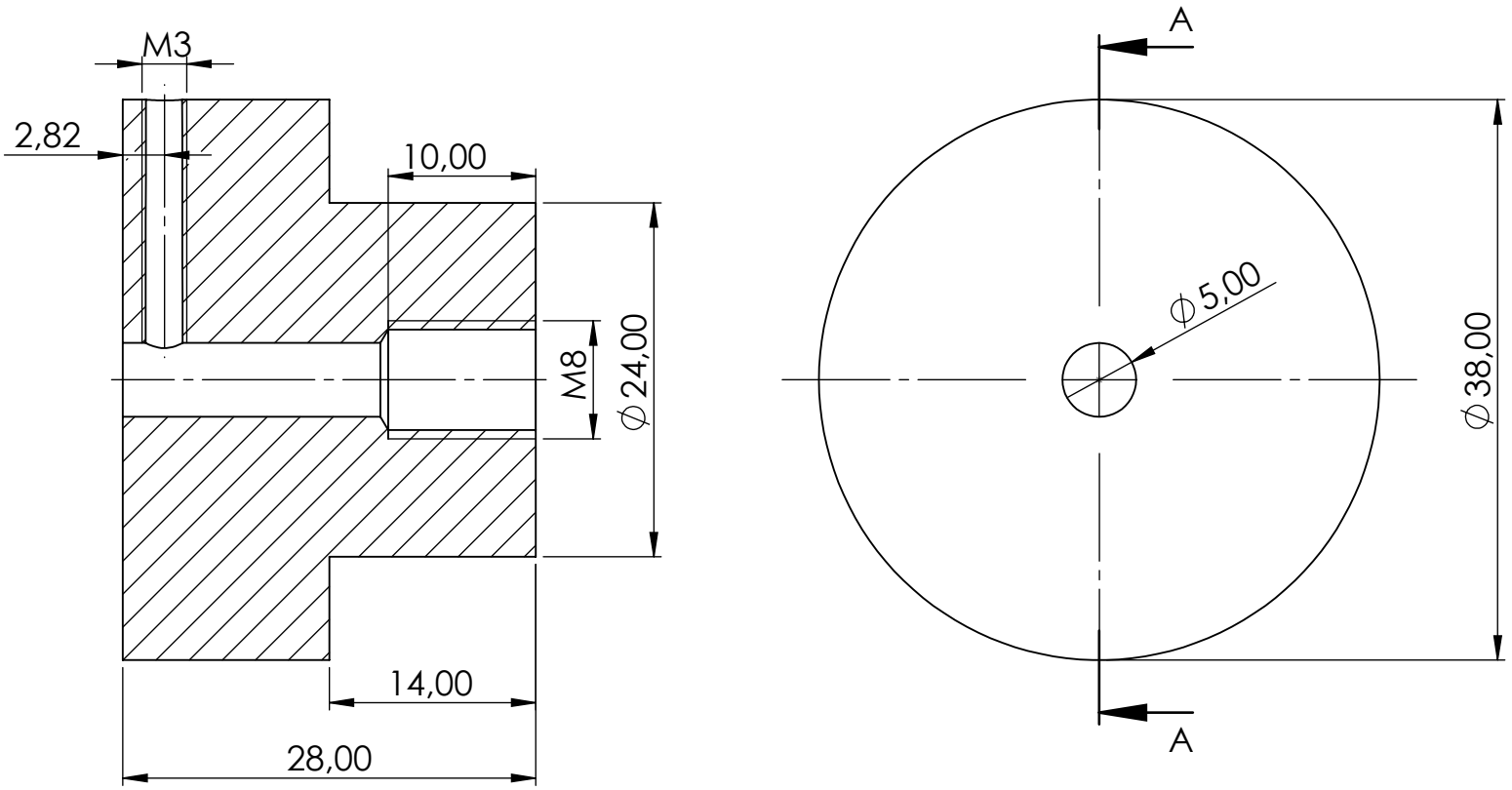
ESCALA: 1:1	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
	EJE CORTO EJE 2				PLANO Nº 21
ISO E					HOJA 21/39





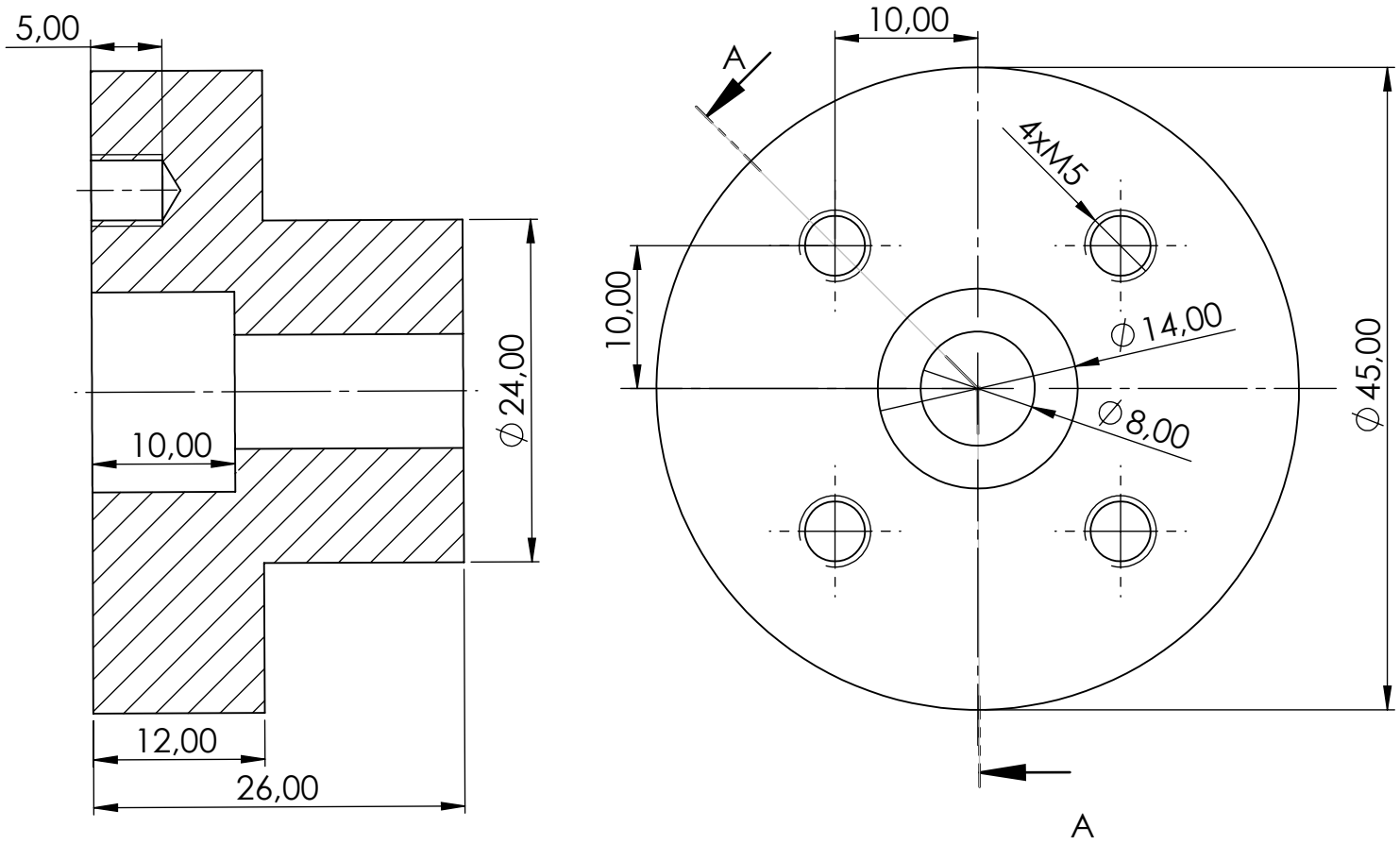
SECCIÓN A-A

ESCALA: 1:1	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
		SUPLEMENTO EJE 2			PLANO Nº 22
ISO E					HOJA 22/39



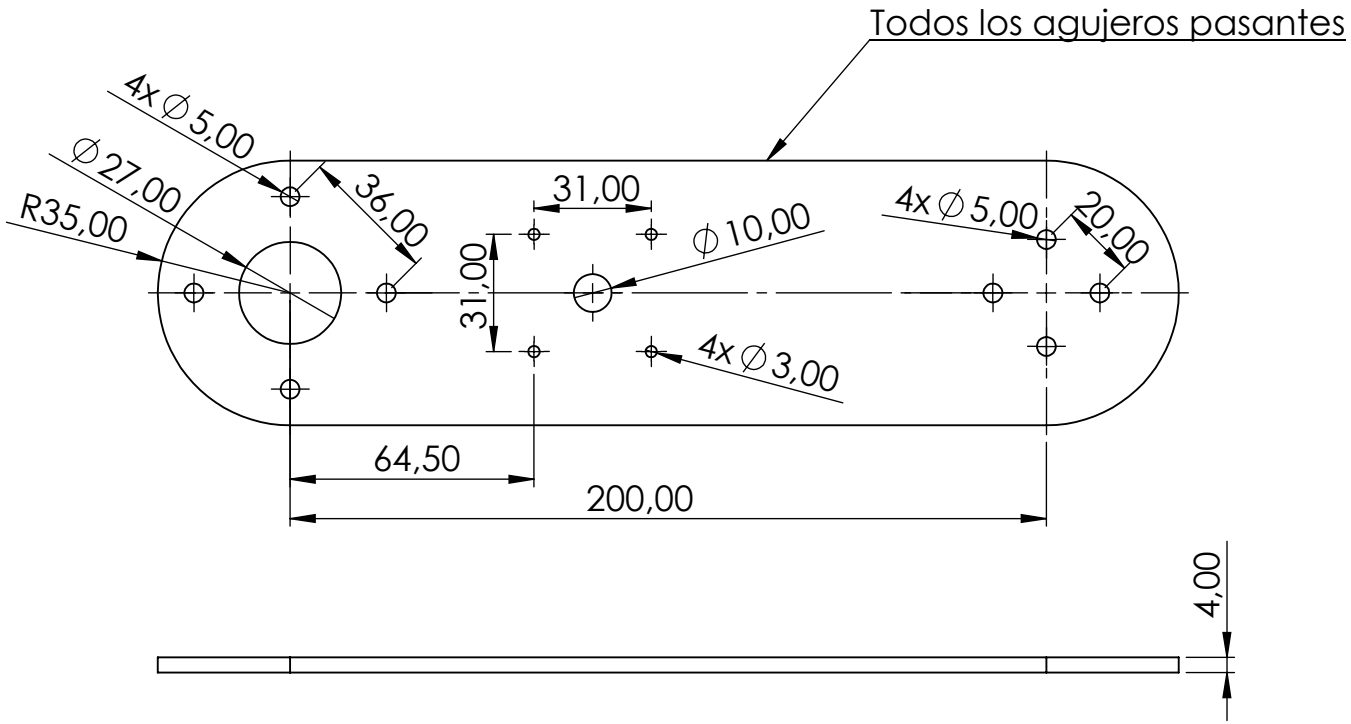
SECCIÓN A-A
ESCALA 2 : 1

ESCALA: 2:1	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
		CIERRE SUPERIOR EJE 2			PLANO Nº 23
ISO E					HOJA 23/39

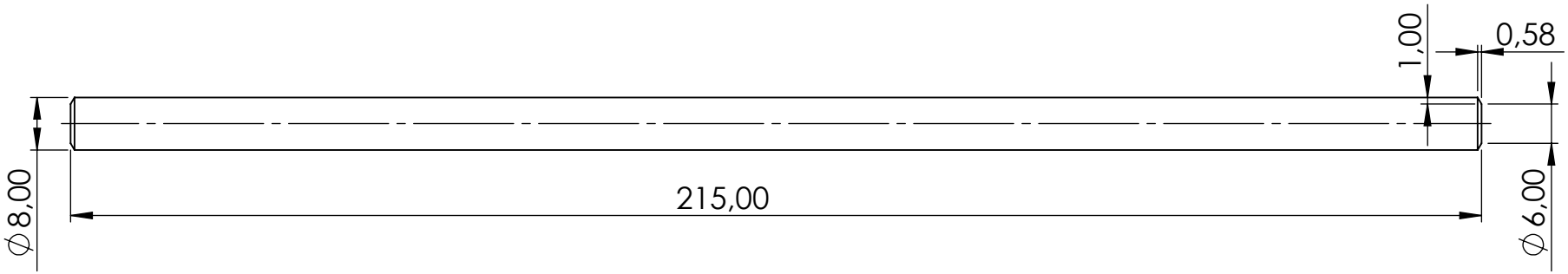


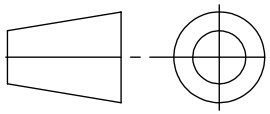
SECCIÓN A-A
ESCALA 2 : 1

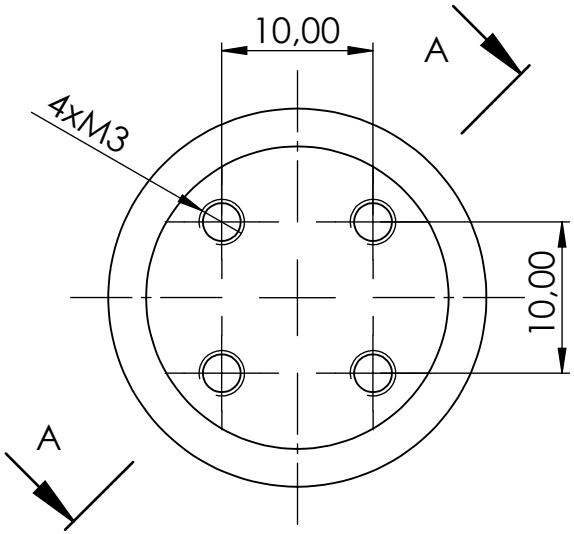
ESCALA: 2:1	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
		CIERRE INFERIOR EJE 2			PLANO Nº 24
ISO E					HOJA 24/39

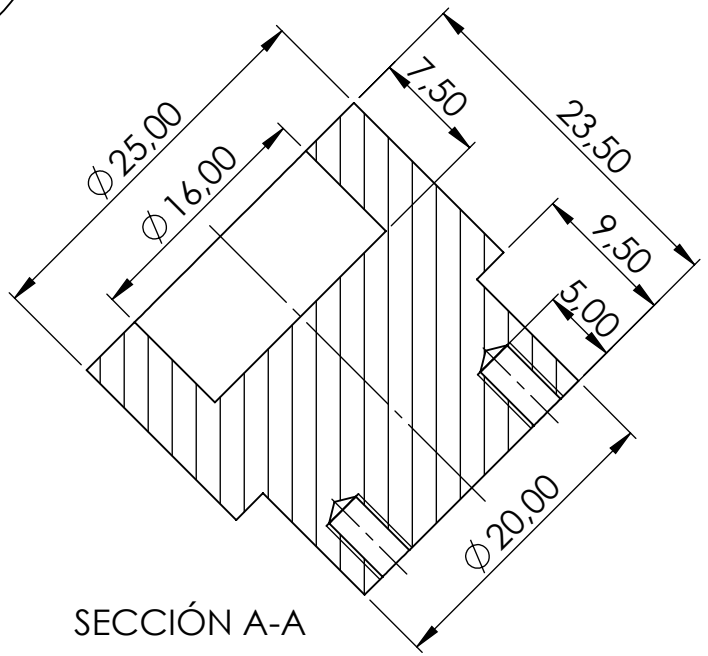


ESCALA: 1:2	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
ISO E	PLETINA EJE 2 A 3				PLANO Nº 25
					HOJA 25/39

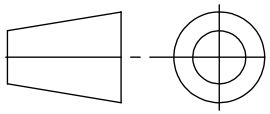


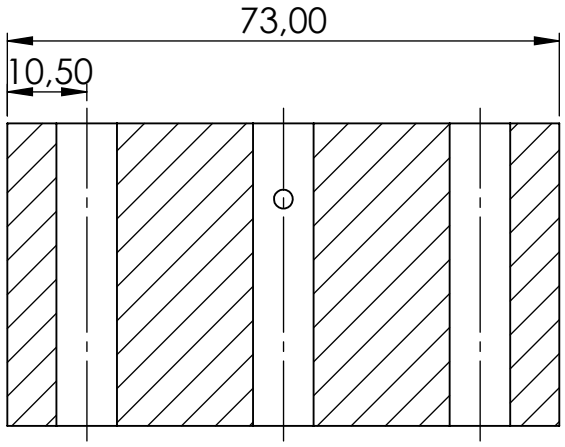
ESCALA: 1:1	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
		GUIA EJE Z			PLANO Nº 26
ISO E					HOJA 26/39



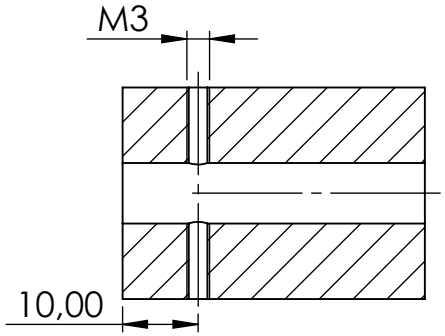


SECCIÓN A-A

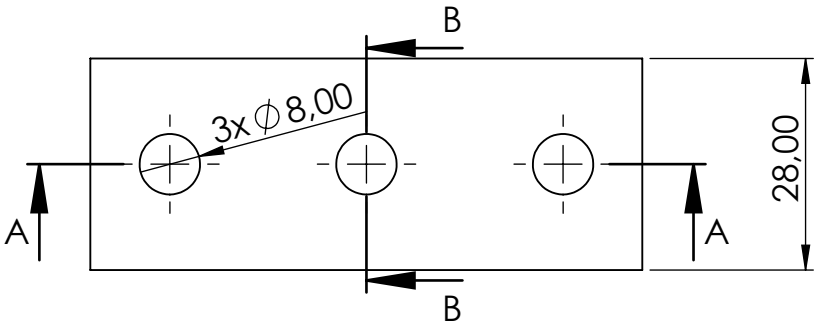
ESCALA: 2:1	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
		CASQUILLO EJE Z			PLANO Nº 27
ISO E					HOJA 27/39

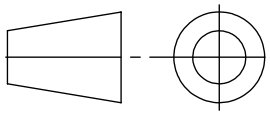


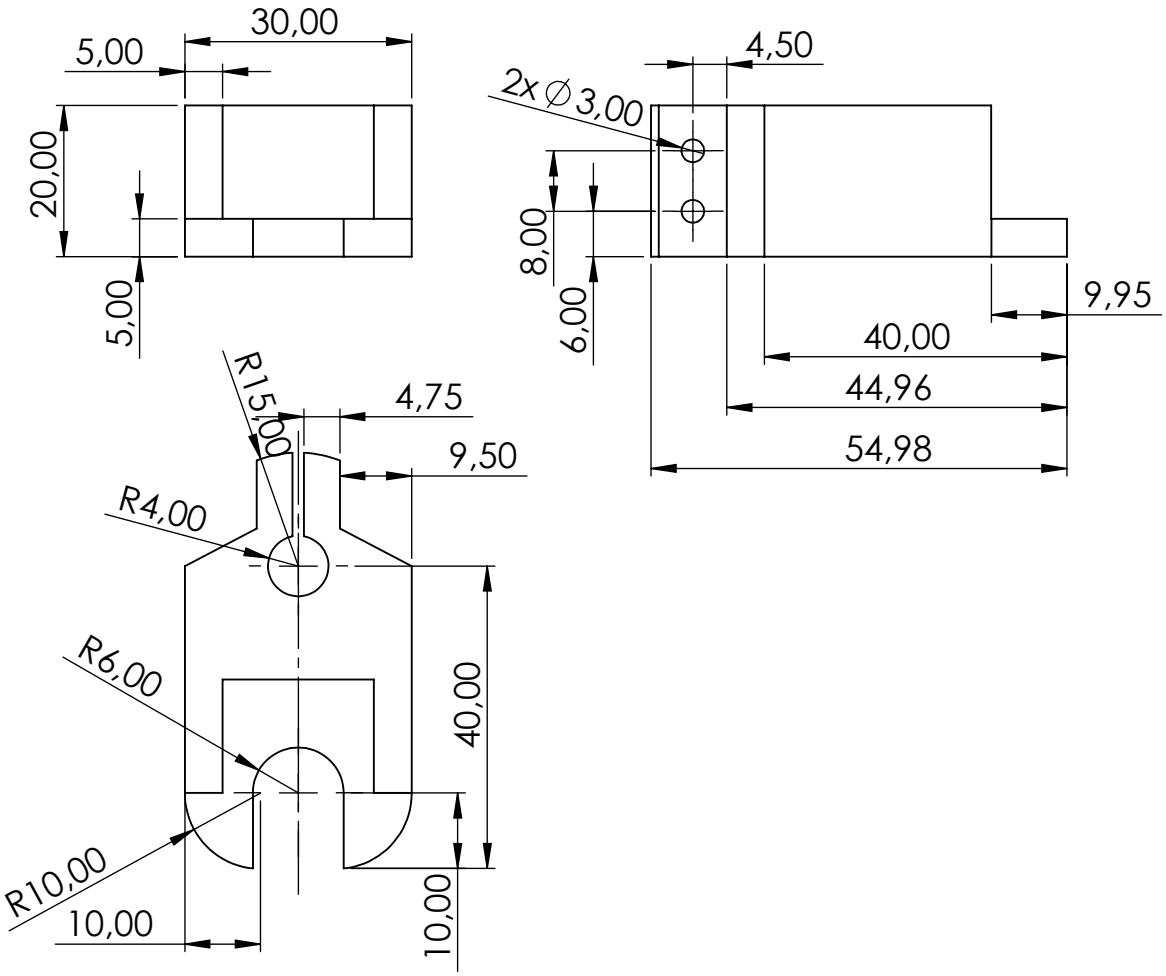
SECCIÓN A-A



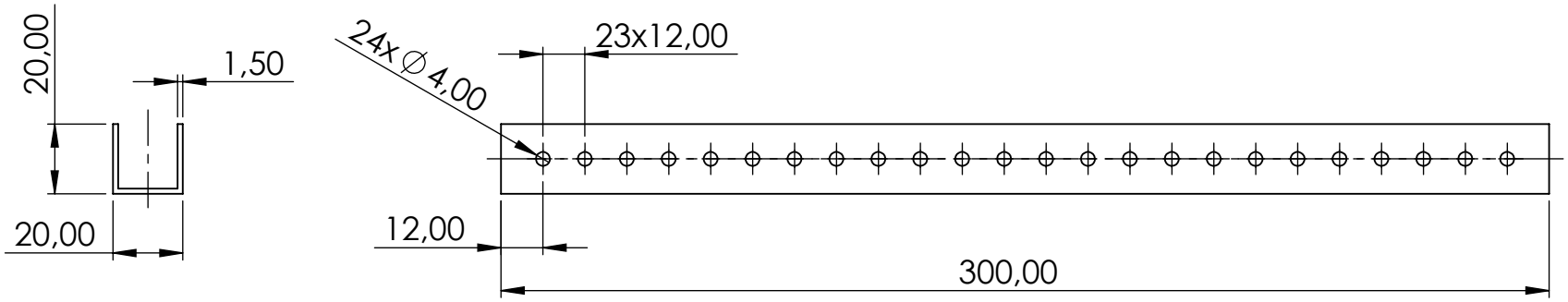
SECCIÓN B-B



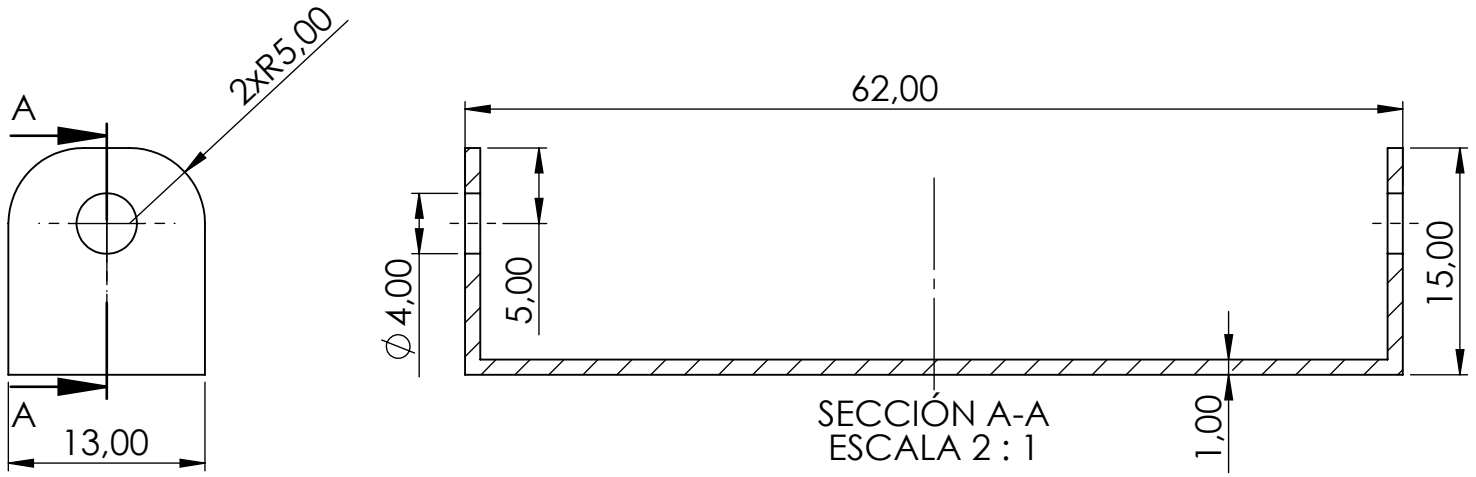
ESCALA: 1:1	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
		SUJECCIÓN HUSILLO EJE Z			PLANO Nº 28
ISO E					HOJA 28/39



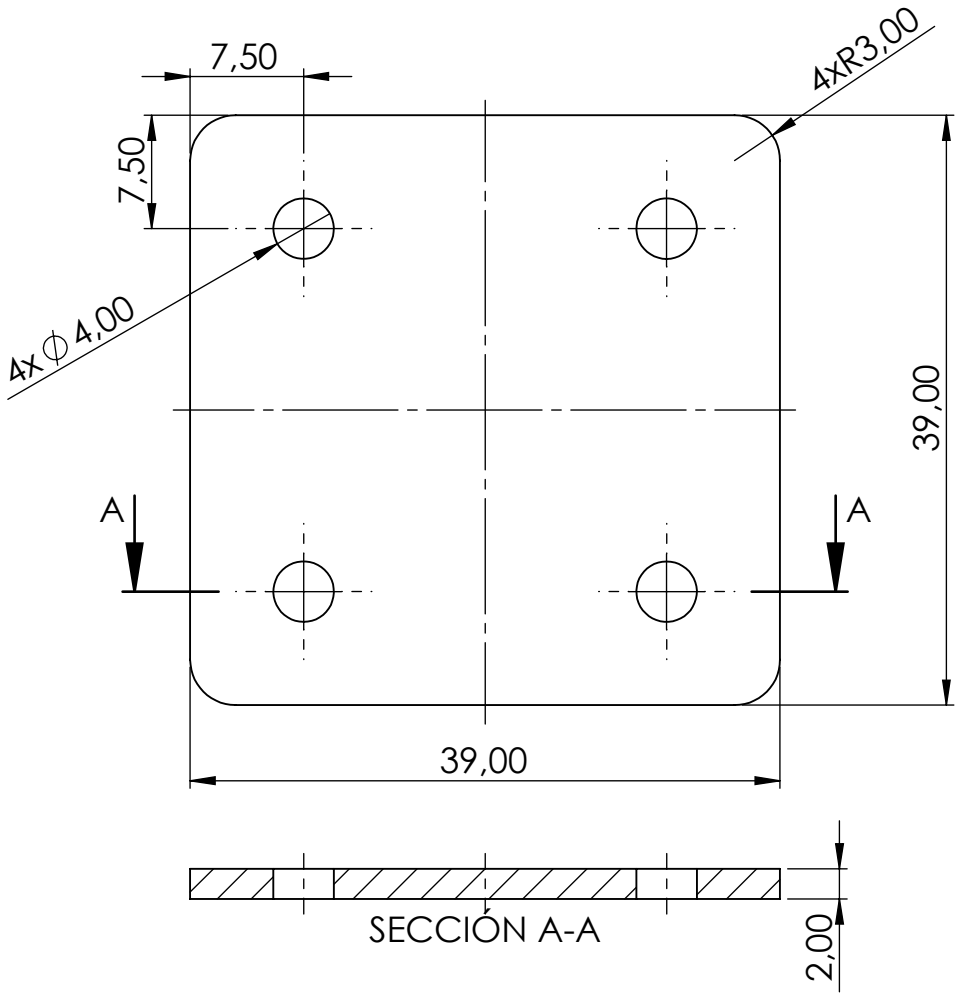
ESCALA: 1:1	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
		PORTAHERRAMIENTAS			PLANO Nº 29
ISO E					HOJA 29/39



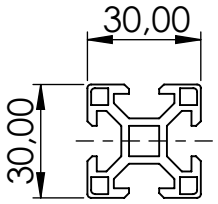
ESCALA: 1:2	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
		PERFIL EN U TALADRADO			PLANO Nº 30
ISO E					HOJA 30/39

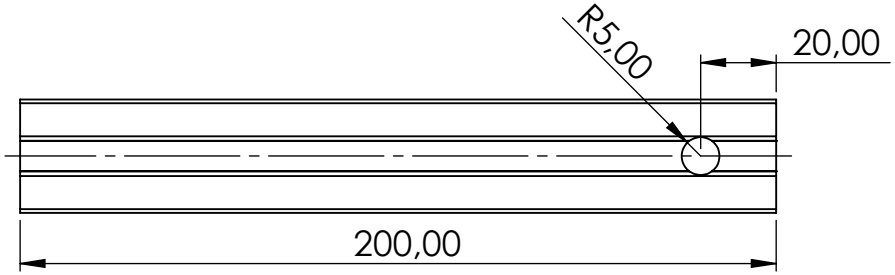


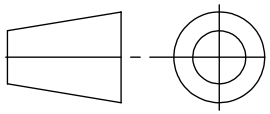
ESCALA: 2:1	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
	ESCUADRA				PLANO Nº 31
ISO E					HOJA 31/39

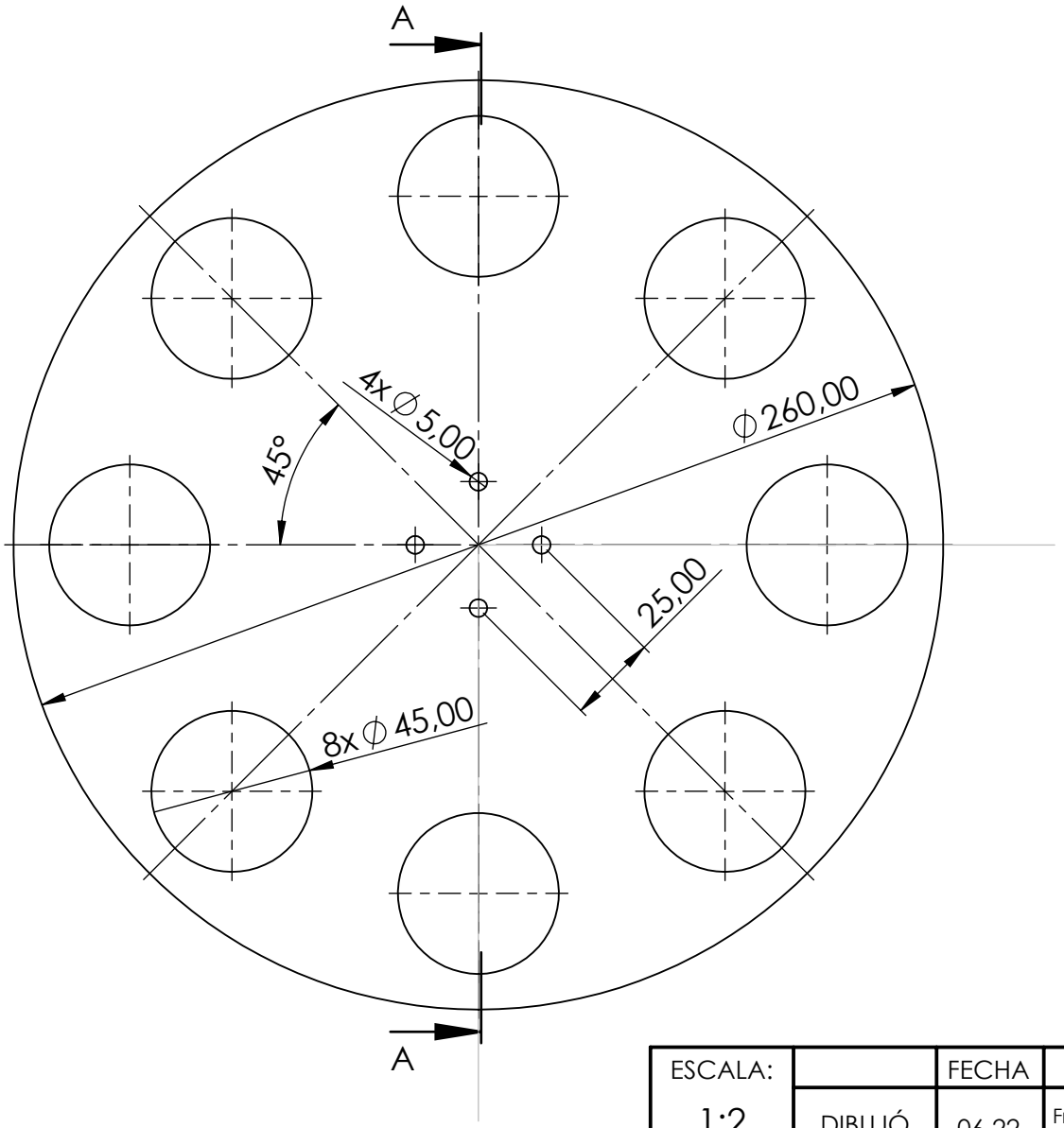


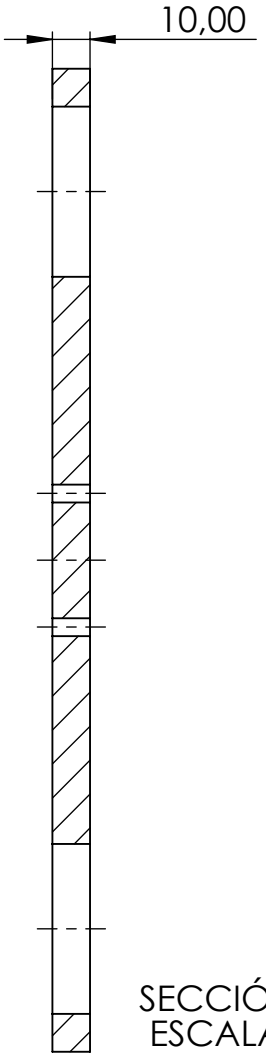
ESCALA: 2:1	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
				PLACA CÁMARA	PLANO Nº 32
ISO E					HOJA 32/39





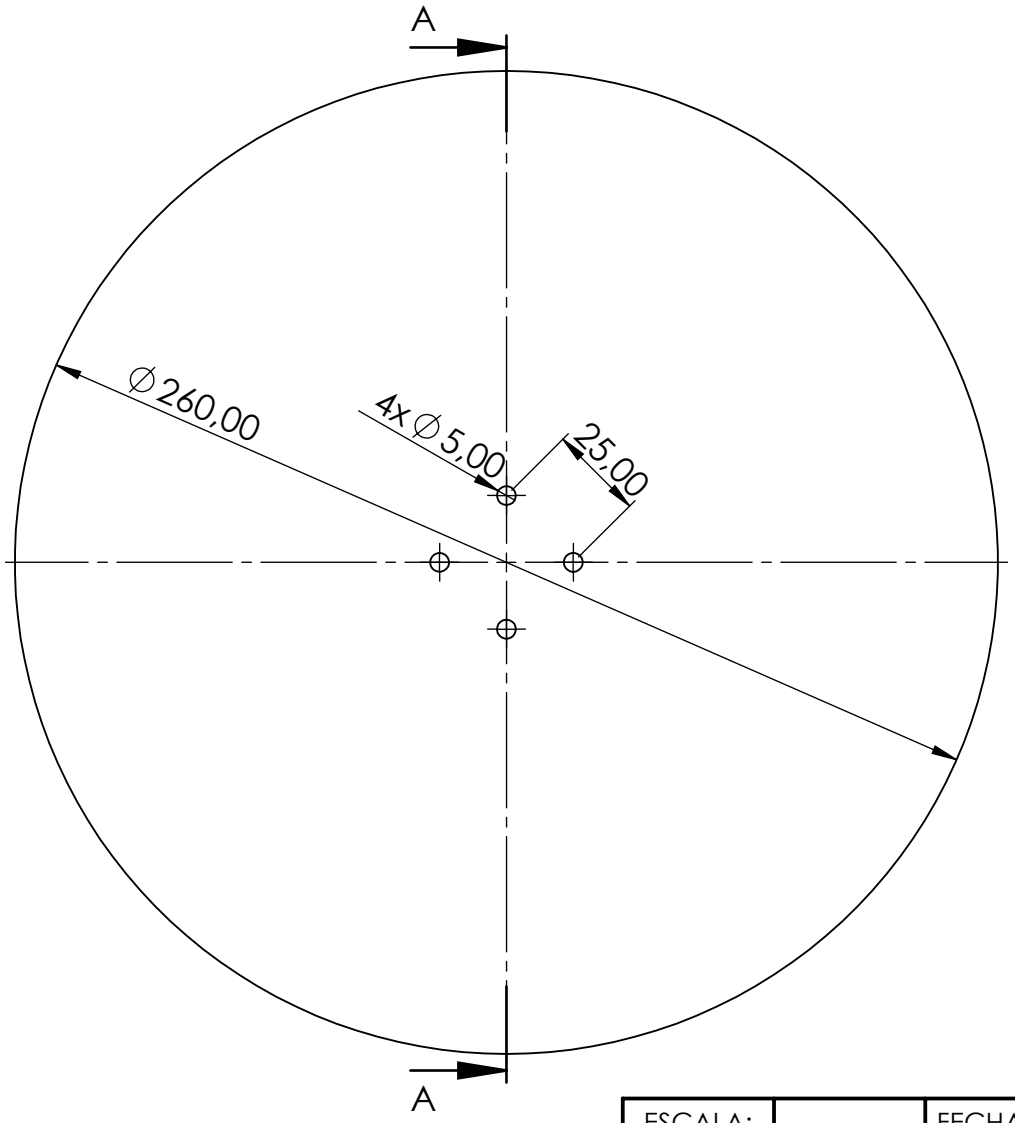
ESCALA: 1:2	DIBUJÓ	FECHA: 06 22	NOMBRE: Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
		PERFIL ESTRUCTURAL 30x30 L-200mm			PLANO Nº 32
ISO E					HOJA 32/39

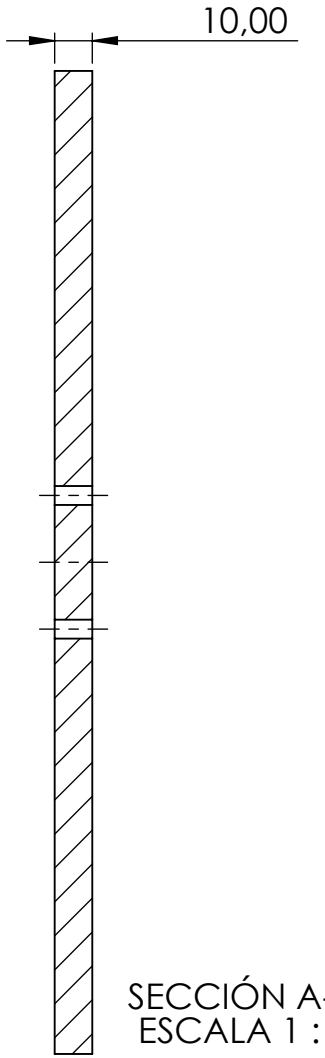




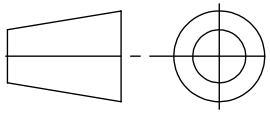
SECCIÓN A-A
ESCALA 1 : 2

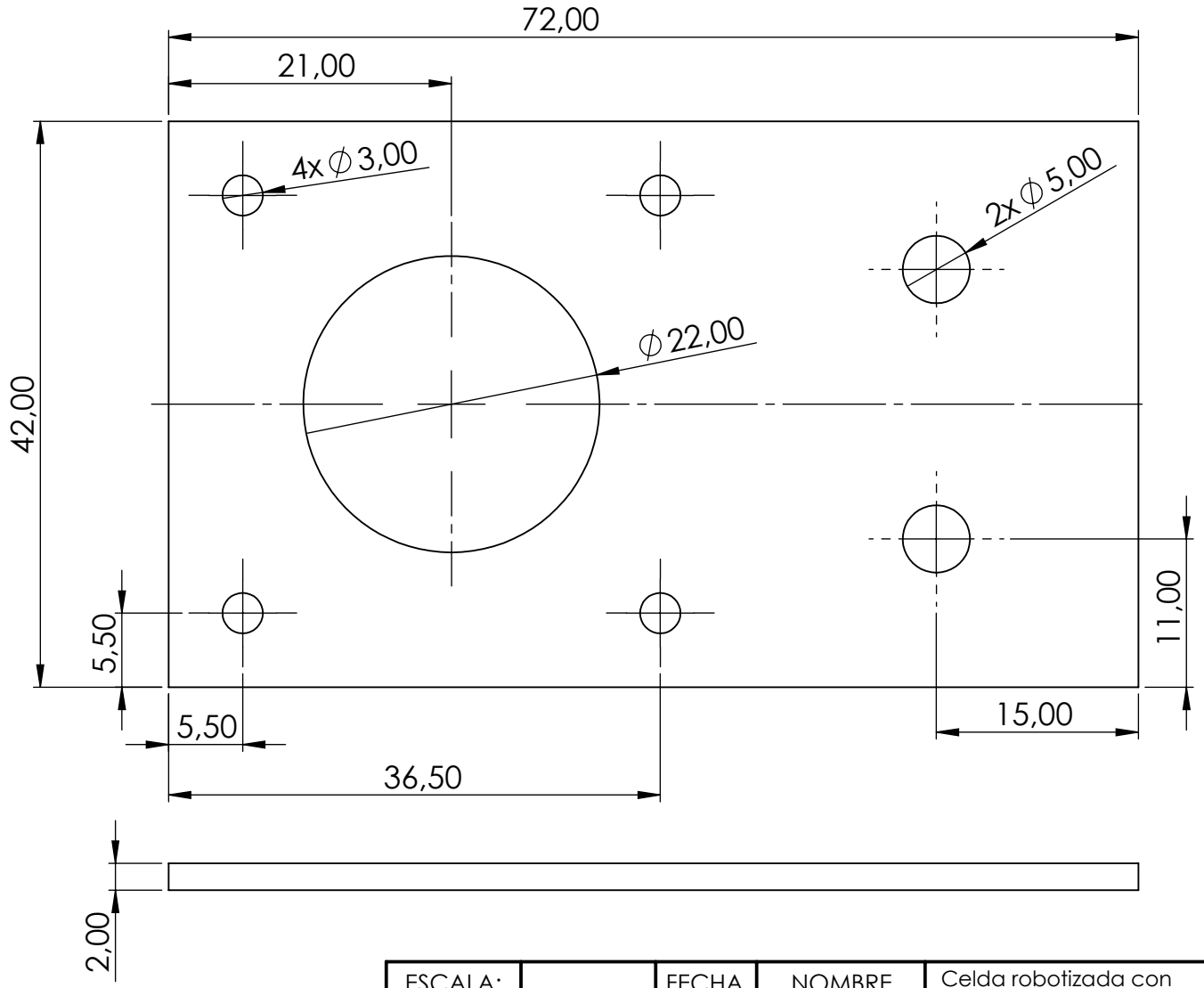
ESCALA: 1:2	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
				PLATO PORTACÁPSULAS	PLANO Nº 34
ISO E					HOJA 34/39



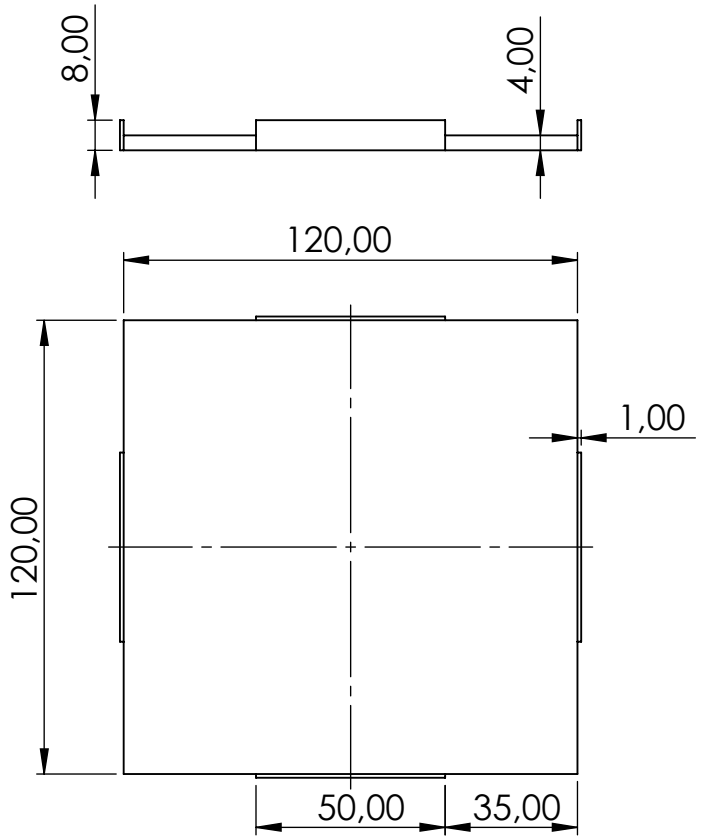


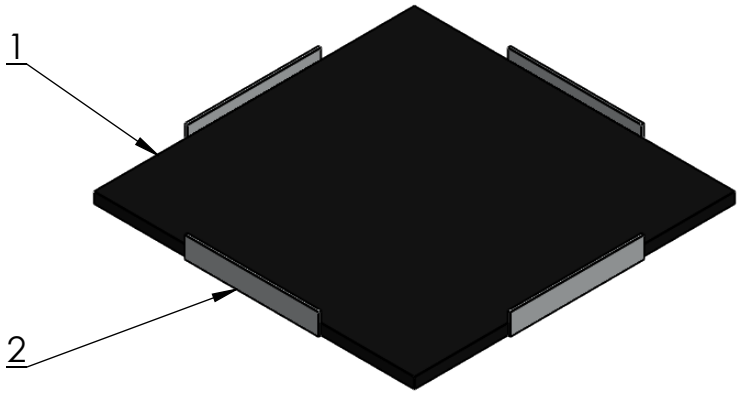
SECCIÓN A-A
ESCALA 1 : 2

ESCALA: 1:2	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
		PLATO PORTACAJAS			PLANO Nº 35
ISO E					HOJA 35/39

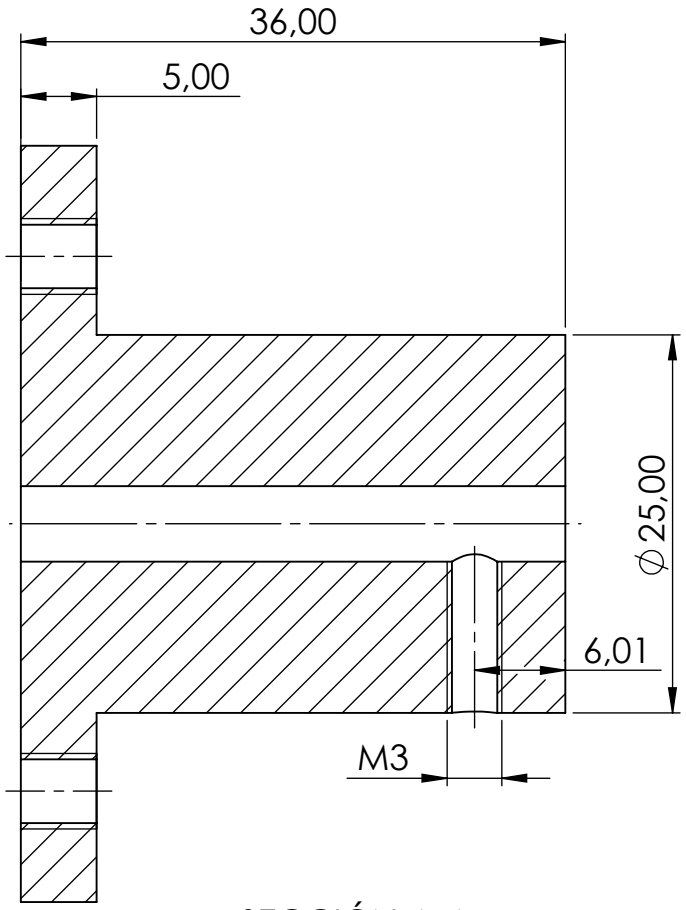


ESCALA: 2:1	DIBUJÓ	FECHA: 06 22	NOMBRE: Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
ISO E	PLETINA SOPORTE MOTOR				PLANO Nº 36
					HOJA 36/39

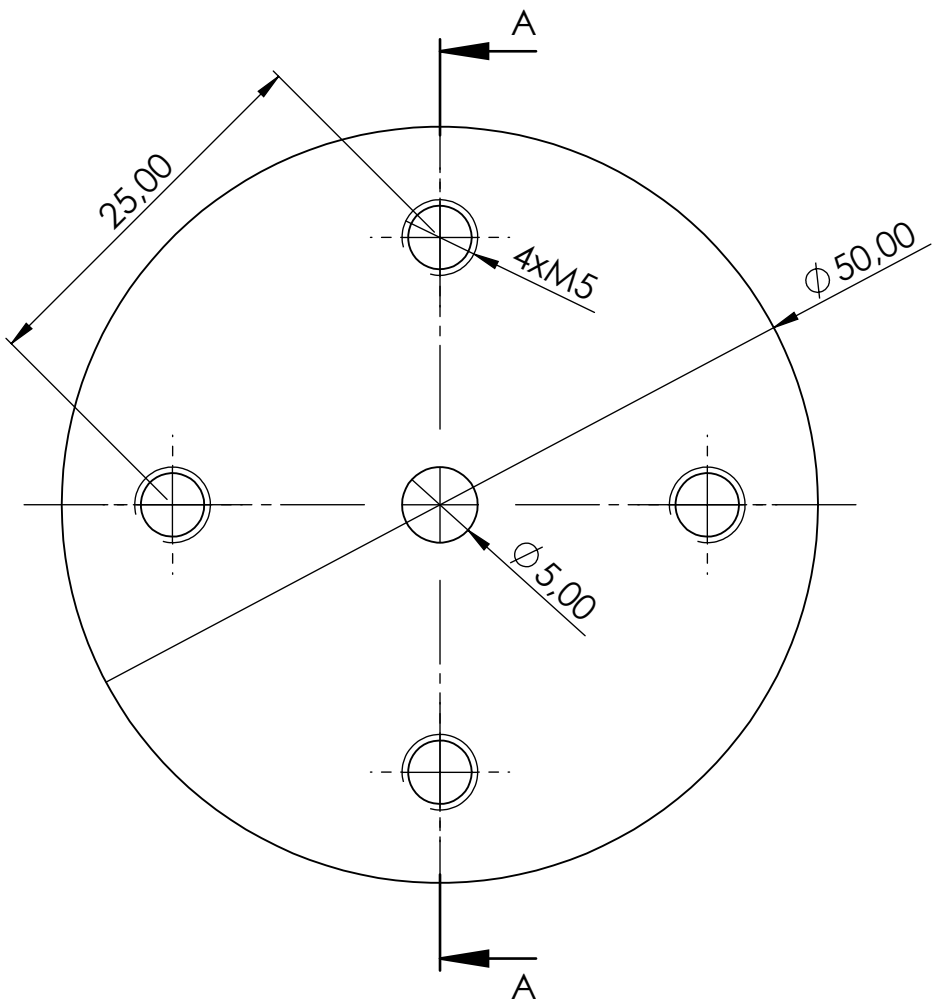




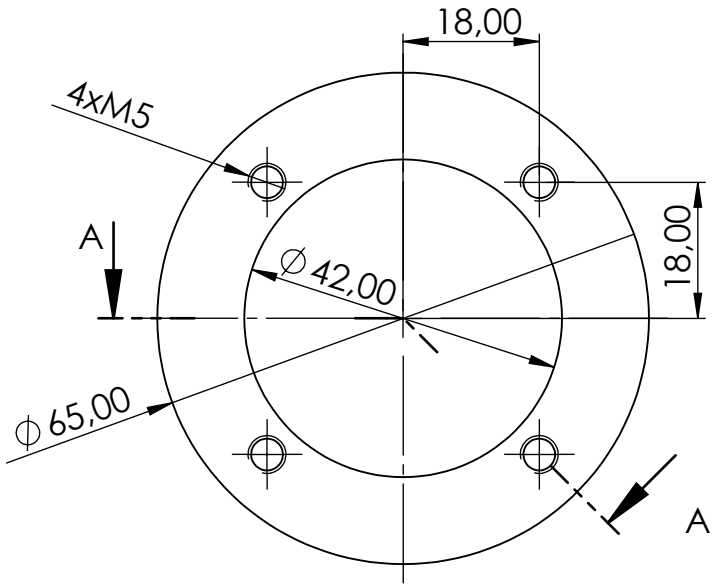
2	4	Pletina aluminio		Plano ,frabricación propia	Aluminio
1	1	Soporte Caja		Plano ,frabricación propia	Madera
Marca	Cantidad	Denominación		Plano y fabricante	Material
ESCALA: 1:2	DIBUJÓ	FECHA: 06 22	NOMBRE: Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
ISO E		ENSAMBLAJE SOPORTE CAJA			PLANO Nº 37
					HOJA 37/39



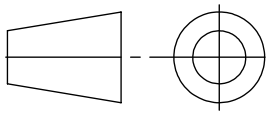
SECCIÓN A-A
ESCALA 2 : 1



ESCALA: 2:1	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
		ACOPLAMIENTO PLATO			PLANO Nº 38
ISO E					HOJA 38/39



SECCIÓN A-A

ESCALA: 1:1	DIBUJÓ	FECHA 06 22	NOMBRE Francisco Javier Navarro Escrivá	Celda robotizada con visión artificial para el empaquetado de cápsulas de café	UNIVERSIDAD POLITÉCNICA DE VALENCIA,E.T.S.I.D
		SUPLEMENTO EJE Z			PLANO Nº 39
ISO E					HOJA 39/39