UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Diseño y desarrollo de un robot cuadrúpedo dedicado al tratamiento de niños con autismo.

Trabajo Fin de Grado

Grado en Ingeniería Electrónica Industrial y Automática

AUTOR/A: Folgado Brisa, Ernest

Tutor/a: Ibáñez Civera, Francisco Javier

CURSO ACADÉMICO: 2022/2023

Documentos contenidos en TFG:

- Memoria
- Anexo 1. Manual de usuario
- Anexo 2. Código
- Planos
- Pliego de condiciones
- Presupuesto

El código y los modelos 3D de las piezas se pueden encontrar en el siguiente enlace:
https://github.com/ErnestFB/RA-01

Agradecimientos

Resulta curioso como a lo largo de nuestra vida pequeños detalles pueden tener una gran importancia en nuestro futuro. Así mismo, las personas que nos rodean pueden marcar la diferencia y generar un impacto positivo en nuestra vida académica.

Es por ello que, una vez finalizado el proyecto, es importante atribuirle los méritos correspondientes a las personas importantes que me rodean. Consecuentemente, que quiero agradecerle:

A mis padres, por orientarme siempre que lo he necesitado, estar a mi lado, criarme y posibilitarme ser quien soy y llegar a donde he llegado.

A mi familia, porque al igual que siempre lo han estado, sé que siempre estarán disponibles para ayudarme cuando lo necesite.

A Marta, por alegrarme cada día y motivarme para seguir adelante.

A mis amigos, por acompañarme a lo largo de infinidad de momentos tanto divertidos como difíciles.

Y finalmente, a mis profesores, especialmente aquellos que han conseguido despertar mi interés.

MEMORIA

Diseño y desarrollo de un robot cuadrúpedo dedicado al tratamiento de niños con autismo.

Trabajo final del:	Grado en Ingeniería electrónica industrial y Automática
Realizado por:	Ernest Folgado Brisa
Tutorizado por:	Javier Ibáñez Civera
Curso académico:	2022/2023

Contenido memoria

Listado de figuras

Listado de tablas

1. Justificación del proyecto

De acuerdo con la OMS, los trastornos del espectro autista (TEA) son un grupo de afecciones caracterizadas por dificultar la interacción social y la comunicación de quienes las padecen [1].

El TEA es uno de los trastornos del neurodesarrollo más frecuentes, y aunque la prevalencia del TEA varía según los distintos estudios, se calcula que 1 de cada 100 niños sufren este trastorno [2]. Otras fuentes como los Centros para el Control y la Prevención de Enfermedades (CDC, por sus siglas en inglés), aseguran que en Estados Unidos el 1 de cada 54 niños padece esta afección [3]. Y en España se estima que hay más de 450.000 personas con autismo y más de 4.500 bebés nacen con TEA cada año [4].

Existen diversos tratamientos con el propósito de mejorar los problemas en el área socio-comunicativa que presentan los niños con TEA. Así mismo, en los últimos años se han empezado a utilizar robots en algunos de estos tratamientos. Tal y como indican Juan Carlos Cruz y Yeliza Salazar, "los niños autistas tienen una gran afinidad hacia los juguetes mecánicos, especialmente los robots. La previsibilidad de comportamiento repentino y monótono del robot es el factor reconfortante que hace que los niños con autistas tengan una gran atracción por robots" [5]. Al ser un ámbito de la robótica emergente, este se encuentra abierto a nuevos tipos de robots y todavía hay mucho por aportar.

Con el fin de diferenciar el robot diseñado de los ya existentes, se ha planteado desarrollar un robot cuadrúpedo; pues se parte con cierta experiencia en el diseño de este tipo de robots al haber realizado previamente el robot RR-01 (figura 1).

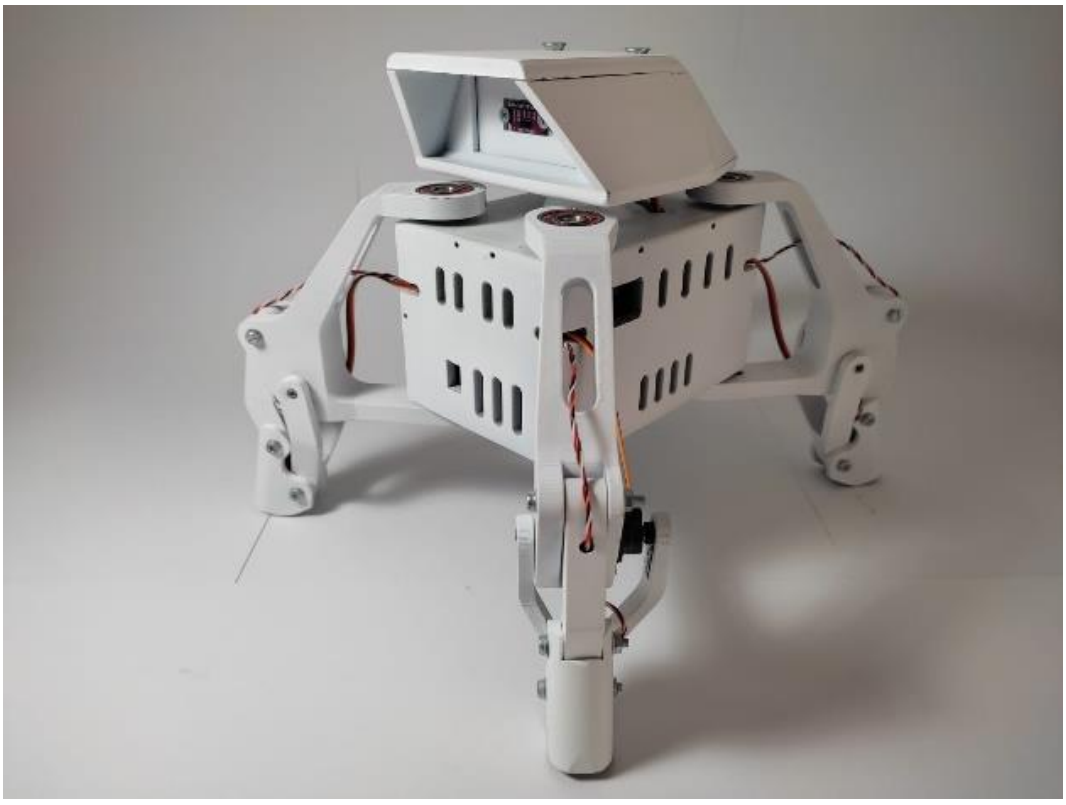


Figura 1. Robot RR-01

En la actualidad se están utilizando distintos tipos de robots en estos tratamientos, pero ninguno de estos robots es cuadrúpedo. Es posible que a ciertos niños les resulten atractiva e interesante esta morfología. Además, a distintos niños les gustarán más o menos los diferentes tipos de robot; por ello, es importante que no todos sean iguales. Otro motivo por el cual se ha optado por esta anatomía, alejándose de la tendencia actual de realizar robots humanoides, es el hecho de que los robots que se asemejan a los humanos pueden llegar a generar rechazo e incomodidad. Existe una explicación psicológica para este proceso al que se le denomina "valle inquietante" [6]. Por ello, es importante que el futuro de los robots dedicados a interactuar con personas sea explorado planteando distintas anatomías. Desde una pantalla empotrada en la pared pasando a robots con morfologías animales e incluso humanas. Siempre teniendo en cuenta el propósito concreto.

1.1. Trastornos del espectro autista.

1.1.1. Definición

Según la CDC: "Los trastornos del espectro autista (TEA) son discapacidades del desarrollo causadas por diferencias en el cerebro. Las personas con TEA con frecuencia tienen problemas con la comunicación y la interacción sociales, y conductas o intereses restrictivos o repetitivos. Las personas con TEA también podrían tener maneras distintas de aprender, moverse o prestar atención. Es importante señalar que algunas personas sin TEA también podrían tener algunos de estos síntomas. Sin embargo, en las personas con TEA, estas características pueden dificultar mucho la vida." [7].

1.1.2. Sintomatología del TEA.

Al hablar de la sintomatología de los trastornos del espectro autista, se debe hacer hincapié en el término "espectro". Pues una de las principales características de esta patología es la significativa variación de los síntomas y su gravedad en los distintos pacientes.

Según el DSM-5 [8], las personas con TEA usualmente tienen:

- Dificultades para comunicarse, tanto de forma verbal como no verbal. Pues suelen tener dificultades a la hora de iniciar o de mantener una conversación, al igual que para entender las señas sociales.
- Problemas a la hora de interactuar con otras personas; dificultades para hacer amigos, o ser partícipe de actividades sociales con normalidad.
- Patrones restringidos y repetitivos de comportamiento, intereses o actividades; movimientos corporales repetitivos, dificultades para cambiar sus rutinas y patrones de pensamiento rígidos.
- Hipersensibilidad o hiposensibilidad sensorial. Esto significa que la persona puede percibir a los estímulos sensoriales, como la luz, el tacto o el sonido, de forma mucho mayor o menor que los demás.

Por otra parte, el Instituto Nacional de Salud Mental (NIMH, por sus siglas en inglés) [9] recoge en su página web un listado de los comportamientos que las personas con TEA suelen mostrar.

En el ámbito socio-comunicativo:

- No responden cuando se les llama por su nombre, o tardar en ello.
- Comparten emociones, intereses u objetivos con poca frecuencia.
- Hablan con frecuencia de sus temas de interés sin darse cuenta de que a los demás no les interesa o sin permitir que otros tengan la oportunidad de responder.
- Realizan poco contacto visual
- Parecen ignorar a quien le habla.
- Realizan expresiones faciales y gesticulaciones no acordes con lo que dicen.
- Tienen tonos de voz inusuales, estos pueden ser monótonos como un robot o melódicos.
- Les es difícil empatizar con otras personas y hacer amigos.
- No les es fácil para entender o predecir los comportamientos de los demás.
- Presentan problemas a la hora de entender el punto de vista de los demás.
- Tienen dificultades para compartir en un juego imaginativo.

Usualmente también tienen conductas limitadas o repetitivas, por ejemplo:

- Recurren reiteradamente a ciertos términos o expresiones.
- Presentan un gran interés en temas específicos de forma duradera.
- Se molestan por pequeños cambios en su rutina y presentan dificultades para cambiarla.
- Tienen mayor o menor sensibilidad que otras personas a la información sensorial (sonido, luz, temperatura o incluso la ropa).

Algunas de estas personas también suelen presentar puntos fuertes, como:

- Tener la capacidad de aprender y recordar información durante mucho tiempo.
- Estar dotados de una gran memoria auditiva y visual.
- Destacar en materias como la música, el arte y las ciencias.

1.1.3. Tratamiento de los TEA

Dado que el autismo es una patología sin cura, el objetivo de los tratamientos es reducir los síntomas que influyen negativamente en el día a día de las personas que lo padecen. Como a cada paciente el TEA le afecta de distinta forma, generalmente el tratamiento debe ser adaptado a cada paciente e incluye diversos profesionales. [10]

En la mayoría de las situaciones esta patología es detectada en una edad temprana, posibilitando la intervención temprana. Pues, durante la juventud resulta más fácil que los niños adquieran las capacidades que necesitarán para desenvolverse tanto en el colegio como en su vida adulta. Estos tratamientos están orientados al desarrollo de habilidades sociales, comunicativas y emocionales entre otras y se pueden diferenciar las siguientes intervenciones:

- Conductuales, centrados en la enseñanza de nuevos comportamientos y habilidades,
- Evolutivas, donde se enseñan técnicas sociales y de comunicación que puedan ser de ayuda para el niño en el desarrollo de conexiones interpersonales enriquecedoras.
- Basadas en terapia, con el fin de trabajar dificultades concretas, como pueden ser habilidades comunicativas o sociales.
- Basadas en la familia, donde se hace participe a la familia.
- Combinadas, en las que se mezclan varios métodos de intervención. [11]

De entre otras, una vía efectiva para llevar a cabo estas intervenciones, especialmente en los pacientes más jóvenes, es el juego. Este puede ser utilizado como herramienta para desarrollar habilidades sociales, de comunicación y de interacción con los demás. A través del juego, se mejora la atención de los niños, se potencia la coordinación motora y se desarrollan capacidades socio comunicativas. Se pueden utilizar juegos de rol para enseñar al niño habilidades de conversación y resolución de conflictos, o juegos de imitación para ayudar al niño a aprender a seguir instrucciones y a desarrollar habilidades motoras. Adicionalmente se les pueden enseñar habilidades necesarias para el día a día de forma divertida como por ejemplo cepillarse los dientes o comer con cubiertos entre otros. Además, el juego ayuda a reducir el estrés y la ansiedad durante la sesión. [12]

Por otro lado, los adultos ya suelen haber desarrollado habilidades sociales, comunicativas y emocionales gracias al haber pasado por tratamientos tempranos, especialmente en los casos más severos. Así mismo estos pueden requerir cierta asistencia por cuestiones relacionadas con la salud mental como puede ser la ansiedad o la depresión [13]. Adicionalmente, los casos más críticos suelen necesitar ayuda para desarrollar habilidades que les permitan vivir por su cuenta [14].

Respecto a los tratamientos farmacológicos, no existe aún ningún fármaco que se haya demostrado eficaz contra los síntomas del TEA. Aun así, sí que existen dos fármacos aprobados por la DFA para su uso en TEA, la risperidona y el aripiprazol. Estos sirven para tratar la irritabilidad, hiperactividad e impulsividad asociada al TEA [15].

1.2. Robots en el tratamiento del TEA

Fruto de la combinación del enfoque psicológico y los progresos tecnológicos, en los últimos años, ha surgido una nueva alternativa en la intervención del autismo; la terapia asistida con robots sociales. En este tipo de terapias orientadas a los pacientes más jóvenes, se utiliza un robot como herramienta de apoyo en la intervención con el fin de hacer que la experiencia sea más cómoda e incentivar la participación.

Se entiende como robots sociales aquellos robots diseñados especialmente para interactuar con las personas, de la misma forma que estas interactúan entre sí. Para ello, estos dispositivos incorporan tecnologías de reconocimiento de voz y visión artificial para percibir su entorno y se les dota de lenguaje y expresiones faciales [16]. En el contexto del tratamiento del TEA, estos suelen ser programados para enseñar a los niños a reconocer expresiones faciales, a seguir instrucciones, a imitar acciones o para ser partícipes de juegos.

Existen distintos tipos de terapias en las que se pueden utilizar este tipo de robots, de entre las cuales podemos destacar el método ABA (Applied Behavior Analysis, análisis conductual aplicado), por ser una de las más frecuentes. Este método consiste en analizar el comportamiento actual del paciente, identificar las habilidades y conductas que se necesitan mejorar y se realiza un plan de tratamiento donde a través del aprendizaje y refuerzos positivos se eliminen o disminuyan los comportamientos negativos del individuo [11]. Aunque no resulta necesario contar con un robot para realizar la terapia sí que puede ser de gran utilidad el uso de un robot en este tipo de tratamientos, al igual que en otros, pues el robot puede ser el encargado de indicar las normas del juego que se quiera hacer, ser participe del mismo, repetir frases como “bien hecho” o incluso ser un ejemplo a seguir diciendo “buenos días” o preguntándole “¿Qué tal te va?”.

Según diversos estudios la terapia asistida con robos sociales está obteniendo resultados favorables, demostrando ser eficaz especialmente en niños, pues los robots posibilitan una interacción social predecible y sencilla, reduciendo la presión y el estrés en los niños durante la intervención. [5] Adicionalmente, la presencia de los robots transforma actividades que usualmente pueden resultar tediosas e incómodas para los niños en algo ameno y divertido. Especialmente a través del juego, en los estudios realizados los niños se muestran más dispuestos a asistir y ser partícipes de las sesiones en las que se dispone de un robot [17]. La mayoría de los robots utilizados en este contexto suelen ser fácilmente programables, posibilitando la adaptación de su comportamiento al carácter de cada niño y posibilitando su participación en las distintas sesiones en las que se desee utilizar [18].

Hasta la fecha los robots han tenido uso en mayoritariamente en el ámbito terapéutico, así mismo también se han realizado algunos estudios en contextos escolares como el proyecto Robot4Autism y Aurora [19].

Cabe destacar, que un robot diseñado para el tratamiento del autismo también puede ser utilizado en otros pacientes. Como en otros trastornos generales del desarrollo o en personas de edad avanzada que hayan sufrido un derrame cerebral o que tengan parálisis parcial de extremidades [20].

2. Antecedentes

2.1. Precursores

En la última década el uso de robots para el tratamiento del autismo se ha popularizado y consecuentemente tanto universidades como empresas privadas han desarrollado sus propios robots. No obstante, ya desde los años 90 se empezó a investigar la posibilidad del uso de robots con este fin, siendo uno de los primeros estudios el realizado por Dautenhahn y Werry. En este estudio [21], se utilizó el robot móvil AuRoRA, Autonomous Robot for Reabilitation (figura 2), en la rehabilitación de niños con autismo.

Figura 2. Robot Aurora [21]

Más adelante, a finales de los 90, se desarrollaron los robots Robota (figura 3) [22]. Estos robots humanoides de 45 cm de altura, construidos utilizando piezas de muñecas comerciales, se utilizaron tanto con propósito terapéutico como educativo. Incorporaban 9 motores para mover sus extremidades y cabeza, y para percibir su entorno contaban con varios sensores, de entre ellos de luz, piroeléctricos e infrarrojos. Para su funcionamiento, los robots precisaban de estar conectados a un ordenador que se encargaba de la síntesis y reconocimiento de voz además de para procesar el video obtenido a través de una cámara ubicada en el cuerpo de la muñeca.

En 1998, se utilizaron los robots Robota en experimentos que consistían en juegos de imitación [23], donde los niños tenían que interactuar con el robot adoptando las mismas posturas que estos (figura 3). Siendo el propósito de la actividad que los niños adoptaran posturas corporales fundamentales para la interacción social.

Figura 3. Robot Robota interactuando con niño con TEA [22]

2.2. Infanoid

Sucediendo a los estudios previos, en 2004 un equipo japonés utilizo el robot Infanoid (figura 4) con el fin de estudiar la interacción de niños normales y niños con TEA con él.

Este robot consistía en una estructura mecánica que imitaba el cuerpo humano de cintura para arriba, siendo su tamaño el de un niño de 3-4 años. Para su movimiento contaba con 29 actuadores, principalmente motores de corriente continua con sus respectivos encoders y sensores de corriente. Sus manos al igual que las humanas tenían 4 dedos y un pulgar permitiéndole señalar y realizar otros gestos. En sus ojos se ubicaban dos pequeñas cámaras a color que se conectaban a un ordenador para procesar el video en tiempo real detectando tanto caras de personas como objetos. Sus cejas y labios también estaban automatizados, permitiéndole hacer varias expresiones faciales. Además. sus labios se movían a la vez que el sintetizador de voz producía sonido y la inclinación de los labios se variaba en función al estado de ánimo del robot. En sus orejas incorporaba 2 micrófonos y aunque no tenía reconocimiento de voz sí que analizaba las frecuencias para determinar el tono en el que se le hablaba y repetir lo que se le decía. [24]

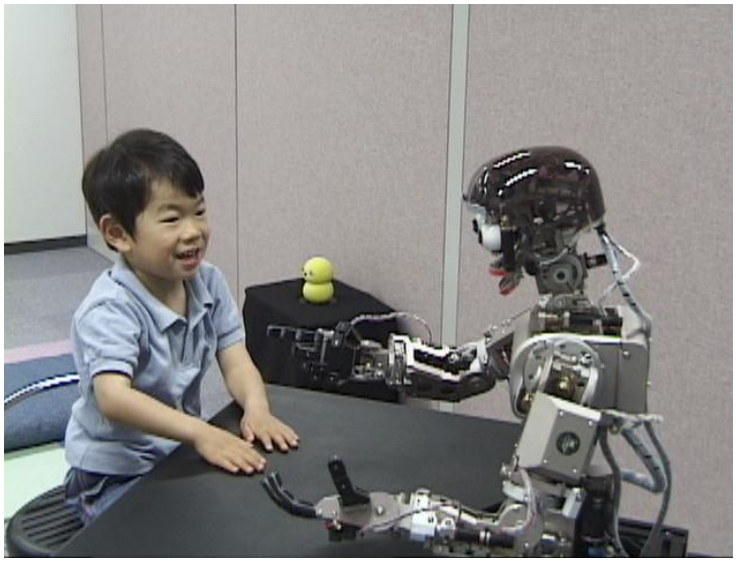


Figura 4. Robot Infanoid interactuando con un niño con TEA [25]

Utilizando este robot se llevaron a cabo varios experimentos tanto en niños normales como en niños con TEA, con edades entre los 6 meses y los 9 años.

El experimento consentía en dejar al niño en un inicio solo junto al robot y observar cómo interactuaba con él. Tras un par de minutos la madre del niño iba y se sentaba a su lado y la interacción con el robot continuaba hasta que el niño perdiera el interés. Los niños normales pasaron por un par de fases, donde primero concebían al robot como una cosa que se movía, pero luego entendieron que tenía comportamiento propio y emociones, empezaron a hablarle y a tratarle como un compañero.

Los niños con autismo por su parte también interactuaron con el robot y se mostraron muy relajados. Esto se debía a lo fácilmente previsible que resultaba el robot, pues no podía levantarse y su comportamiento era limitado. Esto permitió a los niños entrar en una interacción lúdica, que iba desde acciones sociales cotidianas (como dar, mostrar y preguntar) hasta un juego de improvisación y escondite. [25]

2.3. KASPAR

A raíz del avance tecnológico, en 2005 el Dr. Ben Robins junto con su equipo en la Universidad de Hertfordshire en el Reino Unido, desarrolló KASPAR (figura 5), Kinesics and Synchronisation in Personal Assistant Robot [26].

Este robot humanoide demostró rápidamente su capacidad para ser utilizado en la terapia de niños con autismo por su repetitividad, simpleza y bajo precio. Su cara realista, pero a la vez simple, es capaz de transmitir fácilmente su estado de ánimo lo cual y gracias a los numerosos motores que incorpora es capaz de realizar múltiples movimientos [27].

Figura 5. Robot KASPAR [28]

Debido a su éxito, desde su primera versión en 2005, KASPER ha sido actualizado en numerosas ocasiones adaptándolo a las nuevas tecnologías y a la evolución en el tratamiento de niños con autismo. Entre otras mejoras, a lo largo de las diferentes versiones, KASPER ha pasado de ser completamente controlado por un humano a ser semiautónomo, se han mejorado sus sensores y su fabricación.

La versión más reciente, la 5.5, se diseñó con el fin da fabricar múltiples unidades y que estas fueran entregadas a varios colegios y a padres de niños con TEA. Por ello se simplificó el funcionamiento del robot haciendo que no se necesite la presencia de un técnico para el funcionamiento de KASPAR.

Esta versión tiene 22 GDL (Grados de Libertad), 3 en cada ojo y ceja, 2 en la boca, 3 en el cuello, 5 en cada brazo y 1 en el torso. Todas las articulaciones se controlan con servomotores; Hitec HS-82MG para los ojos y cejas, y Dongbu Robot Herkulex drs-0101 y drs-0201 para el resto de las articulaciones.

Respecto a los sensores, el robot incorpora 15 células de carga distribuidos por el cuerpo que dotan al robot de sentido del tacto permitiéndole saber si se le coge la mano, el brazo o le tocan la cara.

A diferencia de los anteriores robots esta versión no necesita estar conectada a un ordenador por cable, sino que incluye conectividad WIFI y para su autonomía incorpora una batería lipo de 12 V y 7 Ah, con la que puede funcionar hasta 7 horas. [29]

2.4. Robots comerciales

Actualmente existen varios robots comerciales que pueden ser utilizados en la terapia de niños con autismo, como es el caso de Milo, NAO y Moxie.

Figura 6. Milo [30]

Figura 7. NAO [35]

Figura 8. Moxie [30]

Milo es un robot humanoide desarrollado y distribuido por la empresa Robokind, especialmente diseñado para interactuar con niños con TEA [31]. Con un tamaño de 60 cm, su aspecto similar a un juguete lo vuelve una muy buena herramienta en el tratamiento del autismo. La cual, al igual que otros robots ha demostrado su eficacia a la hora de captar la atención de los niños con TEA [32].

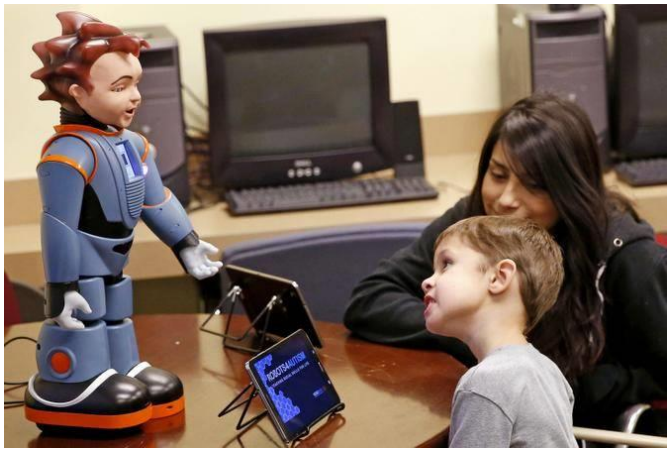


Figura 9. Robot Nilo utilizado en la terapia del autismo [31]

El robot destaca por tener musculatura facial detallada, pues su cara está fabricada con un material flexible debajo del cual se ubican múltiples motores que mueven sus labios, mandíbula, mejillas, ojos parpados y cejas (figura 10).

Figura 10. Expresiones faciales robot Milo [31]

Por otro lado, el robot viene con una gran cantidad de contenido, según la página web más de 140 lecciones dirigidas por el robot y 260 actividades. Desafortunadamente únicamente en inglés [31].

Cabe destacar que, aunque la empresa no comercializa unidades sueltas y solo trabaja con centros educativos, recientemente en 2022, la universidad de Cartagena firmó un convenio de colaboración y compró una unidad por más de 16.000€. Con el propósito de programarlo con su propio protocolo y poder comparar como los niños reaccionan a este robot en comparación con Pepper otro robot humanoide que en vez de cara articulada cuenta con una pantalla en el pecho (figura 11). [33] En caso de no que no se adquiera el robot con fines investigativos, se requiere la compra de un mínimo de 5 unidades.

Figura 11. Robot Pepper adquirido por la uiversidad de cartagena [32]

NAO, diseñado y fabricado por la compañía francesa Aldebaran Robotics, y con una altura de aproximadamente 60 cm, es posiblemente el robot social que más éxito ha tenido en los últimos años. Este robot ha sido utilizado para realizar múltiples tareas, desde jugar partidos de futbol [34] (figura 12), a participar en el tratamiento de niños con autismo (figura 13). Existiendo un kit específico para su uso en el tratamiento del autismo con un precio de aproximadamente 20.000 $ [35].

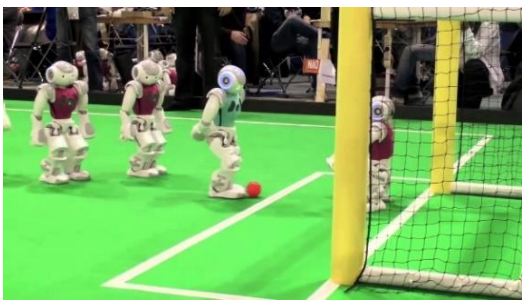


Figura 12. Robot NAO jugando futbol [33]

Figura 13. Robot NAO participando en la terapia de niños con autismo [36]

Si bien es cierto que el robot no cuenta con una cara articulada ni una pantalla a través de la cual pueda expresar expresiones faciales, NAO tiene un software de reconocimiento y síntesis de voz que lo vuelve una herramienta muy útil a la hora de interactuar con los niños. Además, la amplia variedad de movimientos que puede realizar hace que pueda participar en múltiples juegos [37].

Diversos estudios [38] [39] han puesto en práctica el uso de este robot, obteniéndose resultados favorables.

Por otro lado, la compañía Emboiled Inc. ofrece una alternativa de robot más simple: Moxie (figura 14), y con un precio más asequible (1500 $), especialmente diseñado para ser fácil de utilizar. Este robot se conecta vía Wifi a una aplicación móvil con la que los padres pueden monitorear el progreso de sus hijos y personalizar la experiencia [30].

Figura 14. Robot Moxie [40]

En comparación con otros robots, Moxie cuenta con un movimiento mucho más limitado, no puede desplazarse y únicamente teniendo 8 grados de libertad, 2 en el torso, 2 en el cuello, y 2 en cada brazo. Pero esto no limita su capacidad de interacción, pues es a través de la pantalla ubicada en su cabeza con la que muestra sus expresiones e interactúa con el niño. Múltiples animaciones (figura 15) dotan al robot de su propia personalidad, mostrando estas el estado de ánimo del robot de forma exagerada con el fin de facilitar la comprensión de este por el niño. [41]

Figura 15. Expresiones de Moxie [30]

Adicionalmente, el robot cuenta con una cámara, varios micrófonos y un altavoz que junto a su software de inteligencia artificial le permite reaccionar a su entorno y comportarse en función de lo que ocurra a su alrededor [30].

2.5. RR-01

En 2022, un año antes del desarrollo de este TFG, se desarrolló el robot RR-01 (figura 16) para el proyecto de la asignatura robótica móvil [42].

Figura 16. Render RR-01

Dicho robot tenía como objetivo el ser capaz de moverse tanto de forma autónoma, evitando obstáculos, como controlado a través de un smartphone vía bluetooth. Además, se intentó sin éxito el implementar un sistema que evitara su caída, ya sea evitar que se cayera por las escaleras o por el borde de la mesa.

Tal y como se puede observar en la figura 17, el robot incorpora 4 patas con dos articulaciones por pata siendo capaz de moverse en todas direcciones (hacia delante, atrás, derecha e izquierda) y rotar sobre sí mismo. Además, de los 8 servomotores MG90S con los que controla sus patas, el robot incorpora un noveno motor en la cabeza.

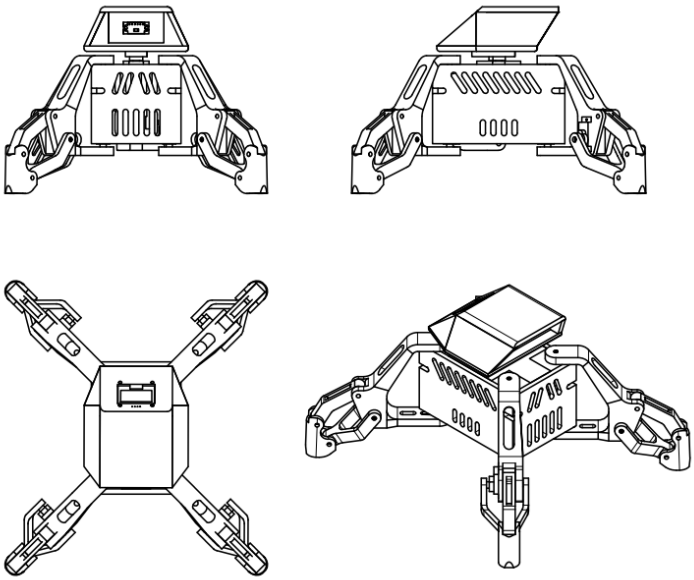


Figura 17. Dibujo RR-01

Para evitar su caída el robot integra 4 sensores de luz infrarroja, situados en los extremos de sus patas orientados hacia abajo (figura 18). Dichos sensores, concretamente los CNY70, constan de un diodo LED emisor de luz infrarroja y un fototransistor. Cuando la luz emitida se refleja en una superficie el fototransistor convierte la luz recibida en una corriente eléctrica proporcional a su intensidad, la cual utilizando una resistencia puede transformarse en una tensión que el microcontrolador puede medir. Cabe destacar que la distancia de funcionamiento de este tipo de sensores es baja, siendo la del CNY70 1 centímetro.[43]

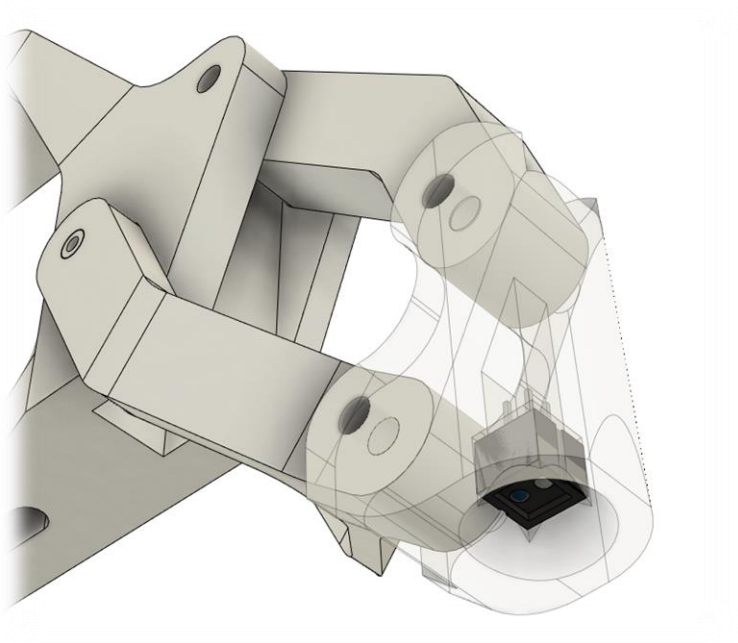


Figura 18. Pata del robot RR-01 con el sensor CNY70

Aunque se sabía de antemano que utilizando este tipos de sensores el sistema anticaída no funcionaría en ambientes con mucha luz ambiental o si la superficie era transparente o de un color muy oscuro, los resultados fueron peores de lo esperado. Cuando el robot da un paso hacia delante y detecta que ha llegado al borde de la mesa ya es demasiado tarde y en la mayoría de las ocasiones acaba cayendo.

Adicionalmente, el robot incluye un sensor de distancia láser VL5LOX (figura 19), una pequeña pantalla monocromática OLED de 1,3'' (figura 20) situada en la parte trasera de la cabeza, donde se muestra las lecturas de sus sensores y un kit microcontrolador ESP32 ESP-WROOM-32 NodeMCU (figura 21) para su control.

Figura 19. Sensor distancia láser VL5LOX [44]

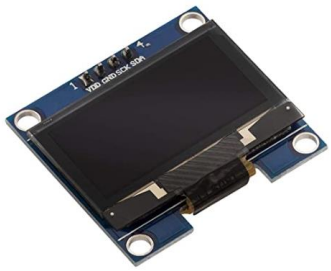


Figura 20. Pantalla monocromática OLED [45]

Figura 21. Microcontrolador ESP32 ESP-WROOM-32 [46]

3. Objeto

El presente proyecto consiste en el diseño, programación y montaje de un prototipo de robot cuadrúpedo, denominado RA-01, que pueda ser utilizado por profesionales sanitarios en el tratamiento de niños con autismo para mejorar sus capacidades sociales y comunicativas.

Dicho robot incorporará: 9 servomotores, con los que se controlarán sus 4 patas y una pinza; una pantalla, en la que se mostrarán expresiones faciales e imágenes; un sistema de reproducción de sonido, luces RGB, un conjunto de sensores de distancia, con los que evitará caerse de la mesa; una cámara, su sistema de control, conectividad WIFI, y su respectiva batería.

Se pretende que quien realice la terapia con el robot sea capaz de controlarlo con un dispositivo de uso general (un móvil, una tableta o un ordenador). Para que esto sea posible el robot actuará como punto de acceso, creando su propia red WIFI con su respectivo SSID y contraseña. De esta forma, un portal de control alojado por el mismo robot permitirá su control y ver una retransmisión de video del punto de vista del robot.

Finalmente, cabe destacar que, dentro de los objetivos de desarrollo sostenible de las Naciones Unidas, el proyecto se enmarca en el tercer objetivo: salud y bienestar.

4. Solución adoptada

4.1. Organigrama

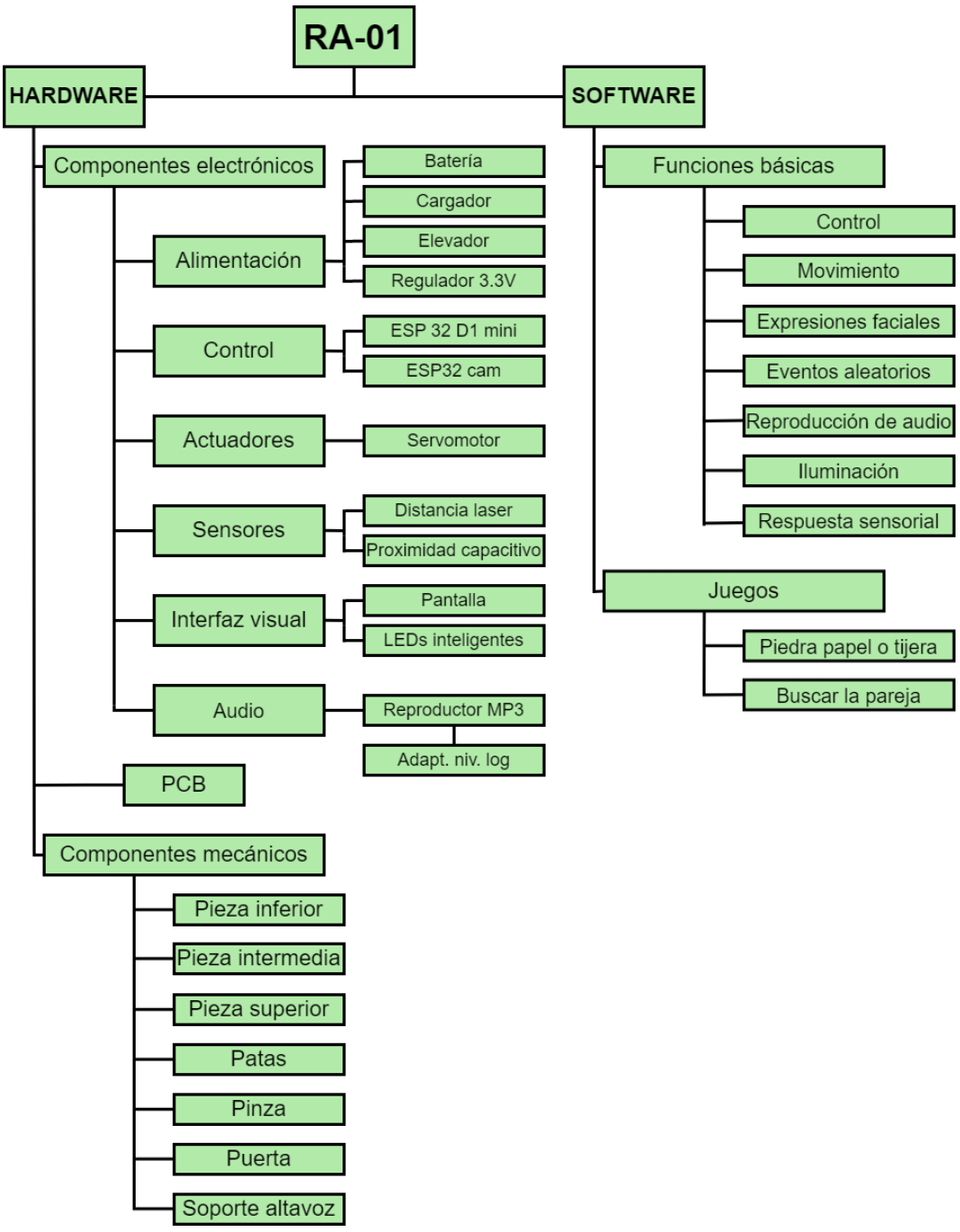


Figura 22. Organigrama RA-01

4.2. Componentes electrónicos

4.2.1. Alimentación

Para el funcionamiento de robot se requieren distintas tensiones: 6V para los servomotores, 5V para el módulo reproductor de MP3 y 3.3V para los microcontroladores, los sensores y la pantalla.

Para lograrlo se ha decidido utilizar una batería de 1 celda de polímero de litio, que proporciona 3.7V nominales y tiene una alta tasa de descarga. Esta se conecta a un módulo que permite su carga y a su salida se conectan 2 elevadores de tensión aumentando el voltaje hasta los 5 y 6V y un regulador que proporciona 3.3V.

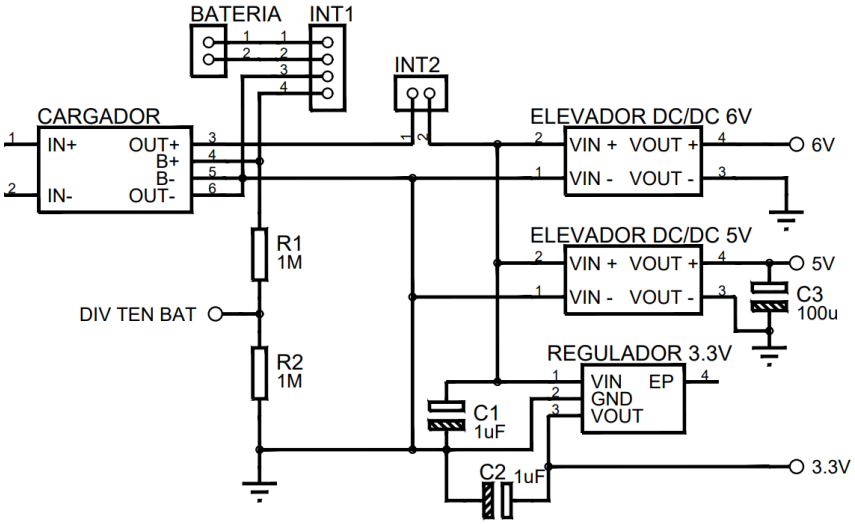


Figura 23. Circuito de alimentación

4.2.1.1. Batería

Como fuente de alimentación se ha decidido emplear una batería lipo por su alta densidad energética.

El distribuidor elegido para la batería, Bricogeek [47], dispone de diferentes modelos de baterías LiPo de 1 celda. De entre ellos concretamente se ha elegido la batería 105050, de 6000 mAh, con unas dimensiones de 50 x 50 x 20 mm y un peso de 110 gramos.

Figura 24. batería seleccionada [47]

4.2.1.2. Cargador

Para poder cargar la batería se ha decidido utilizar el módulo “Cargador LiPo Micro USB TP4056” [48] el cual incorpora el circuito integrado TP4056 [49].

De entre las distintas características del módulo cabe destacar que su conexión de entrada es del tipo micro USB, que alcanza una corriente máxima de carga de 1000mAH y que protege de descarga a la batería. Además, incorpora varios LEDs para indicar el estado. Significando rojo fijo que la carga está activa, verde fin de carga y parpadeo rojo que la batería se ha agotado. [48]

4.2.1.3. Elevador

Con el propósito de elevar la tensión de 3.7V a 5 y 6V se ha decidido utilizar 2 módulos “MT3608 DC/DC Elevador 2-24V/5-28 2A” conectados en paralelo a la salida del cargador.

Tal y como su nombre indica: incorpora el circuito integrado MT3608, funciona con tensiones de entrada de 2V a 24V y es capaz de proporcionar tensiones de salida de 5V a 28V con un máximo de 2A. Aunque no se recomienda superar de forma continua 1A ya que el módulo se puede calentar.

Asimismo, también es importante mencionar que incorpora un potenciómetro multivuelta para ajustar la tensión de salida, su frecuencia de conmutación es de 1,2MHz y su eficiencia media es del 93%.[50]

4.2.1.4. Regulador a 3.3V

Para alimentar el resto de los dispositivos se requiere una tensión de 3,3V. Por ello se ha decidido utilizar el regulador “LDO MCP1755S”. Un regulador de voltaje lineal de baja caída y salida estable de 3.3V, que puede funcionar a partir de una fuente de entre 3.6 y 16V, que proporciona una corriente de salida de hasta 300mA.[51]

Cabe tener en cuenta, que para su salida sea estable se requiere de 1µF de capacitancia mínima de entrada y salida, siendo posible utilizar condensadores cerámicos, de Tantalio o electrolíticos.

Figura 25. Cargador LiPo Micro USB TP4056 [48]

Figura 26. MT3608 DC/DC Elevador 2-24V/5-28 2ª [50]

Figura 27. Regulador a 3,3V, LDO MCP1755S.[51]

4.2.2. Control, ESP32

En un principio se plantó utilizar un único dispositivo para controlar el robot. No obstante, debido a las distintas tareas que se deben realizar y a la cantidad de pines que se requieren; finalmente, se ha decidido utilizar 2 microcontroladores ESP32. Concretamente, la placa de desarrollo “ESP32 D1 mini” y la “ESP32 cam”.

Los microcontroladores ESP32, desarrollados por la empresa china Espressif Systems, surgieron en 2016 como la evolución directa del microcontrolador ESP8266, y al igual que este, han tenido una gran popularidad.

De entre las diversas aplicaciones que se le pueden dar al ESP32 destacan las relacionadas con el internet de las cosas (IOT, por sus siglas en inglés), pues incorpora una antena impresa en su circuito que le permite llevar a cabo conexiones inalámbricas tanto WIFI como bluetooth.

De este microcontrolador cabe destacar su procesador. El ESP32 incorpora un procesador “Tensilica Xtensa LX6” de 32 bits el cual funciona a una frecuencia máxima de 240 MHz y en la mayoría de las modelos es “dual core”[52]. Al disponer de doble núcleo se pueden programar 2 bucles totalmente independientes que funcionan simultáneamente, facilitando el realizar varias tareas a la vez.

Con respecto a las señales PWM, cuenta con 16 canales independientes que pueden configurarse para generar señales PWM. Todos los pines que pueden actuar como salidas se pueden utilizar como pines PWM; excepto los pines del 34 al 39, ya que estos no disponen de resistencias “pull up” o “pull down” y por ello solo pueden utilizarse como entradas y no como salidas.[53]

Además, del microcontrolador ESP32 destaca por tener una memoria SRAM de 520 KB, Soportar hasta 16 MB de memoria FLASH, sus 36 pines GPIO e incorporar un convertidor analógico digital de 12 bits. [52]

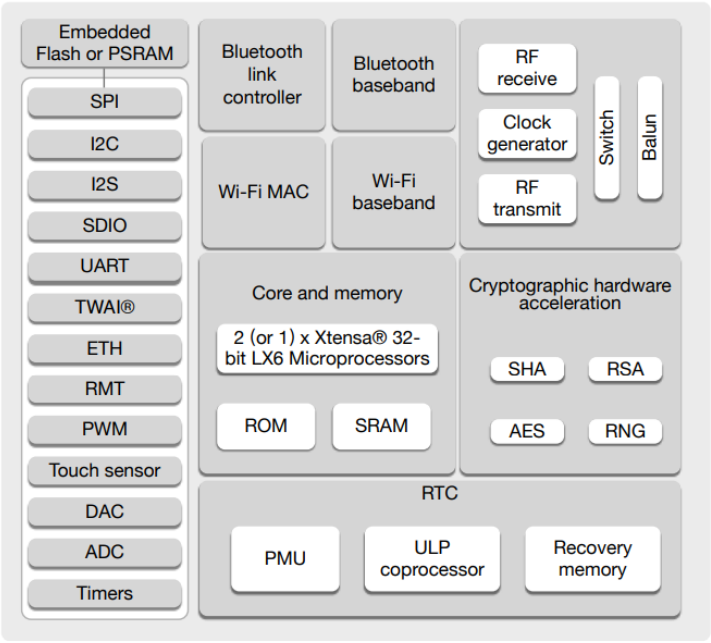


Figura 28. Diagrama de bloques ESP32 [52]

4.2.2.1. ESP32 cam

Actualmente se ha vuelto muy popular utilizar el microcontrolador ESP32 en proyectos relacionados con la visión por computador o con a la videovigilancia, pues existen varios kits de desarrollo que incorporan una cámara conectada al módulo y varios códigos de ejemplo que facilitan mucho su uso.

De entre los distintos kits de desarrollo disponibles uno de los más destacables es el ESP32 cam (figura 29). Este es uno de los kits más económicos y ampliamente utilizados; debido a ello y a su pequeño tamaño se ha elegido para el proyecto.

Su función es crear su propia red Wifi y alojar un portal web a través del cual se pueda controlar el robot. Adicionalmente, se pretende que grabe y retransmita video a través de dicho portal, que controle la pantalla (que funciona con un protocolo SPI de 4 hilos) y que se comunique con el otro microcontrolador via UART.

Dado que el microcontrolador no incorpora conexión USB se requiere de utilizar un nódulo para poder actualizar su código. Para ello se ha decidió utilizar el módulo de conversión serial USB-TTL incluido con el microcontrolador (figura 29). Con respecto a este módulo cabe tener en cuenta que de normal impide el fácil acceso al reto de pines. No obstante, dado que se pretende conectar tanto la placa de desarrollo como el módulo a una PCB esto no supone ningún problema.

Figura 29. ESP32 cam [54]

Figura 30. Cámara ESP32 cam utilizada [55]

La placa de desarrollo incluye una cámara OV2640 de 2 megapíxeles que es capaz de capturar imagen a una resolución de hasta 1600x1200 píxeles [56], siendo más que suficiente para el caso de aplicación. No obstante, dado que el lugar donde se pretende ubicar la cámara se encuentra alejado de la misma se ha tenido que sustituir la cámara incluida por otra igual, pero con el conector más largo, siendo este de 75mm en vez de 8mm (figura 30).

Cabe destacar, que al módulo se le puede conectar una antena externa, y que incorpora una memoria PSRAM de 4 MB. [57]

Respecto a las dimensiones del ESP32 cam, estas se pueden observar en la figura 31.

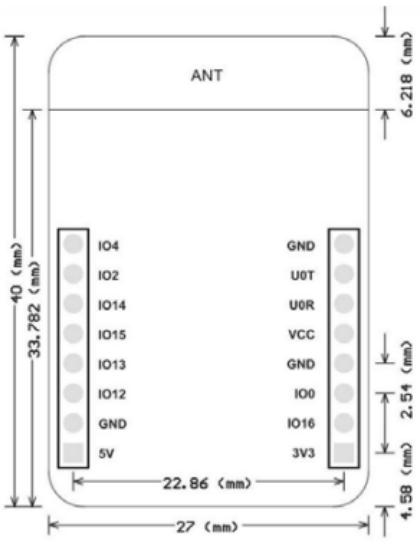


Figura 31. Dimensiones ESP32 cam [57]

En total, el kit microcontrolador dispone de 10 pines de propósito general disponibles, GPIO. No obstante, existen ciertas aspectos que se deben de tener en cuenta al usarlo:

- El GPIO 0 debe estar a nivel bajo para poder subir código. Para que la placa funcione de forma normal este se debe de conectar a nivel alto o sin conexión.
- Los GPIO 1 y 3 son los pines seriales (TX y RX, respectivamente) y deben conectarse a el módulo utilizado de conversión USB-TTL.
- El GPIO 4 está conectado a un LED ultra luminoso. Este LED puede utilizarse como flash y cabe considerar que se calienta mucho si se mantiene encendido.
- Los GPIO 2, 4, 12, 13, 14 y 15 no se pueden utilizar si se están utilizando las funciones de la tarjeta de memoria.
- El GPIO 33 está conectado a un pequeño LED rojo.
- Para el correcto funcionamiento del módulo este se debe alimentar a 5V. [47]

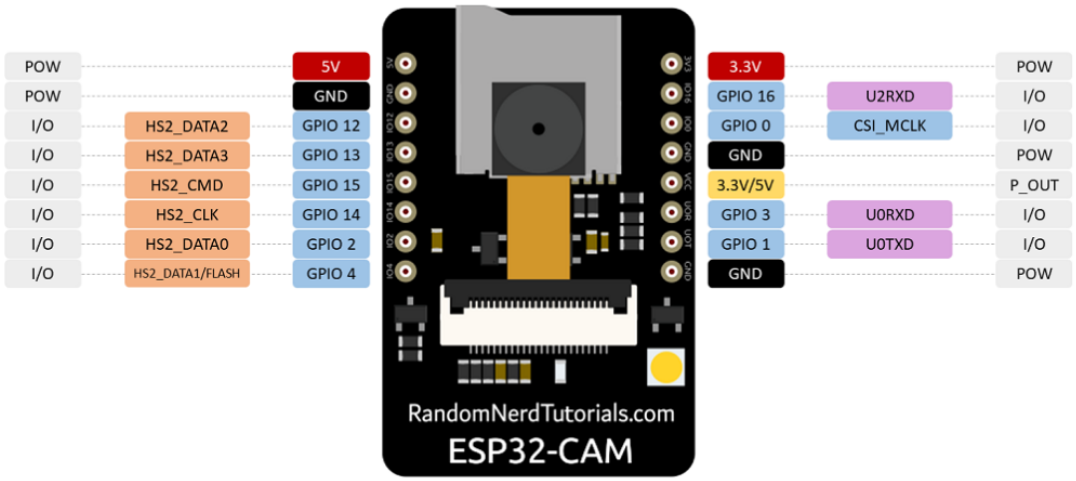


Figura 32. Pinout ESP32 cam [58]

4.2.2.2. ESP32 D1 mini

Debido a su pequeño tamaño (39x28x6mm) y a la gran cantidad de pines disponibles se ha decidido utilizar el kit microcontrolador “ESP32 D1 mini” para el proyecto.

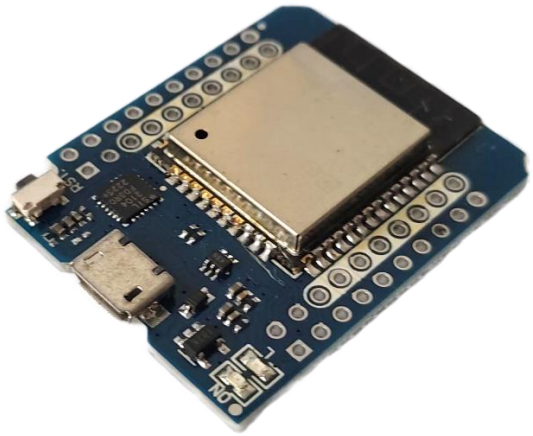


Figura 33. ESP32 D1 mini

En el robot, este microcontrolador se encargará de controlar:

- 9 servomotores mediante señales PWM.
- Las barras de luces LED a través de una salida digital
- Los 2 sensores capacitivos, con dos entradas digitales.
- Los sensores de distancia láser, siendo necesarios 3 salidas digitales para la activación y desactivación de los mismo y 2 pines más para la comunicación I2C.
- 2 pines para la comunicación UART con el ESP 32 cam.
- 1 entrada analógica para medir la tensión de la bacteria.

Esta placa de desarrollo incluye la conexión micro USB para poder conectar el dispositivo a un ordenador y subir el código correspondiente. Aun así, al igual que con el ESP32 cam, hay que evitar utilizar los GPIO 0, 1 y 3 por su papel a la hora de subir código al microcontrolador y los GPIO 34, 35, 36 y 29 solo pueden ser utilizados como entrada.

Descontando estos pines, en total el dispositivo ofrece 19 pines con propósito general: los GPIO 2, 4, 5, 12, 13, 14, 15, 16, 17, 18, 19, 21, 22, 23, 25, 26, 27, 32 y 33.

Para evitar posibles problemas en la comunicación I2C se han utilizado los pines predeterminados: GPIO 21 (SDA) y GPIO (SCL).[59]

Al igual que el ESP32 cam este módulo también incorpora una memoria PSRAM de 4MB.

El “pinout” del dispositivo este se encuentra en la figura 57, en la siguiente página.

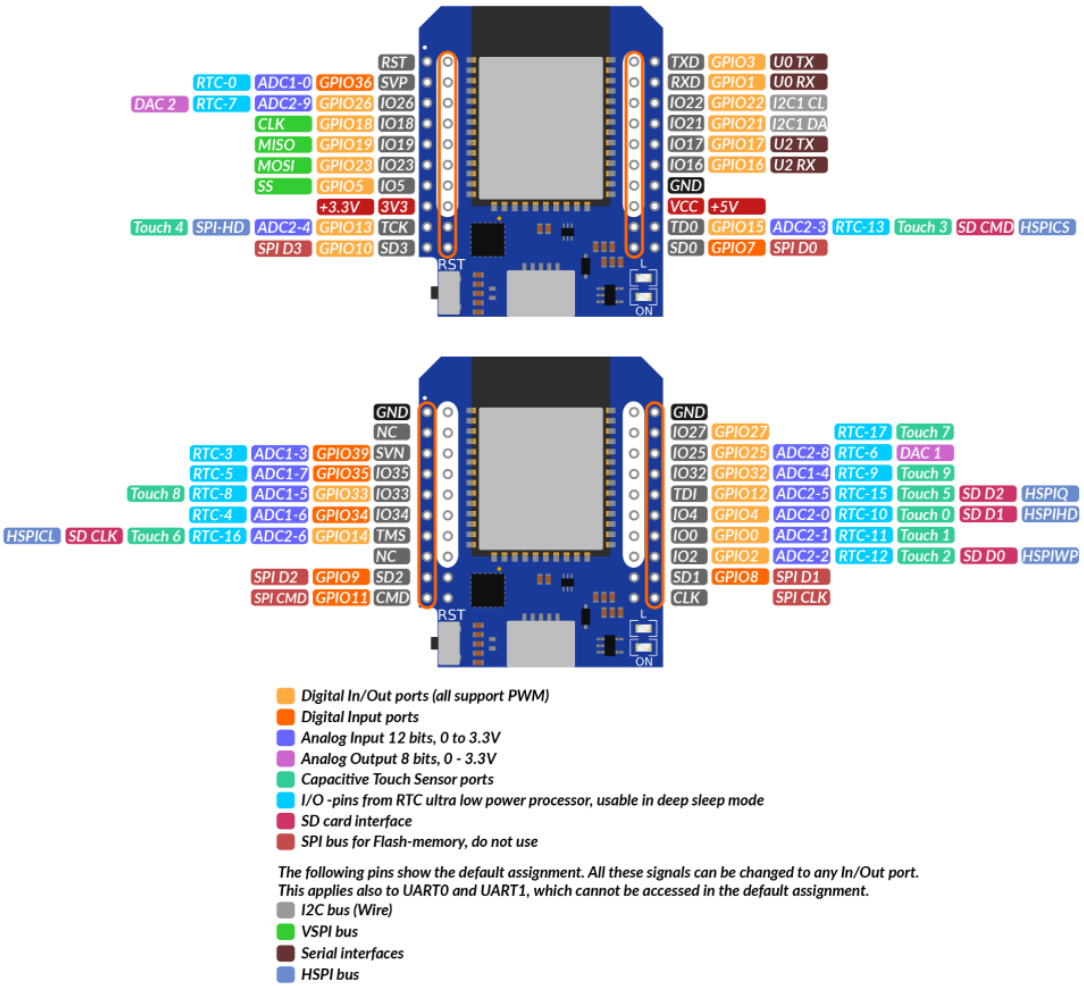


Figura 34. Pinout ESP 32 D1 mini [59]

En la figura 35 se muestra el esquema de conexión de los microcontroladores con los componentes:

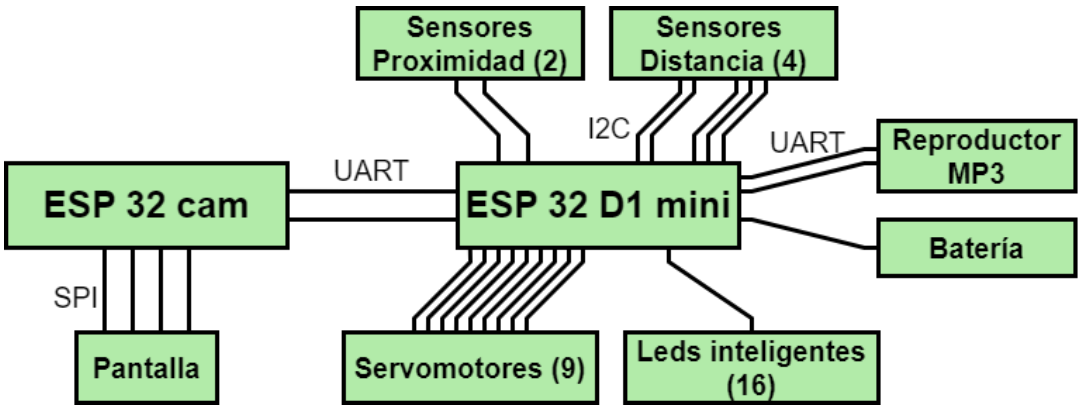


Figura 35. Conexión microcontroladores con los componentes

4.2.3. Sensores

4.2.3.1. Sensores proximidad capacitivos

Con el fin de detectar cuando se toca o acaricia el robot se ha decidido usar sensores capacitivos basados en el integrado TTP223. Existen diversos modelos comercializados, de entre ellos se ha decidido utilizar el denominado como “mini” (figura 36), por su reducido tamaño.

Figura 36. Pulsador táctil TTP223 mini

Este dispositivo cumple la misma función que un interruptor o pulsador, con la diferencia de que no necesita de ningún accionamiento o contacto físico. Pudiendo ser utilizado tras una superficie delgada no conductiva.

Su funcionamiento se basa en la medición de la variación de la capacitancia, e incorpora una lámina conductora conectada a la alimentación. Cuando acercamos nuestro dedo este actúa como otra capa conductora conectada a una masa virtual y el aire actúa como dieléctrico, formando un “condensador”, almacenándose una pequeña carga eléctrica. La cual el circuito integrado TTP223 detecta y en respuesta genera una señal digital. [60]

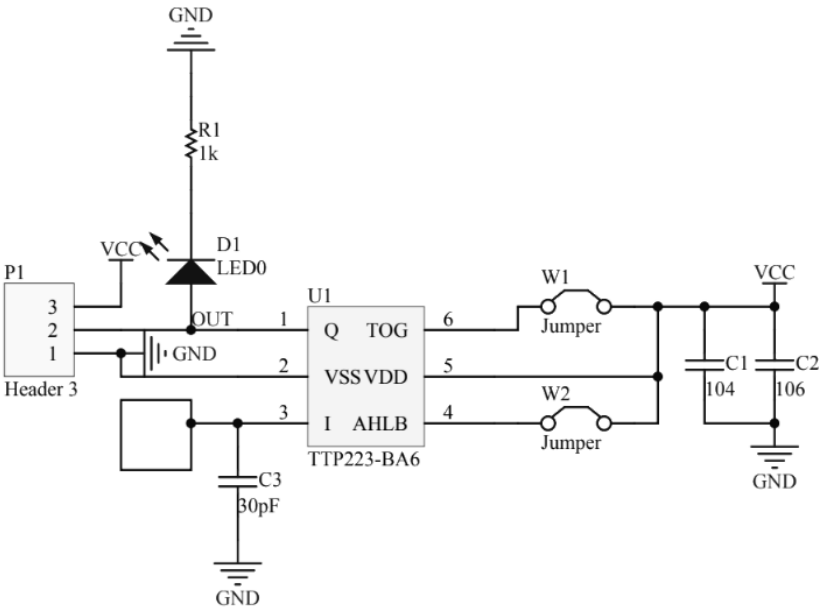


Figura 37. Circuito TTP223 mini [61]

En un principio el dispositivo funciona como un pulsador normalmente cerrado. No obstante, cortocircuitando los puentes A y B (figura 38) se pueden obtener los siguientes modos de funcionamiento:

- Pulsador normalmente abierto, cortocircuitando el puente A.
- Interruptor, con estado inicial bajo, cortocircuitado el puente B.
- Interruptor, con estado inicia alto, conectando ambos puentes. [62]

Figura 38. Puentes A y B

Finalmente, del sensor capacitivo TTP223 mini, se deben destacar las siguientes características:

- Se alimenta con una tensión de entre 2,5 y 5,5 V.
- La tensión de la señal digital es la de alimentación.
- Su consumo es de 11µA en reposo y 2.3mA cuando está activo.
- El área de detección es de 11 x 10mm aproximadamente.
- Incluye un led que indica el estado de la salida.
- Se puede variar su modo de funcionamiento.

4.2.3.2. Sensores distancia láser

Para evitar que el robot se caiga se ha decidido utilizar 4 sensores de distancia láser VL53L0X (figura 39). Este sensor se comercializa en varios módulos, de entre ellos se ha elegido el GY-530 (figura 40) por su reducido tamaño.

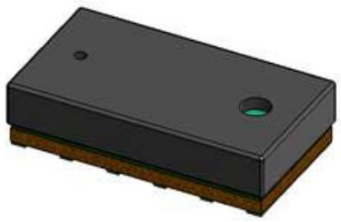


Figura 39. Sensor VL53L0X [63]

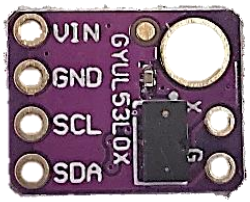


Figura 40. Módulo GY-530

Se ha optado por un sensor láser y no por el típico sensor de ultrasonidos HC-SR04 ya que el sensor láser obtiene resultados incluso cuando la superficie está inclinada respecto al sensor. Además, de por su menor tamaño, funcionar a 3,3V, su mayor velocidad de medición y la posibilidad de poder conectar varios sensores en un mismo bus de comunicación I2C. [64]

El dispositivo utiliza la tecnología TOF, “time of flight”, o tiempo de vuelo. Para medir la distancia emite un haz de luz de 940 nm y recibe la luz reflejada, mide el tiempo entre ambos sucesos y calcula la distancia teniendo en cuenta la velocidad de la luz. En el caso de el sensor de ultrasonidos este es proceso es más lento, pues el sonido viaja más rápido que la luz. [65]

Según la información disponible en la red, el alcance del sensor varía en función del ambiente, siendo capaz de alcanzar 1 metro en sitios con mucha luz natural y hasta 2 metros en interiores[65].

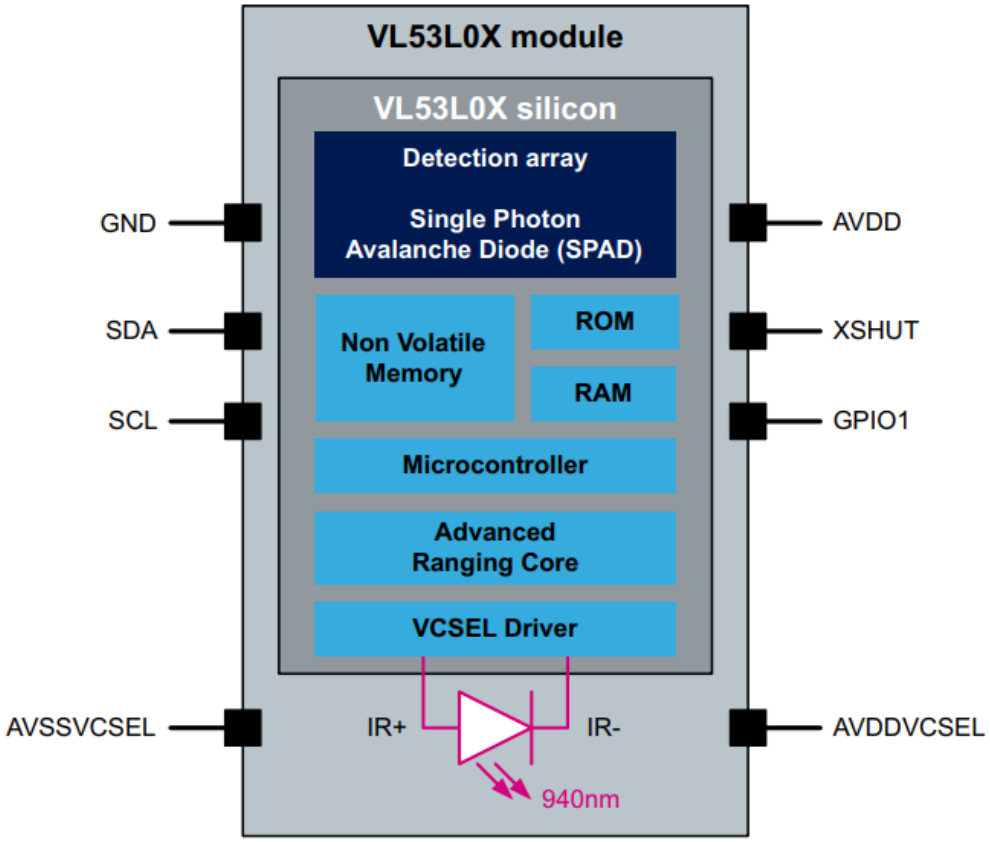


Figura 41. Diagrama de bloques VL53LOX [63]

A la hora de conectar varios sensores al mismo bus I2C aparece el problema de que todos los sensores tienen en un principio la misma dirección. Por suerte mediante código se les puede mandar una instrucción con la que se les asigna otra dirección de memoria.

Para poder mandar esta dirección únicamente a un sensor estos incorporan el pin XSHUT. Este pin se debe conectar a una salida digital del microcontrolador correspondiente. Cuando la salida digital está a nivel bajo el sensor funciona con normalidad, pero cuando esta se conecta a nivel alto el sensor entra en modo de bajo consumo. Para asignar la dirección a un sensor lo que se debe hacer es hacer que antes el resto de los sensores entren en modo bajo consumo, haciendo que solo el sensor deseado reciba la instrucción. [66]

Desafortunadamente, esta nueva dirección I2C se almacena en la memoria RAM, perdiéndose dicha información al apagarse el dispositivo. Consecuentemente se deberá asignar distintas direcciones a los sensores en la secuencia de encendido, cuyo funcionamiento se explica más adelante.

4.2.4. Actuadores: servomotores

A la hora de mover las extremidades del robot los servomotores son la mejor opción pues incorporan control angular de una forma sencilla. Utilizando este tipo de motores se evita la necesidad de colocar codificadores para controlar el movimiento de los ejes, simplificando el diseño del hardware. Así mismo, dado que únicamente se requiere de una señal PWM para indicar el ángulo deseado, el software también se simplifica.

Dado que se pretende que el robot sea lo más pequeño y económico posible se ha optado por utilizar motores MG90S. Estos motores se pueden alimentar con tensiones de entre 4,8 y 6V, siendo su par de 1,8 kg/cm a 4,8V y de 2,2 kg/cm a 6V [67]. Son muy similares a los clásicos motores SG90, pero estos incorporan engranajes metálicos, aumentando su durabilidad.

Debido a como se ha diseñado el robot, únicamente los motores situados en el segundo eje de cada extremidad cargan con el peso del robot. Teniendo en cuenta que su respectivo eslabón mide 3,5cm y que el robot debe de ser capaz de sostener su peso con 2 patas; se plantea el siguiente cálculo para comprobar si el par de los motores es suficiente:

$$F_{motor} = 2{,}2\ {}^{kg}/_{cm} \cdot \frac{1}{3{,}5\ cm} \cdot 9{,}81\ {}^{m}/_{s^2} = 6{,}166\ N$$

$$F_{necesaria} = \frac{1\ kg}{2} \cdot 9{,}81\ {}^{m}/_{s^2} = 4{,}905\ N$$

Como se puede observar, la fuerza del motor es mayor a la fuerza necesaria, por ello podemos afirmar que los motores disponen de par suficiente. No obstante, por la naturaleza del robot, para levantarlo no únicamente hay que ser capaz de vencer a la fuerza de la gravedad, sino también hay que vencer la fuerza que genera el rozamiento con el suelo. Dado que estos cálculos resultan notablemente más complejos no se han realizado y se comprobará si el robot es capaz de levantarse utilizando el prototipo.

Figura 42. Servomotor utilizado

Respecto a las dimensiones de los servomotores, dependiendo del fabricante es posible que varíen ligeramente, para diseñar el robot se utilizó un pie de rey para medir el motor, obteniéndose las dimensiones de la figura 43.

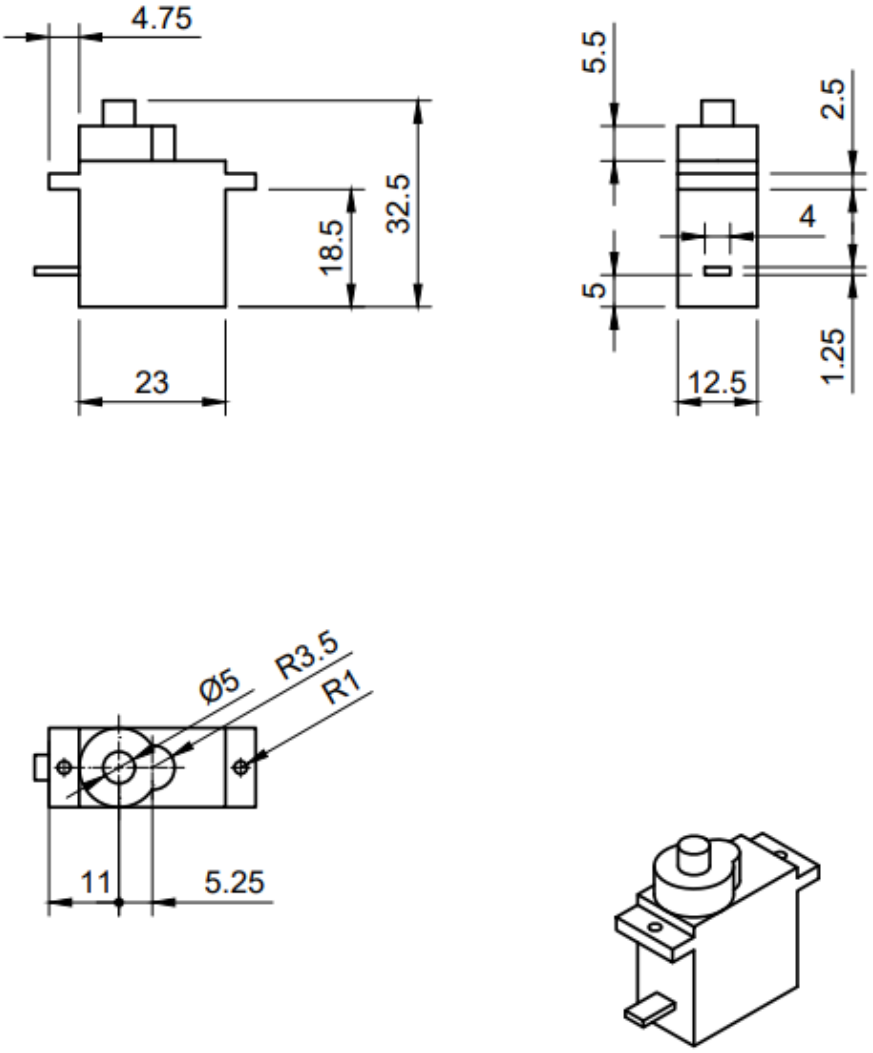


Figura 43. Dimensiones servomotor utilizado

4.2.5. Interfaz visual

4.2.5.1. Pantalla

De entre las distintas pantallas disponibles en el mercado se ha optado por la pantalla “1.5 inch RGB OLED Module” por sus dimensiones, por ser a color y OLED.

En la tecnología OLED, “Organic Light-Emiting Diode”, cada pixel se ilumina de forma independiente, ofreciendo un mayor brillo y contraste con menor consumo energético que las pantallas LCD, “Light Emiting Diode”. Debido a sus características, las pantallas OLED se ven mejor en condiciones donde la luz ambiental dificulta la lectura en otro tipo de pantallas y ofrecen ángulos de visualización más amplios.

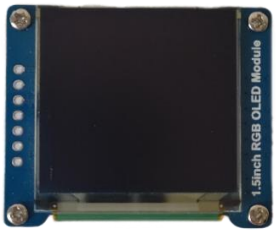


Figura 44. Pantalla, vista frontal

Figura 45. Pantalla, vista trasera

Para comunicarse con el microcontrolador se requiere de conexión SPI de 4 o de 3 líneas. Funcionando con 4 hilos de forma predeterminada. En la parte posterior de la pantalla hay un puente que se puede soldar para alternar a una interfaz de 3 hilos. Se ha decidido utilizar una interfaz de 4 líneas ya que la biblioteca utilizada no era compatible con 3 hilos.

De la pantalla, cabe destacar las siguientes características:

- De 3.3V a 5V de tensión de alimentación, la tensión de alimentación debe ser la misma que la de control.
- Controlador SSD1351.
- Resolución 128x128.
- Tamaño de pixel 0.045x0.194 mm.
- Color 65k.

Respecto a las dimensiones, estas se pueden observar en la figura 46.

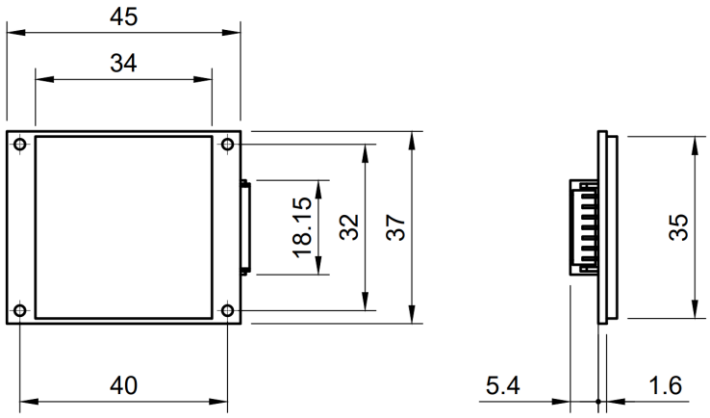


Figura 46. Dimensiones pantalla utilizada

4.2.5.2. Cada LEDs inteligentes

Que el robot incorpore luces de colores puede aumentar el atractivo del robot a los niños pequeños captando su atención y fomentando su participación. Así mismo, los colores pueden utilizarse como una forma de comunicación no verbal, utilizándolos para transmitir el estado de ánimo del robot.

Para el funcionamiento de un led RGB común se requieren 3 salidas PWM de un microcontrolador. Dentro de un LED RGB hay 3 LEDs independientes, uno rojo uno verde y otro azul, que juntando sus luces dan la sensación de ser una única luz.

Es por ello, que si se quisiera controlar varios LEDs RGB con el ESP32 no se podría conectar nada más. Es por ello que se ha decidido utilizar LEDs inteligentes.

El chip controlador LED WS2812, también conocido como “NeoPixel” por los productos de Adafruit que incorporan este chip, son unos circuitos capaces de realizar el control individual de un LED RGB. Para el control únicamente se requiere de una salida digital, pudiéndose conectar tantos LEDs como se desee de forma encadenada y pudiéndolos controlar de forma independiente.

Estos circuitos integrados se comercializan tanto de forma individual como en módulos que integran varios chips ya conectados. De estos, se pueden encontrar tanto con forma lineal, en barra o en tira, como con forma circular.

Dada la anatomía del robot, se ha decidido utilizar 2 módulos “Barra LED RGB Digital WS2812 8×5050” situados a los laterales de la parte superior del robot.

Figura 47. Barra LED RGB Digital WS2812 8×5050

4.2.6. Audio: reproductor MP3

Dado que sintetizar una voz es una tarea muy compleja, para que el robot “hable” se ha decidido utilizar un módulo capaz de reproducir audios grabados previamente.

Se puede conectar el microcontrolador ESP 32 a un lector de tarjetas SD y a un altavoz para que este se encargue de esta función, no obstante, ya se le exigen suficientes tareas y no las puede hacer todas simultáneamente, es por ello que se ha decidido utilizar un módulo para reproducir los audios.

Existen varios módulos en el mercado que pueden cumplir esta función, de entre ellos se ha seleccionado el “DFPlayer Mini MP3 Player” (figura 48) por su reducido tamaño, su precio y su fácil implementación en el proyecto. Este módulo dispone de una librería para Arduino compatible con ESP32 y únicamente se requieren de un par de comandos para su uso.

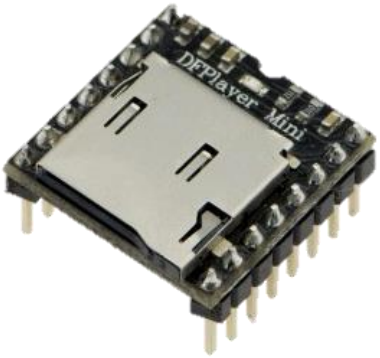


Figura 48. DFPlayer Mini MP3 Player

Figura 49. Altavoz utilizado

Para su control, se debe conectar al microcontrolador utilizando comunicación UART. Según el fabricante para reducir el ruido se recomienda la colocación de resistencias de 1kΩ en las líneas tanto RX como TX.

El dispositivo también puede funcionar sin la necesidad de ser controlado por un microcontrolador e incluye pines con los cuales conectando interruptores se puede subir el volumen, pasar al siguiente archive, pausar la reproducción, etc.

A la hora de conectar el reproductor a un altavoz se tienen 2 opciones:

- Utilizar los pines DAC_R y DAC_L para conectar un amplificador de sonido.
- Utilizar los pines SPK_1 y SPK_2 para conectar directamente un altavoz que no debe tener más de 3W. [68]

Tras probar con un par de altavoces se ha decidido utilizar un altavoz de 8Ω, 2W y 40 mm de diámetro, figura 49.

Que el módulo funcione con una lógica de 5V resulta ideal para trabajar con muchos microcontroladores, no obstante, con el ESP32 no es el caso. Dado que el microcontrolador empleado funciona con una lógica de 3.3V se requerirá el uso de un convertidor de nivel lógico para adaptar los voltajes.

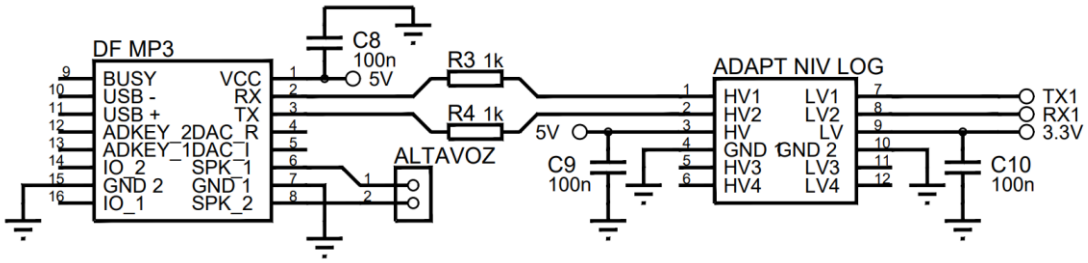


Figura 50. Bloque de audio del circuito electrónico

4.2.6.1. Convertidor de nivel lógico

Dado que el reproductor de MP3 funciona con una lógica de 5V y el microcontrolador ESP32 de 3.3V se requiere del uso de un convertidor de nivel lógico bidireccional. Concretamente se ha utilizado el módulo TE291.

Tal y como su nombre indica, los convertidores de nivel lógico son circuitos capaces de elevar o reducir el voltaje de una señal digital, siendo indispensables para conectar dispositivos que funcionan a diferentes tensiones.[69]

Estos circuitos se comercializan en módulos con capacidad para 2, 4 u 8 canales. Para el proyecto solo se requieren 2 pero estos módulos son más rebuscados y resultan difíciles de obtener, es por ello que se ha utilizado un módulo de 4 canales, concretamente figura 51.

Figura 51. Adaptador de nivel lógico TE291

En la figura 52 se puede observar del adaptador de nivel lógico.

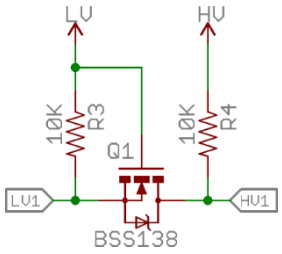


Figura 52. Circuito Adaptador de nivel lógico [69]

4.2.7. Condensadores y resistencias

Con el fin de reducir el posible ruido electrónico se ha decidido colocar condensadores de desacoplo entre la alimentación y la masa de los distintos módulos utilizados. En total se han colocado 9 condensadores con este propósito, siendo estos de 100nF y de tipo cerámico.

También se han colocado condensadores electrolíticos para estabilizar la tensión de alimentación de los motores y absorber y filtrar el ruido que estos generan. Para ello, se han colocado 2 condensadores electrolíticos en paralelo de 470µF, pues se recomienda 100µF por motor y el robot incorpora 9.

Adicionalmente, se han colocado 2 condensadores de 1µF para estabilizar el regulador de 3.3V y otro de 100µF a la salida del elevador a 5Vcon el mismo propósito.

Respecto a las resistencias, las resistencias R1 y R2 forman un divisor de tensión que permite la medida del nivel de la batería. Por otro lado, las resistencias R3 y R4 se han ubicado en las líneas de transmisión y recepción UART que conectan el adaptador de nivel lógico con el módulo “DFPlayer Mini MP3 Player”. Según el fabricante, estas ayudan a reducir el ruido.[68]

4.3. PCB

Con el propósito de conectar los componentes electrónicos entre sí de forma compacta y profesional se ha decidido diseñar una PCB.

De entre los distintos programas disponibles se ha utilizado el Proteus 8, ya que al tener experiencia previa el proceso de diseño ha sido más corto que utilizando un programa nuevo.

A la hora de diseñar el circuito, se han colocado tanto el puerto micro USB del cargador y de los microcontroladores como la ranura SD del reproductor de audio en un mismo lateral de la placa, con el objetivo de que pudieran ser accesibles por la parte de atrás del robot. Lugar donde se ha colocado una puerta.

Así mismo se ha colocado la placa de desarrollador centrada en el extremo contrario para que esta se encuentre lo más cerca posible del sitio donde se coloca la cámara.

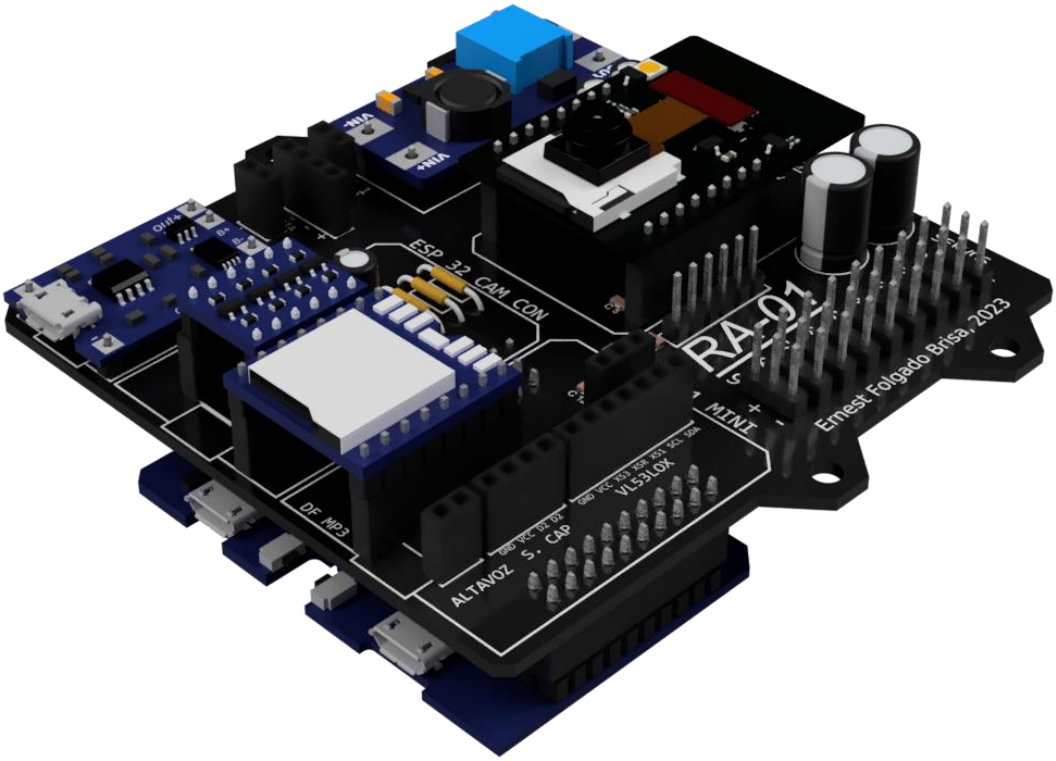


Figura 53. Render realizado en Fusion 360 de la PCB deseñada

Para las pistas se han utilizado las opciones predeterminadas de Proteus, siendo las de alimentación C30 y el resto T15. Dado la cantidad y densidad de conexiones únicamente se ha necesitado de dos capas.

Cabe destacar, que para el diseño de las pistas no se ha utilizado el “auto-trazado”, sino que estas se han enrutado de forma manual siguiendo las indicaciones de la asignatura cursada “Electrónica Orgánica y Procesos en el Diseño Electrónico” (12195). Además, se han situado dos planos de masa tipo entramado, tanto en la capa inferior como en la inferior con el fin de evitar interferencias electromagnéticas.

Se ha decidido situar las pistas de alimentación en la capa superior y las pistas de señal en la capa inferior. No obstante, dado que no ha habido espacio en la capa inferior para todas las pistas de señal en la capa inferior, hay algunas que se han situado en la capa superior. En un principio se ha intentado evitar el uso de vías, a pesar de ello, se ha requerido el uso de 4 vías.

En las figura 54, y 55 se pueden observar las capas de cobre superior e inferior.

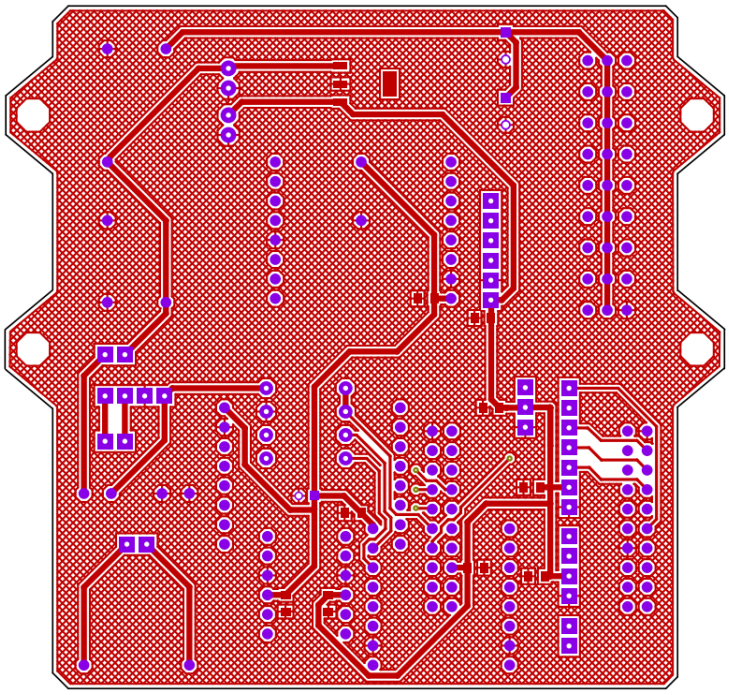


Figura 54. Capa cobre superior PCB

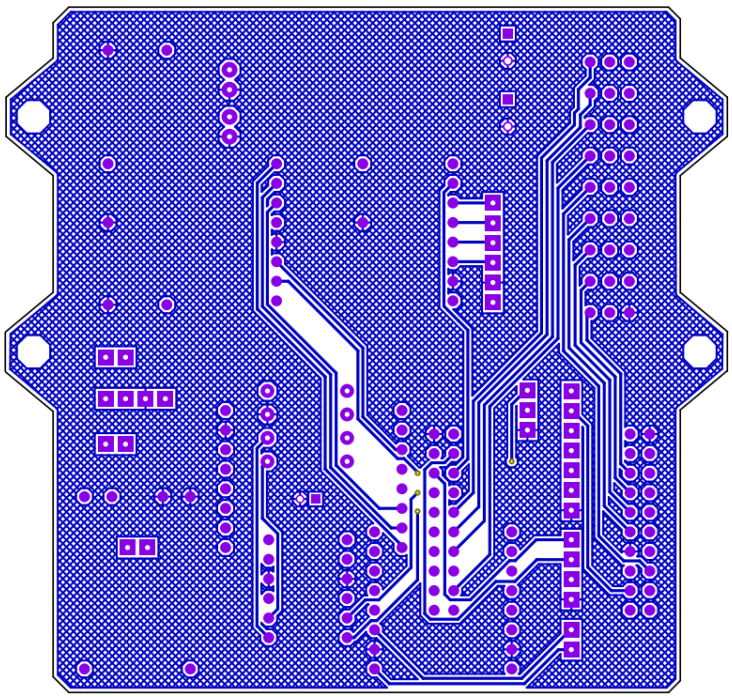


Figura 55. Capa cobre inferior PCB

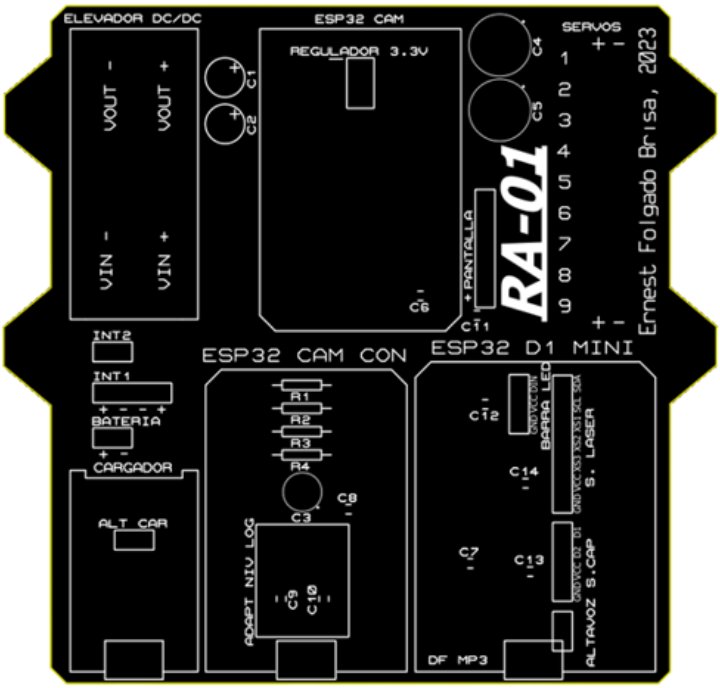


Figura 56. Serigrafía PCB

Con respecto a las dimensiones de la placa (figura 57), esta se ha hecho lo más pequeña posible, cumpliendo los objetivos previos.

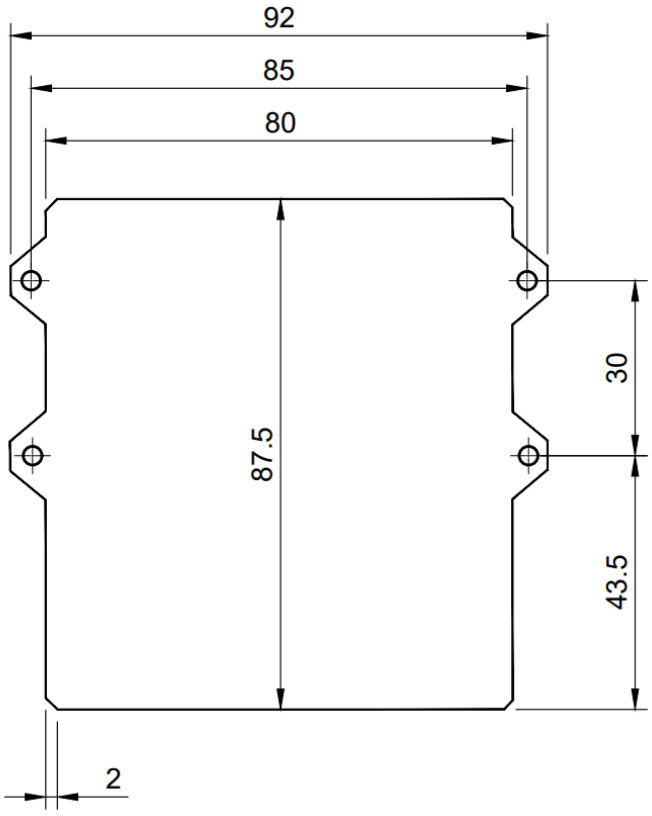


Figura 57. Dimensiones PCB

4.4. Componentes mecánicos

Para el diseño de las distintas piezas que forman el robot se ha utilizado el programa Fusion 360 de Autodesk. Además de para diseñar las piezas, se ha utilizado el programa también para renderizar el diseño y para crear los dibujos de las piezas. Una vez obtenidos los dibujos estos se han exportado en formato “dwg” y se han hecho los planos en Autocad.

Para fabricar el primer prototipo se ha hecho uso del servicio de impresión 3D de la escuela; donde se ha decidido utilizar distintos métodos de impresión para las distintas piezas. Concretamente se ha decido usar impresión por estereolitografía (SLA) para la pieza superior e inferior, mientras que para el resto de las piezas se ha utilizado impresión por deposición de material fundido (FDM, por sus siglas en inglés).

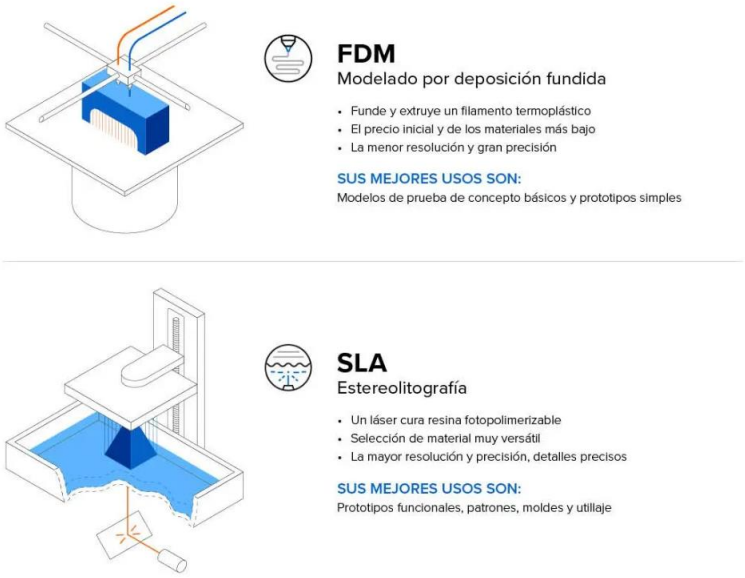


Figura 58. Tecnologías de impresión 3D para plásticos [70]

La impresión SLA utiliza resinas líquidas fotosensibles que se solidifican al exponerse a la luz ultravioleta. La impresora utiliza una luz láser ultravioleta para solidificar selectivamente la resina requerida para formar capa a capa el objeto deseado.

En cambio, en la impresión en la impresión FDM se utiliza un filamento plástico que se derrite en la boquilla y se deposita formando capas sucesivas que constituyen la pieza impresa.

Este último método resulta más económico y rápido que la impresión SLA. La cual además requiere de cierto post procesamiento de las piezas. En la impresión SDA una vez impresa las piezas estas deben ser lavadas en un solvente, para eliminar el exceso de resina, y curadas utilizando luz ultravioleta para obtener una superficie suave ya acabada.[70]

Es por ello que las piezas más complejas han sido impresas en resina mientras que para el resto se ha utilizado filamento.

A continuación, se describe el propósito de cada pieza y se pueden observar sus respectivos renderizados. Cabe destacar, que se han diseñado las piezas para que se unan utilizando tornillos roscados de métrica 3. Sus planos se pueden encontrar en el anexo 4.

4.4.1. Pieza inferior

En primer lugar, se ha diseñado la pieza inferior. Esta pieza alberga: la batería, los motores, que mueven los primeros ejes de las patas; los sensores de distancia y los interruptores de encendido.

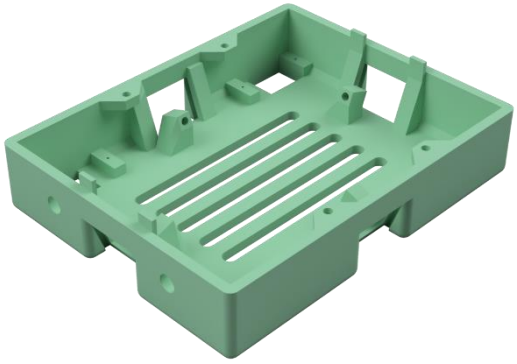


Figura 59. Render pieza inferior

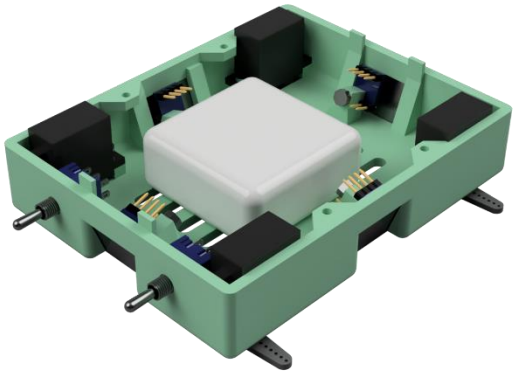


Figura 60. Render pieza inferior con los componentes que contiene

Tal y como se puede observar en la figura 59, la pieza cuenta con 4 soportes, en los cuales se sitúan los sensores de distancia, en frente de estos se han hecho aberturas para posibilitar su funcionamiento. Con el fin de protegerlos, se ha decidido cerrar las aberturas con ventanas de metacrilato de 26 x 18mm con 2mm de grosor. Para poder colocar las ventanas se han diseñado unas ranuras de 2,25mm.

El metacrilato es un material transparente y resistente, con propiedades similares al vidrio, pero más resistente a impactos y ligero. La facilidad con la que se puede cortar lo hace ideal para su uso en el presente proyecto, pudiéndose cortar con un cúter. Sin embargo, en este caso se ha hecho uso de los servicios de la universidad y se ha cortado por medio de corte láser.

Sobre esta pieza se construye el resto del robot y a la hora de montarlo, antes de colocar el resto de las piezas, se deben de situar las patas. En la figura 61 se puede observar la parte inferior del robot en su conjunto.

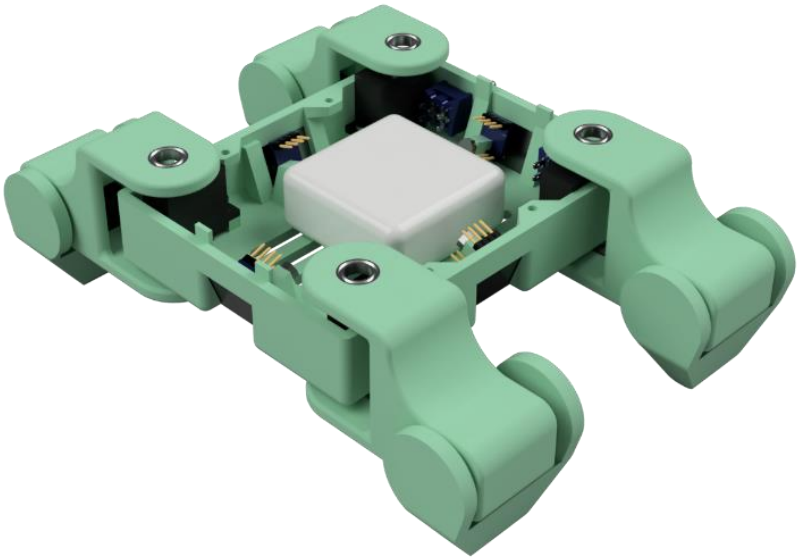


Figura 61. Render parte inferior

4.4.2. Pieza intermedia

Como su nombre indica, esta pieza se ubica entre la pieza inferior y la superior. Además de servir como unión entre estas, sobre la pieza intermedia se coloca el soporte del altavoz, la pinza y la PCB.

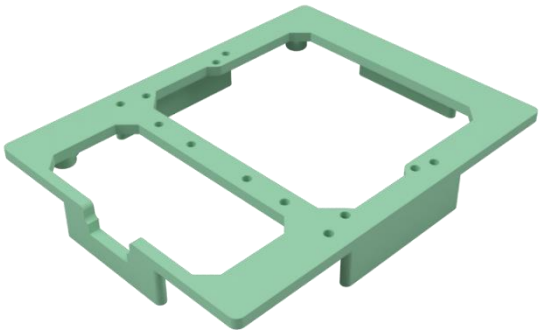


Figura 62. Render pieza intermedia

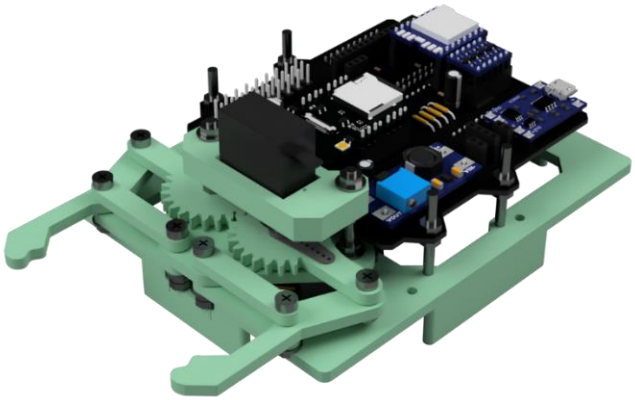


Figura 63. Render pieza intermedia con PCB y pinza

Adicionalmente, esta pieza también cumple un papel crucial en la sujeción de las patas del robot.

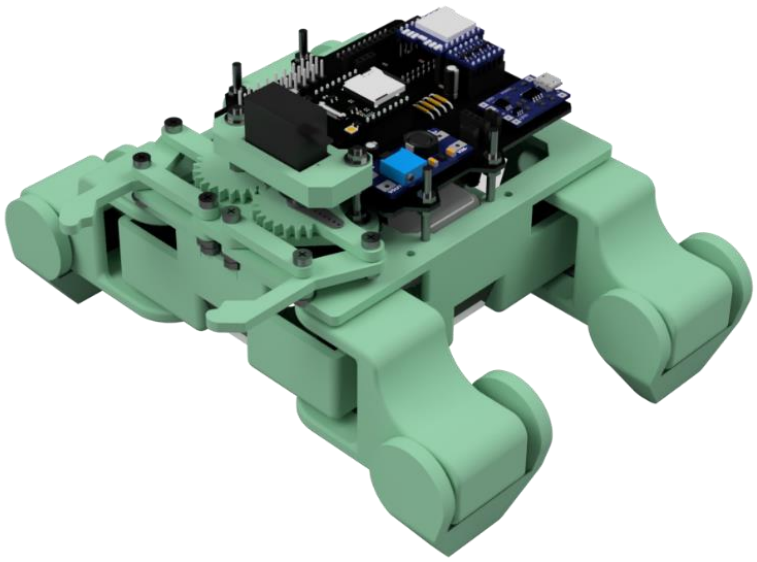


Figura 64. Render parte inferior e intermedia

4.4.3. Pieza superior

La pieza superior es la que mayor impacto tiene sobre la estética del dispositivo. Es por ello por lo que se ha buscado que sea lo más “natural” posible, evitando las formas poligonales y tratando que su superficie fuera suave, continua y estilizada.

Para realizar el diseño de esta pieza se ha utilizado la herramienta “crear forma de Fusion 360. En esta herramienta, se aplica la técnica de modelado T-spline, permitiendo crear superficies complejas y orgánicas de forma sencilla. Obteniendo la pieza que se muestra en la figura 65.

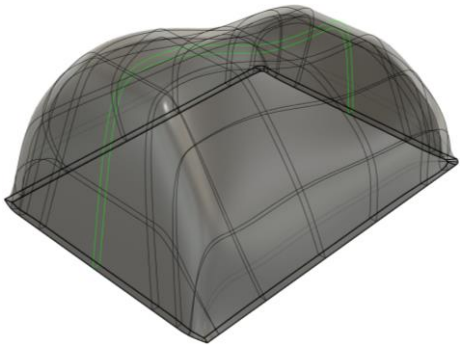


Figura 65. Forma creada para la pieza superior

Posteriormente, se ha editado esta pieza añadiendo las aberturas correspondientes y la bisagra de la puerta, figura 66.

Figura 66. Render pieza superior

En esta pieza se ubica la pantalla, la cámara, las barras LED, las unas piezas impresas en 3d con material traslucido que cubren las luces, los sensores de proximidad y la puerta.

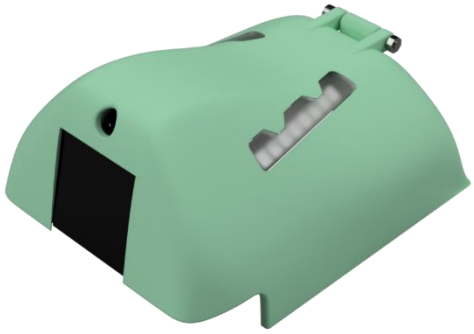


Figura 67. Render conjunto parte superior

4.4.4. Patas

A la hora de diseñar el robot se planteó la posibilidad de que sus patas tuvieran 2 o 3 articulaciones, para finalmente decidir que el robot tuviera 2 articulaciones por pata.

Si RA-01 hubiera tenido 3 articulaciones por pata se habrían necesitado más pines para el control de los servomotores requeridos. Para solucionar el problema se habría tenido que implementar un controlador para los motores. Más articulaciones por pata habría resultado en unas patas más largas y frágiles, resultando en un robot menos robusto y con una estética seguramente no tan amigable. Además, no se podría haber reutilizado el modelo de movimiento del robot RR-01 y se habría tenido que desarrollar la cinemática inversa de las patas. No solo añadiéndose el coste de los materiales sino el del tiempo de desarrollo, encareciendo el proyecto.

En definitiva, las patas diseñadas cuentan con 2 piezas, el eslabón 1 y 2.

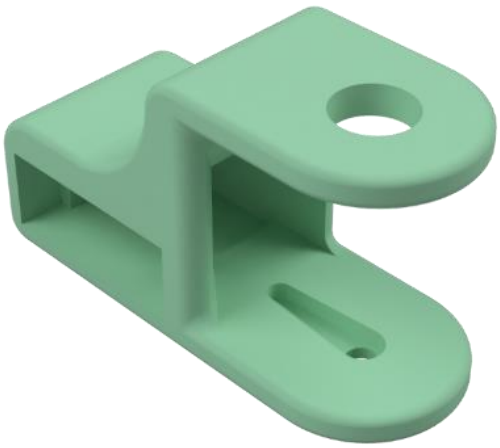


Figura 68. Render eslabón 1 pata, vista 1

Figura 69. Render eslabón 1 pata, vista 2

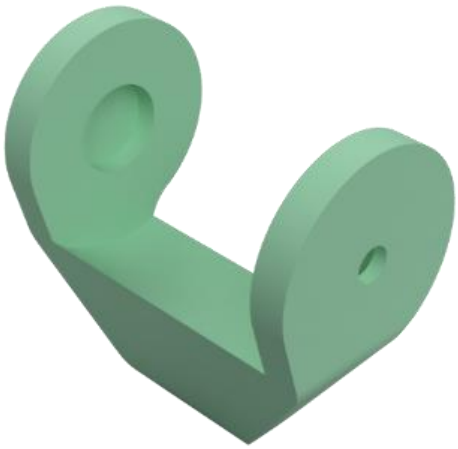


Figura 70. Render eslabón 2 pata, vista 1

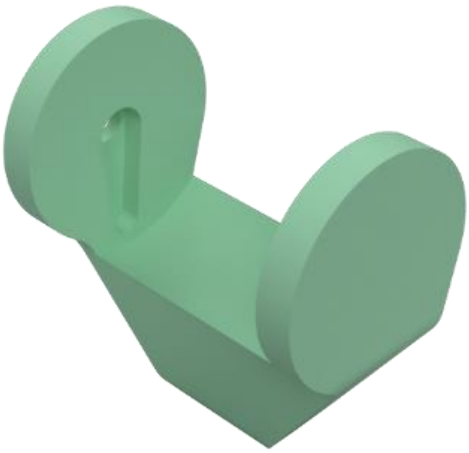


Figura 71. Render eslabón 2 pata, vista 2

El servomotor que controla el eslabón 1 se sitúa en la pieza inferior, mientras que el que gobierna el eslabón 2 se halla dentro del eslabón 1 (figura 72). Para mayor simplicidad, se han ubicado los motores directamente en el eje sobre el que actúan. Adicionalmente, se han colocado rodamientos para que el movimiento de las patas sea lo más fluido posible.

Figura 72. Colocación servomotor

Figura 73. Colocación rodamiento

En la figura 74, se puede observar la pata completa.

Figura 74. Render pata completa

Finalmente cabe destacar que todas las patas no son idénticas, siendo simétricas las de la derecha respecto las de las izquierdas.

4.4.5. Pinza

A diferencia del resto de piezas, no se considera la pinza un elemento indispensable del robot, planteándose en su momento no incluirla (figura 75). No obstante, finalmente se incluyó, ya que le aporta carácter y puede ser un elemento útil a la hora de realizar juegos con el robot.

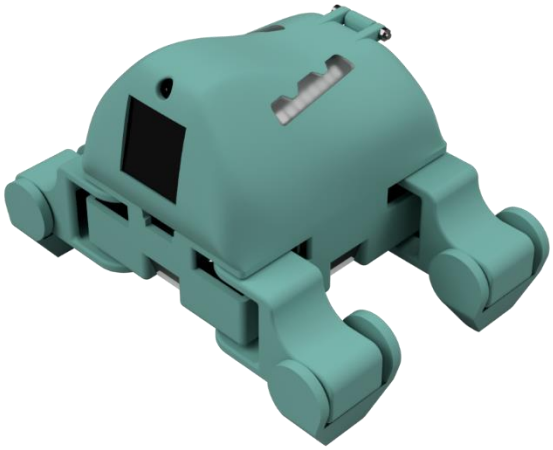


Figura 75. Robot sin pinza

La pinza, deseñada imitando distintos modelos encontrados en internet, consta de un total de 10 piezas:

- Pieza pinza 1 x1
- Pieza pinza 2 x1
- Pieza pinza 3 x4
- Pieza pinza 4 x2
- Base pinza x1
- Soporte servo x1

Una vez montadas todas las piezas la pinza queda tal y como se muestra en la figura 76. Para unirlas, al igual que en el resto del robot se ha decidido utilizar tornillos de métrica 3. Las dimensiones, posición y cantidad de los mismo se pueden encontrar en el anexo 4 planos.

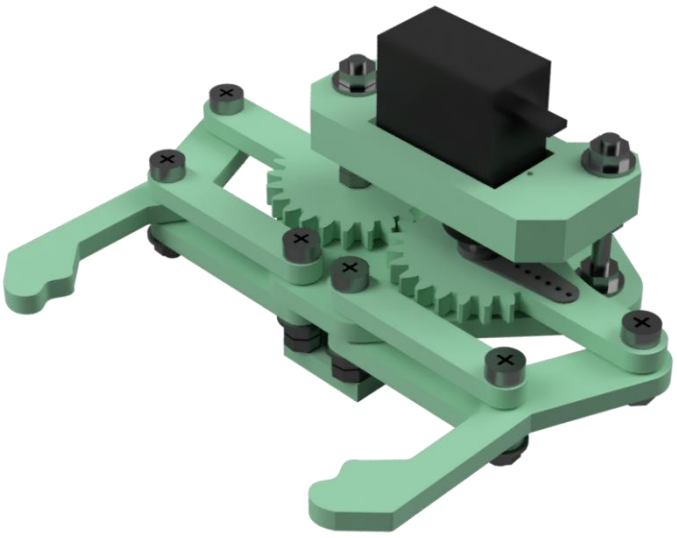


Figura 76. Render pinza completa

La pieza pinza 1 (figura 77) es la que se conecta al motor y transmite el movimiento a uno de los dos extremos de la pinza por medio de un cuadrilátero articulado. Así mismo, esta pieza también incorpora varios dientes con los que hace rotar a la pieza 2 (figura 77). De esta forma, simétricamente, la pieza 2 mueve la otra mitad de la pinza

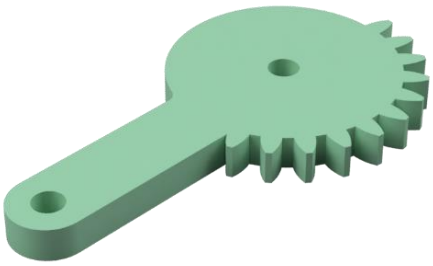


Figura 77. Render pieza pinza 2

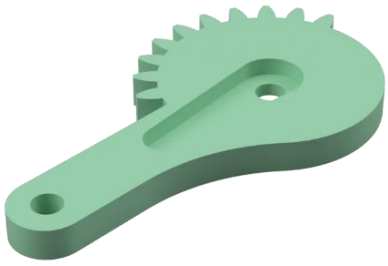


Figura 78. Render pieza pinza 1

El cuadrilátero articulado queda formado por la base de la pinza, la pieza 3 y la pieza 4, figuras 79 y 80 respectivamente.

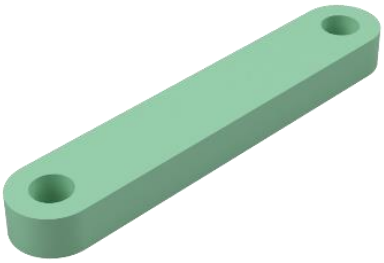


Figura 79. Render pieza pinza

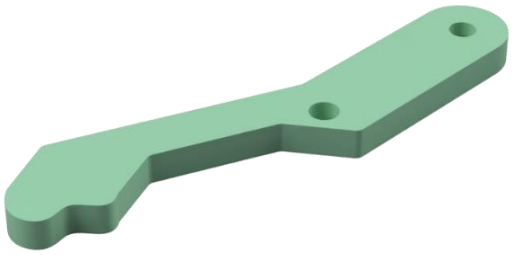


Figura 80. Render pieza pinza 4

Por otro lado, la pinza se construye sobre la pieza: soporte pinza, cuya forma cumple la función de encajar con el resto del chasis del robot (figura81).

Finalmente, para la fijación del motor se ha diseñado el soporte del servomotor (figura 82). Esta pieza se ubica suspendida sobre la pinza gracias a dos tonillos roscados.

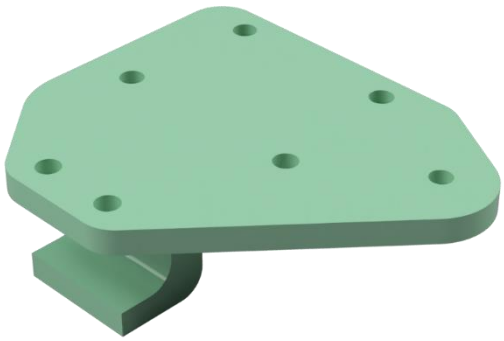


Figura 81. Render soporte pinza

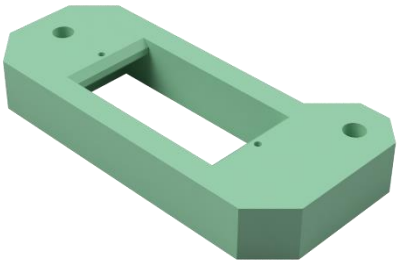


Figura 82. Render soporte servomotor

4.4.6. Puerta

Dado que las conexiones micro USB de los microcontroladores, puerto de carga y la tarjeta SD se ubican en la parte trasera del robot, se ha decidido situar una puerta que permita su fácil acceso. (figura 83)

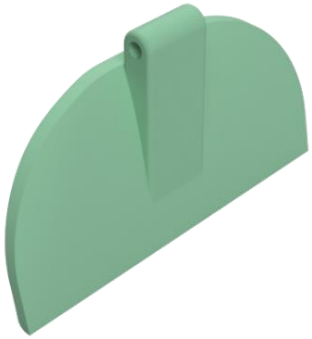


Figura 83. Render puerta

La puerta se abre alcanzando un ángulo de 120º, quedándose apoyada encima del robot. En las figuras 84 y 85 se puede observar la puerta cerrada y abierta respectivamente.

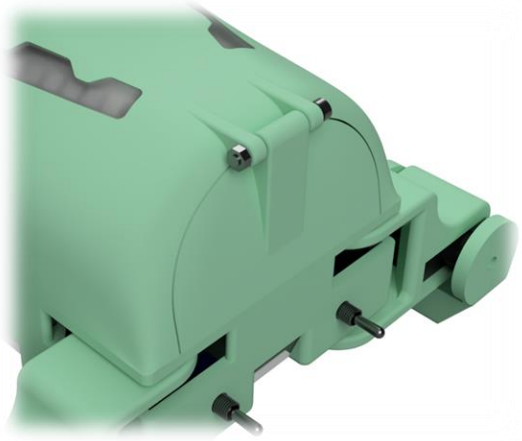


Figura 84. Render puerta abierta

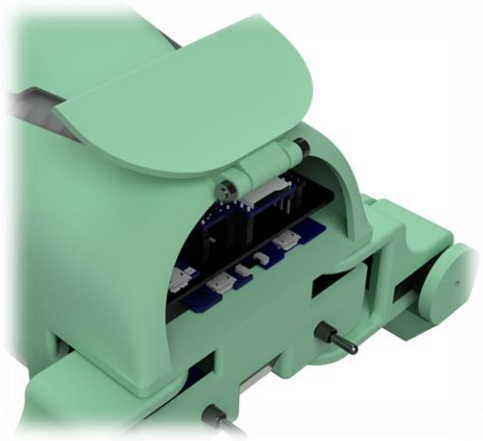


Figura 85. Render puerta cerrada

4.4.7. Soporte del altavoz

Esta pequeña pieza tiene el propósito de sujetar el altavoz. Se han hecho ranuras en su base para que el sonido pueda pasar sin problemas. También se ha hecho una obertura en la parte frontal.

Figura 86. Render soporte altavoz

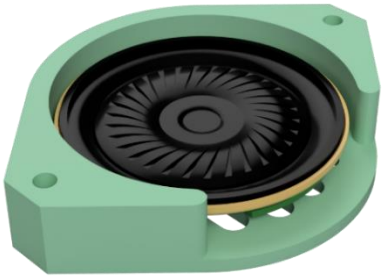


Figura 87. Render soporte con altavoz

4.5. Diseño completo

Una vez montado el robot, este alcanza las siguientes dimensiones erguido y con la pinza recogida: 142 mm de alto, 205.5 mm de ancho y 175 mm de profundo (figura 88).

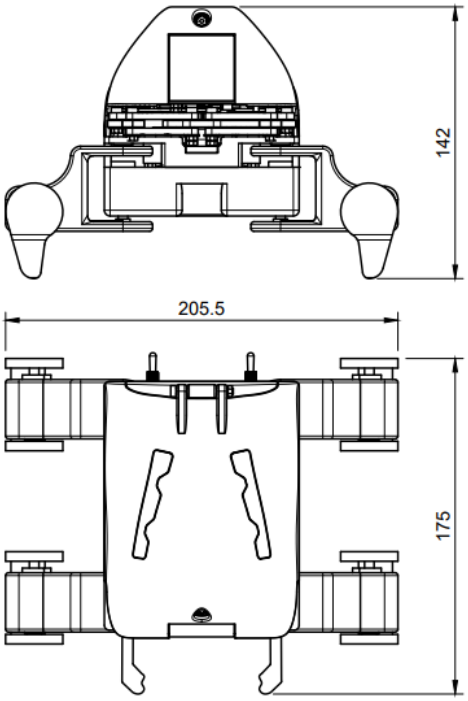


Figura 88. Dimensiones RA-01

En la figura 89, se muestra el robot teniendo las piezas exteriores transparentes. De esta forma, se puede observar las piezas que contiene. Al y como se ha comentado previamente, se ha diseñado el robot para que este sea lo más pequeño posible. Una vez diseñado, se puede afirmar que si se quisiera encoger el robot se debería utilizar una batería más pequeña, diseñar una PCB más compacte y prescindir de la pinza.

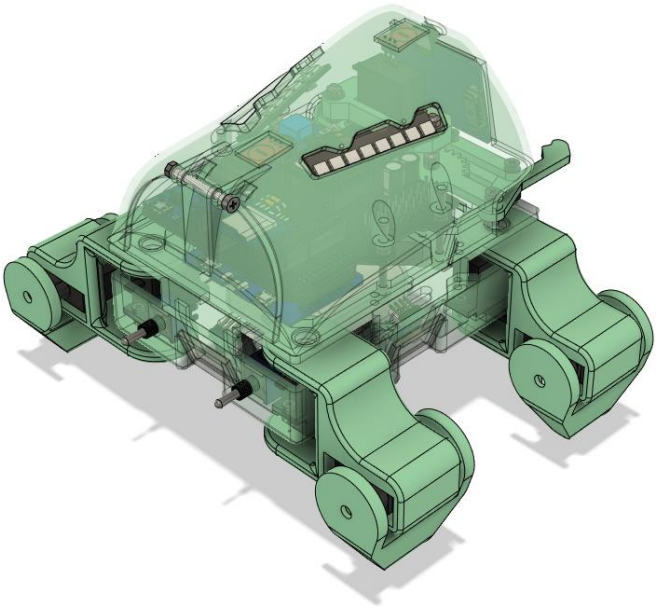


Figura 89. Diseño completo, piezas exteriores transparentes

Finalmente, el render del diseño completo del robot RA-01 se puede observar en las figuras 90 y 91.

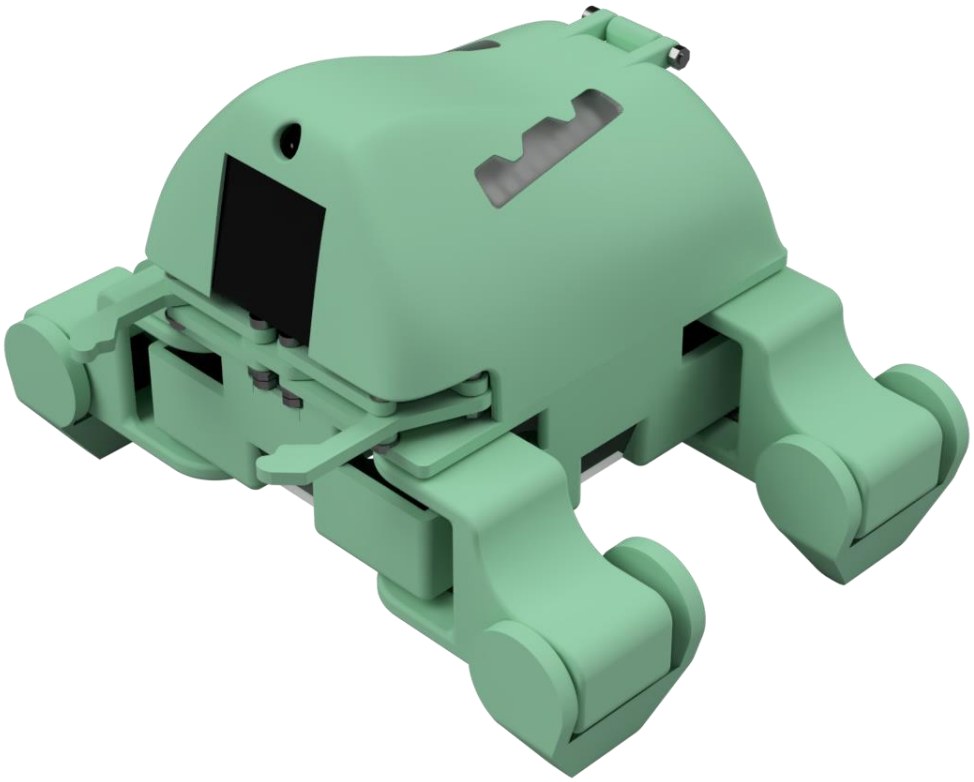


Figura 90. Render diseño completo 1

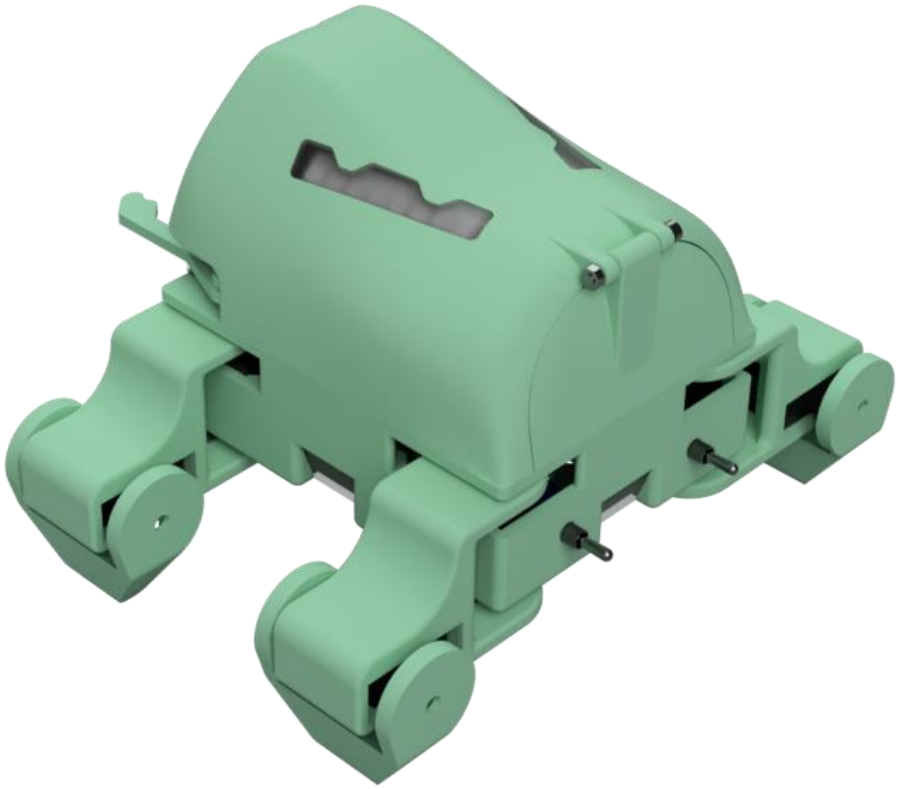


Figura 91. Render diseño completo 2

4.6. Software

Con el fin de comprobar que el diseño realizado funciona correctamente, se ha desarrollado un software que incorpora las funciones básicas del robot y un par de juegos de prueba.

El robot ha sido programado para:

- Generar su propia red WIFi.
- Generar un servidor web local, donde albergar un portal de control.
- Ser controlado a través del portal de control.
- Retransmitir video grabado en tiempo real.
- Mostrar distintas expresiones faciales, que representan emociones básicas.
- Variar su expresión facial a causa de la interacción con el mismo
- Mover los ojos de forma natural y parpadear.
- Mostrar imágenes en pantalla para realizar juegos.
- Caminar, tanto hacia delante y hacia detrás como rotando sobre sí mismo.
- Reaccionar cuando detecta el borde de la mesa o que se le ha levantado.
- Mover la pinza.
- Reproducir audios grabados previamente.
- Iluminarse de distintos colores.
- Detectar, diferenciar y reaccionar, cuando se mantiene la mano apoyada en la parte delantera o en la trasera, o cuando se acaricia el robot de delante a detrás o del revés.

Cabe destacar, que el código desarrollado es únicamente una versión provisional y no se plantea como código definitivo del robot.

4.6.1. Herramientas utilizadas para la programación.

Para programar el código del robot se ha utilizado el IDE de Arduino 2.0. La configuración necesaria para programar microcontroladores ESP32 en este entorno se especifica en el *Anexo 2. Pliego de condiciones*.

Adicionalmente, para realizar el código HTML del portal de control se ha utilizado el entorno de programación online *"CUBIC FACTORY"* [71], figura 92.

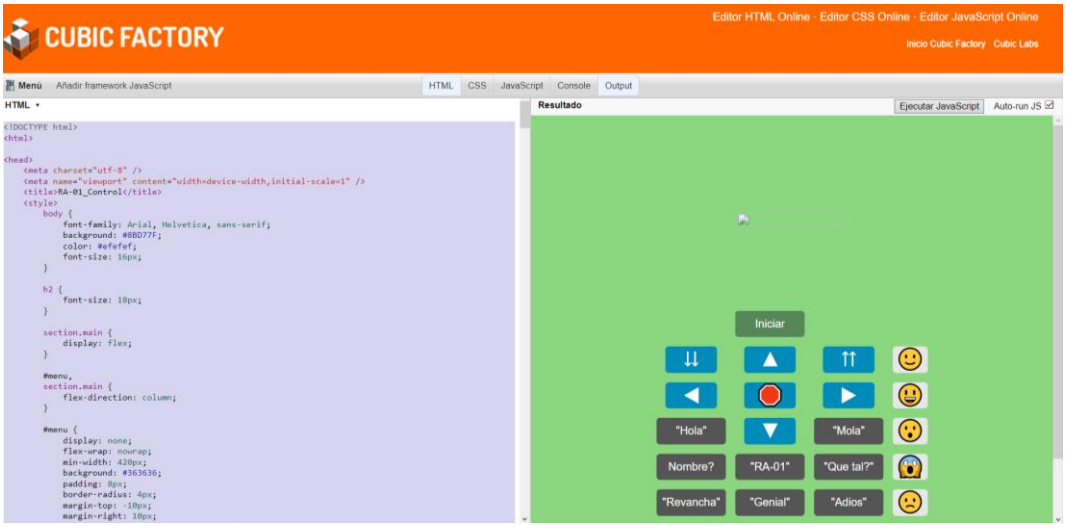


Figura 92. Entorno online de programación HTM utilizado [71]

Una vez realizado el código HTML, para introducirlo en el código del microcontrolador ESP32 cam este se ha "minificado". La "minificación" es un proceso mediante el cual se eliminan los caracteres que entorpecen la lectura del programa por parte de la máquina: los espacios espaciados, saltos de línea, comentarios, etc. Para ello, se ha utilizado la página web *"HTML Minifier"* [72], figura 93.

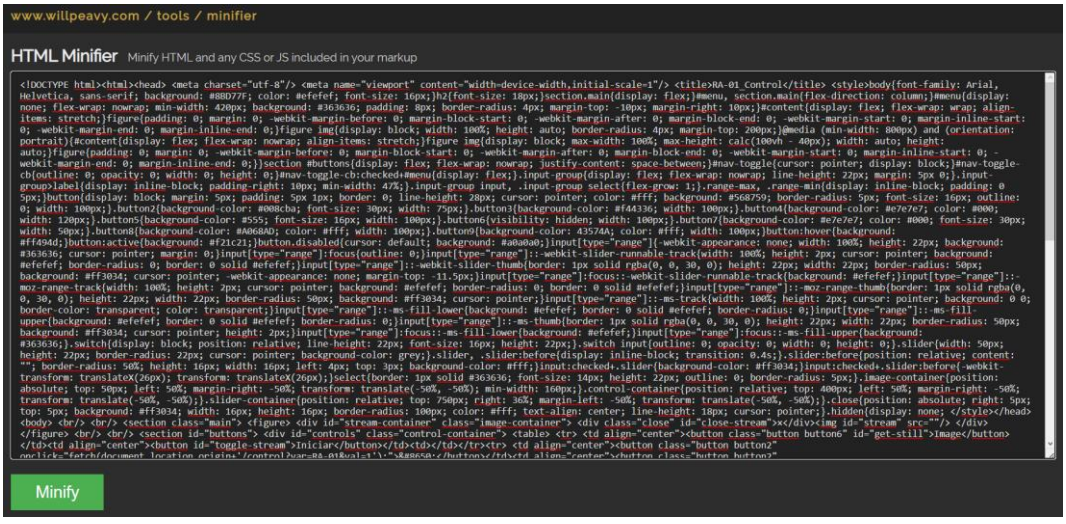


Figura 93. "Minificador" HTML online utilizado [72]

Una vez “minificado” el código este es difícil de interpretar y modificarlo resulta una tarea imposible. Es por ello por lo que cuando se ha deseado modificar el programa se ha utilizado la página web *“unminify 2”* [73], figura , que realiza el proceso inverso.

Figura 94. “Unminificador” online utilizado [73]

Por otro lado, también es importante destacar las herramientas utilizadas para crear las imágenes que incorpora el código.

En primer lugar, para la creación de las imágenes se ha utilizado el programa “GIMP”, figura 95.

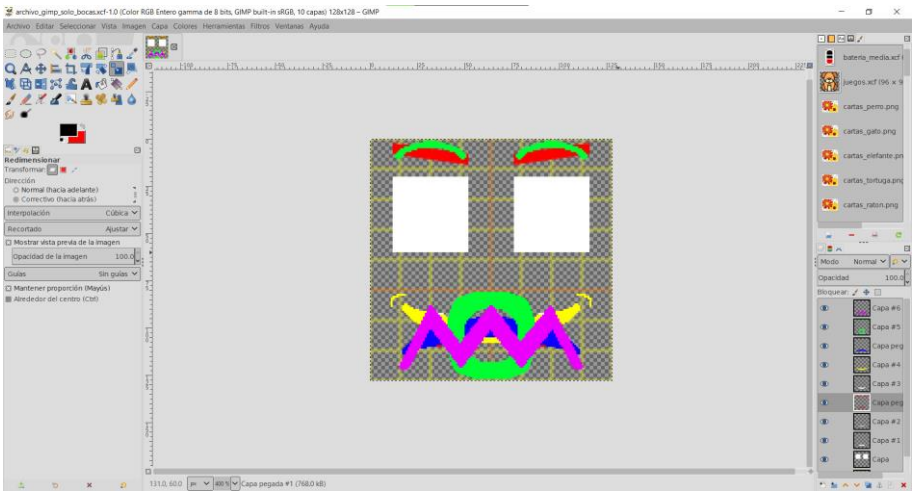


Figura 95. Editor de imagenes "GIMP"

En segundo lugar, dependiendo de si se deseaba una imagen binaria o con distintos colores se han utilizado conversores online que transforman la imagen la imagen en una cadena de datos.

Cuando se ha deseado una imagen binaria se ha utilizado la página web “image2cpp” [74], figura 96. En cambio, cuando la imagen tenia varios colores se ha utilizado la página web “ImageConverter (UTFT)” [75], figura 97.

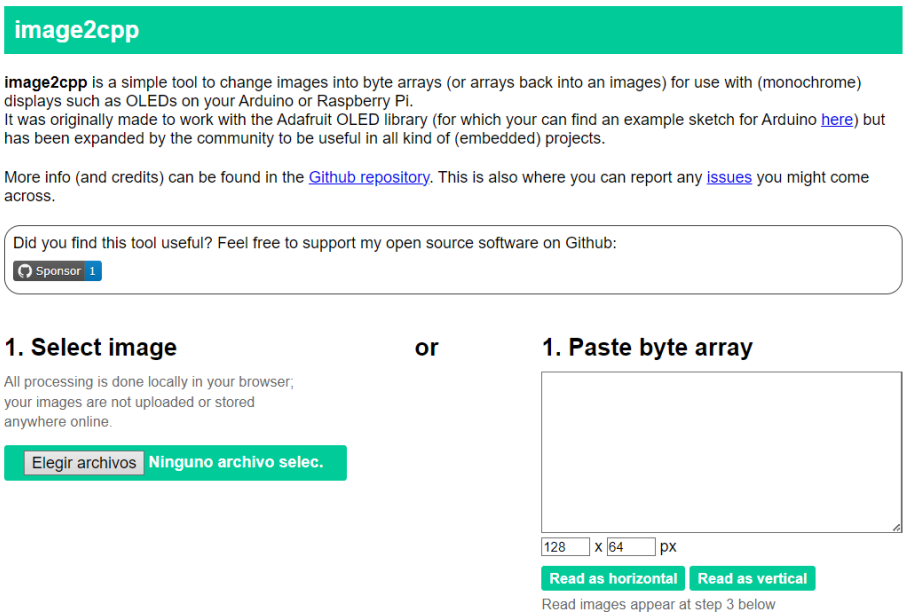


Figura 96. Image2cpp [74]

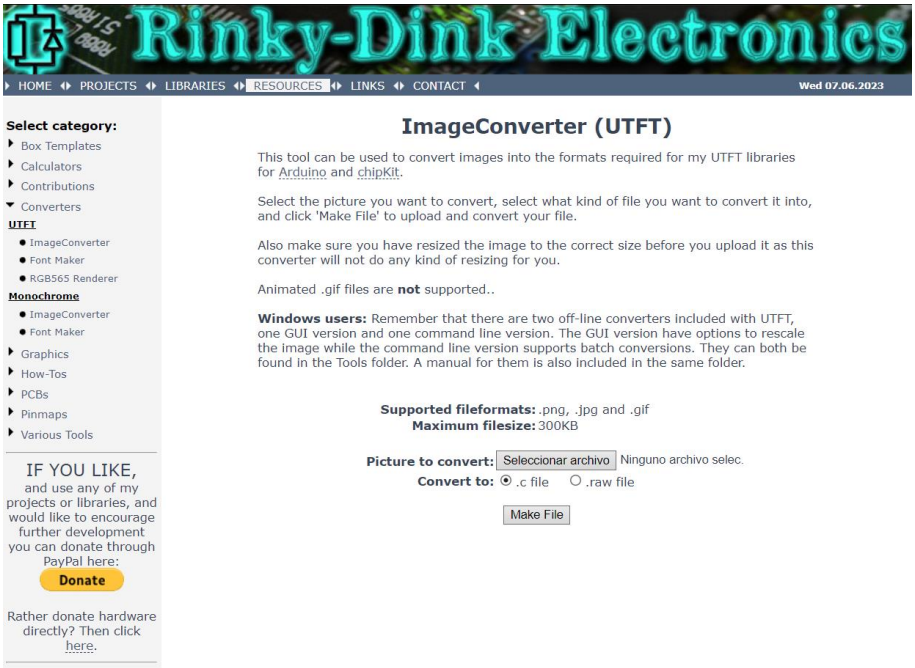


Figura 97. ImageConverter (UTFT) [75]

4.6.2. Diagrama de flujo ESP 32 cam

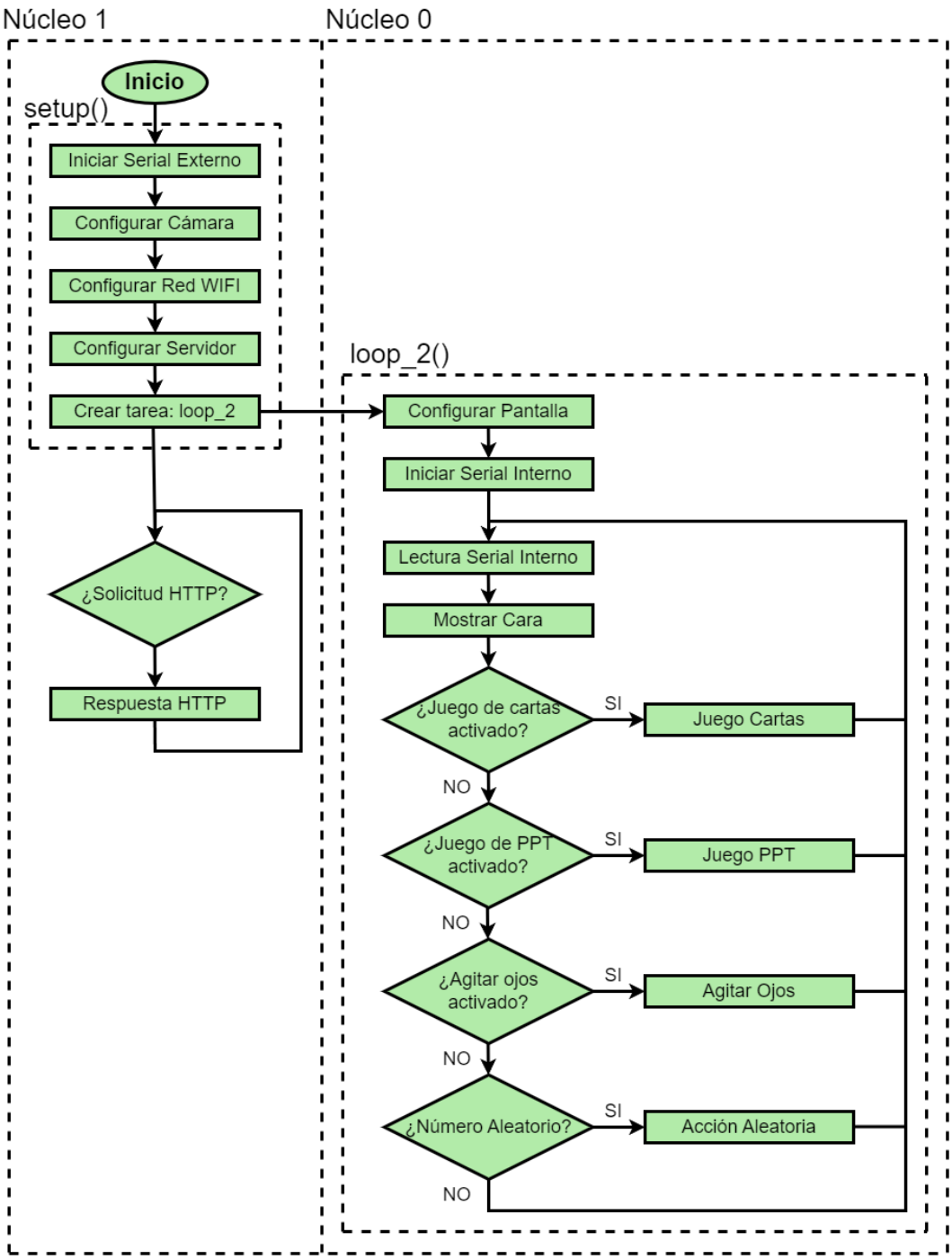


Figura 98. Diagrama de flujo ESP32 cam

4.6.3. Diagrama de flujo ESP 32 d1 mini

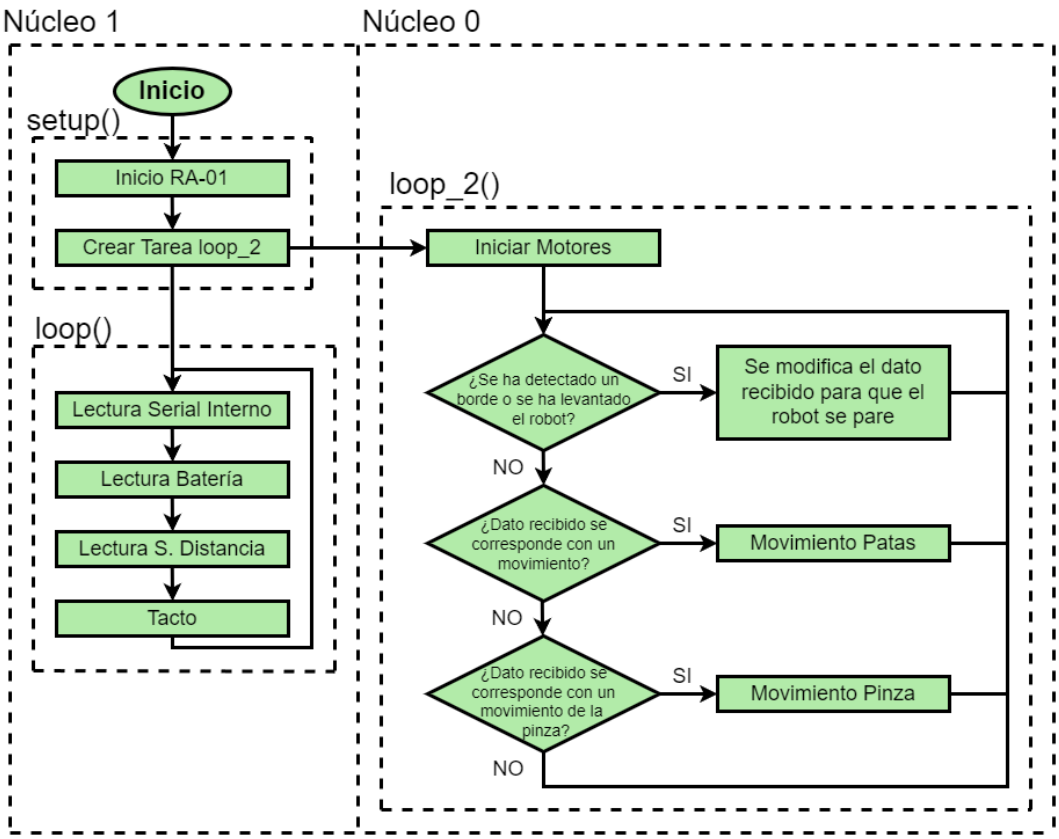


Figura 99. Diagrama de flujo ESP32 d1 mini

4.6.4. Software, funciones básicas

4.6.4.1. Control

Tal y como se ha planteado previamente, el robot crea una red wifi a través de la cual se le puede controlar.

Concretamente, es el microcontrolador ESP 32 cam quien genera su propia red wifi con el siguiente SSID: “RA_01” (El SSID es el nombre de la red wifi que nos aparece cuando en la lista de redes WIFI disponibles). Con su respectiva contraseña: “5555444433”. La cual al igual que SSID se puede modificar a voluntad en el código del dispositivo (líneas 74 y 75 del archivo “ESP_32_cam.ino”).

El microcontrolador, además de generar una red WIFI alberga un portal web de control. Para acceder a este, una vez conectados a la red WIFI, se debe de buscar en el navegador la siguiente dirección IP: “192.168.4.1”. Esta es una dirección IP privada que es utilizada comúnmente como dirección de puerta de enlace predeterminada para una red local.

Es importante destacar, que para la realización del programa correspondiente con el control y la retransmisión se ha partido del código creado por “DroneBot Workshop” [76].

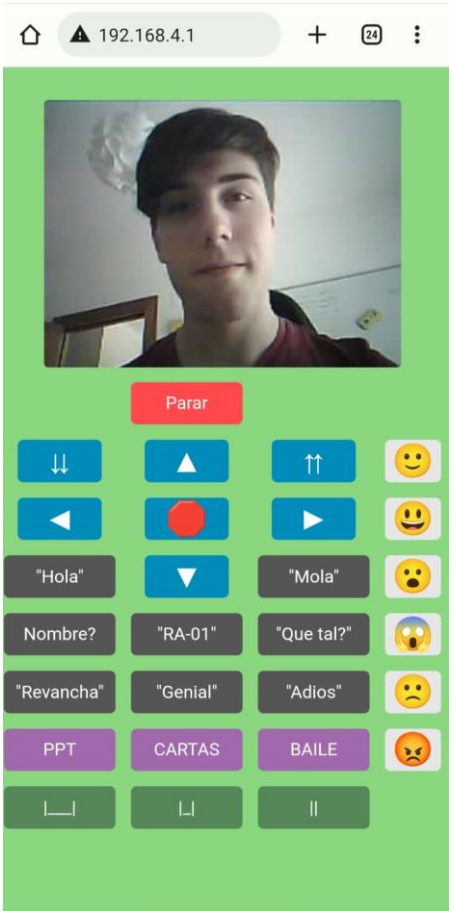


Figura 100. Portal de control

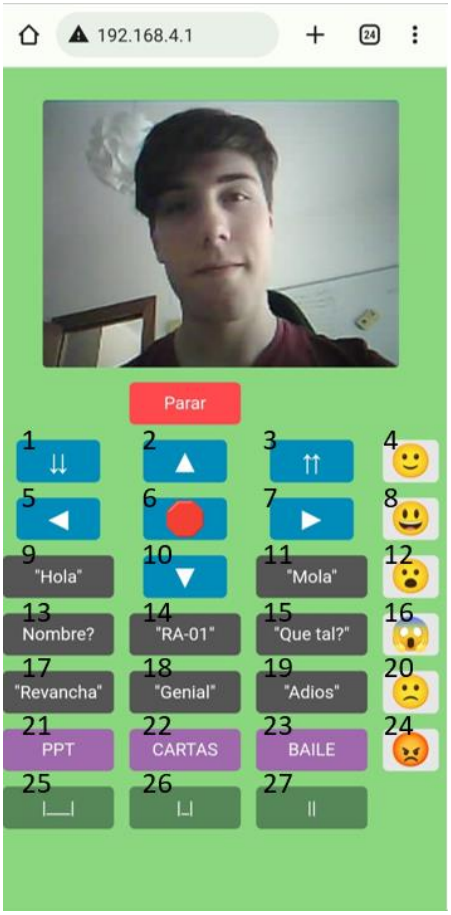


Figura 101. Numeración botones

Una vez el usuario se ha conectado al portal, se puede iniciar la retransmisión y controlar el robot a través de los botones disponibles. Estos botones han sido numerados tal y como se muestra en la figura 101, y en el código cada uno una función asignada (`void Boton_Pulsado_X()`) que se ejecuta cuando el botón se pulsa.

Tal y como se puede observar, para diferenciar la función de los distintos botones estos tienen distintos colores. Siendo los botones azules para controlar el movimiento de la pinza, los grises para reproducir audios, los morados para realizar actividades especiales como juegos o bailes, los verdes para el movimiento de la pinza y los blancos para las emociones del robot.

A continuación, se indica la función de cada botón:

N.º Botón	Función	N.º Botón	Función
1	Movimiento: tumbar	15	Reproducir audio: "¿Qué tal el día"?
2	Movimiento: caminar hacia delante	16	Emoción, asustado
3	Movimiento: erguido	17	Reproducir audio: "¿Jugamos otra vez?"
4	Emoción: normal	18	Reproducir audio: "Genial"
5	Movimiento: rotar sentido antihorario	19	Reproducir audio: "Adios"
6	Movimiento: parar	20	Emoción: triste
7	Movimiento: rotar sentido horario	21	Iniciar y finalizar juego piedra papel o tijera
8	Emoción: feliz	22	Iniciar y finalizar juego cartas
9	Reproducir audio: "Hola"	23	Iniciar baile
10	Movimiento: caminar hacia detrás	24	Emoción: enfadado
11	Reproducir audio: "Mola"	25	Posición pinza: abierta
12	Emoción: sorprendido	26	Posición pinza: a medio cerrar
13	Reproducir audio: "¿Como te llamas?"	27	Posición pinza: cerrada
14	Reproducir audio: "Yo me llamo RA-01"		

Tabla 1. Funciones botones

Dado que el robot es controlado por dos microcontroladores, para su funcionamiento estos se deben de comunicar entre sí. Para ello, se conectan por comunicación UART.

De forma cíclica y frecuente ambos microcontroladores comprueban si han recibido un mensaje del otro, comunicándose a través de números enteros.

Siendo el ESP 32 cam el microcontrolador 1 y el ESP 32 d1 mini el 2, en la tabla 2 se indica lo que significa cada número cuando se lo envía el uno al otro.

Número	Microcontrolador 1 a 2	Microcontrolador 2 a 1
1	Movimiento: tumbar	Mostrar icono: batería cargando
2	Movimiento: erguido	Mostrar icono: batería baja
3	Movimiento: caminar hacia delante	Emoción: normal, agitar ojos arriba
4	Movimiento: caminar hacia detrás	Emoción: enfadado, centrar ojos
5	Movimiento: rotar sentido antihorario	Emoción: feliz, agitar ojos arriba
6	Movimiento: rotar sentido horario	Emoción: sorprendido, centrar ojos
7	Movimiento: parar	Emoción: asustado, bajar ojos
8	Posición pinza: abierta	
9	Posición pinza: a medio cerrar	
10	Posición pinza: cerrada	
11	Reproducir audio: "Hola"	
12	Reproducir audio: "Mola"	
13	Reproducir audio: "¿Como te llamas?"	
14	Reproducir audio: "Yo me llamo RA-01"	
15	Reproducir audio: "¿Qué tal el día"?	
16	Reproducir audio: "¿Jugamos otra vez?"	
17	Reproducir audio: "Genial"	
18	Reproducir audio: "Adios"	
19	Reproducir audio: "Piedra, papel o tijera"	
20	Iniciar baile	

Tabla 2. Mensajes comunicación entre microcontroladores

4.6.4.2. Movimiento de las patas

Para el movimiento de cada pata se utilizan dos servomotores, el primero controla la rotación y el segundo la elevación vertical.

En primer lugar, se han buscado los ángulos reales que se le deben de asignar a los distintos servomotores para que estos alcancen las posiciones deseadas. En la figura 102, se puede observar los ángulos correspondientes con los motores que controlan la rotación de las patas.

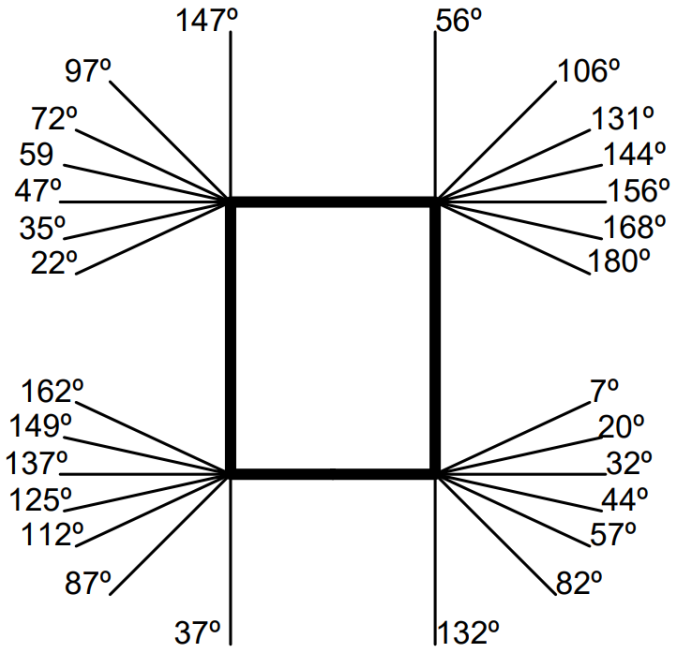


Figura 102. Ángulos rotación patas

Los ángulos de la figura 98 se corresponden con las siguientes diferencias angulares (figura 103).

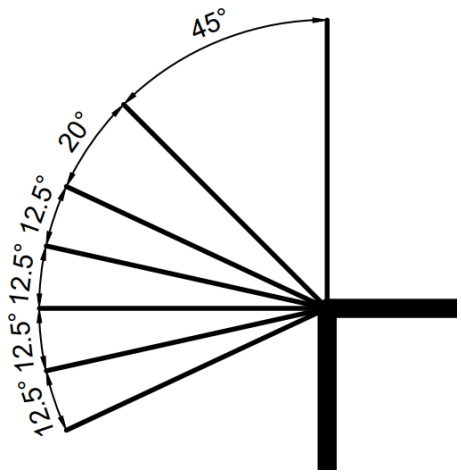


Figura 103. Diferencias angulares rotación partas

Posteriormente, se han determinado sobre el papel todas las secuencias de posiciones que se deben adoptar para hacer que el robot se levante, avance (hacia delate y hacia detrás) y rote sobre sí mismo (sentido horario y antihorario). Así mismo, también se han determinado las secuencias necesarias para alcanzar las posiciones iniciales necesarias desde las distintas posiciones que el robot alcanza.

En el código todas estas posiciones se almacenarán en cadenas 8 cadenas (una por articulación). Para una mejor organización, se le ha asignado a cada conjunto de ángulos un valor numérico (de 0 a 38), que será utilizado en la programación de las secuencias. En la figura 104, podemos ver la secuencia de movimientos completa utilizando pictogramas. Más adelante, se detallan las diferentes secuencias.

Cabe destacar que, en los pictogramas, la posición del segundo eslabón se indica por medio de una flecha, significando la flecha hacia abajo que la pata debe de estar estirada hacia abajo y la flecha señalando hacia arriba que la pata debe de estar levantada.

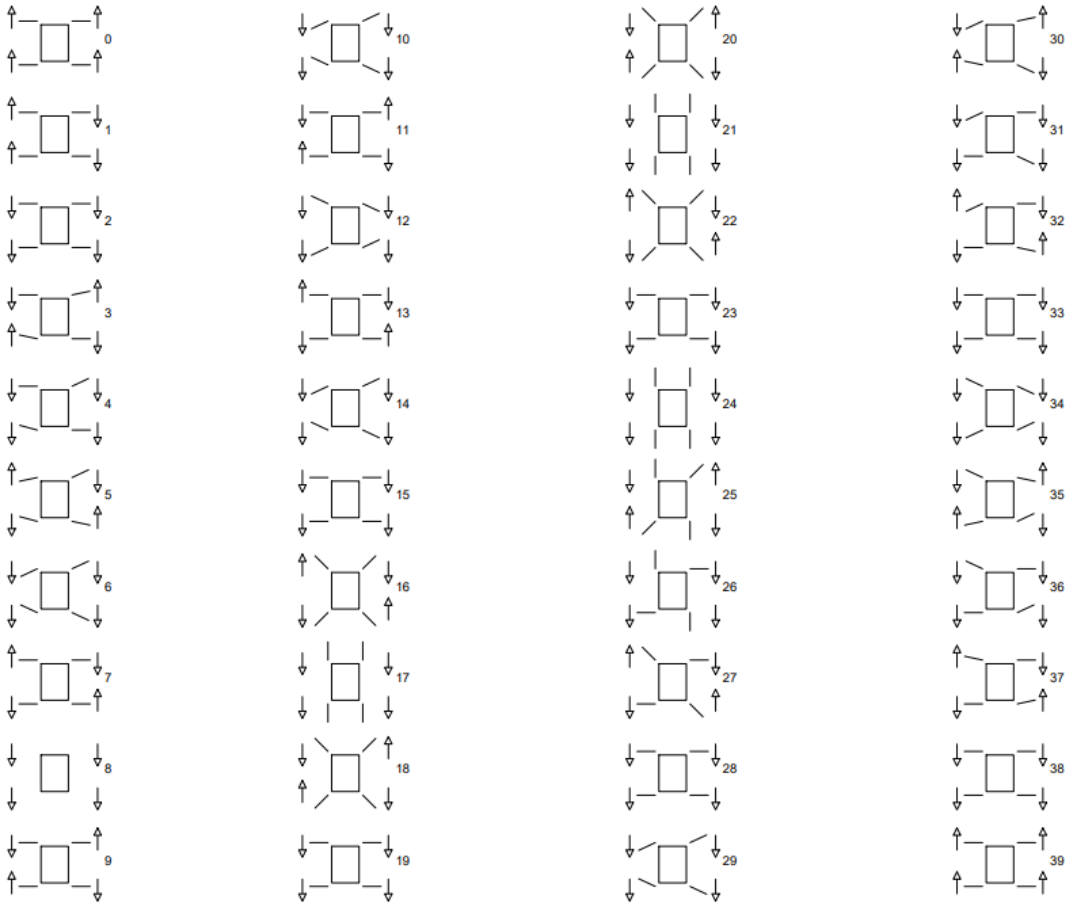


Figura 104. Secuencia completa de movimiento

Respecto al código de control del movimiento, se han establecido los siguientes estados de movimiento:

- Tumbado (0)
- Posición erguida (1)
- Caminar hacia delante (2)
- Caminar hacia detrás (3)
- Rotar antihorario (4)
- Rotar horario (5)
- Parar (6)

Durante el movimiento del robot, existen varias posiciones que se repiten en múltiples ocasiones donde se apoyan en el suelo las 4 patas:

- Tumbado (0)
- Levantado centrado (1)
- Primer paso (2)
- Segundo paso (3)
- Levantado estirado (4)

Las posiciones se corresponden con los siguientes pictogramas (figura 105):

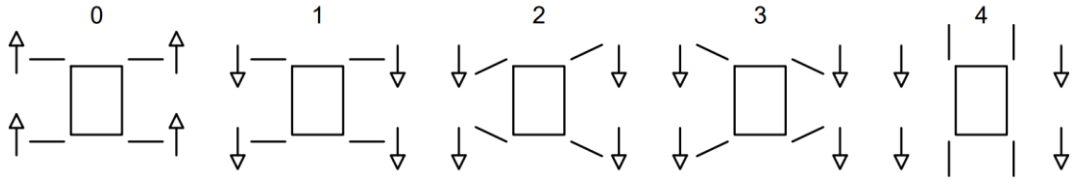


Figura 105. Posiciones críticas

Dado que, estas posiciones se repiten a lo largo de las distintas animaciones, se ha decidido utilizar estas como punto de partida de un estado de movimiento a otro. Habiéndose realizado secuencias de movimientos que sirven a modo de transición entre los distintos estados de movimiento.

Es decir, dependiendo del estado de funcionamiento y de la posición de sus articulaciones, el robot realizará una secuencia de movimientos u otra, siguiendo el siguiente diagrama de flujo (figura 106):

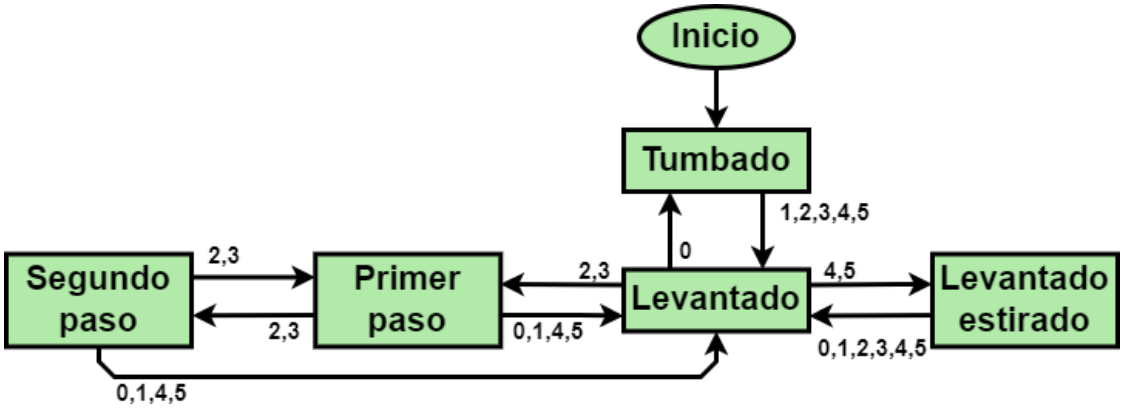


Figura 106. Diagrama de bloques, variación de la posición del robot

Cabe destacar que, aunque en el diagrama de bloques la variación entre posiciones puede ser la misma para varios estados de movimiento, la secuencia de movimiento no es siempre la misma.

Por ejemplo, si el estado de movimiento es caminar hacia delante o caminar hacia detrás, las posiciones alcanzadas entre los pasos son las mismas, pero sus pasos son distintos, pues la secuencia de movimientos es distinta.

Tal y como se ha comentado previamente, todas las colocaciones que el robot debe alcanzar para realizar sus movimientos se han situado, en 8 listas de 40 ángulos (0-39), una para cada motor. A continuación (tabla 3), se detalla la correspondencia entre secuencia, movimiento realizado, posición inicial y posición final. Siendo estas dos últimas las indicadas previamente en la figura 105.

Secuencia	Tipo de movimiento	Posición inicial	Posición final
0 → 2	Levantar	0	1
2 → 6	Transición	1	2
6 → 8	Caminar hacia delante	2	3
8 → 10	Caminar hacia delante	3	2
10 → 12	Caminar hacia detrás	2	3
12 → 14	Caminar hacia detrás	3	2
15 → 17	Rotar sentido horario	1	4
17 → 19	Rotar sentido horario	4	1
19 → 21	Rotar sentido antihorario	1	4
21 → 23	Rotar sentido antihorario	4	1
24 → 28	Transición	4	1
29 → 33	Transición	2	1
34 → 38	Transición	3	1
38 → 39	Tumbar	1	0

Tabla 3. Secuencias de movimiento.

Como se mueven los motores para lograr un movimiento suave y simultaneo de las articulaciones del robot se detalla en el apartado 4.6.3.4 *Control de movimiento*.

4.6.4.3. Movimiento de la pinza

Siendo controlada por únicamente un motor, en comparación con el movimiento de las patas el control de la pinza resulta más sencillo.

Se ha asignado una posición inicial (pinza abierta) y en el portal de control hay 3 botones que sirven para abrir la pinza, dejarla medio cerrada o cerrarla completamente. Teniendo 110° de movimiento

Si la pinza se encuentra ya en la posición deseada no se mueve.

Al igual que con el movimiento de las patas el control del movimiento se detalla en el apartado *4.6.3.4 Control de movimiento*.

4.6.4.4. Control del movimiento

En un principio, se pretendía que los servomotores rotaran a velocidad constante. No obstante, al comprobar el funcionamiento del robot se experimentaron 2 problemas.

En primer lugar, de forma relativamente frecuente, la corriente pico, generada por el inicio del movimiento de varios motores simultáneamente, causaba que el circuito de protección de la batería hiciera que esta se desconectara, apagando el robot.

En segundo lugar, a lo largo de la mayoría de los movimientos, las patas se agitaban de forma descontrolada cuando estas estaban en el aire.

Para mejorar el movimiento del robot, minimizando estos problemas, se ha decidido implementar un control eje a eje con trayectoria polinomial en el proyecto.

Utilizando este control, las articulaciones, en vez de moverse a velocidad constante a lo largo de todo el recorrido, inician este con una velocidad nula, que augmenta hasta alcanzar una velocidad máxima y antes de llegar al final se reduce para llegar a la posición deseada deteniéndose de forma suave.

Este método de control se ha estudiado en la asignatura de *Sistemas robotizados* [77] y se puso en práctica en el proyecto de esta, donde se construyó y se programó un pequeño brazo robótico. Es por ello, que las fórmulas utilizadas, son las de dicho proyecto, de cuyo enunciado también se ha obtenido la figura 107. Esta imagen muestra cómo evolucionan las posiciones angulares a lo largo del tiempo, teniendo una posición inicial y otra final en un tiempo dado.

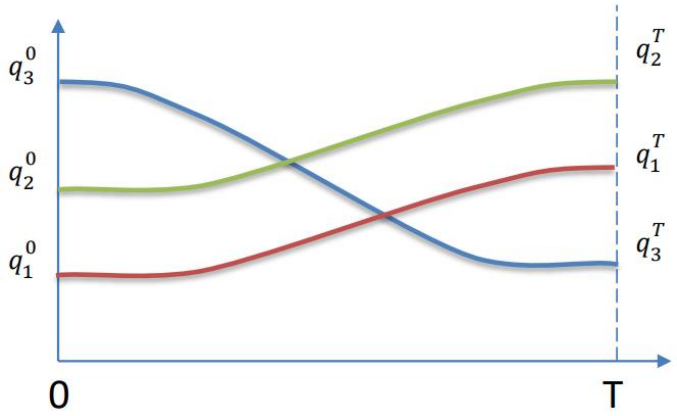


Figura 107. Representación gráfica posiciones angulares [77]

A continuación, se describen las fórmulas matemáticas empleadas:

$$q_i(t) = a_i \cdot t^3 + b_i \cdot t^2 + c_i \cdot t + d_i$$

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

Siendo T el tiempo designado para el movimiento, q_i(t) la posición angular en un instante concreto, ${q_i}^0$ la posición angular inicial, ${q_i}^T$ la posición angular final y a_i, b_i, c_i y d_i condiciones de contorno para obtener una trayectoria cúbica.

4.6.4.5. Expresiones faciales

Para representar las diferentes emociones que incorpora el robot se han creado las siguientes caras:

Figura 108. Expresión normal

Figura 109. Expresión feliz

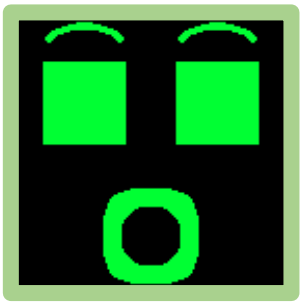


Figura 110. Expresión sorprendido

Figura 111. Expresión asustado

Figura 112. Expresión triste

Figura 113. Expresión enfadado

Al encender el robot, este se encuentra muestra la expresión normal. Cuando se pulsan los botones correspondientes del panel de control o en respuesta a ciertos estímulos la emoción del robot cambiará y con ello la cara que muestra en pantalla.

Respecto a la programación, para mostrar la cara del robot en pantalla se ha creado la función *"void Mostrar_Cara()"*. En esta función, dependiendo de la emoción del robot se establece el color y se muestra la boca y las cejas si la emoción las incluye. La posición de la boca es fija. A diferencia de, la posición de las cejas que depende de la de los ojos y se borran y dibujan cada vez que se mueven los ojos,desafortunadamente de forma apreciable a simple vista.

Para el movimiento de los ojos la función *"void Mostrar_Cara()"* llama a la función *"void Mover_Ojos()"*.

Los ojos son 2 cuadrados que se posicionan en función de las coordenadas del centro del ojo izquierdo respecto la esquina superior izquierda (figura 114). Para su movimiento se han creado 4 variables globales: 2 para la posición actual en x e y, y otras 2 para las posiciones deseadas.

La función *"void Mover_Ojos()"* dispone de dos modos de funcionamiento.

En el primero, se determina la dirección del movimiento de los ojos comparando la posición actual y la deseada. Acto seguido, se dibuja una fila, columna o ambas (dependiendo de si el movimiento es horizontal, vertical o diagonal), del color de los ojos en la dirección del movimiento, y se dibuja otra de color negro en el extremo opuesto de los cuadrados. Con múltiples iteraciones, los ojos alcanzan su posición deseada y se detienen.

En el segundo modo de funcionamiento, ambos ojos desaparecen dibujando un cuadrado negro sobre ellos y vuelven a aparecer en la posición deseada. Este método es útil si se desea mover los ojos de forma rápida y da la sensación de que el robot ha parpadeado a la vez que ha cambiado su mirada.

Respecto al movimiento de los ojos, la posición deseada se modifica reaccionando a ciertos estímulos, con el movimiento del robot y por causa de eventos aleatorios (detallados en el apartado 4.6.3.6. Eventos aleatorios).

En la siguiente figura 114 se puede observar la posición de los ojos en la pantalla. Siendo la zona inferior de 48x128 pixeles la dedicada a la boca y el rectángulo ubicado encima de los ojos donde se ubican las cejas.

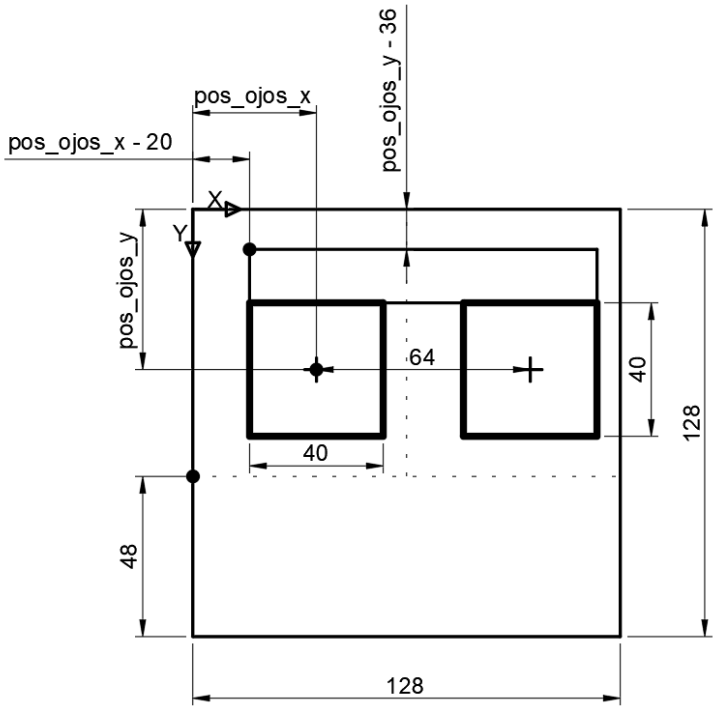


Figura 114. Disposición elementos en pantalla

A la hora de programar la disposición de los distintos elementos en pantalla, se debe indicar la posición de la esquina superior izquierda del cuadrado línea o imagen a dibujar izquierda respecto el origen de coordenadas ubicado en la esquina superior izquierda de la pantalla.

Teniendo en cuenta esto, se definen las posiciones con las siguientes formulas:

- Posición en x rectángulo ojo izquierdo = pos_ojos - 20
- Posición en y rectángulo ojo derecho = pos_ojos - 20
- Posición en x rectángulo ojo izquierdo = pos_ojos + 44
- Posición en y rectángulo ojo derecho = pos_ojos – 20
- Posición en x imagen cejas = pos_ojos – 20
- Posición en y imagen cejas = pos_ojos - 36
- Posición en x imagen boca = 0
- Posición en y imagen boca = 80

Finalmente, cabe destacar que los ojos han sido programados de forma que cuando miran hacia arriba, abajo, derecha e izquierda se aplastan contra el borde, dando mayor sensación de movimiento. A continuación, se muestran varias imágenes de la expresión normal con los ojos en distintas posiciones.

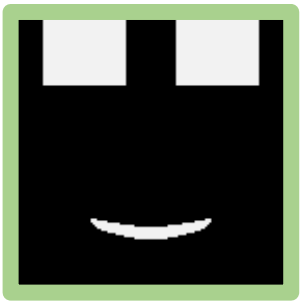


Figura 115. Mirada 1

Figura 116. Mirada 2

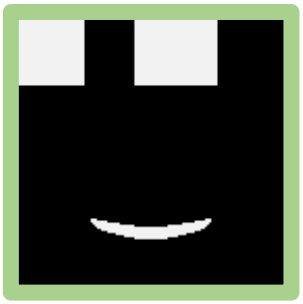


Figura 117. Mirada 3

Figura 118. Mirada 4

Figura 119. Mirada 4

Figura 120. Mirada 4

4.6.4.6. Eventos aleatorios

Con el objetivo de aportarle mayor realismo al robot se ha decidido que de forma aleatoria parpadee y mueva sus ojos.

Para lograrlo, en cada iteración del bucle de la función *"void loop_2()",* del ESP 32 cam, se genera un número aleatorio, que cuando siendo dividido por 500 su resto es 0 ejecuta la función *"Accion_Aleatoria()".*

Esta función genera otro número aleatorio y lo divide entre 40, y dependiendo del resto de esta división se mueven los ojos, o se parpadea. Teniendo un 55% de posibilidades de parpadear, un 25% de posibilidades de mirar al centro y un 20% de mirar a su alrededor.

En la siguiente tabla se puede ver la acción para cada número resultante.

Número	Posibilidad	Acción
0	2.5%	Posición deseada ojos → abajo izquierda
1	2.5%	Posición deseada ojos → izquierda
2	2.5%	Posición deseada ojos → arriba izquierda
3	2.5%	Posición deseada ojos → abajo
4-13	25%	Posición deseada ojos → centro
14	2.5%	Posición deseada ojos → arriba
15	2.5%	Posición deseada ojos → abajo derecha
16	2.5%	Posición deseada ojos → derecha
17	2.5%	Posición deseada ojos → arriba derecha
18-39	55%	Parpadear

Tabla 4. Posibilidades de las distintas acciones aleatorias

4.6.4.7. Reproducción de audios

Tanto cuando se enciende como cuando se le indica a través del portal de control el robot reproduce palabras o frases grabadas previamente.

Estos audios se encuentran en la tarjeta SD accesible a través de la puerta trasera del robot y pueden ser modificados a voluntad.

En un principio el robot incorpora las siguientes frases (tabla 5):

Número asignado:	Frase:
1	"Hola"
2	"Mola"
3	"¿Cómo te llamas?"
4	"Yo me llamo RA-01"
5	"¿Qué tal el día"?
6	"¿Jugamos otra vez?"
7	"Genial"
8	"Adios"
9	"Piedra papel o tijera"

Tabla 5. Asignación numérica de las frases grabadas

Para grabar los audios, tras probar con varias aplicaciones de modificación de voz con resultados decepcionantes, se utilizó una aplicación de la play store (Voice Editor, figura 121) cuyo efecto "Baby Robot" encajaba con el tono de voz deseado.

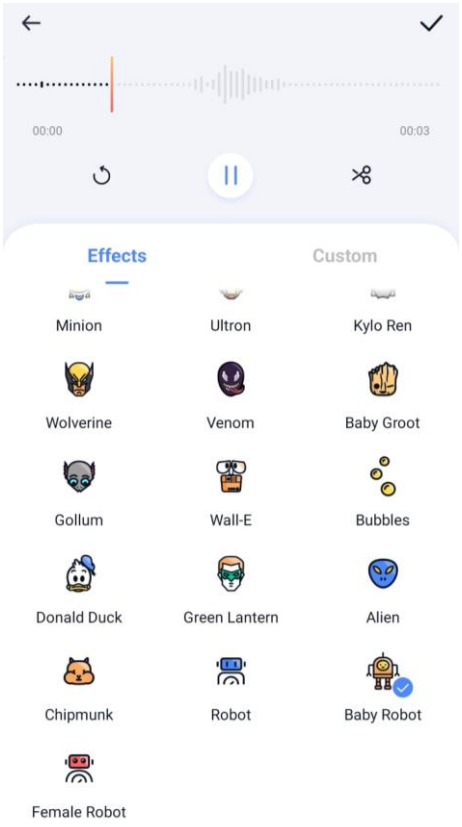


Figura 121. Programa Voice Editor

Una vez grabados los audios, utilizado el programa “Audacity” y se recortó su duración y se amplificaron para que se escucharan con más caridad.

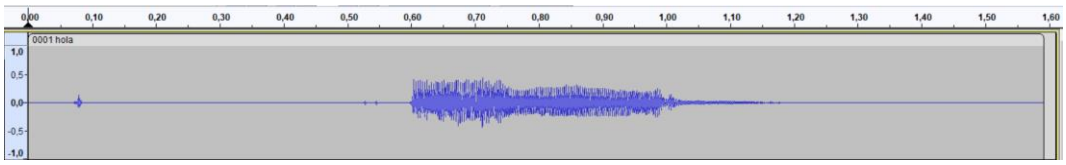


Figura 122. Sonido antes de ser editado

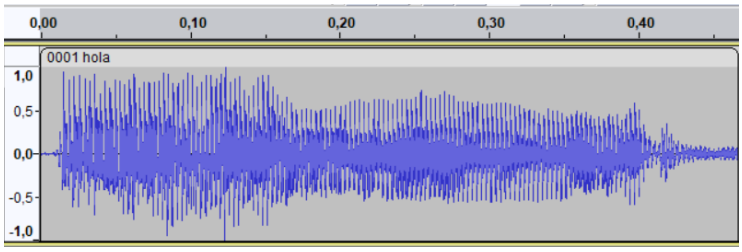


Figura 123. Sonido editado

Finalmente, los archivos de sonido se numeraron y se grabaron en la tarjeta SD.

4.6.4.8. Iluminación

Para controlar la iluminación se han programado dos funciones:

- void Luces_Onda(uint8_t R, uint8_t G, uint8_t B)
- void Luces_Color_fijo(uint8_t R, uint8_t G, uint8_t B)

La primera función, tal y como su nombre indica enciende las luces y las apaga en forma de onda, dando la impresión de movimiento.

Por otro lado, la segunda las enciende todas a la vez y las mantiene encendidas.

Los parámetros de entrada son los valores RGB del color deseado.

4.6.4.9. Respuesta sensorial

El robot incorpora dos tipos de sensores:

- Sensores de distancia laser
- Sensores de proximidad capacitivos

Sensores de distancia laser incluye 4 y se ubican en la pieza inferior del robot mirando hacia delante atrás, derecha e izquierda con una inclinación de 10 grados hacia abajo, figura 124.

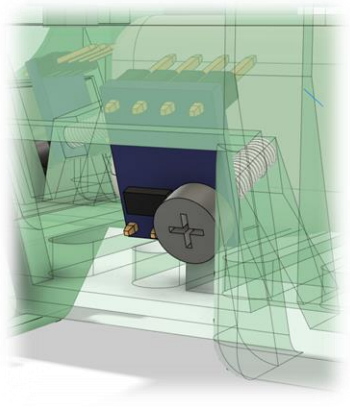


Figura 124. Sensor de distancia colocado

Se han ubicado de esta forma para detectar el borde de la mesa y evitar que el robot se caiga de la misma. Para esta función, solo se requieren el sensor frontal y el trasero. A pesar de ello, el robot incorpora los 4 porque en un principio se planteó que el robot caminara también de lado.

Adicionalmente, estos sensores también sirven para saber si el robot ha sido levantado.

En el código del ESP32 d1 mini, estos sensores funcionan de forma continua (función *"void Lectura_Sensores_Distancia"). C*uando uno de ellos detecta una distancia superior a la esperada el robot se detiene, la expresión de la cara cambia a asustado y se iluminan las luces de color morado.

Respecto los sensores de proximidad capacitivos, de forma periódica el ESP 32 d1 mini ejecuta la función "*void tacto*". En esta función se realizan las operaciones lógicas necesarias para diferenciar los siguientes contactos con el robot:

- Se ha mantenida la mano apoyada sobre la parte frontal del robot.
- Se ha mantenida la mano apoyada sobre la parte trasera del robot
- Se ha acariciado el robot de delante hacia detrás
- Se ha acariciado el robot de detrás hacia delante

En la siguiente tabla se muestra como reacción el robot a los diferentes estímulos.

Estimulo	Respuesta
Se acerca a un borde o se levanta.	Emoción asustado, se detiene.
Se ha mantenida la mano sobre la parte frontal.	Emoción: normal, levanta la mirada y agita los ojos.
Se ha mantenida la mano sobre la parte trasera.	Emoción: enfadado y centra los ojos.
Se ha acariciado de delante hacia detrás.	Emoción: feliz, levanta la mirada y agita los ojos.
Se ha acariciado de detrás hacia delante.	Emoción: sorprendido y centra los ojos.

Tabla 6. Respuesta a los estímulos del robot

4.6.5. Software, juegos

Además de las funciones básicas, en el código desarrollado se han incorporado dos juegos. No se debe pasar por alto que se desconoce si dichos juegos tienen utilidad terapéutica. El propósito de estos es demostrar las posibilidades del dispositivo no proponerlos como terapia.

4.6.5.1. Piedra, papel o tijera

Cuando se pulsa el botón correspondiente en el panel de control se inicia el juego de piedra papel o tijera, y no se detiene hasta que se vuelve a presionar el botón.

En este juego el robot reproduce un audio pregrabado de “piedra papel o tijera” mientras muestra en pantalla la figura 125, para luego mostrar de forma aleatoria una imagen de una piedra, un papel o unas tijeras. Figuras 126, 127 y 129 respectivamente.

Figura 125. Inicio PPT

Figura 126. Piedra

Figura 127. Papel

Figura 128. Tijeras

Queda como tarea para quien controle el robot como esté reaccione a el resultado de la partida y guiar la misma. Pudiendo decir cosas como: “venga, al mejor de 3” y reproduciendo el audio de “genial” o de “revancha” dependiendo de quien gane o pierda, al igual que cambiando la emoción del robot dependiendo de cómo reaccione el niño.

4.6.5.2. Buscar la pareja

Este juego se basa en el típico juego para niños, donde se distribuyen en la mesa múltiples cartas con distintas ilustraciones repetidas dos veces y se deben de buscar las cartas idénticas solo pudiendo darles la vuelta a dos cartas a la vez.

A diferencia del juego original, para hacer partícipe al robot este es el que muestra que dibujo hay que buscar. Se propone, que solo pudiendo darle la vuelta a una carta simultáneamente, se trate de buscar la carta que muestra en pantalla, teniendo un tiempo limitado para encontrarla antes de que el robot muestre otra imagen distinta, necesitando recordar donde estaban las cartas para encontrarlas.

Para realizar el juego se han elegido las siguientes imágenes de animales:

Figura 129. Elefante

Figura 130. Gato

Figura 131 León

Figura 132. Mono

Figura 133. Pájaro

Figura 134. Perro

Figura 135. Pez

Figura 136. Ratón

Figura 137. Tigre

Figura 138. Tortuga

5. Implementación práctica

El robot ha sido montado y puesto en funcionamiento, comprobando el correcto funcionamiento del código y que todas las piezas encajen correctamente. Así como, las fortalezas y flaquezas del diseño.

Las piezas se imprimieron en el servicio de impresión de la universidad.

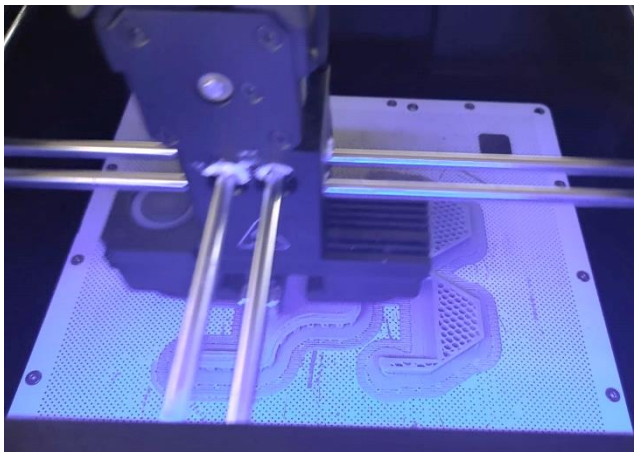


Figura 139. Piezas siendo impreas en impresora FDM

Una vez impresas las pizas (figura 140) se lijaron y pintaron dándole un color verde menta y un acabado suave al robot (figura 141).

Figura 140. Piezas impresas sin pintar

Figura 141. Piezas impresas pintadas

Por otro lado, para la fabricación de la PCB, se ha recurrido a el uso de empresas especializadas. Habiéndose y fabricado dos placas con problemas en el diseño antes de obtener la PCB definitiva.

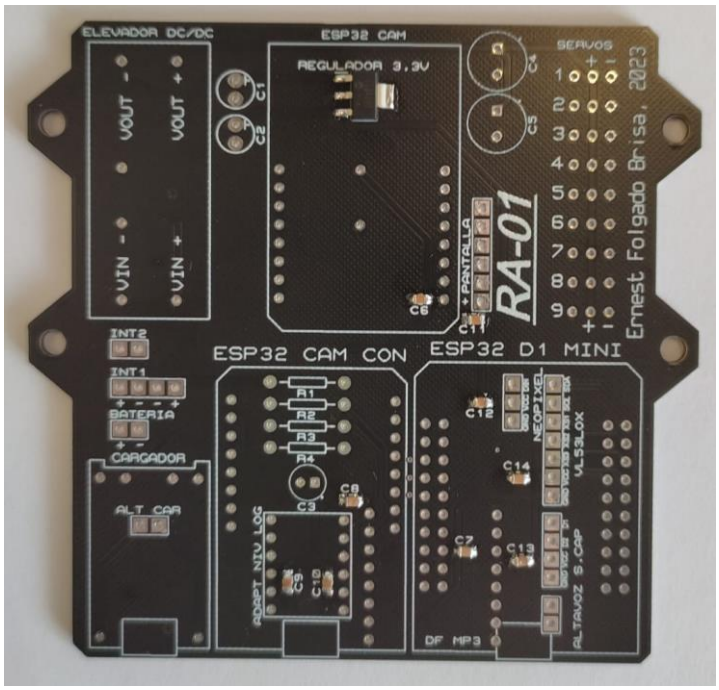


Figura 142. PCB

Una vez recibidas las PCBs, los distintos componentes integrados en ella han sido soldados manualmente, utilizando un soldador convencional.

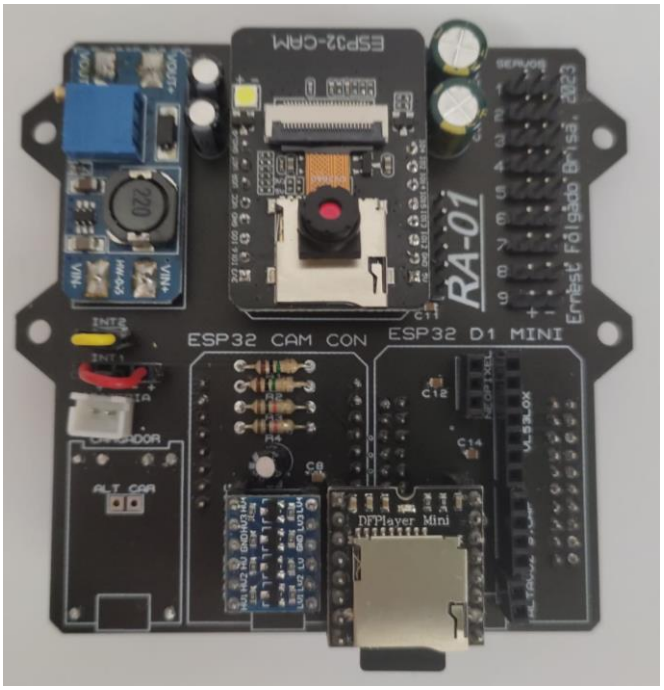


Figura 143. PCB con componentes soldados

Finalmente, se han colocado todos los componentes en su lugar y se ha montado el dispositivo. En el pliego de condiciones se detalla el proceso. Las figuras 144, 145 y 146 muestran cómo se ha llevado a cabo el montaje.

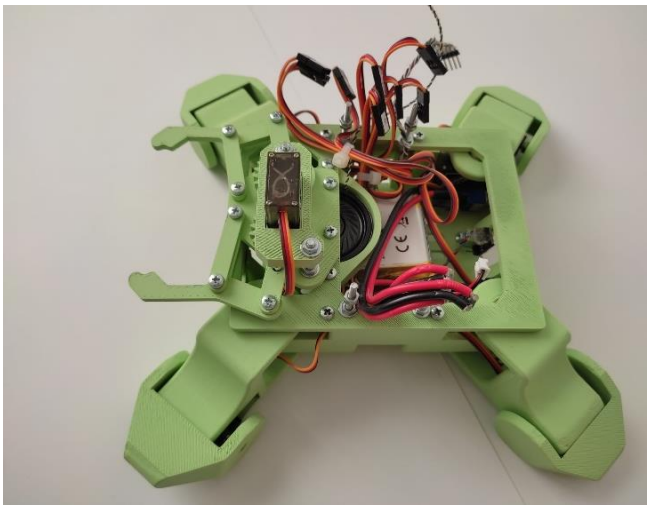


Figura 144. Montaje 1

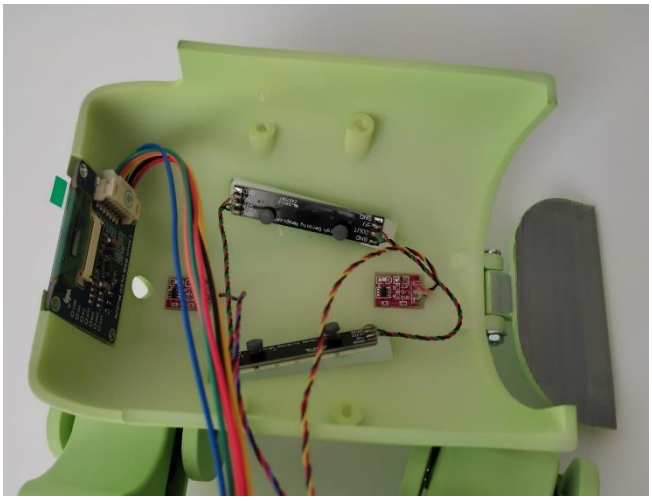


Figura 145. Montaje 2

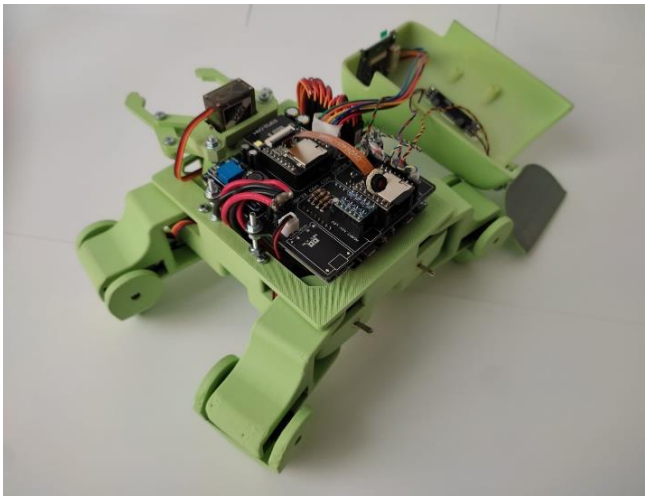


Figura 146. Montaje 3

En la figura 147 y 148, se puede observar el robot montado y encendido.

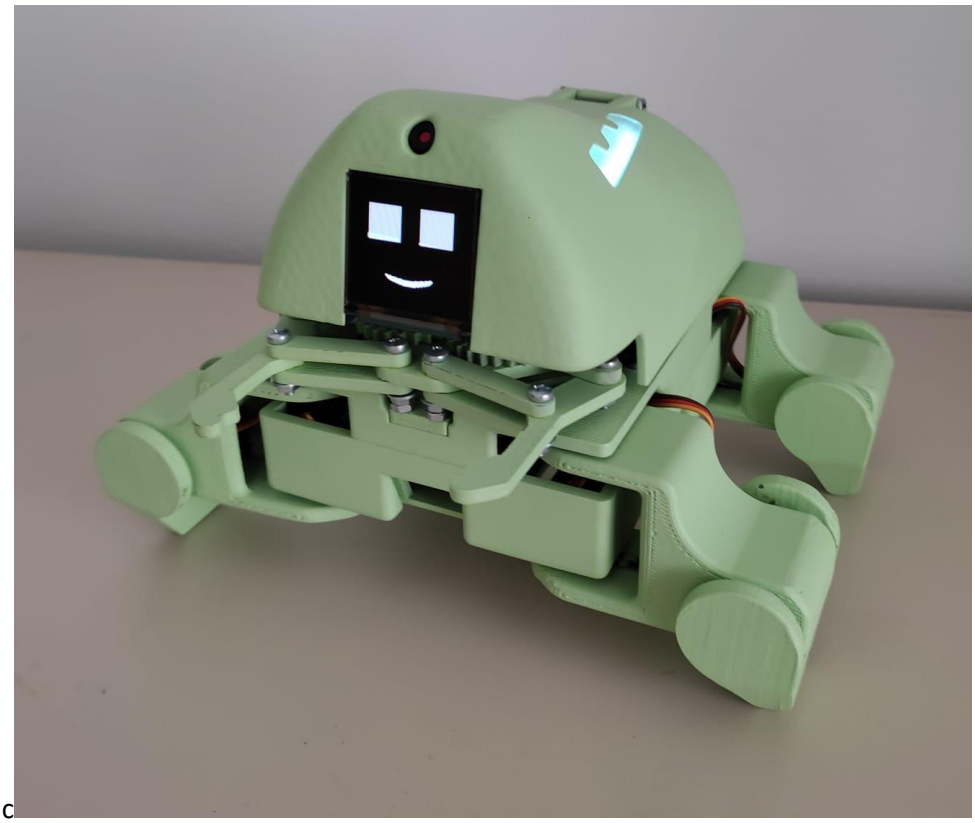
c 

Figura 147. Robot montado y en funcionamiento1

Figura 148. Robot montado y en funcionamiento 2

6. Análisis de fallos y mejoras propuestas

Una vez montado, programado y puesto a prueba el robot, se han encontrado varios aspectos que deben de ser corregidos de cara a realizar una futura versión del dispositivo.

6.1. Problemas con la alimentación

Utilizando la batería seleccionada (*apartado 6.2.1.1 Batería*) se presentaron los siguientes problemas:

- De forma frecuente, cuando se accionan los interruptores de encendido, un pico de corriente causa que la batería se desconecte y se apaga el robot.
- Durante el movimiento de las patas, si se estas se bloquean, aumenta la corriente requerida, se desconecta la batería y se apaga el robot.
- De forma aleatoria se producen picos de corriente durante el movimiento de las patas que causan que la batería se desconecte y se apague el robot.

Para solucionar dicho problema, se optó por probar el prototipo utilizando otra batería similar (figura 149).

Figura 149. Batería utilizada

Utilizando la nueva batería se solucionó el problema de encendido, aun así, los otros dos problemas persisten (con menor frecuencia dependiendo de la carga de la batería).

Se ha utilizado un osciloscopio con una sonda de corriente para medir el consumo de corriente y los datos obtenidos se han exportado a Excel para su análisis. Obteniendo los siguientes datos y gráficas.

En la figura 150, se muestra la corriente medida cuando se ha encendido y apagado el robot de forma continua, de esta forma se puede apreciar el pico de corriente generado al encender el robot. En estos casos concretos dichos valores máximos son: 8.8A, 7,2A, 6,6A y 7,2A.

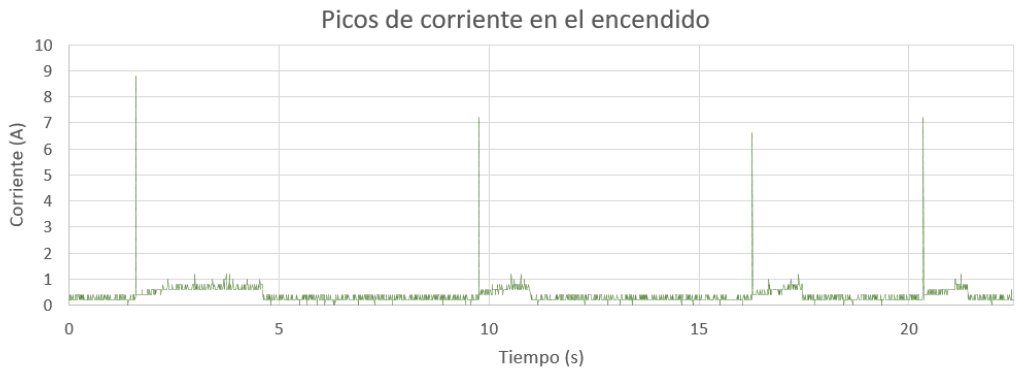


Figura 150. Picos de corriente en el encendido

Cabe destacar que, aunque este pico de corriente es frecuente no ocurre en todas las ocasiones. Se considera, que es por ello por lo que con la anterior batería sí que se lograba encender el robot de forma ocasional.

Durante el encendido del robot, en caso de que estas no estén colocadas en la posición inicial estas se mueven una a una de forma brusca, se ha medido la corriente durante la secuencia de encendido teniendo las patas descolocadas, obteniendo la figura 151, en esta se puede observar los picos de corriente correspondientes a la colocación de las patas, que alcanzan 3.84A.

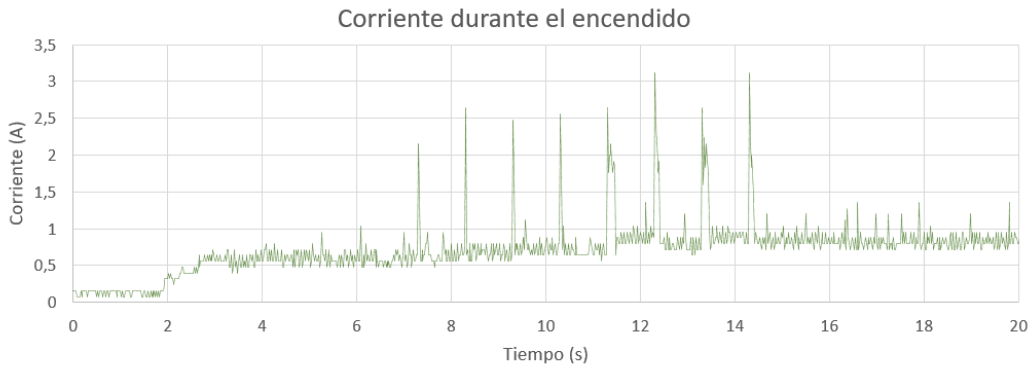


Figura 151. Corriente durante el encendido

En la figura 152, se puede observar la corriente medida cuando el robot caminaba hacia delante obteniéndose como valor máximo 2.88A y como valor promedio 1.1194A.

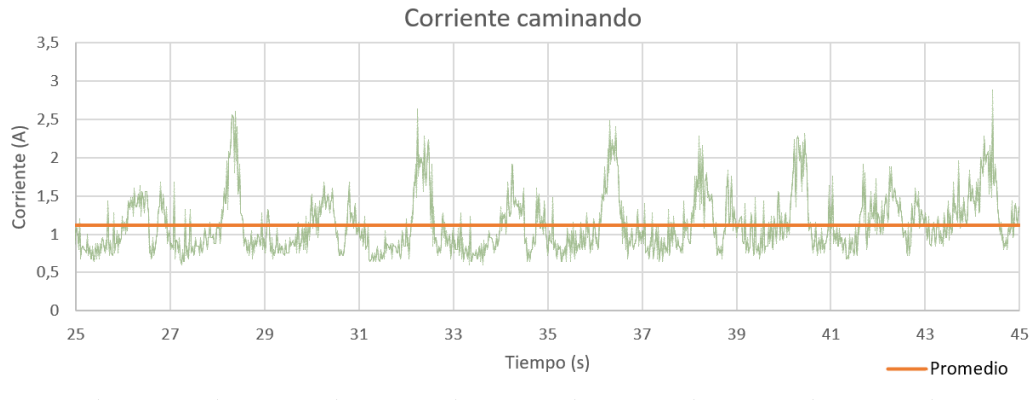


Figura 152. Corriente caminando

Tal y como se ha comentado previamente, de forma aleatoria, durante el movimiento de las patas el robot se apaga. Esto se debe a picos de corriente generados por problemas mecánicos como que una pata se bloquee. En la figura 153 se puede observar la corriente correspondiente a encender el robot con las patas ya colocadas, que este camine y que se apague a causa de un pico de corriente causado por el movimiento de los motores (9.2A).

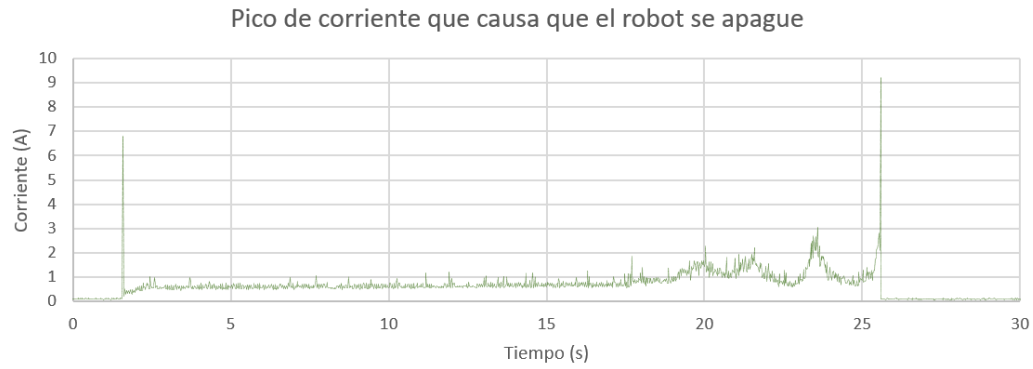


Figura 153. Pico de corriente que causa que el robot se apague

Por otro lado, en la figura 154 se, muestra el consumo de corriente al reproducir audios. Siendo el valor medio 0.8385A y alcanzando un máximo de 1.6A.

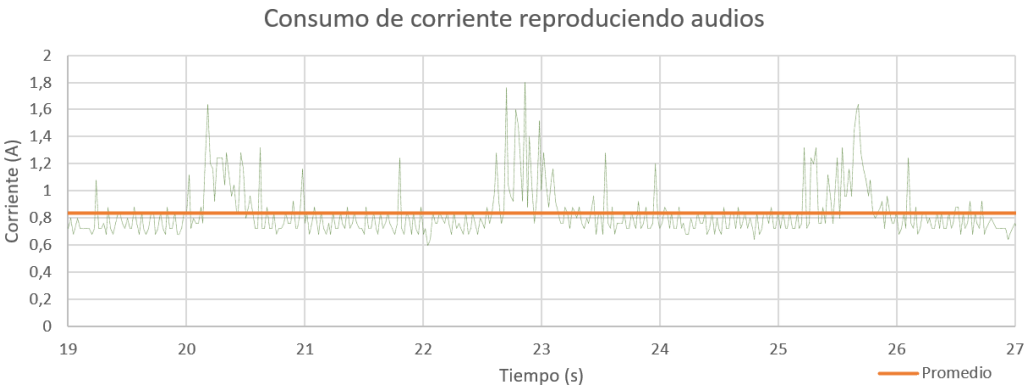


Figura 154. Consumo de corriente reproduciendo audios

Finalmente, también se ha medido la corriente promedio del robot en funcionamiento sin reproducir audios ni caminar. Siendo esta de 0.8243A.

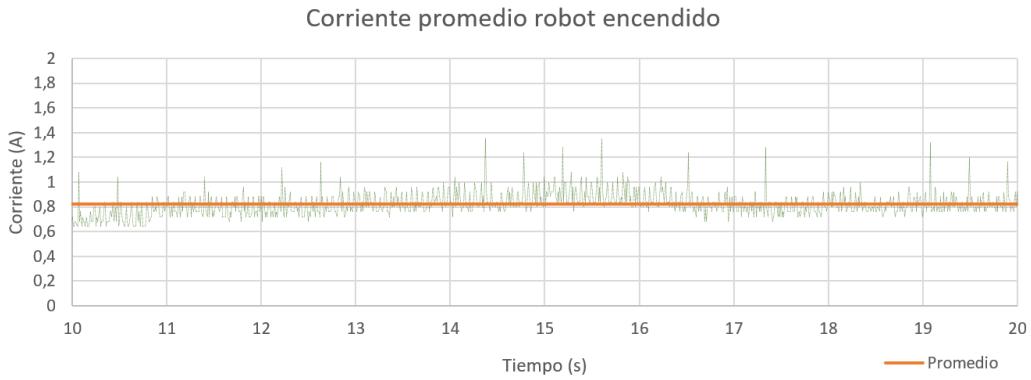


Figura 155. Corriente promedio robot encendido

6.2. Problemas en el diseño del chasis

A la hora de acceder a la batería se precisa de desmontar el robot por completo. Por otro lado, la pieza superior no está unida al resto de piezas con la suficiente rigidez (se desmonta muy fácil) y la puerta no acaba de cerrar y cuando lo hace saca la tarjeta SD del módulo de reproducción de sonido.

Para solucionar estos problemas se deberán de rediseñar las piezas inferior, intermedia y superior.

Otro aspecto que considerar es que las ventanas de metacrilato no permiten el funcionamiento de los sensores de distancia. Se deberá de buscar un material a través del cual, sí que puedan medir, o en caso de que no sea posible minimizar el tamaño de la obertura.

7. Conclusión

Junto a tiempo y dedicación, múltiples aspectos fundamentales en el trabajo de un ingeniero han sido requeridos, puestos a prueba y desarrollados para la realización del proyecto. La investigación por cuenta propia, la búsqueda de piezas y componentes, el diseño de piezas tridimensionales, el diseño electrónico, el desarrollo de software, la planificación de plazos y la superación de problemas, entre otros.

Se ha llevado a cabo el desarrollo del robot planteado. Sin que esto signifique, que el dispositivo no incorpore defectos. Los defectos encontrados se especifican en el apartado *8. Análisis de fallos y mejoras propuestas* y aunque no se descarta realizar por cuenta propia una segunda versión mejorada del robot, se invita a quien acepte el reto a desarrollar una versión del robot que solucione los problemas de la versión actual.

Así mismo, queda público el proyecto y libre de uso y modificación tanto el código como los modelos 3d y la electrónica diseñada para quien desee montar por cuenta propia una réplica del robot o pretenda modificarlo a su gusto. Siempre atribuyéndose los créditos correspondientes al autor original.

Desafortunadamente, no se considera el dispositivo desarrollado como un producto acabado que pueda ser puesto en práctica en el tratamiento de niños con autismo. Quedando pendiente la realización de una versión mejorada para dicho fin.

Finalmente, aunque no se haya desarrollado una versión perfecta y sea importante tener en cuenta los fallos, también es importante destacar lo obtenido. Se ha desarrollado de forma exitosa: un diseño completo, estético y funcional; una PCB, que conecta los distintos componentes electrónicos entre sí; y un software, que permite controlar el robot y poner en práctica el dispositivo demostrando sus posibilidades.

El código y los modelos 3D de las piezas se pueden encontrar en el siguiente enlace: https://github.com/ErnestFB/RA-01

8. Grado de relación del trabajo con los Objetivos de Desarrollo Sostenibles

Objetivo de desarrollo sostenible	Alto	Medio	Bajo	No procede
ODS 1. Fin de la pobreza				X
ODS 2. Hambre cero				X
ODS 3. Salud y bienestar	X			
ODS 4. Educación de calidad		X		
ODS 5. Igualdad de género				X
ODS 6. Agua limpia y saneamiento				X
ODS 7. Energía asequible y no contaminante				X
ODS 8. Trabajo decente y crecimiento económico				X
ODS 9. Industria, innovación e infraestructuras				X
ODS 10. Reducción de las desigualdades		X		
ODS 11. Ciudades y comunidades sostenibles				X
ODS 12. Producción y consumo responsables				X
ODS 13. Acción por el clima				X
ODS 14. Vida submarina				X
ODS 15. Vida de ecosistemas terrestres				X
ODS 16. Paz, justicia e instituciones sólidas				X
ODS 17. Alianzas para lograr objetivos				X

Tabla 7. Grado de relación del trabajo con los Objetivos de Desarrollo Sostenibles (ODS)

Dado que el propósito para el cual se le ha realizado el proyecto es realizar un dispositivo que pueda ser utilizado en el tratamiento de niños con autismo se puede afirmar que principalmente se alinea con el objetivo número 3, “Salud y bienestar”.

Así mismo, dado que la terapia del autismo es indispensable para que quienes lo padecen se puedan integrar en el sistema educativo y en la sociedad también se considera que el proyecto está relacionado con el objetivo 4 y 10, “educación de calidad” y “reducción de las desigualdades”.

9. Bibliografía

[1] Organización Mundial de la Salud, «Autismo», 29 de mayo de 2023. https://www.who.int/es/news-room/fact-sheets/detail/autism-spectrum-disorders (accedido 30 de marzo de 2023).

[2] J. Zeidan *et al.*, «Global prevalence of autism: A systematic review update», *Autism Research*, vol. 15, n.º 5, pp. 778-790, 2022.

[3] Centros para el Control y la Prevención de Enfermedades, «En las comunidades monitoreadas por los CDC se identifica un aumento en la prevalencia del autismo», 31 de marzo de 2020. https://www.cdc.gov/spanish/mediosdecomunicacion/comunicados/p_autismo_033020.html (accedido 5 de abril de 2023).

[4] Confederación Autismo España, «¿Qué es el autismo? Características y diagnóstico», *Autismo España*. https://autismo.org.es/el-autismo/que-es-el-autismo/ (accedido 30 de marzo de 2023).

[5] J. C. C. Ardila y Y. A. Salazar, «APLICACIÓN ROBÓTICA PARA REALIZAR TERAPIAS EN NIÑOS CON AUTISMO», 2014.

[6] M. Mori, K. F. MacDorman, y N. Kageki, «The Uncanny Valley [From the Field]», *IEEE Robotics & Automation Magazine*, vol. 19, n.º 2, pp. 98-100, jun. 2012, doi: 10.1109/MRA.2012.2192811.

[7] CDC, «Signos y síntomas de los trastornos del espectro autista», *Centers for Disease Control and Prevention*, 27 de abril de 2022. https://www.cdc.gov/ncbddd/spanish/autism/signs.html (accedido 30 de marzo de 2023).

[8] American Psychiatric Association, Ed., *Guía de consulta de los criterios diagnósticos del DSM-5*. Arlington, VA: American Psychiatric Publishing, 2014.

[9] National Institute of Mental Health, «Trastornos del espectro autista», *National Institute of Mental Health (NIMH)*. https://www.nimh.nih.gov/health/publications/espanol/trastornos-del-espectro-autista (accedido 30 de marzo de 2023).

[10] CDC, «Tratamiento y servicios de intervención para el trastorno del espectro», *Centers for Disease Control and Prevention*, 28 de enero de 2022. https://www.cdc.gov/ncbddd/spanish/autism/treatment.html (accedido 30 de marzo de 2023).

[11] F. Mulas Delgado, G. Ros Cervera, M. G. Millá Romero, M. C. Etchepareborda Simonini, L. Abad Mas, y M. Téllez de Meneses Lorenzo, «Modelos de intervención en niños con autismo», *RevNeurol*, vol. 50, n.º S03, p. 77, 2010, doi: 10.33588/rn.50S03.2009767.

[12] C. X. González-Moreno, «Intervención en un niño con autismo mediante el juego», *Rev. Fac. Med.*, vol. 66, n.º 3, pp. 365-374, jul. 2018, doi: 10.15446/revfacmed.v66n3.62355.

[13] V. Ruggieri, «Autismo, depresión y riesgo de suicidio», *Medicina (Buenos Aires)*, vol. 80, pp. 12-16, mar. 2020.

[14] V. Sabater, «Autismo en adultos: retos psicológicos y sociales para alcanzar el bienestar», *La Mente es Maravillosa*, 24 de enero de 2020. https://lamenteesmaravillosa.com/autismo-en-adultos-retos-psicologicos-y-sociales-para-alcanzar-el-bienestar/ (accedido 30 de marzo de 2023).

[15] A. H. Zúñiga, N. Balmaña, y M. Salgado, «Los trastornos del espectro autista (TEA)».

[16] C. Breazeal, K. Dautenhahn, y T. Kanda, «Social Robotics», en *Springer Handbook of Robotics*, B. Siciliano y O. Khatib, Eds., en Springer Handbooks. Cham: Springer International Publishing, 2016, pp. 1935-1972. doi: 10.1007/978-3-319-32552-1_72.

[17] V. Pinel, L. A. Rendón, y D. Adrover-Roig, «Los robots sociales como promotores de la comunicación en los Trastornos del Espectro Autista (TEA)», *Let. Hoje*, vol. 53, pp. 39-47, mar. 2018, doi: 10.15448/1984-7726.2018.1.28920.
[18] J.-J. Cabibihan, H. Javed, M. Ang, y S. M. Aljunied, «Why Robots? A Survey on the Roles and Benefits of Social Robots in the Therapy of Children with Autism», *Int J of Soc Robotics*, vol. 5, n.º 4, pp. 593-618, nov. 2013, doi: 10.1007/s12369-013-0202-2.
[19] B. Scassellati, H. Admoni, y M. Matarić, «Robots for Use in Autism Research», *Annual review of biomedical engineering*, vol. 14, pp. 275-94, may 2012, doi: 10.1146/annurev-bioeng-071811-150036.
[20] E. S. Kim *et al.*, «Social Robots as Embedded Reinforcers of Social Behavior in Children with Autism», *J Autism Dev Disord*, vol. 43, n.º 5, pp. 1038-1049, may 2013, doi: 10.1007/s10803-012-1645-2.
[21] I. Werry, I. P. Werry, K. Dautenhahn, y K. Dautenhahn, «Applying Mobile Robot Technology to the Rehabilitation of Autistic Children».
[22] A. Billard, Ed., «Robota: Clever Toy and Educational Tool», *Robotics and Autonomous Systems*, 2003, doi: 10.1016/S0921-8890(02)00380-9.
[23] K. Dautenhahn y A. Billard, Eds., «Games children with autism can play with robota, a humanoid robotics doll», *Universal Access and Assistive Technology*, 2002, doi: 10.1007/978-1-4471-3719-1_18.
[24] H. Kozima, C. Nakagawa, N. Kawai, D. Kosugi, y Y. Yano, «A humanoid in company with children», en *4th IEEE/RAS International Conference on Humanoid Robots, 2004.*, Santa Monica, CA, USA: IEEE, 2004, pp. 470-477. doi: 10.1109/ICHR.2004.1442138.
[25] «CareBots Project (Interaction)». https://www.ei.tohoku.ac.jp/xkozima/carebots/interaction-eng.html (accedido 4 de junio de 2023).
[26] K. Dautenhahn *et al.*, «KASPAR - a minimally expressive humanoid robot for human-robot interaction research», 2009, Accedido: 5 de abril de 2023. [En línea]. Disponible en: http://uhra.herts.ac.uk/handle/2299/9346
[27] B. Robins, K. Dautenhahn, y J. Dubowski, «Does appearance matter in the interaction of children with autism with a humanoid robot?», *Interaction Studies*, vol. 7, n.º 3, pp. 479-512, nov. 2006, doi: 10.1075/is.7.3.16rob.
[28] «Kaspar», *ROBOTS: Your Guide to the World of Robotics*. https://robotsguide.com/robots/kaspar (accedido 4 de junio de 2023).
[29] L. J. Wood, A. Zaraki, B. Robins, y K. Dautenhahn, «Developing Kaspar: A Humanoid Robot for Children with Autism», *Int J of Soc Robotics*, vol. 13, n.º 3, pp. 491-508, jun. 2021, doi: 10.1007/s12369-019-00563-6.
[30] «Embodied Moxie», *Embodied, Inc.* https://embodied.com/ (accedido 5 de abril de 2023).
[31] RoboKind, «Assistive Technology & Curriculum for Autistic Students | RoboKind». https://www.robokind.com/ (accedido 5 de abril de 2023).
[32] X. E. Báez Sánchez, «Efectividad del "Robot Milo" en el desarrollo de habilidades sociales y comunicación en niños de 5 a 7 años con trastorno del espectro del autismo de grado 1», bachelorThesis, Quito, 2018. Accedido: 5 de abril de 2023. [En línea]. Disponible en: http://repositorio.usfq.edu.ec/handle/23000/7136
[33] «Así es Milo, el primer robot con emociones que interactúa con el autismo y que ya está en Cartagena», *El Español*, 13 de julio de 2022. https://www.elespanol.com/omicrono/tecnologia/20220713/mylo-primer-robot-emociones-interactua-autismo-cartagena/687431519_0.html (accedido 5 de abril de 2023).

[34] Z. Bi, Q. Yang, M. Zhang, A. Goupil, y L. Feng, «Accurate Football Detection and Localization for Nao Robot with the Improved HOG-SVM Approach», en *2018 Chinese Automation Congress (CAC)*, nov. 2018, pp. 567-571. doi: 10.1109/CAC.2018.8623727.
[35] «NAO Autism Pack». https://www.robotlab.com/store/robotlab-nao-autism-pack (accedido 5 de abril de 2023).
[36] J. Penalva, «El robot Nao ayudará ahora a niños autistas», *Xataka*, 15 de mayo de 2013. https://www.xataka.com/robotica-e-ia/el-robot-nao-ayudara-ahora-a-ninos-autistas (accedido 4 de junio de 2023).
[37] «NAO le robot humanoïde et programmable | Aldebaran». https://www.aldebaran.com/es/nao (accedido 5 de abril de 2023).
[38] A. Tapus *et al.*, «Children with autism social engagement in interaction with Nao, an imitative robot: A series of single case experiments», *Interaction Studies*, vol. 13, n.º 3, pp. 315-347, ene. 2012, doi: 10.1075/is.13.3.01tap.
[39] M. A. Miskam, M. A. C. Hamid, H. Yussof, S. Shamsuddin, N. A. Malik, y S. N. Basir, «Study on Social Interaction between Children with Autism and Humanoid Robot NAO», *Applied Mechanics and Materials*, vol. 393, pp. 573-578, 2013, doi: 10.4028/www.scientific.net/AMM.393.573.
[40] «Moxie Robot», *Pluck*. https://www.pluckstudio.com/blog/moxie-robot (accedido 5 de junio de 2023).
[41] «Moxie the Robot Helps Children With Autism Through AI - dot.LA». https://dot.la/autism-robot-moxie-ai-2645867544.html (accedido 5 de abril de 2023).
[42] Leopoldo Armesto, «Robótica movil (14670)».
[43] «CNY70 Sensor óptico de reflexión con Arduino», *HeTPro-Tutoriales*, 21 de diciembre de 2017. https://hetpro-store.com/TUTORIALES/cny70-sensor-optico/ (accedido 15 de mayo de 2023).
[44] «Sensor de distancia láser de tiempo de vuelo (ToF) VL53L0X compatible con Arduino», *AZ-Delivery*. https://www.az-delivery.de/es/products/vl53l0x-time-of-flight-tof-laser-abstandssensor (accedido 5 de junio de 2023).
[45] «Pantalla OLED I2C de 128 x 64 píxeles de 1.3 pulgadas compatible con Arduino y Raspberry Pi», *AZ-Delivery*, 5 de mayo de 2018. https://www.az-delivery.de/es/products/1-3zoll-i2c-oled-display (accedido 5 de junio de 2023).
[46] «ESP32 NodeMCU Módulo WLAN WiFi Development Board con CP2102 (modelo sucesor de ESP8266) compatible con Arduino», *AZ-Delivery*, 29 de abril de 2018. https://www.az-delivery.de/es/products/esp32-developmentboard (accedido 5 de junio de 2023).
[47] Bricogeek, «Batería LiPo 6000mAh / 3.7V Sparkfun PRT-08484 | BricoGeek.com». https://tienda.bricogeek.com/baterias-lipo/418-bateria-lipo-6000mah-37v.html (accedido 13 de abril de 2023).
[48] «Cargador LiPo Micro USB TP4056 BricoGeek | BricoGeek.com». https://tienda.bricogeek.com/cargadores/1232-cargador-lipo-usb-tp4056.html (accedido 13 de abril de 2023).
[49] «TP4056.pdf». Accedido: 13 de abril de 2023. [En línea]. Disponible en: https://dlnmh9ip6v2uc.cloudfront.net/datasheets/Prototyping/TP4056.pdf
[50] «MT3608 DC/DC elevador 2-24V/5-28V 2A (ref: 0364) – electronperdido.com». https://electronperdido.com/shop/fuentes-alimentacion/dc-dc/elevador/mt3608-dc-dc-elevador-2-24v-5-28v-2a/ (accedido 13 de abril de 2023).
[51] Microchip Technologi Inc., «Datasheet MCP1755».
[52] «esp32_datasheet_en.pdf». Accedido: 14 de abril de 2023. [En línea]. Disponible en: https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf
[53] *ESP32 Salidas PWM*, (26 de enero de 2022). Accedido: 16 de mayo de 2023. [En línea Video]. Disponible en: https://www.youtube.com/watch?v=azFFLbQvsRc

[54] «Diymore Placa de Desarrollo Bluetooth ESP32-CAM-MB,ESP32 CAM de Doble Núcleo de Desarrollo Inalámbrico con Cámara 2640 Módulo de Tarjeta TF : Amazon.es: Electrónica». https://www.amazon.es/diymore-desarrollo-Bluetooth-ESP32-CAM-MB-inal%C3%A1mbrico/dp/B08X3GRK22/ref=sr_1_8?__mk_es_ES=%C3%85M%C3%85%C5%BD%C3%95%C3%91&crid=2SW8MJHLQ1GG0&keywords=esp32+cam&qid=1685963734&sprefix=esp32+cam%2Caps%2C102&sr=8-8 (accedido 5 de junio de 2023).
[55] «Módulo de cámara OV2640 para ESP32 CAM - AliExpress», *aliexpress.com*. //es.aliexpress.com/item/1005003279204315.html?src=ibdm_d03p0558e02r02&sk=&aff_platform=&aff_trace_key=&af=&cv=&cn=&dp= (accedido 5 de junio de 2023).
[56] D. Expst, B. Agnd, y V. Strobe, «OV2640 Color CMOS UXGA (2.0 MegaPixel) CAMERACHIP Sensor with OmniPixel2 Technology», 2007.
[57] «ESP-32 cam Developement Board». Accedido: 17 de mayo de 2023. [En línea]. Disponible en: https://media.digikey.com/pdf/Data%20Sheets/DFRobot%20PDFs/DFR0602_Web.pdf
[58] «ESP32-CAM AI-Thinker Pinout Guide: GPIOs Usage Explained | Random Nerd Tutorials», 10 de marzo de 2020. https://randomnerdtutorials.com/esp32-cam-ai-thinker-pinout/ (accedido 17 de mayo de 2023).
[59] AZ-Delivery, «Datasheet ESP32 D1 mini».
[60] L. Llamas, «Interruptor táctil con Arduino y sensor capacitivo touchless», *Luis Llamas*, 15 de julio de 2016. https://www.luisllamas.es/interruptor-touchless-con-arduino-y-sensor-capacitivo/ (accedido 18 de mayo de 2023).
[61] «TTP223-Capacitive-Touch-Sensor-Module-Schematic-Diagram.pdf». Accedido: 17 de mayo de 2023. [En línea]. Disponible en: https://www.rhydolabz.com/documents/27/TTP223-Capacitive-Touch-Sensor-Module-Schematic-Diagram.pdf
[62] *Touch Switch?? Aprende a usarlo!! TTP223 - Arduino | Review en ESPAÑOL*, (29 de junio de 2020). Accedido: 18 de mayo de 2023. [En línea Video]. Disponible en: https://www.youtube.com/watch?v=9dxp33ESX8M
[63] «VL53L0X pdf, VL53L0X Description, VL53L0X Datasheet, VL53L0X view ::: ALLDATASHEET »: https://pdf1.alldatasheet.com/datasheet-pdf/view/948120/STMICROELECTRONICS/VL53L0X.html (accedido 17 de mayo de 2023).
[64] *Programar Sensor Laser de Distancia VL53L0X Con Arduino*, (16 de febrero de 2021). Accedido: 5 de abril de 2023. [En línea Video]. Disponible en: https://www.youtube.com/watch?v=BbJYIfF8qHw
[65] «Medir distancia con precisión con Arduino y sensor láser VL53L0X y VL6180X», *Luis Llamas*, 9 de enero de 2018. https://www.luisllamas.es/arduino-sensor-distancia-vl53l0x/ (accedido 5 de abril de 2023).
[66] «Combinación de sensores de proximidad de un solo Bus I^2C | DigiKey». https://www.digikey.com.mx/es/articles/combining-multiple-proximity-sensors-using-a-single-i2c-bus (accedido 5 de abril de 2023).
[67] «MG90S Metal Gear High Speed Micro Servo for RC Car Helicopter Plane», *DIYMORE*. https://www.diymore.cc/products/mg90s-servo-motors-metal-gear-9g-for-rc-helicopter-airplane-car-boat-robot-controls (accedido 17 de mayo de 2023).
[68] «DFPlayer Mini Mp3 Player - DFRobot Wiki». https://wiki.dfrobot.com/DFPlayer_Mini_SKU_DFR0299 (accedido 16 de mayo de 2023).
[69] «Bi-Directional Logic Level Converter Hookup Guide - SparkFun Learn». https://learn.sparkfun.com/tutorials/bi-directional-logic-level-converter-hookup-guide/all (accedido 18 de mayo de 2023).
[70] «Comparación de tecnologías de impresión 3D: FDM, SLA o SLS», *Formlabs*. https://formlabs.com/es/blog/fdm-sla-sls-como-elegir-tecnologia-impresion-3d-adecuada/ (accedido 17 de mayo de 2023).

[71] «Editor JavaScript online - www.cubicfactory.com - Diseño web Zaragoza». https://www.cubicfactory.com/jseditor/ (accedido 7 de junio de 2023).
[72] «HTML Minifier - Minify HTML and any CSS or JS included in your markup». https://www.willpeavy.com/tools/minifier/ (accedido 5 de junio de 2023).
[73] «Unminify JS, CSS, HTML». https://www.unminify2.com/ (accedido 7 de junio de 2023).
[74] «image2cpp». http://javl.github.io/image2cpp/ (accedido 7 de junio de 2023).
[75] «Rinky-Dink Electronics». http://www.rinkydinkelectronics.com/t_imageconverter565.php (accedido 7 de junio de 2023).
[76] D. Workshop, «Build an ESP32CAM Robot Car», *DroneBot Workshop*, 13 de abril de 2021. https://dronebotworkshop.com/esp32cam-robot-car/ (accedido 7 de junio de 2023).
[77] Leopoldo Armesto, «Sistemas robotizados (12157)».

ANEXO 1.
MANUAL DE USUARIO

Diseño y desarrollo de un robot cuadrúpedo dedicado al tratamiento de niños con autismo.

Trabajo final del:	Grado en Ingeniería electrónica industrial y Automática
Realizado por:	Ernest Folgado Brisa
Tutorizado por:	Javier Ibáñez Civera
Curso académico:	2022/2023

Contenido manual de usuario

1. Objeto

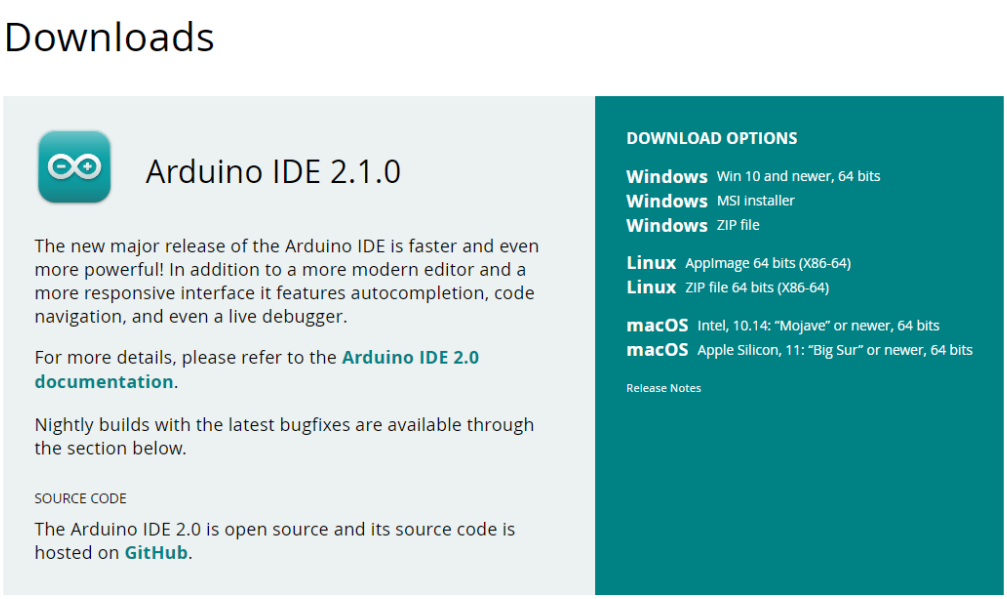
En el presente manual de usuario se detalla como descargar el entorno de programación de Arduino y configurarlo para subir código a microcontroladores ESP 32. Adicionalmente, se detalla la ubicación de la ranura SD y de la entrada de carga.

2. Programación del dispositivo

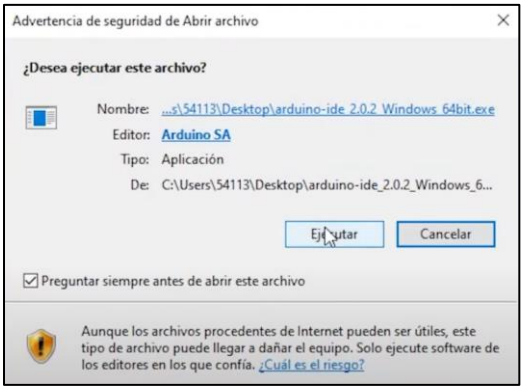
2.1. Descarga del IDE de Arduino

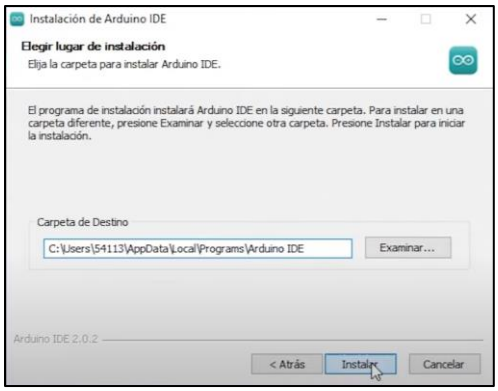
Para descargar el entorno de programación se puede buscar este por internet, o bien accediendo al siguiente enlace a la página oficial de Arduino:

https://www.arduino.cc/en/software

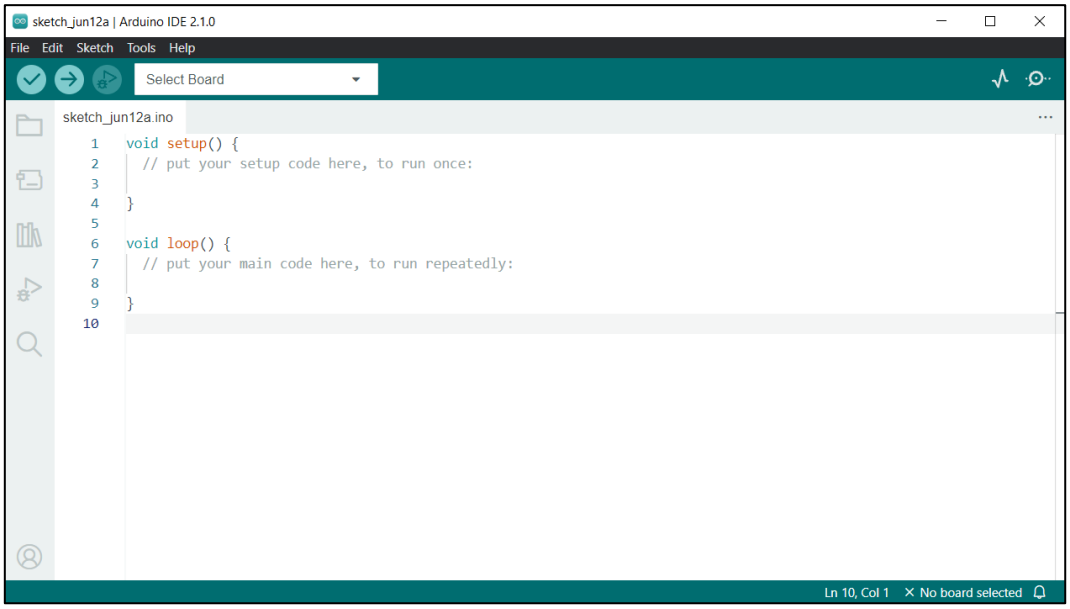


Una vez descargado, se debe de ejecutar el instalador, se deben de aceptar los términos se selecciona la ruta de instalación y esperamos a que finalice el proceso.



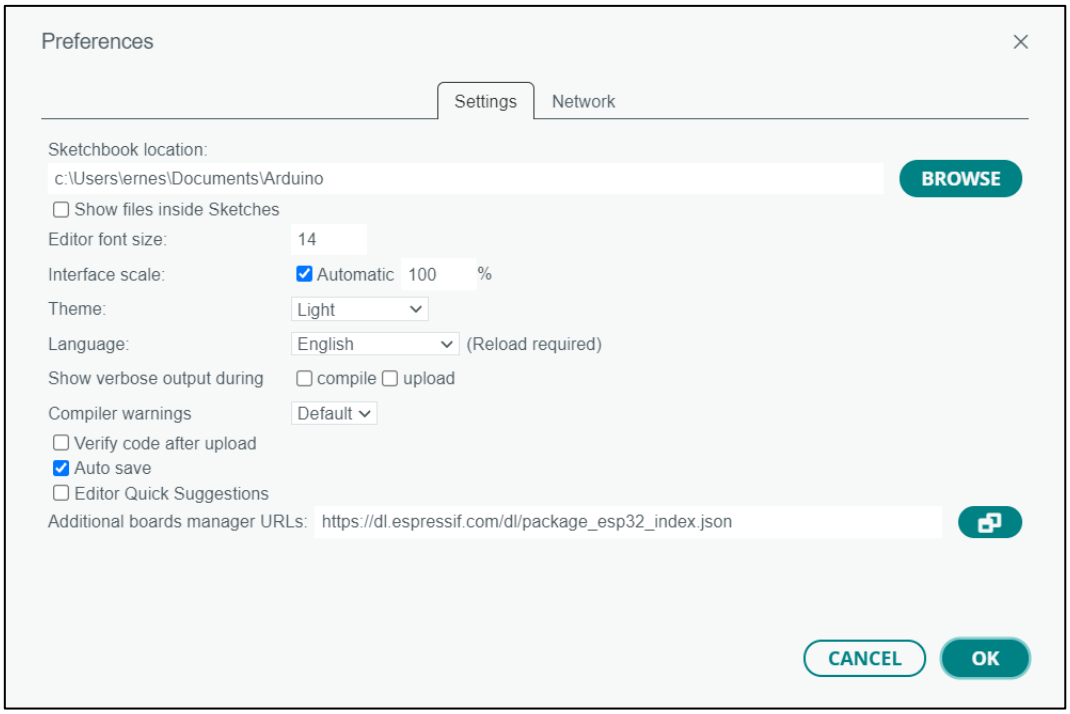


Una vez instalado el programa se debería de reconocer el siguiente entorno:



2.2.Incluir placas ESP32

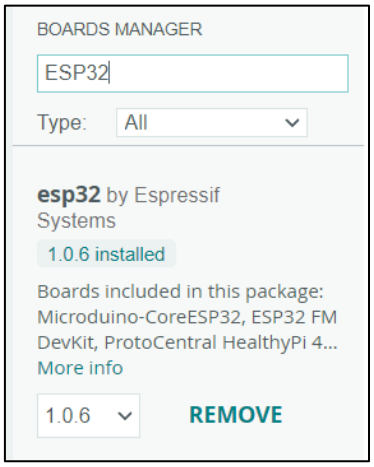
Para incluir las placas ESP 32 en el IDE de Arduino se deberá de ir a archivos, preferencias. (Files, Preferences).



Una vez en las preferencias se deberá de copiar y pegar el siguiente enlace donde pone ”Gestor de URLs adicionales de tarjetas” o “Additional boards manager URLs”

https://dl.espressif.com/dl/package_esp32_index.json

Acto seguido, se deberá de ir a herramientas, gestor de tarjetas y se deberán de buscar e instalar las placas ESP32



2.3. Instalación de librerías

Para poder programar el dispositivo se requieren de varias librerías. Estas se pueden descargar en formato “ZIP” desde los siguientes enlaces.

https://github.com/adafruit/Adafruit-GFX-Library

https://github.com/adafruit/Adafruit-SSD1351-library/blob/master/Adafruit_SSD1351.h

https://github.com/jkb-git/ESP32Servo

https://github.com/adafruit/Adafruit_VL53L0X

https://github.com/adafruit/Adafruit_NeoPixel

https://github.com/DFRobot/DFRobotDFPlayerMini

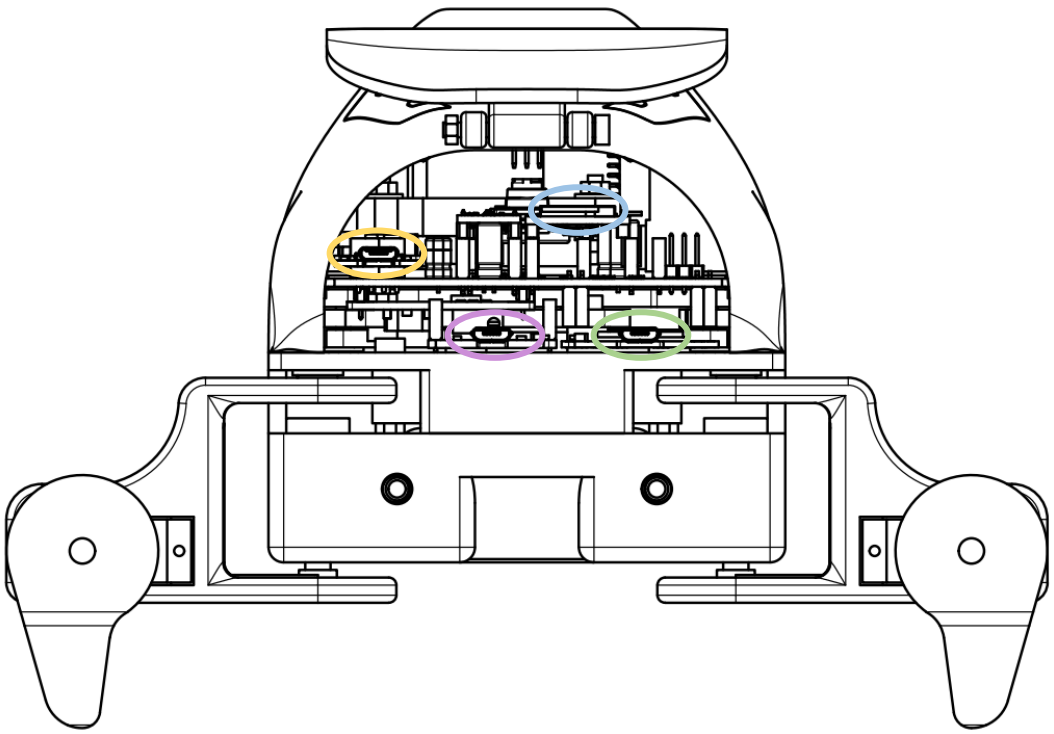
Una vez descargadas, se incorporarán a Arduino yendo a Sketch, incluir librería, incluir librería .ZIP.

2.4. Subir código

Tal y como se puede observar en la siguiente imagen, el robot en su parte trasera cuenta con 3 entradas micro USB y una ranura SD.

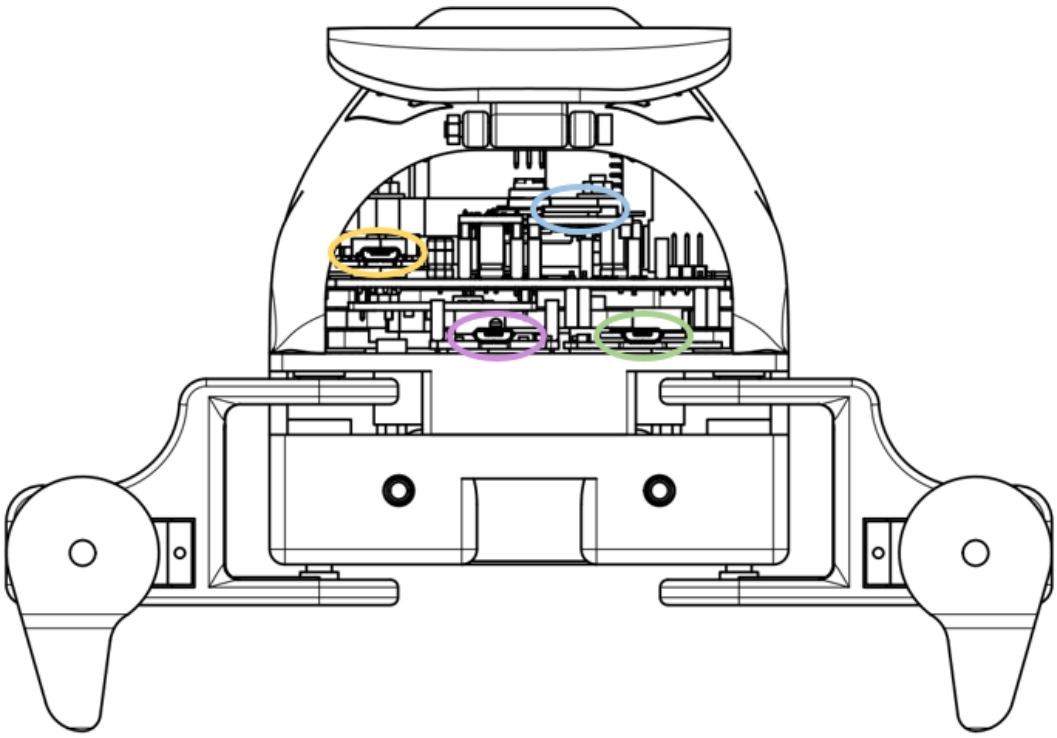
Siendo el conector rodeado de morado el correspondiente con el ESP 32 cam, se conectará a este un cable usb-micro USB desde el ordenadZor para subir código al microcontrolador ESP 32 cam.

Por otro lado, para subir código al ESP 32 d1 mini se utilizará el conector rodeado de color verde.



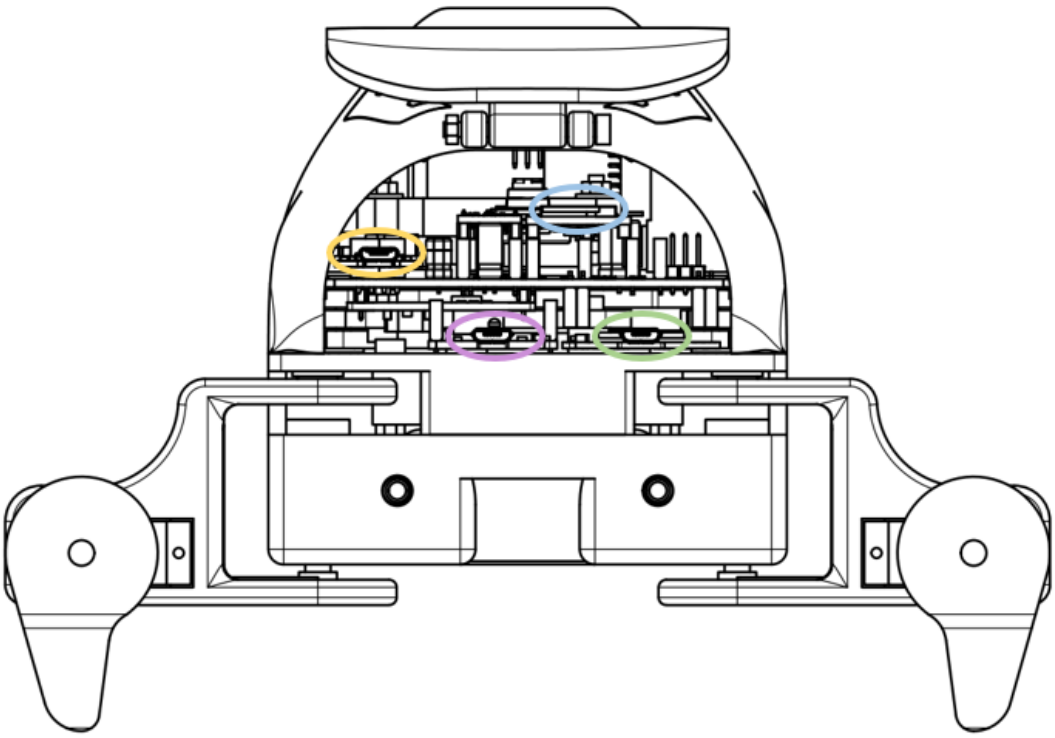
3. Cambiar los audios

Si se pretende modificar los archivos de audio que le robot reproduce, se deberá de extraer la tarjeta SD, (ovalo azul).



4. Cargar el dispositivo

Para cargar el dispositivo se deberá de conectar el cargador al puerto micro USB rodeado de amarillo en la siguiente imagen.



ANEXO 2.
CÓDIGO

Diseño y desarrollo de un robot cuadrúpedo dedicado al tratamiento de niños con autismo.

Trabajo final del:	Grado en Ingeniería electrónica industrial y Automática
Realizado por:	Ernest Folgado Brisa
Tutorizado por:	Javier Ibáñez Civera
Curso académico:	2022/2023

Contenido código

1. Objeto

Dado que el robot incorpora dos microcontroladores, se han realizado 2 códigos separados. Así mismo, para mayor claridad se han dividido dichos códigos en varios archivos.

- ESP_32_cam.ino
    - app_httpd.h
    - app_httpd.cpp
    - funciones.h
    - funciones.cpp
    - imágenes.h
- ESP_32_D1_mini.ino
    - funciones_2.h
    - funciones_2.cpp

Cabe destacar, que para que para poder compilar, todos los archivos correspondientes a un mismo código deben de estar en la misma carpeta cuyo nombre tiene que ser igual al del archivo ".ino".

En el siguiente enlace se pueden encontrar los archivos correspondientes:

https://github.com/ErnestFB/RA-01

2. ESP_32_cam.ino

```
#include <WiFi.h>

#include <soc/rtc_cntl_reg.h>

#include <dl_lib_matrix3d.h>      //Librería para el funcionamiento de la cámara
#include <img_converters.h>

#include <esp_http_server.h>      //Librería para la creación del portal de control

#include <Adafruit_GFX.h>         //Libreríias para la pantalla
#include <Adafruit_SSD1351.h>
#include <SPI.h>

#include "app_httpd.h"            //Librerías propias
#include "funciones.h"
#include "imagenes.h"

// Dimensiones de la pantalla
#define ANCHO_PANTALLA  128
#define ALTO_PANTALLA 128

//pines pantalla
#define MOSI_PIN 12 //DIN
#define SCLK_PIN 13 //CLK
#define CS_PIN   15 //CS
#define DC_PIN   14 //MISO

//colores
#define NEGRO           0x0000
#define AZUL            0x001F
#define ROJO            0xF800
#define VERDE           0x07E0
#define MORADO          0xF81F
#define AMARILLO        0xFFE0
#define BLANCO          0xFFFF

#define RX_1a2 16
#define TX_1a2 2


Adafruit_SSD1351 pantalla = Adafruit_SSD1351(ANCHO_PANTALLA, ALTO_PANTALLA, CS_PIN, DC_PIN,
MOSI_PIN, SCLK_PIN);

HardwareSerial Serial_1a2(1);

TaskHandle_t  Loop_2;

// SSID y contraseña para acceder a la red WIFI creada
const char* ssid1 = "RA_01";
const char*   password = "5555444433";

uint8_t pos_ojos_actual_x=32;
uint8_t pos_ojos_actual_y=40;
uint8_t pos_ojos_deseada_x=32;
uint8_t pos_ojos_deseada_y=40;

uint8_t emocion=0;
uint8_t emocion_previa=99;

bool juego_cartas = false;
bool juego_PPT = false;
bool agitar_ojos = false;
```

```
void setup()
{
  Serial.begin(115200);
   Configurar_Camara();
   Configurar_Red_WIFI();
   Configurar_Servidor();
  delay(50);
  xTaskCreatePinnedToCore(loop_2,     //Nombre de la función
                          "loop_2",   //Nombre, puede ser cualquier cosa
                          5000,       //Tamaño de la pila
                          NULL,       //Parámetro que le quiera pasar a la tarea
                          1,          //Prioridad
                          & Loop_2,   //Nombre de la tarea
                          0);         //Nucleo
}

void loop()
{
  delay(1);
  yield();
}

void  loop_2(void *parameter) //Se ejecuta en el núcelo 0
{
  pantalla.begin();
  pantalla.fillScreen(NEGRO);
  Serial_1a2.begin(115200, SERIAL_8N1, RX_1a2, TX_1a2);
  Serial.println("loop_2 setup finalizado");
  while(1)
  {
    Lectura_Serial_2a1();
    Mostrar_Cara();
    if(juego_cartas) Juego_Cartas();
    else if (juego_PPT) Juego_PPT ();
    else if(esp_random() % 500==0) Accion_Aleatoria();
    else if(agitar_ojos) Agitar_Ojos();
    delay(5);
    yield();
  }
}
```

2.1. app_httpd.h

```
#ifndef APP_HTTPD_H
#define APP_HTTPD_H

void  Configurar_Servidor();  //Configura e inicia el servidor local donde se alberga el
portal de control

#endif
```

2.2. app_httpd.cpp

```
#include <WiFi.h>

#include <soc/rtc_cntl_reg.h>

#include <dl_lib_matrix3d.h>
#include <esp_http_server.h>
#include <img_converters.h>

#include <Adafruit_GFX.h>         //Librerias de la pantalla
#include <Adafruit_SSD1351.h>
#include <SPI.h>

#include "funciones.h"
#include "app_httpd.h"

#include "funciones.h"
```

```
typedef struct {
  httpd_req_t *req;
  size_t len;
} jpg_chunking_t;

#define PART_BOUNDARY "123456789000000000000987654321"
static const char* _STREAM_CONTENT_TYPE = "multipart/x-mixed-replace;boundary=" PART_BOUNDARY;
static const char* _STREAM_BOUNDARY = "\r\n--" PART_BOUNDARY "\r\n";
static const char* _STREAM_PART = "Content-Type: image/jpeg\r\nContent-Length: %u\r\n\r\n";

httpd_handle_t stream_httpd = NULL;
httpd_handle_t camera_httpd = NULL;

static size_t jpg_encode_stream(void * arg, size_t index, const void* data, size_t len)
{
  //Serial.println("jpg_encode_stream en nucleo ->"+String(xPortGetCoreID()));
  jpg_chunking_t *j = (jpg_chunking_t *)arg;
  if (!index) {
    j->len = 0;
  }
  if (httpd_resp_send_chunk(j->req, (const char *)data, len) != ESP_OK) {
    return 0;
  }
  j->len += len;
  return len;
}

static esp_err_t stream_handler(httpd_req_t *req) //Esta función se encarga de la
retransmisión
{
  Serial.println("stream_handler en nucleo ->"+String(xPortGetCoreID()));
  camera_fb_t * fb = NULL;
  esp_err_t res = ESP_OK;
  size_t _jpg_buf_len = 0;
  uint8_t * _jpg_buf = NULL;
  char * part_buf[64];
  dl_matrix3du_t *image_matrix = NULL;

  static int64_t last_frame = 0;
  if (!last_frame) {
    last_frame = esp_timer_get_time();
  }

  res = httpd_resp_set_type(req, _STREAM_CONTENT_TYPE);
  if (res != ESP_OK) {
    return res;
  }

  while (true) {
    //Serial.println("Bucle retransmision en nucleo ->"+String(xPortGetCoreID()));
    fb = esp_camera_fb_get();
    if (!fb) {
      Serial.println("Camera capture failed");
      res = ESP_FAIL;
    } else {
      {
        if (fb->format != PIXFORMAT_JPEG) {
          //Serial.println("Se mete aqui 228");
          bool jpeg_converted = frame2jpg(fb, 80, &_jpg_buf, &_jpg_buf_len);
          esp_camera_fb_return(fb);
          fb = NULL;
          if (!jpeg_converted) {
            Serial.println("JPEG compression failed");
            res = ESP_FAIL;
          }
        } else {
          _jpg_buf_len = fb->len;
          _jpg_buf = fb->buf;
        }
      }
    }
    if (res == ESP_OK) {
      size_t hlen = snprintf((char *)part_buf, 64, _STREAM_PART, _jpg_buf_len);
```

```
      res = httpd_resp_send_chunk(req, (const char *)part_buf, hlen);
    }
    if (res == ESP_OK) {
      res = httpd_resp_send_chunk(req, (const char *)_jpg_buf, _jpg_buf_len);
    }
    if (res == ESP_OK) {
      res = httpd_resp_send_chunk(req, _STREAM_BOUNDARY, strlen(_STREAM_BOUNDARY));
    }
    if (fb) {
      esp_camera_fb_return(fb);
      fb = NULL;
      _jpg_buf = NULL;
    } else if (_jpg_buf) {
      free(_jpg_buf);
      _jpg_buf = NULL;
    }
    if (res != ESP_OK) {
      break;
    }
    int64_t fr_end = esp_timer_get_time();
    int64_t frame_time = fr_end - last_frame;
    last_frame = fr_end;
    frame_time /= 1000;
    /*Serial.printf("MJPG: %uB %ums (%.1ffps)\n",
                  (uint32_t)(_jpg_buf_len),
                  (uint32_t)frame_time, 1000.0 / (uint32_t)frame_time);*/
  }
  last_frame = 0;
  return res;
}

enum state {fwd, rev, stp};
state actstate = stp;

static esp_err_t cmd_handler(httpd_req_t *req)
{
  Serial.println("cmd_handler en nucleo ->"+String(xPortGetCoreID()));
  char*  buf;
  size_t buf_len;
  char variable[32] = {0,};
  char value[32] = {0,};

  buf_len = httpd_req_get_url_query_len(req) + 1;
  if (buf_len > 1) {
    buf = (char*)malloc(buf_len);
    if (!buf) {
      httpd_resp_send_500(req);
      return ESP_FAIL;
    }
    if (httpd_req_get_url_query_str(req, buf, buf_len) == ESP_OK) {
      if (httpd_query_key_value(buf, "var", variable, sizeof(variable)) == ESP_OK &&
          httpd_query_key_value(buf, "val", value, sizeof(value)) == ESP_OK) {
      } else {
        free(buf);
        httpd_resp_send_404(req);
        return ESP_FAIL;
      }
    } else {
      free(buf);
      httpd_resp_send_404(req);
      return ESP_FAIL;
    }
    free(buf);
  } else {
    httpd_resp_send_404(req);
    return ESP_FAIL;
  }
  int val = atoi(value);
  int res = 0;
  // Look at values within URL to determine function
  if (!strcmp(variable, "RA-01")) {
    Serial.println("Dato recibido:");
    Serial.println(val);
    switch(val)
```

```
    {
      case 1:
        Serial.println("Se ha pulsado el boton 1");
        Boton_Pulsado_1();
      break;

      case 2:
        Serial.println("Se ha pulsado el boton 2");
        Boton_Pulsado_2();
      break;

      case 3:
        Serial.println("Se ha pulsado el boton 3");
        Boton_Pulsado_3();
      break;

      case 4:
        Serial.println("Se ha pulsado el boton 4");
        Boton_Pulsado_4();
      break;

      case 5:
        Serial.println("Se ha pulsado el boton 5");
        Boton_Pulsado_5();
      break;

      case 6:
        Serial.println("Se ha pulsado el boton 6");
        Boton_Pulsado_6();
      break;

      case 7:
        Serial.println("Se ha pulsado el boton 7");
        Boton_Pulsado_7();
      break;

      case 8:
        Serial.println("Se ha pulsado el boton 8");
        Boton_Pulsado_8();
      break;

      case 9:
        Serial.println("Se ha pulsado el boton 9");
        Boton_Pulsado_9();
      break;

      case 10:
        Serial.println("Se ha pulsado el boton 10");
        Boton_Pulsado_10();
      break;
      case 11:
        Serial.println("Se ha pulsado el boton 11");
        Boton_Pulsado_11();
      break;

      case 12:
        Serial.println("Se ha pulsado el boton 12");
        Boton_Pulsado_12();
      break;

      case 13:
        Serial.println("Se ha pulsado el boton 13");
        Boton_Pulsado_13();
      break;

      case 14:
        Serial.println("Se ha pulsado el boton 14");
        Boton_Pulsado_14();
      break;

      case 15:
        Serial.println("Se ha pulsado el boton 15");
        Boton_Pulsado_15();
      break;
```

```
      case 16:
        Serial.println("Se ha pulsado el boton 16");
        Boton_Pulsado_16();
      break;

      case 17:
        Serial.println("Se ha pulsado el boton 17");
        Boton_Pulsado_17();
      break;

      case 18:
        Serial.println("Se ha pulsado el boton 18");
        Boton_Pulsado_18();
      break;

      case 19:
        Serial.println("Se ha pulsado el boton 19");
        Boton_Pulsado_19();
      break;

      case 20:
        Serial.println("Se ha pulsado el boton 20");
        Boton_Pulsado_20();
      break;
      case 21:
        Serial.println("Se ha pulsado el boton 21");
        Boton_Pulsado_21();
      break;
      case 22:
        Serial.println("Se ha pulsado el boton 22");
        Boton_Pulsado_22();
      break;
      case 23:
        Serial.println("Se ha pulsado el boton 23");
        Boton_Pulsado_23();
      break;
      case 24:
        Serial.println("Se ha pulsado el boton 24");
        Boton_Pulsado_24();
      break;
      case 25:
        Serial.println("Se ha pulsado el boton 25");
        Boton_Pulsado_25();
      break;
      case 26:
        Serial.println("Se ha pulsado el boton 26");
        Boton_Pulsado_26();
      break;
      case 27:
        Serial.println("Se ha pulsado el boton 27");
        Boton_Pulsado_27();
      break;
    }
  }
  else
  {
    Serial.println("variable");
    res = -1;
  }

  if (res) {
    return httpd_resp_send_500(req);
  }

  httpd_resp_set_hdr(req, "Access-Control-Allow-Origin", "*");
  return httpd_resp_send(req, NULL, 0);
}

static esp_err_t status_handler(httpd_req_t *req)
{
  Serial.println("status_handler en nucleo ->"+String(xPortGetCoreID()));
  static char json_response[1024];
```

```
  sensor_t * s = esp_camera_sensor_get();
  char * p = json_response;
  *p++ = '{';

  p += sprintf(p, "\"framesize\":%u,", s->status.framesize);
  p += sprintf(p, "\"quality\":%u,", s->status.quality);
  *p++ = '}';
  *p++ = 0;
  httpd_resp_set_type(req, "application/json");
  httpd_resp_set_hdr(req, "Access-Control-Allow-Origin", "*");
  return httpd_resp_send(req, json_response, strlen(json_response));
}

static const char PROGMEM INDEX_HTML[] = R"rawliteral(
<!DOCTYPE html><html><head> <meta charset="utf-8"/> <meta name="viewport"
content="width=device-width,initial-scale=1"/> <title>RA-01_Control</title> <style>body{font-
family: Arial, Helvetica, sans-serif; background: #8BD77F; color: #efefef; font-size:
16px;}h2{font-size: 18px;}section.main{display: flex;}#menu, section.main{flex-direction:
column;}#menu{display: none; flex-wrap: nowrap; min-width: 420px; background: #363636;
padding: 8px; border-radius: 4px; margin-top: -10px; margin-right: 10px;}#content{display:
flex; flex-wrap: wrap; align-items: stretch;}figure{padding: 0; margin: 0; -webkit-margin-
before: 0; margin-block-start: 0; -webkit-margin-after: 0; margin-block-end: 0; -webkit-
margin-start: 0; margin-inline-start: 0; -webkit-margin-end: 0; margin-inline-end: 0;}figure
img{display: block; width: 100%; height: auto; border-radius: 4px; margin-top: 200px;}@media
(min-width: 800px) and (orientation: portrait){#content{display: flex; flex-wrap: nowrap;
align-items: stretch;}figure img{display: block; max-width: 100%; max-height: calc(100vh -
40px); width: auto; height: auto;}figure{padding: 0; margin: 0; -webkit-margin-before: 0;
margin-block-start: 0; -webkit-margin-after: 0; margin-block-end: 0; -webkit-margin-start: 0;
margin-inline-start: 0; -webkit-margin-end: 0; margin-inline-end: 0;}}section
#buttons{display: flex; flex-wrap: nowrap; justify-content: space-between;}#nav-toggle{cursor:
pointer; display: block;}#nav-toggle-cb{outline: 0; opacity: 0; width: 0; height: 0;}#nav-
toggle-cb:checked+#menu{display: flex;}.input-group{display: flex; flex-wrap: nowrap; line-
height: 22px; margin: 5px 0;}.input-group>label{display: inline-block; padding-right: 10px;
min-width: 47%;}.input-group input, .input-group select{flex-grow: 1;}.range-max, .range-
min{display: inline-block; padding: 0 5px;}button{display: block; margin: 5px; padding: 5px
1px; border: 0; line-height: 28px; cursor: pointer; color: #fff; background: #568759; border-
radius: 5px; font-size: 16px; outline: 0; width: 100px;}.button2{background-color: #008cba;
font-size: 30px; width: 75px;}.button3{background-color: #f44336; width:
100px;}.button4{background-color: #e7e7e7; color: #000; width: 120px;}.button5{background-
color: #555; font-size: 16px; width: 100px;}.button6{visibility: hidden; width:
100px;}.button7{background-color: #e7e7e7; color: #000; font-size: 30px; width:
50px;}.button8{background-color: #A068AD; color: #fff; width: 100px;}.button9{background-
color: 43574A; color: #fff; width: 100px;}button:hover{background:
#ff494d;}button:active{background: #f21c21;}button.disabled{cursor: default; background:
#a0a0a0;}input[type="range"]{-webkit-appearance: none; width: 100%; height: 22px; background:
#363636; cursor: pointer; margin: 0;}input[type="range"]:focus{outline:
0;}input[type="range"]::-webkit-slider-runnable-track{width: 100%; height: 2px; cursor:
pointer; background: #efefef; border-radius: 0; border: 0 solid
#efefef;}input[type="range"]::-webkit-slider-thumb{border: 1px solid rgba(0, 0, 30, 0);
height: 22px; width: 22px; border-radius: 50px; background: #ff3034; cursor: pointer; -webkit-
appearance: none; margin-top: -11.5px;}input[type="range"]:focus::-webkit-slider-runnable-
track{background: #efefef;}input[type="range"]::-moz-range-track{width: 100%; height: 2px;
cursor: pointer; background: #efefef; border-radius: 0; border: 0 solid
#efefef;}input[type="range"]::-moz-range-thumb{border: 1px solid rgba(0, 0, 30, 0); height:
22px; width: 22px; border-radius: 50px; background: #ff3034; cursor:
pointer;}input[type="range"]::-ms-track{width: 100%; height: 2px; cursor: pointer; background:
0 0; border-color: transparent; color: transparent;}input[type="range"]::-ms-fill-
lower{background: #efefef; border: 0 solid #efefef; border-radius: 0;}input[type="range"]::-
ms-fill-upper{background: #efefef; border: 0 solid #efefef; border-radius:
0;}input[type="range"]::-ms-thumb{border: 1px solid rgba(0, 0, 30, 0); height: 22px; width:
22px; border-radius: 50px; background: #ff3034; cursor: pointer; height:
2px;}input[type="range"]:focus::-ms-fill-lower{background:
#efefef;}input[type="range"]:focus::-ms-fill-upper{background: #363636;}.switch{display:
block; position: relative; line-height: 22px; font-size: 16px; height: 22px;}.switch
input{outline: 0; opacity: 0; width: 0; height: 0;}.slider{width: 50px; height: 22px; border-
radius: 22px; cursor: pointer; background-color: grey;}.slider, .slider:before{display:
inline-block; transition: 0.4s;}.slider:before{position: relative; content: ""; border-radius:
50%; height: 16px; width: 16px; left: 4px; top: 3px; background-color:
#fff;}input:checked+.slider{background-color: #ff3034;}input:checked+.slider:before{-webkit-
transform: translateX(26px); transform: translateX(26px);}select{border: 1px solid #363636;
font-size: 14px; height: 22px; outline: 0; border-radius: 5px;}.image-container{position:
absolute; top: 50px; left: 50%; margin-right: -50%; transform: translate(-50%, -50%); min-
width: 160px;}.control-container{position: relative; top: 400px; left: 50%; margin-right: -
50%; transform: translate(-50%, -50%);}.slider-container{position: relative; top: 750px;
```

```
right: 36%; margin-left: -50%; transform: translate(-50%, -50%);}.close{position: absolute;
right: 5px; top: 5px; background: #ff3034; width: 16px; height: 16px; border-radius: 100px;
color: #fff; text-align: center; line-height: 18px; cursor: pointer;}.hidden{display: none;
</style></head><body> <br/> <br/> <section class="main"> <figure> <div id="stream-container"
class="image-container"> <div class="close" id="close-stream">×</div><img id="stream" src=""/>
</div></figure> <br/> <br/> <section id="buttons"> <div id="controls" class="control-
container"> <table> <tr> <td align="center"><button class="button button6" id="get-
still">Image</button></td><td align="center"><button id="toggle-
stream">Iniciar</button></td><td></td></tr><tr> <td align="center"><button class="button
button2" onclick="fetch(document.location.origin+'/control?var=RA-
01&val=1');">&#8650;</button></td><td align="center"><button class="button button2"
onclick="fetch(document.location.origin+'/control?var=RA-01&val=2');">&#9650;</button></td><td
align="center"><button class="button button2"
onclick="fetch(document.location.origin+'/control?var=RA-01&val=3');">&#8648;</button></td><td
align="center"><button class="button button7"
onclick="fetch(document.location.origin+'/control?var=RA-
01&val=4');">&#128578;</button></td></tr><tr> <td align="center"><button class="button
button2" onclick="fetch(document.location.origin+'/control?var=RA-
01&val=5');">&#9664;</button></td><td align="center"><button class="button button2"
onclick="fetch(document.location.origin+'/control?var=RA-
01&val=6');">&#128721;</button></td><td align="center"><button class="button button2"
onclick="fetch(document.location.origin+'/control?var=RA-01&val=7');">&#9654;</button></td><td
align="center"><button class="button button7"
onclick="fetch(document.location.origin+'/control?var=RA-
01&val=8');">&#128515;</button></td></tr><tr> <td align="center"><button class="button
button5" onclick="fetch(document.location.origin+'/control?var=RA-
01&val=9');">"Hola"</button></td><td align="center"><button class="button button2"
onclick="fetch(document.location.origin+'/control?var=RA-
01&val=10');">&#9660;</button></td><td align="center"><button class="button button5"
onclick="fetch(document.location.origin+'/control?var=RA-01&val=11');">"Mola"</button></td><td
align="center"><button class="button button7"
onclick="fetch(document.location.origin+'/control?var=RA-
01&val=12');">&#x1F62E;</button></td></tr><tr> <td align="center"><button class="button
button5" onclick="fetch(document.location.origin+'/control?var=RA-
01&val=13');">Nombre?</button></td><td align="center"><button class="button button5"
onclick="fetch(document.location.origin+'/control?var=RA-01&val=14');">"RA-
01"</button></td><td align="center"><button class="button button5"
onclick="fetch(document.location.origin+'/control?var=RA-01&val=15');">"Que
tal?"</button></td><td align="center"><button class="button button7"
onclick="fetch(document.location.origin+'/control?var=RA-
01&val=16');">&#x1F631;</button></td></tr><tr> <td align="center"><button class="button
button5" onclick="fetch(document.location.origin+'/control?var=RA-
01&val=17');">"Revancha"</button></td><td align="center"><button class="button button5"
onclick="fetch(document.location.origin+'/control?var=RA-
01&val=18');">"Genial"</button></td><td align="center"><button class="button button5"
onclick="fetch(document.location.origin+'/control?var=RA-
01&val=19');">"Adios"</button></td><td align="center"><button class="button button7"
onclick="fetch(document.location.origin+'/control?var=RA-
01&val=20');">&#128577;</button></td></tr><tr> <td align="center"><button class="button
button8" onclick="fetch(document.location.origin+'/control?var=RA-
01&val=21');">PPT</button></td><td align="center"><button class="button button8"
onclick="fetch(document.location.origin+'/control?var=RA-01&val=22');">CARTAS</button></td><td
align="center"><button class="button button8"
onclick="fetch(document.location.origin+'/control?var=RA-01&val=23');">BAILE</button></td><td
align="center"><button class="button button7"
onclick="fetch(document.location.origin+'/control?var=RA-
01&val=24');">&#x1F621;</button></td></tr><tr> <td align="center"><button class="button
button9" onclick="fetch(document.location.origin+'/control?var=RA-
01&val=25');">|___|</button></td><td align="center"><button class="button button9"
onclick="fetch(document.location.origin+'/control?var=RA-01&val=26');">|_|</button></td><td
align="center"><button class="button button9"
onclick="fetch(document.location.origin+'/control?var=RA-
01&val=27');">||</button></td></table> </div><br/> </section> </section>
<script>document.addEventListener("DOMContentLoaded", function(){function b(B){let C; switch
(B.type){case "checkbox": C=B.checked ? 1 : 0; break; case "range": case "select-one":
C=B.value; break; case "button": case "submit": C="1"; break; default: return;}const
D=`${c}/control?var=${B.id}&val=${C}`; fetch(D).then((E)=>{console.log(`request to
${D}finished, status: ${E.status}`);});}var c=document.location.origin; const
e=(B)=>{B.classList.add("hidden");}, f=(B)=>{B.classList.remove("hidden");},
g=(B)=>{B.classList.add("disabled"), (B.disabled=!0);},
h=(B)=>{B.classList.remove("disabled"), (B.disabled=!1);}, i=(B, C, D)=>{D=!(null !=D) || D;
let E; "checkbox"===B.type ? ((E=B.checked), (C=!!C), (B.checked=C)) : ((E=B.value),
(B.value=C)), D && E !==C ? b(B) : !D && ("aec"===B.id ? (C ? e(v) : f(v)) : "agc"===B.id ? (C
? (f(t), e(s)) : (e(t), f(s))) : "awb_gain"===B.id ? (C ? f(x) : e(x)) :
```

```
"face_recognize"===B.id && (C ? h(n) : g(n)));};
document.querySelectorAll(".close").forEach((B)=>{B.onclick=()=>{e(B.parentNode);};}),
fetch(`${c}/status`).then(function(B){return
B.json();}).then(function(B){document.querySelectorAll(".default-action").forEach((C)=>{i(C,
B[C.id], !1);});});}); const j=document.getElementById("stream"),
k=document.getElementById("stream-container"), l=document.getElementById("get-still"),
m=document.getElementById("toggle-stream"), n=document.getElementById("face_enroll"),
o=document.getElementById("close-stream"), p=()=>{window.stop(), (m.innerHTML="Iniciar");},
q=()=>{(j.src=`${c + ":81"}/stream`), f(k), (m.innerHTML="Parar");}; (l.onclick=()=>{p(),
(j.src=`${c}/capture?_cb=${Date.now()}`), f(k);}), (o.onclick=()=>{p(), e(k);}),
(m.onclick=()=>{const B="Stop"===m.innerHTML; B ? p() : q();}), (n.onclick=()=>{b(n);}),
document.querySelectorAll(".default-action").forEach((B)=>{B.onchange=()=> b(B);}); const
r=document.getElementById("agc"), s=document.getElementById("agc_gain-group"),
t=document.getElementById("gainceiling-group"); r.onchange=()=>{b(r), r.checked ? (f(t), e(s))
: (e(t), f(s));}; const u=document.getElementById("aec"),
v=document.getElementById("aec_value-group"); u.onchange=()=>{b(u), u.checked ? e(v) : f(v);};
const w=document.getElementById("awb_gain"), x=document.getElementById("wb_mode-group");
w.onchange=()=>{b(w), w.checked ? f(x) : e(x);}; const
y=document.getElementById("face_detect"), z=document.getElementById("face_recognize"),
A=document.getElementById("framesize"); (A.onchange=()=>{b(A), 5 < A.value && (i(y, !1), i(z,
!1));}), (y.onchange=()=>{return 5 < A.value ? (alert("Please select CIF or lower resolution
before enabling this feature!"), void i(y, !1)) : void(b(y), !y.checked && (g(n), i(z,
!1)));}), (z.onchange=()=>{return 5 < A.value ? (alert("Please select CIF or lower resolution
before enabling this feature!"), void i(z, !1)) : void(b(z), z.checked ? (h(n), i(y, !0)) :
g(n));});}); </script></body></html>
)rawliteral";

static esp_err_t index_handler(httpd_req_t *req)
{
    Serial.println("index_handler en nucleo ->"+String(xPortGetCoreID()));
    httpd_resp_set_type(req, "text/html");
    return httpd_resp_send(req, (const char *)INDEX_HTML, strlen(INDEX_HTML));
}

void  Configurar_Servidor()
{
    httpd_config_t config = HTTPD_DEFAULT_CONFIG();

    httpd_uri_t index_uri = {
        .uri       = "/",
        .method    = HTTP_GET,
        .handler   = index_handler,
        .user_ctx  = NULL
    };

    httpd_uri_t status_uri = {
        .uri       = "/status",
        .method    = HTTP_GET,
        .handler   = status_handler,
        .user_ctx  = NULL
    };

    httpd_uri_t cmd_uri = {
        .uri       = "/control",
        .method    = HTTP_GET,
        .handler   = cmd_handler,
        .user_ctx  = NULL
    };

   httpd_uri_t stream_uri = {
        .uri       = "/stream",
        .method    = HTTP_GET,
        .handler   = stream_handler,
        .user_ctx  = NULL
    };

    Serial.printf("Starting web server on port: '%d'\n", config.server_port);
    if (httpd_start(&camera_httpd, &config) == ESP_OK) {
        httpd_register_uri_handler(camera_httpd, &index_uri);
        httpd_register_uri_handler(camera_httpd, &cmd_uri);
        httpd_register_uri_handler(camera_httpd, &status_uri);
    }

    config.server_port += 1;
```

```
    config.ctrl_port += 1;
    Serial.printf("Starting stream server on port: '%d'\n", config.server_port);
    if (httpd_start(&stream_httpd, &config) == ESP_OK) {
        httpd_register_uri_handler(stream_httpd, &stream_uri);
    }
}
```

2.3. funciones.h

```
#ifndef FUNCIONES_H
#define FUNCIONES_H

void Configurar_Camara();           //Configura e inicia la cámara.
void Configurar_Red_WIFI();         //Configura e inicia la red WIFI.
void mover_ojos(uint8_t modo);      //Modo 0: mueve los ojos de forma lineal, modo 1: parpadea
y los ojos aparecen en la posicion directamente.
void Mostrar_Cara();                //Dependiendo de la empoción, esta función cambia el
color, la imagen de la boca y en algunos casos las cejas.
void Agitar_Ojos ();                //Esta función hace que la posición de los ojos alterne
entre derecha e izquierda.
void Mostrar_Bateria(uint8_t num);  //Dependiendo de el parámetro de entrada, se mostrará un
icono u otro de forma intermitente 3 vecces.
void Mostrar_Carta(uint8_t num);    //Dependiendo de el dato de entrada muestra una imagen u
otra del juego de encontrar la carta.
void Mostrar_PPT(uint8_t num);      //Dependiendo de el dato de entrada muestra una imagen u
otra del juego de piedra pepel o tijera (PPT).
void Juego_Cartas();                //Primero genera un número aleatorio, si este es distinto
al previo se muestra la imagen correspondiente. Finalmente se pone la pantalla en negro.
void Juego_PPT ();                  //Primero genera un número aleatorio, si este es distinto
al previo se muestra la imagen correspondiente. Finalmente se pone la pantalla en negro.
void Accion_Aleatoria();            //Cuando es llamada esta funcion se genera un número
aleatorio de 0 a 39. Dependiendo del valor de este se moveran los ojos, o se parpadeará.
void Lectura_Serial_2a1();          //Lee los datos que el microcontrolador ESP 32 d1 mini
envia y toma las decisiones correspondientes llamando a otras funciones o cambiando variables
globales
void Boton_Pulsado_1();       //Tumbar
void Boton_Pulsado_2();       //Caminar hacia delante
void Boton_Pulsado_3();       //Erguido
void Boton_Pulsado_4();       //Emocion nomral
void Boton_Pulsado_5();       //Rotar antihorario
void Boton_Pulsado_6();       //Parar
void Boton_Pulsado_7();       //Rotar horario
void Boton_Pulsado_8();       //Emocion feliz
void Boton_Pulsado_9();       //Frase "Hola"
void Boton_Pulsado_10();      //Caminar hacia detrás
void Boton_Pulsado_11();      //Frase "Mola"
void Boton_Pulsado_12();      //Emocion sorprendido
void Boton_Pulsado_13();      //Frase "¿Como te llamas?"
void Boton_Pulsado_14();      //Frase "Yo me llamo RA-01"
void Boton_Pulsado_15();      //Frase "Que tal el dia"
void Boton_Pulsado_16();      //Emocion asustado
void Boton_Pulsado_17();      //Frase "¿Revancha?"
void Boton_Pulsado_18();      //Frase "Genial"
void Boton_Pulsado_19();      //Frase "Adios"
void Boton_Pulsado_20();      //Emocion triste
void Boton_Pulsado_21();      //Juego Piedra Papel o Tijera, PPT
void Boton_Pulsado_22();      //Juego Cartas
void Boton_Pulsado_23();      //Baile
void Boton_Pulsado_24();      //Emocion enfadado
void Boton_Pulsado_25();      //Pinza abierta
void Boton_Pulsado_26();      //Pinza abierta a la mitad de su recorrido
void Boton_Pulsado_27();      //Cerrar pinza y mirar abajo

#endif
```

2.4. funciones.cpp

```
#include <WiFi.h>

#include <soc/rtc_cntl_reg.h>

#include <dl_lib_matrix3d.h>      //Librería para el funcionamiento de la cámara
#include <img_converters.h>

#include <esp_http_server.h>      //Librería para la creación del portal de control

#include <Adafruit_GFX.h>         //Libreríias para la pantalla
#include <Adafruit_SSD1351.h>
#include <SPI.h>

#include "app_httpd.h"            //Librerías propias
#include "funciones.h"
#include "imagenes.h"

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

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

//colores
#define NEGRO           0x0000
#define AZUL            0x001F
#define ROJO             0xF800
#define VERDE           0x07E0

#define MORADO         0xF81F
#define AMARILLO          0xFFE0
#define BLANCO           0xFFFF

#define incremento 1

extern HardwareSerial Serial_1a2;
extern Adafruit_SSD1351 pantalla;

// SSID y contraseña para acceder a la red WIFI creada
extern const char* ssid1;
extern const char*   password;

extern uint8_t pos_ojos_actual_x;
extern uint8_t pos_ojos_actual_y;
extern uint8_t pos_ojos_deseada_x;
extern uint8_t pos_ojos_deseada_y;

extern uint8_t emocion;
extern uint8_t emocion_previa;

uint16_t color=BLANCO;

extern bool juego_cartas;
extern bool juego_PPT;

extern bool agitar_ojos;

bool cejas=false;
bool derecha = false;
```

```
uint8_t longitud_de_la_linea = 40;
uint8_t  contador_agitar_ojos = 0;
uint8_t num_random_previo;


void  Configurar_Camara()   //Configura e inicia la cámara.
{
  camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0; ////zxxxxxxxxxxxxxxx
  config.ledc_timer = LEDC_TIMER_0;     ///xxxxxxxxxxxxxxxxx
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
  config.pin_d6 = Y8_GPIO_NUM;
  config.pin_d7 = Y9_GPIO_NUM;
  config.pin_xclk = XCLK_GPIO_NUM;
  config.pin_pclk = PCLK_GPIO_NUM;
  config.pin_vsync = VSYNC_GPIO_NUM;
  config.pin_href = HREF_GPIO_NUM;
  config.pin_sscb_sda = SIOD_GPIO_NUM;
  config.pin_sscb_scl = SIOC_GPIO_NUM;
  config.pin_pwdn = PWDN_GPIO_NUM;
  config.pin_reset = RESET_GPIO_NUM;
  config.xclk_freq_hz = 20000000;
  config.pixel_format = PIXFORMAT_JPEG;
  //init with high specs to pre-allocate larger buffers
  if(psramFound()){
    config.frame_size = FRAMESIZE_QVGA;
    config.jpeg_quality = 10;
    config.fb_count = 2;
  } else {
    config.frame_size = FRAMESIZE_QVGA;
    config.jpeg_quality = 12;
    config.fb_count = 1;
  }

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

  //drop down frame size for higher initial frame rate
  sensor_t * s = esp_camera_sensor_get();
  s->set_framesize(s, FRAMESIZE_QVGA);
  s->set_vflip(s, 0);
  s->set_hmirror(s, 1);
}
void  Configurar_Red_WIFI()   //Configura e inicia la red WIFI.
{
  WiFi.softAP(ssid1,   password);
  IPAddress myIP = WiFi.softAPIP();
  Serial.print("AP IP address: ");
  Serial.println(myIP);
}

void mover_ojos(uint8_t modo) //Modo 0: mueve los ojos de forma lineal, modo 1: parpadea y los
ojos aparecen en la posicion directamente.
{
  switch (modo)
  {
    case 0:
      if(pos_ojos_actual_y < pos_ojos_deseada_y)    //MOVIMIENTO VERTICAL (BAJAR OJOS)
      {
        pantalla.drawFastHLine(pos_ojos_actual_x-20, pos_ojos_actual_y-20, 40, NEGRO);
        pantalla.drawFastHLine(pos_ojos_actual_x+44, pos_ojos_actual_y-20, 40, NEGRO);
        pos_ojos_actual_y = pos_ojos_actual_y+incremento;
        if(pos_ojos_actual_y > 60) longitud_de_la_linea = 40 + 60 - pos_ojos_actual_y;
        if(pos_ojos_actual_y <= 60)
```

```
        {
          pantalla.drawFastHLine(pos_ojos_actual_x-20, pos_ojos_actual_y+19, 40, color);
          pantalla.drawFastHLine(pos_ojos_actual_x+44, pos_ojos_actual_y+19, 40, color);
        }
      }
      if(pos_ojos_actual_y > pos_ojos_deseada_y)  //MOVIMIENTO VERTICAL (SUBIR OJOS)
      {
        if(pos_ojos_actual_y <= 60)
        {
          pantalla.drawFastHLine(pos_ojos_actual_x-20, pos_ojos_actual_y+19, 40, NEGRO);
          pantalla.drawFastHLine(pos_ojos_actual_x+44, pos_ojos_actual_y+19, 40, NEGRO);
        }
        pos_ojos_actual_y = pos_ojos_actual_y-incremento;
        if(pos_ojos_actual_y >= 60) longitud_de_la_linea = 40 + 60 - pos_ojos_actual_y;
        if(pos_ojos_actual_y >= 20)
        {
          pantalla.drawFastHLine(pos_ojos_actual_x-20, pos_ojos_actual_y-20, 40, color);
          pantalla.drawFastHLine(pos_ojos_actual_x+44, pos_ojos_actual_y-20, 40, color);
        }

      }

      if(pos_ojos_actual_x>pos_ojos_deseada_x)    //MOVIMIENTO HORIZONTAL (IZQUIERDA)
      {
        pantalla.drawFastVLine(pos_ojos_actual_x+19, pos_ojos_actual_y-20,
longitud_de_la_linea, NEGRO);
        pantalla.drawFastVLine(pos_ojos_actual_x+83, pos_ojos_actual_y-20,
longitud_de_la_linea, NEGRO);
        pos_ojos_actual_x = pos_ojos_actual_x-incremento;
        if(pos_ojos_actual_x >= 20)
        {
          pantalla.drawFastVLine(pos_ojos_actual_x-20, pos_ojos_actual_y-20,
longitud_de_la_linea, color);
        }
          pantalla.drawFastVLine(pos_ojos_actual_x+44, pos_ojos_actual_y-20,
longitud_de_la_linea, color);
      }

      if(pos_ojos_actual_x < pos_ojos_deseada_x)    //MOVIMIENTO HORIZONTAL (DERECHA)
      {
        pantalla.drawFastVLine(pos_ojos_actual_x-20, pos_ojos_actual_y-20,
longitud_de_la_linea, NEGRO);
        pantalla.drawFastVLine(pos_ojos_actual_x+44, pos_ojos_actual_y-20,
longitud_de_la_linea, NEGRO);
        pos_ojos_actual_x = pos_ojos_actual_x+incremento;
        pantalla.drawFastVLine(pos_ojos_actual_x+19, pos_ojos_actual_y-20,
longitud_de_la_linea, color);
        if(pos_ojos_actual_x <= 44)
        {
          pantalla.drawFastVLine(pos_ojos_actual_x+83, pos_ojos_actual_y-20,
longitud_de_la_linea, color);
        }
      }
    break;

    case 1:
      if(pos_ojos_actual_x != pos_ojos_deseada_x || pos_ojos_actual_y != pos_ojos_deseada_y)
      {
        if(pos_ojos_actual_y > 60) longitud_de_la_linea = 40 + 60 - pos_ojos_actual_y;
        {
          pantalla.fillRect(0,0,128,80,NEGRO);
          pos_ojos_actual_x = pos_ojos_deseada_x;
          pos_ojos_actual_y = pos_ojos_deseada_y;
          pantalla.fillRect(pos_ojos_actual_x, pos_ojos_actual_y, 40, longitud_de_la_linea,
color);
          pantalla.fillRect(pos_ojos_actual_x+64, pos_ojos_actual_y, 40, longitud_de_la_linea,
color);
        }
      }
    break;
  }
}
```

```
void Mostrar_Cara()   //Dependiendo de la empoción, esta función cambia el color, la imagen de
la boca y en algunos casos las cejas.
{
  if (emocion_previa != emocion)
  {
    pantalla.fillRect(pos_ojos_actual_x-32, pos_ojos_actual_y-36, 128, 16, NEGRO);
    switch(emocion)
    {
      case 0:
        color=BLANCO;
        pantalla.fillRect(0, 80, 128, 48, NEGRO);
        pantalla.drawBitmap(0, 82, boca_normal, 128, 48, color);
        cejas=false;
        //Serial.print("Emocion: normal");
      break;

      case 1:
        color=AMARILLO;
        pantalla.fillRect(0, 80, 128, 48, NEGRO);
        pantalla.drawBitmap(0, 82, boca_contento, 128, 48, color);
        cejas=false;
        //Serial.print("Emocion: contento");
      break;

      case 2:
        color=VERDE;
        pantalla.fillRect(0, 80, 128, 48, NEGRO);
        pantalla.drawBitmap(0, 82, boca_sorprendido, 128, 48, color);
        pantalla.drawBitmap(pos_ojos_actual_x-20, pos_ojos_actual_y-36, cejas_sorprendido,
104, 16, color);
        cejas=true;
        //Serial.print("Emocion: sorprendido");
      break;

      case 3:
        color=MORADO;
        pantalla.fillRect(0, 80, 128, 48, NEGRO);
        pantalla.drawBitmap(0, 82, boca_asustado, 128, 48, color);
        cejas=false;
        //Serial.print("Emocion: asustado");
      break;

      case 4:
        color=AZUL;
        pantalla.fillRect(0, 80, 128, 48, NEGRO);
        pantalla.drawBitmap(0, 82, boca_triste, 128, 48, color);
        cejas=false;
        //Serial.print("Emocion: triste");
      break;

      case 5:
        color=ROJO;
        pantalla.fillRect(0, 80, 128, 48, NEGRO);
        pantalla.drawBitmap(0, 82, boca_enfadado, 128, 48, color);
        pantalla.drawBitmap(pos_ojos_actual_x-20, pos_ojos_actual_y-36, cejas_enfadado, 104,
16, color);
        cejas=true;
        //Serial.print("Emocion: enfadado");
      break;
    }
    emocion_previa=emocion;
    pantalla.fillRect(pos_ojos_actual_x-20, pos_ojos_actual_y-20, 40, longitud_de_la_linea,
color);
    pantalla.fillRect(pos_ojos_actual_x+44, pos_ojos_actual_y-20, 40, longitud_de_la_linea,
color);
  }
  if(cejas)
  {
    if(pos_ojos_actual_x != pos_ojos_deseada_x || pos_ojos_actual_y != pos_ojos_deseada_y)
    {
      pantalla.fillRect(pos_ojos_actual_x-32, pos_ojos_actual_y-36, 128, 16, NEGRO);
      if(emocion==2) {pantalla.drawBitmap(pos_ojos_actual_x-20, pos_ojos_actual_y-36,
cejas_sorprendido, 104, 16, color);}
```

```
      if(emocion==5) {pantalla.drawBitmap(pos_ojos_actual_x-20, pos_ojos_actual_y-36,
cejas_enfadado, 104, 16, color);}
    }
  }
  mover_ojos(0);
}

void Agitar_Ojos () //Esta función hace que la posición de los ojos alterne entre derecha e
izquierda.
{
  if (pos_ojos_actual_x < 10) pos_ojos_deseada_x = 60; //derecha=false;
  if( pos_ojos_actual_x>54) pos_ojos_deseada_x = 5;  //derecha=true;
  contador_agitar_ojos++;
  if(contador_agitar_ojos>200)
  {
    pos_ojos_deseada_x = 32;
    pos_ojos_deseada_y = 40;
    contador_agitar_ojos=0;
    agitar_ojos=false;
  }
}

void Mostrar_Bateria(uint8_t num) //Dependiendo de el parámetro de entrada, se mostrará un
icono u otro de forma intermitente 3 vecces.
{
  switch(num)
  {
    case 0:
      pantalla.drawRGBBitmap(112,0,bateria_baja,16,32);
    break;
    case 1:
      pantalla.drawRGBBitmap(112,0,bateria_media,16,32);
    break;
    case 2:
      pantalla.drawRGBBitmap(112,0,bateria_cargada,16,32);
    break;
    case 3:
      pantalla.drawRGBBitmap(112,0,bateria_cargando,16,32);
    break;
  }
}

void Mostrar_Carta (uint8_t num)  //Dependiendo de el dato de entrada muestra una imagen u
otra del juego de encontrar la carta.
{
  switch(num)
  {
    case 0:
       pantalla.drawRGBBitmap(16,16,cartas_elefante,96,96);
    break;
    case 1:
       pantalla.drawRGBBitmap(16,16,cartas_gato,96,96);
    break;
    case 2:
       pantalla.drawRGBBitmap(16,16,cartas_leon,96,96);
    break;
    case 3:
       pantalla.drawRGBBitmap(16,16,cartas_mono,96,96);
    break;
    case 4:
       pantalla.drawRGBBitmap(16,16,cartas_pajaro,96,96);
    break;
    case 5:
       pantalla.drawRGBBitmap(16,16,cartas_perro,96,96);
    break;
    case 6:
       pantalla.drawRGBBitmap(16,16,cartas_pez,96,96);
    break;
    case 7:
       pantalla.drawRGBBitmap(16,16,cartas_raton,96,96);
    break;
    case 8:
       pantalla.drawRGBBitmap(16,16,cartas_tigre,96,96);
    break;
```

```
    case 9:
       pantalla.drawRGBBitmap(16,16,cartas_tortuga,96,96);
    break;
  }
}

void Mostrar_PPT (uint8_t num)  //Dependiendo de el dato de entrada muestra una imagen u otra
del juego de piedra pepel o tijera (PPT).
{
  switch(num)
  {
    case 0:
       pantalla.drawRGBBitmap(16,16,PPT_piedra,96,96);
    break;
    case 1:
       pantalla.drawRGBBitmap(16,16,PPT_papel,96,96);
    break;
    case 2:
       pantalla.drawRGBBitmap(16,16,PPT_tijera,96,96);
    break;
  }
}

void Juego_Cartas()   //Primero genera un número aleatorio, si este es distinto al previo se
muestra la imagen correspondiente. Finalmente se pone la pantalla en negro.
{
  Serial.print("Juego cartas, carta: ");
  uint8_t num_random=esp_random() % 10;
  Serial.println(num_random);
  if(num_random!=num_random_previo)
  {
    pantalla.fillRect(0, 0, 128, 128, NEGRO);
    Mostrar_Carta (num_random);
    delay(5000);
    pantalla.fillRect(0, 0, 128, 128, NEGRO);
  }
  num_random_previo=num_random;     //Se almacena en una variable global el numero previo
  emocion_previa=99;                //Dado que la emovión y la emoción previa son distintas,
una vez finalice el juego se volverá a mostrar la cara del robot
}

void Juego_PPT ()     //Primero genera un número aleatorio, si este es distinto al previo se
muestra la imagen correspondiente. Finalmente se pone la pantalla en negro.
{
  Serial.print("Juego PPT: ");
  delay(1000);
  uint8_t num_random=esp_random() % 3;
  //Serial.println(num_random);
  pantalla.fillRect(0, 0, 128, 128, NEGRO);
  pantalla.drawRGBBitmap(16,16,PPT,96,96);
  delay(500);
  Serial_1a2.write(19);
  delay(2000);
  pantalla.fillRect(0, 0, 128, 128, NEGRO);
  Mostrar_PPT(num_random);
  delay(4000);
  pantalla.fillRect(0, 0, 128, 128, NEGRO);
  emocion_previa=99;                //Dado que la emovión y la emoción previa son distintas,
una vez finalice el juego se volverá a mostrar la cara del robot
}

void Accion_Aleatoria()   //Cuando es llamada esta funcion se genera un número aleatorio de 0
a 39. Dependiendo del valor de este se moveran los ojos, o se parpadeará.
{
  uint8_t num_random=esp_random() %40;
   if(num_random == 0)                      //Miar abajo izquierda
  {
    pos_ojos_deseada_x=20;
    pos_ojos_deseada_y=60;
  }
  else if(num_random == 1)                    //Miar izquierda
  {
    pos_ojos_deseada_x=20;
    pos_ojos_deseada_y=40;
```

```
  }
  else if(num_random == 2)                    //Miar arriba izquierda
  {
    pos_ojos_deseada_x=20;
    pos_ojos_deseada_y=20;
  }
  else if(num_random == 3)                    //Miar abajo
  {
    pos_ojos_deseada_x=32;
    pos_ojos_deseada_y=60;
  }
  else if(num_random >=4 && num_random <= 13)  //Miar centro
  {
    pos_ojos_deseada_x=32;
    pos_ojos_deseada_y=40;
  }
  else if(num_random == 14)                    //Miar arriba
  {
    pos_ojos_deseada_x=32;
    pos_ojos_deseada_y=20;
  }
  else if(num_random == 15)                    //Miar abajo derecha
  {
    pos_ojos_deseada_x=44;
    pos_ojos_deseada_y=60;
  }
  else if(num_random == 16)                    //Miar derecha
  {
    pos_ojos_deseada_x=44;
    pos_ojos_deseada_y=40;
  }
  else if(num_random == 17)                    //Miar arriba derecha
  {
    pos_ojos_deseada_x=44;
    pos_ojos_deseada_y=20;
  }
  else if(num_random > 17)                    //Parpadear
  {
    pantalla.fillRect(pos_ojos_actual_x-20, pos_ojos_actual_y-20, 104, 40, NEGRO);
    delay(250);
    pantalla.fillRect(pos_ojos_actual_x-20, pos_ojos_actual_y-20, 40, longitud_de_la_linea,
color);
    pantalla.fillRect(pos_ojos_actual_x+44, pos_ojos_actual_y-20, 40, longitud_de_la_linea,
color);
  }
}

void Lectura_Serial_2a1()   //Lee los datos que el microcontrolador ESP 32 d1 mini envia y
toma las decisiones correspondientes llamando a otras funciones o cambiando variables globales
{
  uint8_t dato_recibido = Serial_1a2.read();
  if (dato_recibido != 255)
  {
    Serial.print("dato_recibido: ");
    Serial.println(dato_recibido);
    switch (dato_recibido)
    {
      case 1:   //Mostrar bateria cargando
        Serial.println("Mostrar bateria cargando");
        Mostrar_Bateria(3);
        delay(500);
        pantalla.fillRect(112, 0, 16, 32, NEGRO);
        delay(500);
        Mostrar_Bateria(3);
        delay(500);
        pantalla.fillRect(112, 0, 16, 32, NEGRO);
        delay(500);
        Mostrar_Bateria(3);
        delay(500);
        pantalla.fillRect(112, 0, 16, 32, NEGRO);
      break;
      case 2:   //Mostrar bateria baja
        Serial.println("Mostrar bateria baja");
        Mostrar_Bateria(1);
```

```
        delay(500);
        pantalla.fillRect(112, 0, 16, 32, NEGRO);
        delay(500);
        Mostrar_Bateria(1);
        delay(500);
        pantalla.fillRect(112, 0, 16, 32, NEGRO);
        delay(500);
        Mostrar_Bateria(1);
        delay(500);
        pantalla.fillRect(112, 0, 16, 32, NEGRO);
      break;
      case 3:   //Se ha mantenido la mano apollada en la cabeza, emocion = normal, ojos arriba
moviendose de lado a lado
        emocion = 0;
        pos_ojos_deseada_x = 60;
        pos_ojos_deseada_y=15;
        agitar_ojos=true;
      break;
      case 4:   //Se ha mantenido la mano apollada en la parte trasera, emocion = enfadado,
ojos centrados
        emocion = 5;
        pos_ojos_deseada_x=32;
        pos_ojos_deseada_y=40;
      break;
      case 5:   //Se ha acariciado de delante hacia detrás, emocion = feliz, ojos arriba
moviendose de lado a lado
        emocion = 1;
        pos_ojos_deseada_x = 60;
        pos_ojos_deseada_y=15;
        agitar_ojos=true;
      break;
      case 6:   //Se ha acariciado de detrás hacia delante, emocion = sorprendido, ojos
centrados
      emocion = 2;
        pos_ojos_deseada_x=32;
        pos_ojos_deseada_y=40;
      break;
      case 7:   //Se ha levantado el robot y se asusta
      emocion = 3;
        pos_ojos_deseada_x=32;
        pos_ojos_deseada_y=70;
      break;
    }
  }
}

void Boton_Pulsado_1()  //Tumbar
{
  Serial.println("Boton_Pulsado_1");
  Serial_1a2.write(1);
}

void Boton_Pulsado_2()  //Caminar hacia delante
{
  Serial.println("Boton_Pulsado_2");
  Serial_1a2.write(3);
}

void Boton_Pulsado_3()  //Erguido
{
  Serial.println("Boton_Pulsado_3");
  Serial_1a2.write(2);
}

void Boton_Pulsado_4()  //Emocion nomral
{
  Serial.println("Boton_Pulsado_4");
  emocion=0;
}

void Boton_Pulsado_5()  //Rotar antihorario
{
  Serial.println("Boton_Pulsado_5");
  pos_ojos_deseada_x=64;
```

```
  pos_ojos_deseada_y=40;
  Serial_1a2.write(5);
}
void Boton_Pulsado_6()  //Parar
{
  Serial.println("Boton_Pulsado_6");
  pos_ojos_deseada_x=32;
  pos_ojos_deseada_y=40;
  Serial_1a2.write(7);
}
void Boton_Pulsado_7()  //Rotar horario
{
  Serial.println("Boton_Pulsado_7");
  pos_ojos_deseada_x=0;
  pos_ojos_deseada_y=40;
  Serial_1a2.write(6);
}
void Boton_Pulsado_8()  //Emocion feliz
{
  Serial.println("Boton_Pulsado_8");
  emocion=1;
}
void Boton_Pulsado_9()  //Frase "Hola"
{
  Serial.println("Boton_Pulsado_9");
  Serial_1a2.write(11);
}
void Boton_Pulsado_10() //Caminar hacia detrás
{
  Serial.println("Boton_Pulsado_10");
  pos_ojos_deseada_x=64;
  pos_ojos_deseada_y=20;
  Serial_1a2.write(4);
}
void Boton_Pulsado_11() //Frase "Mola"
{
  Serial.println("Boton_Pulsado_11");
  Serial_1a2.write(12);
}
void Boton_Pulsado_12() //Emocion sorprendido
{
  Serial.println("Boton_Pulsado_12");
  emocion=2;
}
void Boton_Pulsado_13() //Frase "¿Como te llamas?"
{
  Serial.println("Boton_Pulsado_13");
  Serial_1a2.write(13);
}
void Boton_Pulsado_14() //Frase "Yo me llamo RA-01"
{
  Serial.println("Boton_Pulsado_14");
  Serial_1a2.write(14);
}
void Boton_Pulsado_15() //Frase "Que tal el dia"
{
  Serial.println("Boton_Pulsado_15");
  Serial_1a2.write(15);
}
void Boton_Pulsado_16() //Emocion asustado
{
  Serial.println("Boton_Pulsado_16");
  emocion=3;
}
void Boton_Pulsado_17() //Frase "¿Revancha?"
{
  Serial.println("Boton_Pulsado_17");
  Serial_1a2.write(16);
}
void Boton_Pulsado_18() //Frase "Genial"
{
  Serial.println("Boton_Pulsado_18");
  Serial_1a2.write(17);
}
```

```
void Boton_Pulsado_19() //Frase "Adios"
{
  Serial.println("Boton_Pulsado_19");
  Serial_1a2.write(18);
}
void Boton_Pulsado_20() //Emocion triste
{
  Serial.println("Boton_Pulsado_20");
  emocion = 4;
}
void Boton_Pulsado_21() //Juego Piedra Papel o Tijera, PPT
{
  Serial.println("Boton_Pulsado_21");
  juego_PPT=!juego_PPT;
  if(juego_PPT) juego_cartas=false;
}
void Boton_Pulsado_22() //Juego Cartas
{
  Serial.println("Boton_Pulsado_22");
  juego_cartas=!juego_cartas;
  if(juego_cartas) juego_PPT=false;
}
void Boton_Pulsado_23() //Baile
{
  Serial.println("Boton_Pulsado_23");
  Serial_1a2.write(20);
}
void Boton_Pulsado_24() //Emocion enfadado
{
  Serial.println("Boton_Pulsado_24");
  emocion = 5;
}
void Boton_Pulsado_25() //Pinza abierta
{
  Serial.println("Boton_Pulsado_25");
  Serial_1a2.write(8);
  pos_ojos_deseada_x=32;
  pos_ojos_deseada_y=40;
}
void Boton_Pulsado_26() //Pinza abierta a la mitad de su recorrido
{
  Serial.println("Boton_Pulsado_26");
  pos_ojos_deseada_x=32;
  pos_ojos_deseada_y=50;
  Serial_1a2.write(9);
}
void Boton_Pulsado_27() //Cerrar pinza y mirar abajo
{
  Serial.println("Boton_Pulsado_27");
  Serial_1a2.write(10);
  pos_ojos_deseada_x=32;
  pos_ojos_deseada_y=60;
}
```

2.5. imagenes.h

Debido a la extensión de este archivo, no se ha puesto aquí su código.
El código y los modelos 3D de las piezas se pueden encontrar en el siguiente enlace:
https://github.com/ErnestFB/RA-01

3. ESP_32_D1_mini.ino

```
#include <ESP32Servo.h>

#include <Adafruit_VL53L0X.h>
#include <Wire.h>

#include <Adafruit_NeoPixel.h>

#include <DFRobotDFPlayerMini.h>

#include "funciones_2.h"

#define velocidad_movimiento_pinza 1

TaskHandle_t loop_2;

extern HardwareSerial Serial_2a1;
extern DFRobotDFPlayerMini DFPlayer;

uint8_t dato_recibido = 1;
uint8_t posicion =1;
bool sen_cap1 = false, sen_cap2 = false;
bool detener_robot = false;

void setup()
{
  Inicio_RA_01();
  delay(2500);
  xTaskCreatePinnedToCore(Loop_2,    //Nombre de la función
                          "loop_2", //Nombre, puede ser cualquier cosa
                          5000,     //Tamaño de la pila
                          NULL,     //Parámetro que le quiera pasar a la tarea
                          1,        //Prioridad
                          &loop_2,   //Nombre de la tarea
                          0);       //Nucleo
}

void loop()
{
  dato_recibido = Lectura_Serial_2a1();
  if(dato_recibido>10 && dato_recibido <20)
  {
    Serial.print("Reproducir_audio: ");
    Serial.println(dato_recibido-10);
    DFPlayer.play(dato_recibido-10);
  }
  Lectura_Bateria();
  Lectura_Sensores_Distancia();
  Tacto();
  delay(5);
}

void Loop_2(void *parameter) //Se ejecuta en el núcelo 0 **********************************
{
  Iniciar_Motores();
  while(1)
  {
    if(detener_robot)
    {
      dato_recibido=2;
      detener_robot=false;
    }
    else
    {
      if(dato_recibido <= 7) Movimiento_Patas(dato_recibido);
      else if(dato_recibido > 7 && dato_recibido <=10) Movimiento_Pinza(dato_recibido,
velocidad_movimiento_pinza);
    }
    delay(5);
    yield();
  }
}
```

3.1. funciones_2.h

```
#ifndef FUNCIONES_2_H
#define FUNCIONES_2_H

void Luces_Onda(int R, int G, int B);
//Enciende y apaga los LEDs en forma de onda
void Luces_Color_Fijo (int R, int G, int B);
//Enciende todos los LEDs a la vez y los mantiene encendido
void Inicio_Sensores_Distancia();
//Inicia los sensores de distancia.
void Sensor_Cap1_Activado();
//Rutina de interrupción del sensor capacitivo frontal
void Sensor_Cap2_Activado();
//Rutina de interrupción del sensor capacitivo trasero
void Inicio_RA_01();
//Iniciala cadena de LEDs, las comunicaciones Serial, los sensores de distancia, el módulo de
reproducción de MP3 y las interrupciones de los sensores capacitivos.
void Iniciar_Motores();
//Coloca los motores en la posición inicial, una por una, realizando movimietnos bruscos en
caso de que no se hayan colocado coorectamente en un inicio.
void Mover_Patas (uint8_t n_pos_inicial, uint8_t n_pos_final, const double tiempo);
//Los datos de entrada de esta función es el número de posición inicial y final y el tiempo
que transcurrirá entre los distintos movimientos. Por medio de varios bucles, realiza los
movimietnso correspondietes entre las posiciones indicadas.
void Movimiento_Patas(uint8_t dato_recibido);
//Teniendo como parametro el dato recibido que el movimiento final deseado, esta función se
encargará de que los movimientos realizados sena los correctos, realizando las transiciones
correspondientes.
void Movimiento_Pinza(uint8_t dato_recibido, const double tiempo);
//Dado el dato recibido que indica la posición final y el tiempo, esta función se encargará
de mover la pinza desde la posición actual hasta la deseada de forma suave.
uint8_t Lectura_Serial_2a1();
//Lee los datos que el microcontrolador ESP 32 cam envia.
void Lectura_Sensores_Distancia();
//Realiza la medida de los  sensores de distancia y cambia la emoción y cambia el esttado de
movimiento cuando detecta que el robot se va a caer o que ha sido levantado.
void Lectura_Bateria();
//Realiza la lectura de la tensión de la batería. Si esta es baja envia un mensaje al otro
microcontrolador. Si se deteta que se  ha conectado el cargador envia otro dato.
void Funcion_Sen_1();
//Esta función es activada por la rutina de interrupción del sensor capacitivo frontal. Sirve
para detectar cuando se toca el robot.
void Funcion_Sen_2();
//Esta función es activada por la rutina de interrupción del sensor capacitivo trasero. Sirve
para detectar cuando se toca el robot.
void Tacto();
//Esta función detecta si el robot ha sido acariciado de deatrás a delante, de delante a
detrás o si se ha mantenido la mano apoyada en sobre el robot. Dependiendo de como haya sido
el contacto con el robot envia un dato u otro al microcontrolador ESP 32 cam que recibe el
mensaje y reacciona al mismo modificando la expresión facila del robot y la posición de sus
ojos.

#endif
```

3.2. funciones_2.cpp

```
#include <ESP32Servo.h>

#include <Adafruit_VL53L0X.h>
#include <Wire.h>

#include <Adafruit_NeoPixel.h>

#include <DFRobotDFPlayerMini.h>

#define Servo__Pin_1 19
#define Servo__Pin_2 23
#define Servo__Pin_3 5
#define Servo__Pin_4 13
#define Servo__Pin_5 2
#define Servo__Pin_6 16
```

```
#define Servo__Pin_7 17
#define Servo__Pin_8 21
#define Servo__Pin_9 22

#define XSHUT_Sen_Dist_D 27
#define XSHUT_Sen_Dist_F 25
#define XSHUT_Sen_Dist_I 32

#define I2C_SDA 4
#define I2C_SCL 15

#define pin_Sen_cap1 39
#define pin_Sen_cap2 36

#define pin_Lectura_Bat 34

#define pin_Neopixel 14

#define RX_MP3 35
#define TX_MP3 33

#define RX_2a1 26
#define TX_2a1 18

#define velocidad_movimiento_1 1
#define velocidad_movimiento_R 2

#define NUM_Motores 8

//                        Levantar    CoDF    CoIT    CoIF    CoDT    CoPriPas
B_Delante(14->18)  B_Detras(18->22)
//                         0   1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39
const uint8_t pos_if_r[] = { 47, 47, 47, 47, 47, 35, 22, 47, 72, 47, 22, 47, 72, 47, 22, 47,
97,147, 97, 47, 97,147, 97, 47,147,147,147, 97, 47, 22, 22, 22, 35, 47, 72, 72, 72, 60, 47,
47};
const uint8_t pos_df_r[] =
{156,156,156,144,131,131,131,156,181,156,131,156,180,156,131,156,106, 56,106,156,106,
56,106,156, 56,106,156,156,156,131,144,156,156,156,180,168,156,156,156,156};
const uint8_t pos_it_r[] = {137,137,137,149,162,162,162,137,112,137,162,137,112,137,162,137,
87, 37, 87,137, 87, 37, 87,137, 37,
87,137,137,137,162,149,137,137,137,112,124,137,137,137,137};
const uint8_t pos_dt_r[] = { 32, 32, 32, 32, 32, 44, 57, 32,  7, 32, 57, 32,  7, 32, 57, 32,
82,132, 82, 32, 82,132, 82, 32,132,132,132, 82, 32, 57, 57, 57, 44, 32,  7,  7,  7, 19, 32,
32};

const uint8_t pos_if_v[] =
{100,100,170,170,170,145,170,145,170,170,170,170,170,145,170,170,120,170,170,170,170,170,120,1
70,170,170,170,120,170,170,170,170,145,170,170,170,170,145,170,100};
const uint8_t pos_df_v[] = { 80, 10, 10, 35, 10, 10, 10, 10, 10, 35, 10, 35, 10, 10, 10, 10,
10, 10, 60, 10, 60, 10, 10, 10, 10, 60, 10, 10, 10, 10, 35, 10, 10, 10, 10, 35, 10, 10, 10,
80};
const uint8_t pos_it_v[] = { 90,
90,160,135,160,160,160,160,160,135,160,135,160,160,160,160,160,160,110,160,110,160,160,160,160
,110,160,160,160,160,135,160,160,160,160,135,160,160,160, 90};
const uint8_t pos_dt_v[] = { 90, 15, 15, 15, 15, 40, 15, 40, 15, 15, 15, 15, 15, 40, 15, 15,
65, 15, 15, 15, 15, 15, 65, 15, 15, 15, 15, 65, 15, 15, 15, 15, 40, 15, 15, 15, 15, 40, 15,
90};

uint8_t pos_pinza_actual=50;
uint8_t contador=0;
uint8_t dato_recibido_previo=0;
uint8_t contador_sen_1 = 0;
uint8_t contador_sen_2 = 0;
uint8_t contador_sensor_previo=150;

extern uint8_t posicion;

uint16_t nivel_bateria_previo=1000;

bool sen_1_previo = false;
bool sen_2_previo = false;

extern bool detener_robot;
```

```
extern bool sen_cap1, sen_cap2;

Servo Servo_IF_R;  //1  Izquierda frontal rotativo
Servo Servo_DF_R;  //2  Derecha frontal rotativo
Servo Servo_IT_R;  //3  Izquierda trasero rotativo
Servo Servo_DT_R;  //4  Derecha trasero rotativo
Servo Servo_IF_V;  //5  Izquierda frontal vertical
Servo Servo_DF_V;  //6  derecha frontal vertical
Servo Servo_IT_V;  //7  Izquierda trasero vertical
Servo Servo_DT_V;  //8  Derecha trasero verticcal
Servo Servo_Pinza; //9  Pinza


Adafruit_VL53L0X Sen_Dist_T = Adafruit_VL53L0X();
Adafruit_VL53L0X Sen_Dist_D = Adafruit_VL53L0X();
Adafruit_VL53L0X Sen_Dist_F = Adafruit_VL53L0X();
Adafruit_VL53L0X Sen_Dist_I = Adafruit_VL53L0X();


Adafruit_NeoPixel LED_Chain = Adafruit_NeoPixel (16, //Number of leds
                                                 pin_Neopixel,  //Pin
                                                 NEO_GRB + NEO_KHZ800);

HardwareSerial SerialDFPlayer(1);
DFRobotDFPlayerMini DFPlayer;

HardwareSerial Serial_2a1(2);


void Luces_Onda (int R, int G, int B) //Enciende y apaga los LEDs en forma de onda.
{
  LED_Chain.setBrightness(20);
  for (int i=0; i<8; i++)
  {
    LED_Chain.setPixelColor(i,R,G,B); // Nº pixel, R,G,B
    LED_Chain.setPixelColor(15-i,R,G,B);
    LED_Chain.show();
    delay(100);
  }
  for (int i=0; i<8; i++)
  {
    LED_Chain.setPixelColor(i,0,0,0); // Nº pixel, R,G,B
    LED_Chain.setPixelColor(15-i,0,0,0);
    LED_Chain.show();
    delay(100);
  }
  LED_Chain.clear();
}

void Luces_Color_Fijo (int R, int G, int B) //Enciende todos los LEDs a la vez y los mantiene
encendido.
{
  LED_Chain.setBrightness(20);
  for (int i=0; i<8; i++)
  {
    LED_Chain.setPixelColor(i,R,G,B); // Nº pixel, R,G,B
    LED_Chain.setPixelColor(15-i,R,G,B);
    LED_Chain.show();
  }
  LED_Chain.show();
}

void Inicio_Sensores_Distancia()  //Inicia los sensores de distancia.
{
  Wire.begin(I2C_SDA, I2C_SCL);
  Serial.println("3");
  pinMode(XSHUT_Sen_Dist_D, OUTPUT);
  pinMode(XSHUT_Sen_Dist_F, OUTPUT);
  pinMode(XSHUT_Sen_Dist_I, OUTPUT);
  digitalWrite(XSHUT_Sen_Dist_D,LOW);
  digitalWrite(XSHUT_Sen_Dist_F,LOW);
  digitalWrite(XSHUT_Sen_Dist_I,LOW); delay(20);
  Serial.println("4");
  delay(50);
```

```
  if (!Sen_Dist_T.begin(0x21))
  {
    Serial.println("Fallo al iniciar Sen_Dist_T");
    Luces_Color_Fijo(125, 0, 0);
    while(1);
  }
  Serial.println("5");
  digitalWrite(XSHUT_Sen_Dist_D,HIGH);  delay(20);
  if (!Sen_Dist_D.begin(0x22))
  {
    Serial.println("Fallo al iniciar Sen_Dist_D");
    Luces_Color_Fijo(125, 0, 0);
    while(1);
  }
  digitalWrite(XSHUT_Sen_Dist_F,HIGH);  delay(20);
  if (!Sen_Dist_F.begin(0x23))
  {
    Serial.println("Fallo al iniciar Sen_Dist_F");
    Luces_Color_Fijo(125, 0, 0);
    while(1);
  }
  digitalWrite(XSHUT_Sen_Dist_I,HIGH);  delay(20);
  if (!Sen_Dist_I.begin(0x24))
  {
    Serial.println("Fallo al iniciar Sen_Dist_I");
    Luces_Color_Fijo(125, 0, 0);
    while(1);
  }
  delay(10);
  Sen_Dist_T.startRangeContinuous();
  delay(10);
  Sen_Dist_D.startRangeContinuous();
  delay(10);
  Sen_Dist_F.startRangeContinuous();
  delay(10);
  Sen_Dist_I.startRangeContinuous();
  delay(10);
  Sen_Dist_T.setMeasurementTimingBudgetMicroSeconds(25000);
  delay(10);
  Sen_Dist_D.setMeasurementTimingBudgetMicroSeconds(25000);
  delay(10);
  Sen_Dist_F.setMeasurementTimingBudgetMicroSeconds(25000);
  delay(10);
  Sen_Dist_I.setMeasurementTimingBudgetMicroSeconds(25000);
  delay(50);
}

void Sensor_Cap1_Activado() //Rutina de interrupción del sensor capacitivo frontal.
{
  sen_cap1=true;
}

void Sensor_Cap2_Activado() //Rutina de interrupción del sensor capacitivo trasero.
{
  sen_cap2=true;
}

void Inicio_RA_01()     //Iniciala cadena de LEDs, las comunicaciones Serial, los sensores de
distancia, el módulo de reproducción de MP3 y las interrupciones de los sensores capacitivos.
{
  LED_Chain.begin();
  LED_Chain.setBrightness(20);
  LED_Chain.setPixelColor(0,50,50,50);
  LED_Chain.show();
  Serial.begin(115200);
  Serial.println("i_0");
  Serial.println("i_Serial");
  LED_Chain.setPixelColor(1,50,50,50);
  LED_Chain.show();
  SerialDFPlayer.begin(9600, SERIAL_8N1, RX_MP3, TX_MP3);
  Serial.println("i_SerialDFPlayer");
  LED_Chain.setPixelColor(2,50,50,50);
  LED_Chain.show();
  Serial_2a1.begin(115200, SERIAL_8N1, RX_2a1, TX_2a1);
```

```
  Serial.println("i_Serial_2a1");
  LED_Chain.setPixelColor(3,50,50,50);
  LED_Chain.show();
  Inicio_Sensores_Distancia();
  LED_Chain.setPixelColor(4,50,50,50);
  LED_Chain.show();
  Serial.println("i_Sensores_Distancia");
  pinMode(pin_Sen_cap1,INPUT);
  pinMode(pin_Sen_cap2,INPUT);
  attachInterrupt(digitalPinToInterrupt(pin_Sen_cap1), Sensor_Cap1_Activado, RISING);
//activamos la interrupcion
  attachInterrupt(digitalPinToInterrupt(pin_Sen_cap2), Sensor_Cap2_Activado, RISING);
//activamos la interrupcion
  LED_Chain.setPixelColor(5,50,50,50);
  LED_Chain.show();
  Serial.println("i_Sensores_Capacitivos");
  while(!DFPlayer.begin(SerialDFPlayer))
  {
    Serial.println("Iniciando DFPlayer");
  }
  LED_Chain.setPixelColor(6,50,50,50);
  LED_Chain.show();
  delay(250);
  DFPlayer.volume(25);  //0-30
  delay(250);
  DFPlayer.play(1);
  LED_Chain.setPixelColor(7,50,50,50);
  LED_Chain.show();
  Serial.println("i_DFMP3");
  Serial.println("i_finalizado");
}

void Iniciar_Motores()    //Coloca los motores en la posición inicial, una por una, realizando
movimietnos bruscos en caso de que no se hayan colocado coorectamente en un inicio.
{
  Servo_IF_R.attach(Servo__Pin_1);
  Servo_DF_R.attach(Servo__Pin_2);
  Servo_IT_R.attach(Servo__Pin_3);
  Servo_DT_R.attach(Servo__Pin_4);
  Servo_IF_V.attach(Servo__Pin_5);
  Servo_DF_V.attach(Servo__Pin_6);
  Servo_IT_V.attach(Servo__Pin_7);
  Servo_DT_V.attach(Servo__Pin_8);
  Servo_Pinza.attach(Servo__Pin_9);


  Servo_IF_R.write(pos_if_r[2]);
  LED_Chain.setPixelColor(8,0,0,100);
  LED_Chain.show();
  delay(1000);
  Servo_DF_R.write(pos_df_r[2]);
  LED_Chain.setPixelColor(9,0,0,100);
  LED_Chain.show();
  delay(1000);
  Servo_IT_R.write(pos_it_r[2]);
  LED_Chain.setPixelColor(10,0,0,100);
  LED_Chain.show();
  delay(1000);
  Servo_DT_R.write(pos_dt_r[2]);
  LED_Chain.setPixelColor(11,0,0,100);
  LED_Chain.show();
  delay(1000);
  Servo_IF_V.write(pos_if_v[2]);
  LED_Chain.setPixelColor(12,0,0,100);
  LED_Chain.show();
  delay(1000);
  Servo_DF_V.write(pos_df_v[2]);
  LED_Chain.setPixelColor(13,0,0,100);
  LED_Chain.show();
  delay(1000);
  Servo_IT_V.write(pos_it_v[2]);
  LED_Chain.setPixelColor(14,0,0,100);
  LED_Chain.show();
  delay(1000);
```

```
  Servo_DT_V.write(pos_dt_v[2]);
  LED_Chain.setPixelColor(15,0,0,100);
  LED_Chain.show();
  delay(1000);
  Servo_Pinza.write(pos_pinza_actual);
  Luces_Color_Fijo (50, 125, 50);
  delay(1000);
  Servo_Pinza.detach();
}

void Mover_Patas (uint8_t n_pos_inicial, uint8_t n_pos_final, const double tiempo)  //Los
datos de entrada de esta función es el número de posición inicial y final y el tiempo que
transcurrirá entre los distintos movimientos.
{                                                                                    //Por
medio de varios bucles, realiza los movimietnso correspondietes entre las posiciones
indicadas.
  uint8_t n_pos, e_IF_R, e_DF_R, e_IT_R, e_DT_R, e_IF_V, e_DF_V, e_IT_V, e_DT_V;
  double q0[NUM_Motores], qT[NUM_Motores], a[NUM_Motores], b[NUM_Motores], c[NUM_Motores],
d[NUM_Motores], t, t0, q;

  for (n_pos = n_pos_inicial; n_pos < n_pos_final; n_pos++)
  {
    q0[0] = pos_if_r[n_pos],
    q0[1] = pos_df_r[n_pos],
    q0[2] = pos_it_r[n_pos],
    q0[3] = pos_dt_r[n_pos],
    q0[4] = pos_if_v[n_pos],
    q0[5] = pos_df_v[n_pos],
    q0[6] = pos_it_v[n_pos],
    q0[7] = pos_dt_v[n_pos];

    qT[0] = pos_if_r[n_pos+1],
    qT[1] = pos_df_r[n_pos+1],
    qT[2] = pos_it_r[n_pos+1],
    qT[3] = pos_dt_r[n_pos+1],
    qT[4] = pos_if_v[n_pos+1],
    qT[5] = pos_df_v[n_pos+1],
    qT[6] = pos_it_v[n_pos+1],
    qT[7] = pos_dt_v[n_pos+1];

    for (int i = 0; i < NUM_Motores; i++)
    {
      a[i] = -2 * (qT[i] - q0[i]) / pow(tiempo, 3);
      b[i] = 3 * (qT[i] - q0[i]) / pow(tiempo, 2);
      c[i] = 0;
      d[i] = q0[i];
    }
    //Serial.println("Parametros calculados");
    t0 = millis() / 1000.0;
    t = 0.0;
    while (t < tiempo)
    {
      for (int i = 0; i < NUM_Motores; i++)
      {
        q = map(a[i] * pow(t, 3) + b[i] * pow(t, 2) + c[i] * t + d[i], 0, 180, 600, 2400);
        switch(i)
        {
          case 0:
            Servo_IF_R.writeMicroseconds(q);
          break;
          case 1:
            Servo_DF_R.writeMicroseconds(q);
          break;
          case 2:
            Servo_IT_R.writeMicroseconds(q);
          break;
          case 3:
            Servo_DT_R.writeMicroseconds(q);
          break;
          case 4:
            Servo_IF_V.writeMicroseconds(q);
          break;
          case 5:
            Servo_DF_V.writeMicroseconds(q);
```

```
          break;
          case 6:
            Servo_IT_V.writeMicroseconds(q);
          break;
          case 7:
            Servo_DT_V.writeMicroseconds(q);
          break;
        }
      }
      t = millis() / 1000.0 - t0;
    }
  }
}

void Movimiento_Patas(uint8_t dato_recibido)  //Teniendo como parametro el dato recibido que
el movimiento final deseado, esta función se encargará de que los movimientos realizados sena
los correctos, realizando las transiciones correspondientes.
{
  switch(dato_recibido-1)
  {
    case 0:
      Serial.println("Estado_movimiento: Tumbado");
      switch (posicion)
      {
        case 0:
        break;

        case 1:
          Mover_Patas(38,39,velocidad_movimiento_1);
          posicion=0;
        break;

        case 2:
          Mover_Patas(29,33,velocidad_movimiento_1);
          posicion=1;
        break;

        case 3:
          Mover_Patas(34,38,velocidad_movimiento_1);
          posicion=1;
        break;

        case 4:
          Mover_Patas(17,19,velocidad_movimiento_1);
          posicion=1;
        break;
      }

    break;
    case 1:
      Serial.println("Estado_movimiento: Posición erguida");
      switch (posicion)
      {
        case 0:
          Mover_Patas(0,2,velocidad_movimiento_1);
          posicion=1;
        break;

        case 1:
        break;

        case 2:
          Mover_Patas(29,33,velocidad_movimiento_1);
          posicion=1;
        break;

        case 3:
          Mover_Patas(34,38,velocidad_movimiento_1);
          posicion=1;
        break;

        case 4:
          Mover_Patas(24,28,velocidad_movimiento_1);
          posicion=1;
```

```
        break;
      }
    break;

    case 2:
      Serial.println("Estado_movimiento: Caminar hacia delante");
      switch (posicion)
      {
        case 0:
          Mover_Patas(0,2,velocidad_movimiento_1);
          posicion=1;
        break;

        case 1:
          Mover_Patas(2,6,velocidad_movimiento_1);
          posicion=2;
        break;

        case 2:
          Mover_Patas(6,8,velocidad_movimiento_1);
          posicion=3;
        break;

        case 3:
          Mover_Patas(8,10,velocidad_movimiento_1);
          posicion=2;
        break;

        case 4:
          Mover_Patas(24,28,velocidad_movimiento_1);
          posicion=1;
        break;
      }
    break;

    case 3:
      Serial.println("Estado_movimiento: Caminar hacia detrás");
      switch (posicion)
      {
        case 0:
          Mover_Patas(0,2,velocidad_movimiento_1);
          posicion=1;
        break;

        case 1:
          Mover_Patas(2,6,velocidad_movimiento_1);
          posicion=2;
        break;

        case 2:
          Mover_Patas(10,12,velocidad_movimiento_1);
          posicion=3;
        break;

        case 3:
          Mover_Patas(12,14,velocidad_movimiento_1);
          posicion=2;
        break;

        case 4:
          Mover_Patas(24,28,velocidad_movimiento_1);
          posicion=1;
        break;
      }
    break;

    case 4:
      Serial.println("Estado_movimiento: Rotar antihorario");
      switch (posicion)
      {
        case 0:
          Mover_Patas(0,2,velocidad_movimiento_1);
          posicion=1;
        break;
```

```
        case 1:
          Mover_Patas(19,21,velocidad_movimiento_R);
          posicion=4;
        break;

        case 2:
          Mover_Patas(29,33,velocidad_movimiento_1);
          posicion=1;

        break;

        case 3:
          Mover_Patas(34,38,velocidad_movimiento_1);
          posicion=1;
        break;

        case 4:
          Mover_Patas(21,23,velocidad_movimiento_R);
          posicion=1;
        break;
      }
    break;

    case 5:
      Serial.println("Estado_movimiento: Rotar horario");
      switch (posicion)
      {
        case 0:
          Mover_Patas(0,2,velocidad_movimiento_1);
          posicion=1;
        break;

        case 1:
          Mover_Patas(15,17,velocidad_movimiento_R);
          posicion=4;
        break;

        case 2:
          Mover_Patas(29,33,velocidad_movimiento_1);
          posicion=1;
        break;

        case 3:
          Mover_Patas(34,38,velocidad_movimiento_1);
          posicion=1;
        break;

        case 4:
          Mover_Patas(17,19,velocidad_movimiento_R);
          posicion=1;
        break;
      }
    break;

    case 6:
      Serial.println("Estado_movimiento: Parado");
    break;
  }
}

void Movimiento_Pinza(uint8_t dato_recibido,  const double tiempo) //Dado el dato recibido que
indica la posición final y el tiempo, esta función se encargará de mover la pinza desde la
posición actual hasta la deseada de forma suave.
{
  double q0, qT, a, b, c, d, t, t0, q;
  Servo_Pinza.attach(Servo__Pin_9);
  switch(dato_recibido)
  {
    case 8:
      qT = 50;
    break;

    case 9:
```

```
      qT = 90;
    break;

    case 10:
      qT = 130;
    break;
  }
  q0=pos_pinza_actual;
  pos_pinza_actual=qT;
  a = -2 * (qT - q0) / pow(tiempo, 3);
  b = 3 * (qT - q0) / pow(tiempo, 2);
  c = 0;
  d = q0;
  //Serial.println("Parametros calculados");
  t0 = millis() / 1000.0;
  t = 0.0;
  while (t < tiempo)
  {
    q = map(a * pow(t, 3) + b * pow(t, 2) + c * t + d, 0, 180, 600, 2400);
    Servo_Pinza.writeMicroseconds(q);
    t = millis() / 1000.0 - t0;
  }
  Servo_Pinza.detach();
}

uint8_t Lectura_Serial_2a1() //Lee los datos que el microcontrolador ESP 32 cam envia.
{
  uint8_t dato_recibido = Serial_2a1.read();
  if (dato_recibido != 255)
  {
    Serial.print("Dato recibido:  ");
    Serial.println(dato_recibido);
    if (dato_recibido > 10 && dato_recibido < 20) dato_recibido_previo = 0;
    else dato_recibido_previo = dato_recibido;
    return (uint8_t)dato_recibido;
  }
    return dato_recibido_previo;
}

void Lectura_Sensores_Distancia() //Realiza la medida de los  sensores de distancia y cambia
la emoción y cambia el esttado de movimiento cuando detecta que el robot se va a caer o que ha
sido levantado.
{
  uint16_t dist_D, dist_I, dist_T, dist_F;

   if (Sen_Dist_D.isRangeComplete())
  {
    dist_D = Sen_Dist_D.readRange();
    //Serial.print(";  Sen_Dist_D : ");
    //Serial.print(dist_D);
  }
  else
  {
    Serial.print("  Sen_Dist_D falla ");
    Luces_Color_Fijo(125, 0, 0);
  }

  if (Sen_Dist_I.isRangeComplete())
  {
    dist_I = Sen_Dist_I.readRange();
    //Serial.print(";  Sen_Dist_I : ");
    //Serial.print(dist_I);
  }

  else
  {
    Serial.println("  Sen_Dist_I falla ");
    Luces_Color_Fijo(125, 0, 0);
  }

  if (Sen_Dist_T.isRangeComplete())
  {
    dist_T=Sen_Dist_T.readRange();
    //Serial.print("  Sen_Dist_T : ");
```

```
    //Serial.print(dist_T);
  }
  else
  {
    Serial.print("  Sen_Dist_T falla ");
    Luces_Color_Fijo(125, 0, 0);
  }

  if (Sen_Dist_F.isRangeComplete())
  {
    dist_F=Sen_Dist_F.readRange();
    //Serial.print(";  Sen_Dist_F : ");
    //Serial.println(dist_F);
  }
  else
  {
    Serial.print("  Sen_Dist_F falla ");
    Luces_Color_Fijo(125, 0, 0);
  }
  if(dist_D > 300 || dist_I > 300 || dist_T > 300 || dist_F > 300)
  {
    detener_robot=true;
    Serial_2a1.write(7);
  }
}

void Lectura_Bateria() //Realiza la lectura de la tensión de la batería. Si esta es baja envia
un mensaje al otro microcontrolador. Si se deteta que se  ha conectado el cargador envia otro
dato.
{
  if (contador>100)
  {
    uint16_t nivel_bateria = analogRead(pin_Lectura_Bat);
    nivel_bateria = map(nivel_bateria, 0, 4095, 0, 6600);
    if(nivel_bateria_previo == 1000) nivel_bateria_previo = nivel_bateria;
    Serial.print("Nivel Bateria: ");
    Serial.println(nivel_bateria);
    if(nivel_bateria - nivel_bateria_previo > 70)  //Se ha conectado la baetría a cargar
    {
      Serial.println("Se ha conectado el cargador");
      Serial_2a1.write(1);
    }
    if(nivel_bateria < 3400)     //Nivel de batria bajo
    {
      Serial_2a1.write(2);
      Serial.println("Se ha detectado bateria baja");
    }
    nivel_bateria_previo = nivel_bateria;
    contador=0;
  }
  else contador++;
}

void Funcion_Sen_1() //Esta función es activada por la rutina de interrupción del sensor
capacitivo frontal. Sirve para detectar cuando se toca el robot.
{
  contador_sen_1++;
  sen_1_previo=true;
  sen_2_previo=false;
  if(!digitalRead(pin_Sen_cap1)) sen_cap1=false;
}

void Funcion_Sen_2() //Esta función es activada por la rutina de interrupción del sensor
capacitivo trasero. Sirve para detectar cuando se toca el robot.
{
  contador_sen_2++;
  sen_1_previo=false;
  sen_2_previo=true;
  if(!digitalRead(pin_Sen_cap2)) sen_cap2=false;
}

void Tacto()  //Esta función detecta si el robot ha sido acariciado de deatrás a delante, de
delante a detrás o si se ha mantenido la mano apoyada en sobre el robot.
```

```
{             //Dependiendo de como haya sido el contacto con el robot envia un dato u otro al
microcontrolador ESP 32 cam que recibe el mensaje y reacciona al mismo modificando la
expresión facila del robot y la posición de sus ojos.
  if (sen_cap1 && sen_2_previo) //Previamente se habia detectado el sensor 2 y ahora se
detecta el sensor 1
  {
    Serial_2a1.write(6);
    Luces_Onda (0, 125, 0);  //Luces_Onda verde
    Serial.println("Se ha acariciado el robot de detras a delante");
  }
  if (sen_cap2 && sen_1_previo) //Previamente se habia detectado el sensor 1 y ahora se
detecta el sensor 2
  {
    Serial_2a1.write(5);
    Luces_Onda (126, 126, 0); //Luces_Onda amarilla
    Serial.println("Se  ha acariciado el robot de delante a detras");
  }

  if(sen_cap1) Funcion_Sen_1();
  else contador_sen_1 = 0;
  if(sen_cap2) Funcion_Sen_2();
  else contador_sen_2 = 0;
  if(contador_sen_1 > 15)
  {
    Serial_2a1.write(3);
    Luces_Onda (126, 126, 126); //Luces_Onda blanca
    Serial.println("Se ha mantenido la mano apollada sobre la parte frontal");
    contador_sen_1 = 0;
  }
  else if(contador_sen_2 > 15)
  {
    Serial_2a1.write(4);
    Luces_Onda (126, 0, 0); //Luces_Onda roja
    Serial.println("Se ha mantenido la mano apoyada sobre la parte trasera");
    contador_sen_2 = 0;
  }
  else if (sen_1_previo || sen_2_previo)
  {
    contador_sensor_previo++;
  }
  if(contador_sensor_previo > 10)
  {
    sen_1_previo=false;
    sen_2_previo=false;
    contador_sensor_previo=0;
  }
}
```

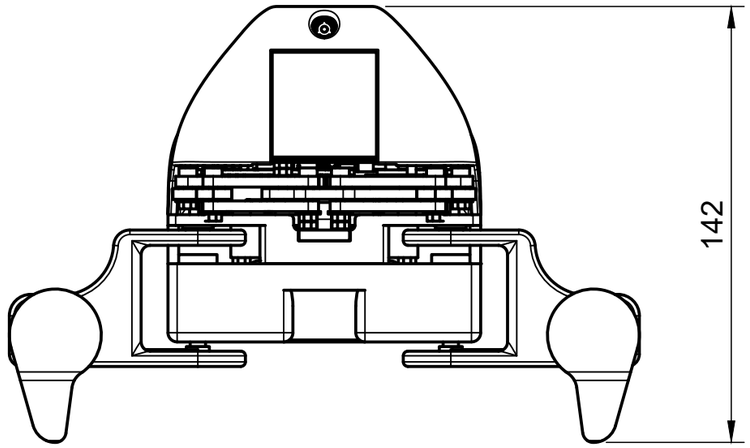
PLANOS

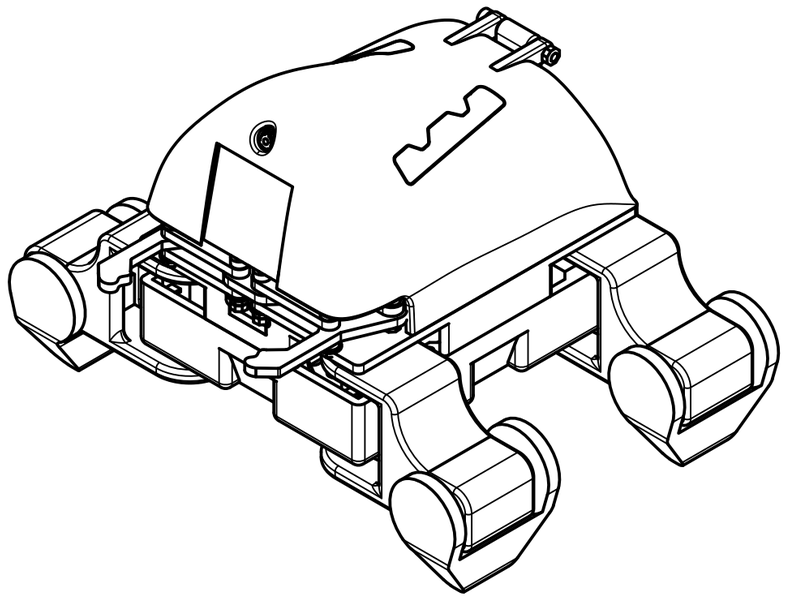
Diseño y desarrollo de un robot cuadrúpedo dedicado al tratamiento de niños con autismo.

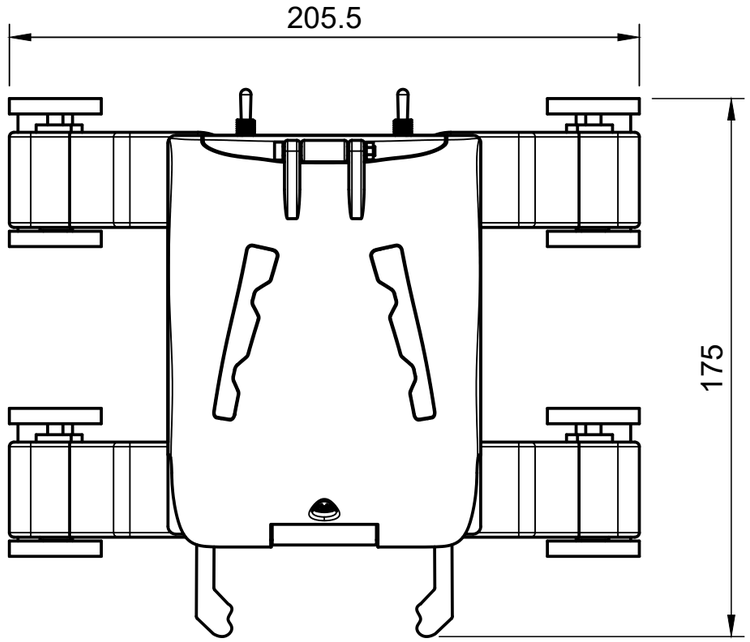
Trabajo final del:	Grado en Ingeniería electrónica industrial y Automática
Realizado por:	Ernest Folgado Brisa
Tutorizado por:	Javier Ibáñez Civera
Curso académico:	2022/2023

Contenidos planos:

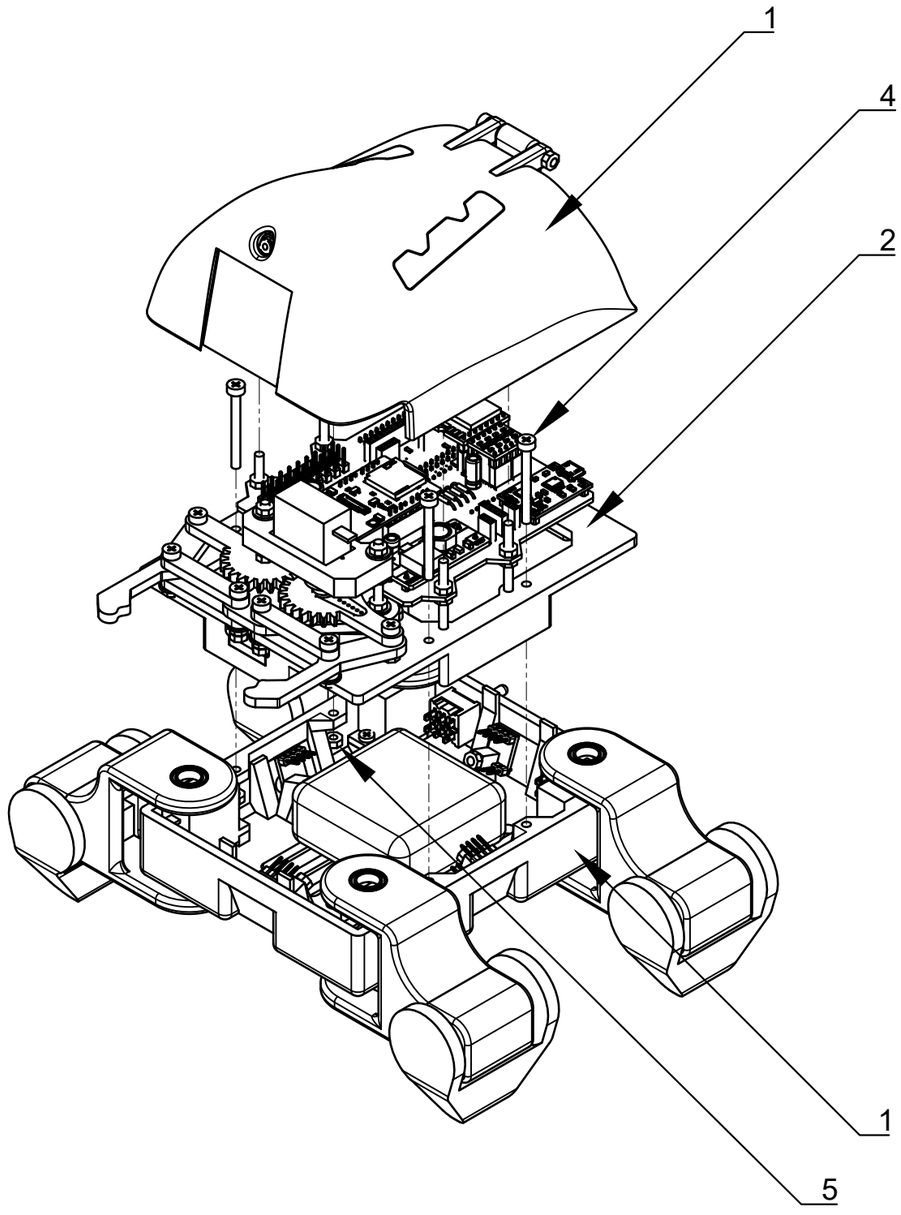
El código y los modelos 3D de las piezas se pueden encontrar en el siguiente enlace: https://github.com/ErnestFB/RA-01





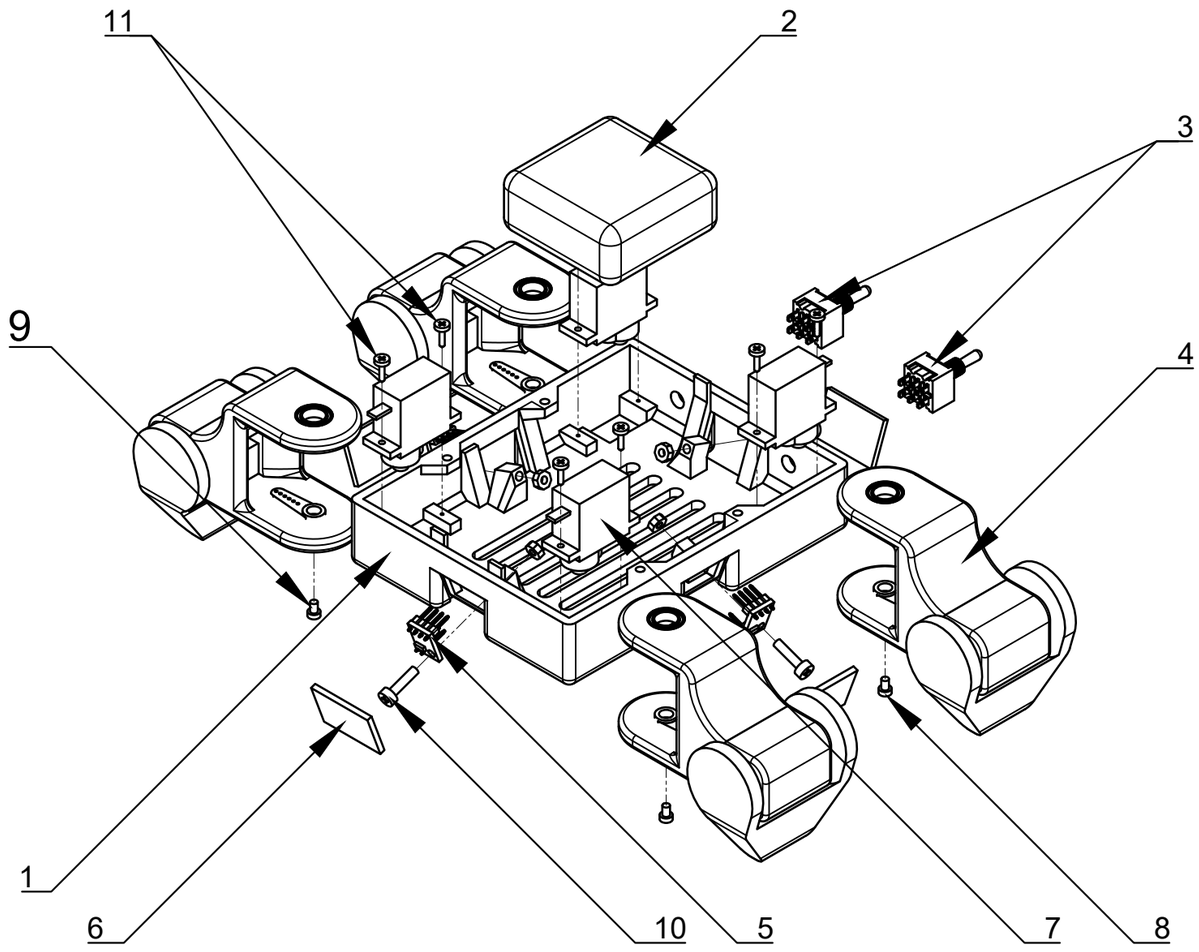


PROYECTO: RA-01	TÍTULO: Conjunto		FECHA: 13/05/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA Escuela Técnica Superior de Ingeniería del Diseño	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: 1:2	UNIDADES: mm
	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 1	



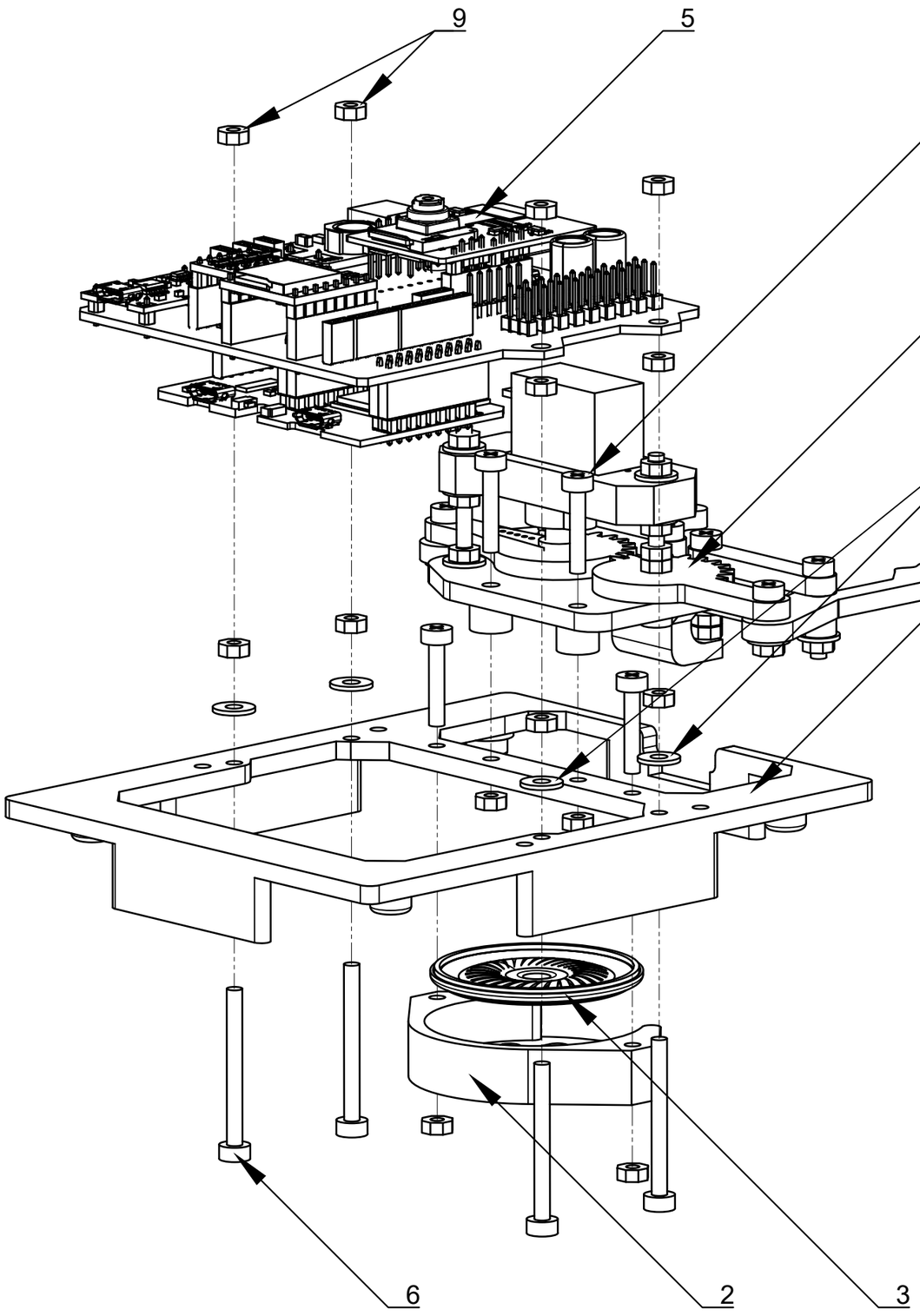
Marca	Cantidad	Denominación	Nº plano, fabricante o distribuidor	Observaciones
5	4	Tuerca M3	Fabricante: STANDERS	
4	4	Tornillo M3 L30	Fabricante: STANDERS	Cabeza cilíndrica
3	1	Parte inferior	Plano Nº5	
2	1	Parte intermedia	Plano Nº 4	
1	1	Parte superior	Plano Nº 3	

PROYECTO: RA-01	TÍTULO: Completo, despiece		FECHA: 13/05/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA Escuela Técnica Superior de Ingeniería del Diseño	AUTOR: Ernest Folgado Brisa	FIRMA: Ernest FB	ESCALA: 1:2	UNIDADES: mm
	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 2	



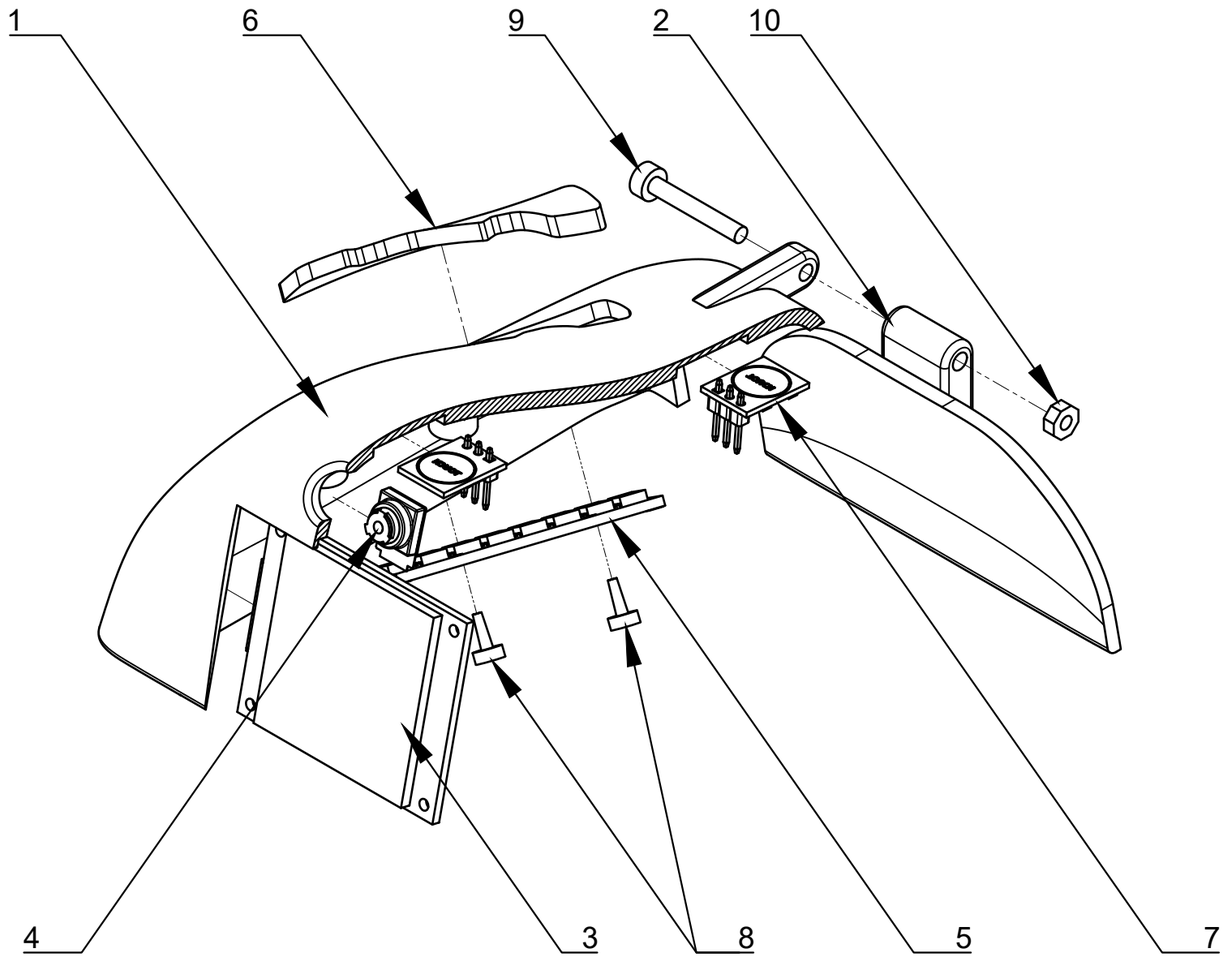
11	8	Tornillo 3		Incluido con el servomotor
10	4	Tornillo 2		Incluido con el servomotor
9	4	Tornillo M3 L12	Fabricante: STANDERS	Cabeza cilíndrica
8	4	Tuerca M3	Fabricante: STANDERS	
7	4	Servomotor MG90S	Plano Nº 24, Fabricante: DM DIY MORE	SKU: 060004
6	4	Ventana metacrilato		26x18x2mm
5	4	Módulo GY-530	Distribuidor: WCMCU Store (aliexpresss)	
4	4	Pata	Plano Nº 6 y 7	
3	2	Interruptor basculante palanca	Distribuidor: Electrón Perdido	Referencia: 0209 125V/3A DPDT 6 Pines
2	1	Batería	Fabricante: Bricogeek	Referencia:BAT-0012 Modelo 105050, 3.7V, 50x50x20mm
1	1	Pienza inferior	Plano Nº 3	
Marca	Cantidad	Denominación	Nº plano, fabricante o distribuidor	Observaciones

PROYECTO	TÍTULO:		FECHA:	
RA-01	Parte inferior, despiece		19/04/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA Escuela Técnica Superior de Ingeniería del Diseño	AUTOR: Ernest Folgado Brisa	FIRMA: Ernest FB	ESCALA: 1:2	UNIDADES: mm
	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 3	



9	16	Tuerca M3	Fabricante: STANDERS	
8	4	Arandela M3	Fabricante: STANDERS	
7	4	Tornillo M3 L16	Fabricante: STANDERS	Cabeza cilíndrica
6	4	Tornillo M3 L30	Fabricante: STANDERS	Cabeza cilíndrica
5	1	PCB	Planos Nº 27-32	
4	1	Pinza	Planos Nº 8 y 9	
3	1	Altavoz	Fabricante: Visaton	8Ω, 2W, Ø40mm Mouser ref: 243-K40-8OHM
2	1	Soporte altavoz	Plano Nº 15	
1	1	Pieza intermedia	Plano Nº 11	
Marca	Cantidad	Denominación	Nº plano, fabricante o distribuidor	Observaciones

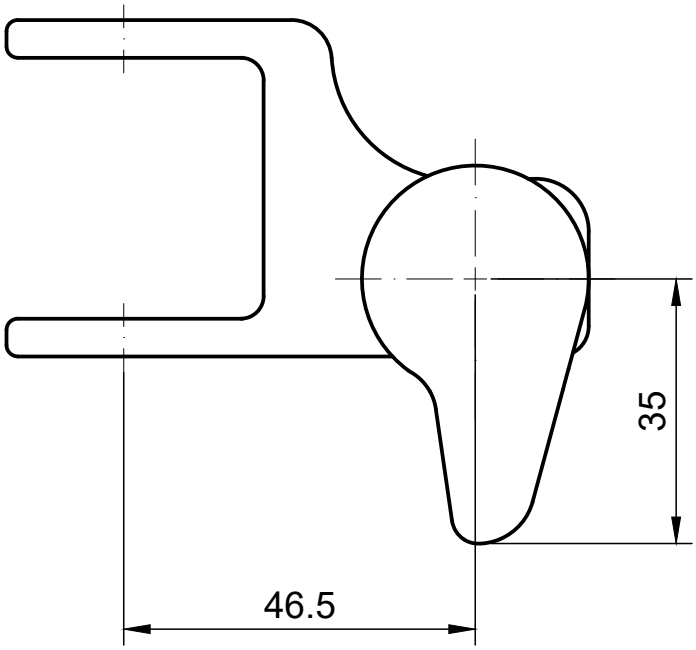
PROYECTO	TÍTULO:		FECHA:	
RA-01	Parte intermedia, despiece		03/05/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA	AUTOR: Ernest Folgado Brisa	FIRMA: EFB	ESCALA: 1:1	UNIDADES: mm
Escuela Técnica Superior de Ingeniería del Diseño	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 4	

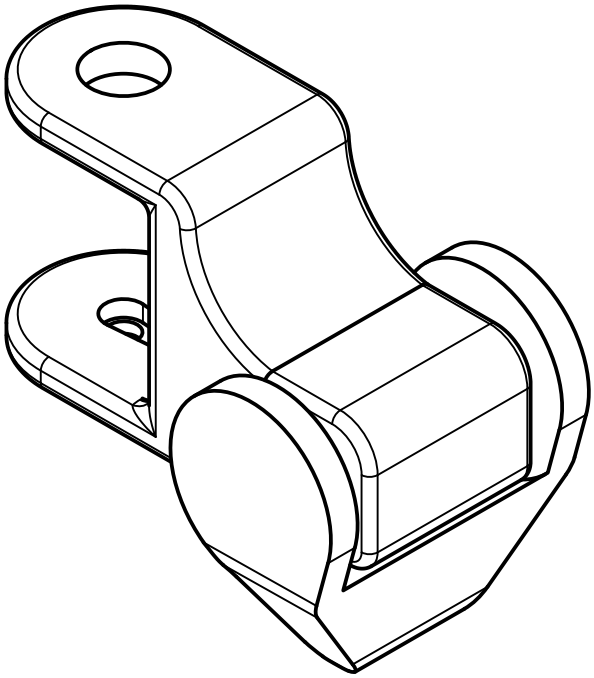


Para mayor claridad se ha cortado la pieza superior.

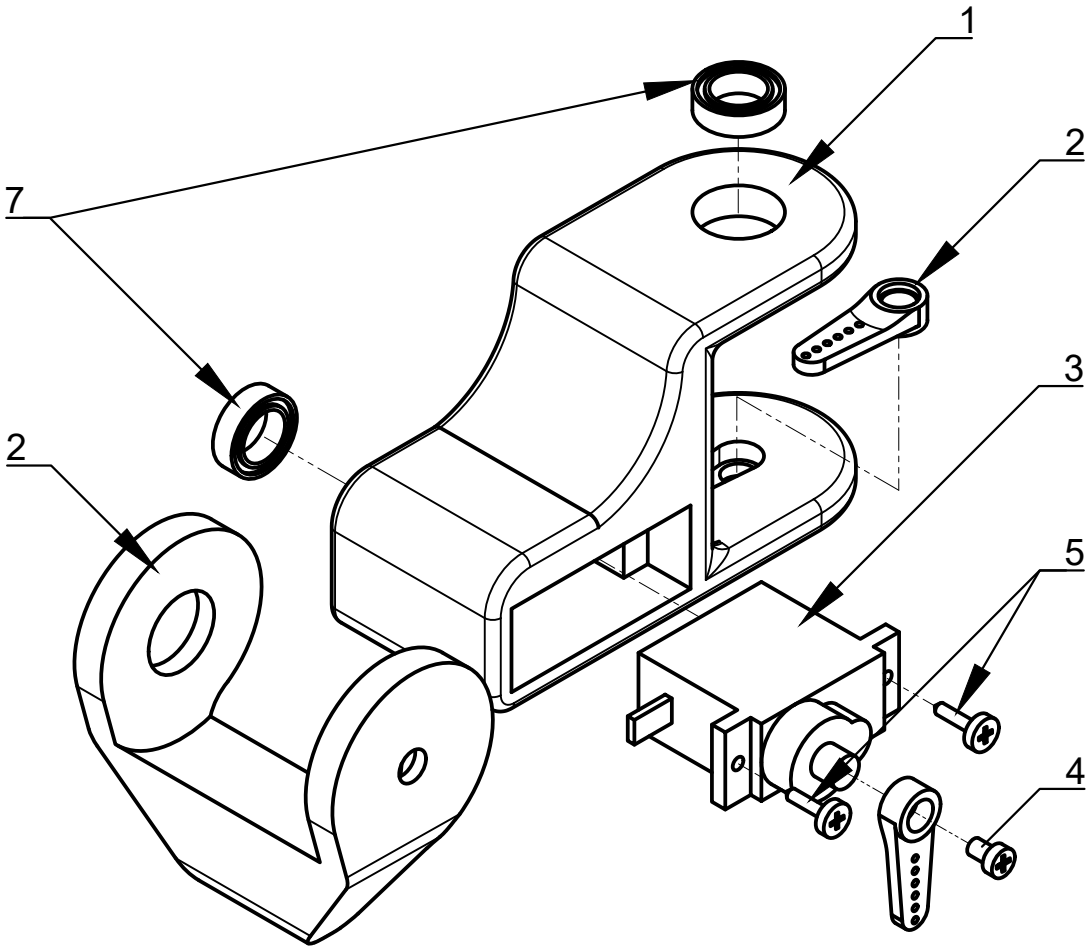
Marca	Cantidad	Denominación	Nº plano, fabricante o distribuidor	Observaciones
10	1	Tuerca M3	Fabricante: STANDERS	
9	1	Tornillo M3 20mm	Fabricante: STANDERS	Cabeza cilíndrica
8	4	Enganche	Plano Nº 25	
7	2	Módulo interruptor tactil TTP223	Distribuidor KYYKA (amazon)	ASIN:B07DB5YFGW
6	2	Pieza luces	Plano Nº 17	
5	2	Barra LED RGB digital WS2812 8x5050	Distribuidor electrón perdido, plano Nº 26	Referencia: 0117
4	1	Cámara	Distribuidor: WCMCU Store (aliexpresss)	OV2640, longitud de conector 75 mm
3	1	Pantalla	Fabricante: Waveshare (amazon), plano Nº 27	1.5" RGB OLED, ASIN:B07DB5YFGW
2	1	Puerta	Plano Nº 16	
1	1	Pieza superior	Plano Nº 12	

PROYECTO: RA-01	TÍTULO: Paste superior, despiece			FECHA: 19/04/2023
UNIVERSITAT POLITÈCNICA DE VALÈNCIA Escuela Técnica Superior de Ingeniería del Diseño	AUTOR: Ernest Folgado Brisa	FIRMA: Ernest FB	ESCALA: 1:1	UNIDADES: mm
	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 5	



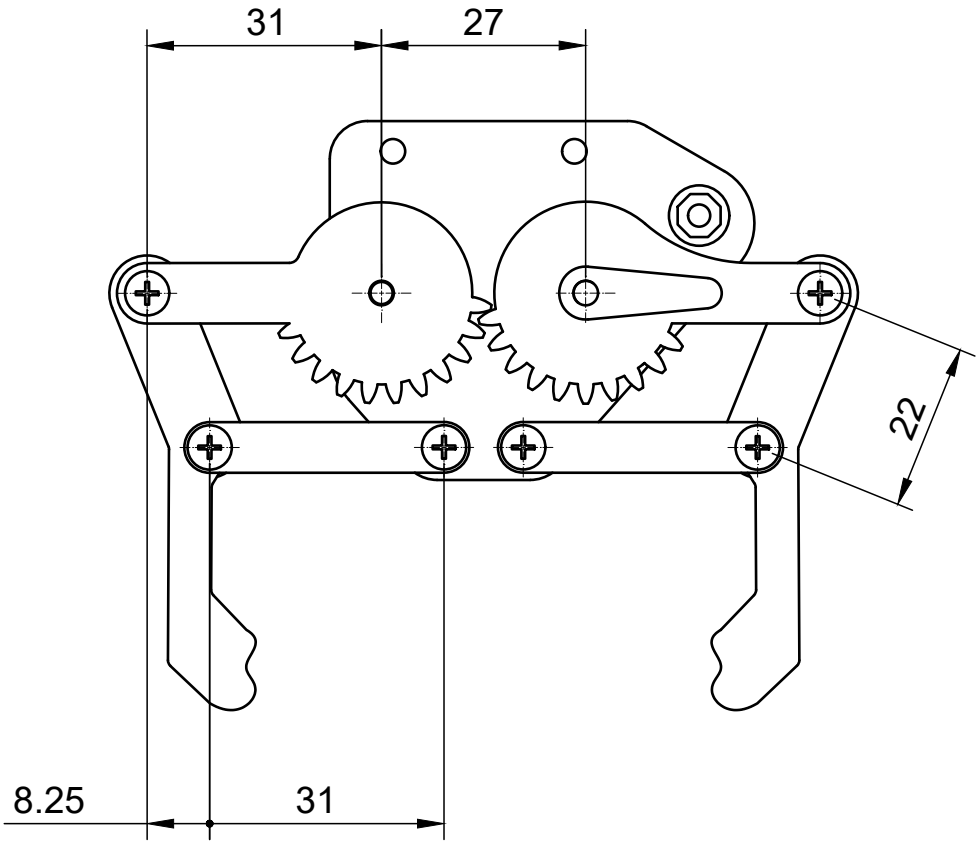


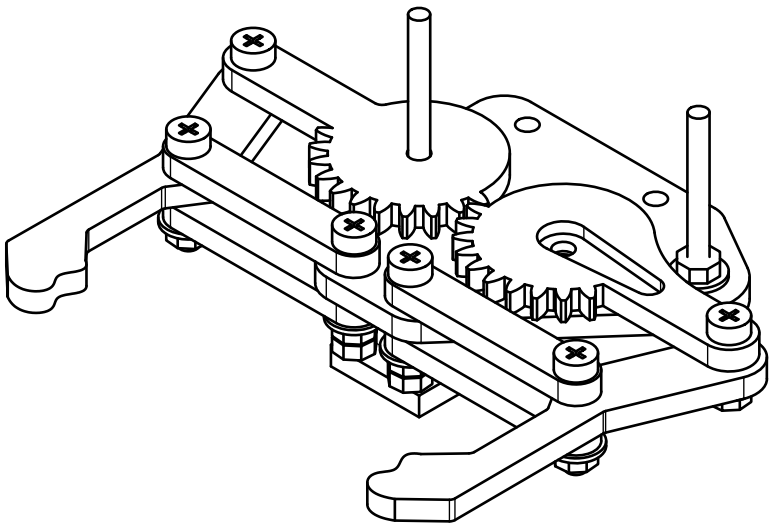
PROYECTO	TÍTULO:		FECHA:	
RA-01	Pata, conjunto		15/04/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA Escuela Técnica Superior de Ingeniería del Diseño	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: 1:1	UNIDADES: mm
	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 6	



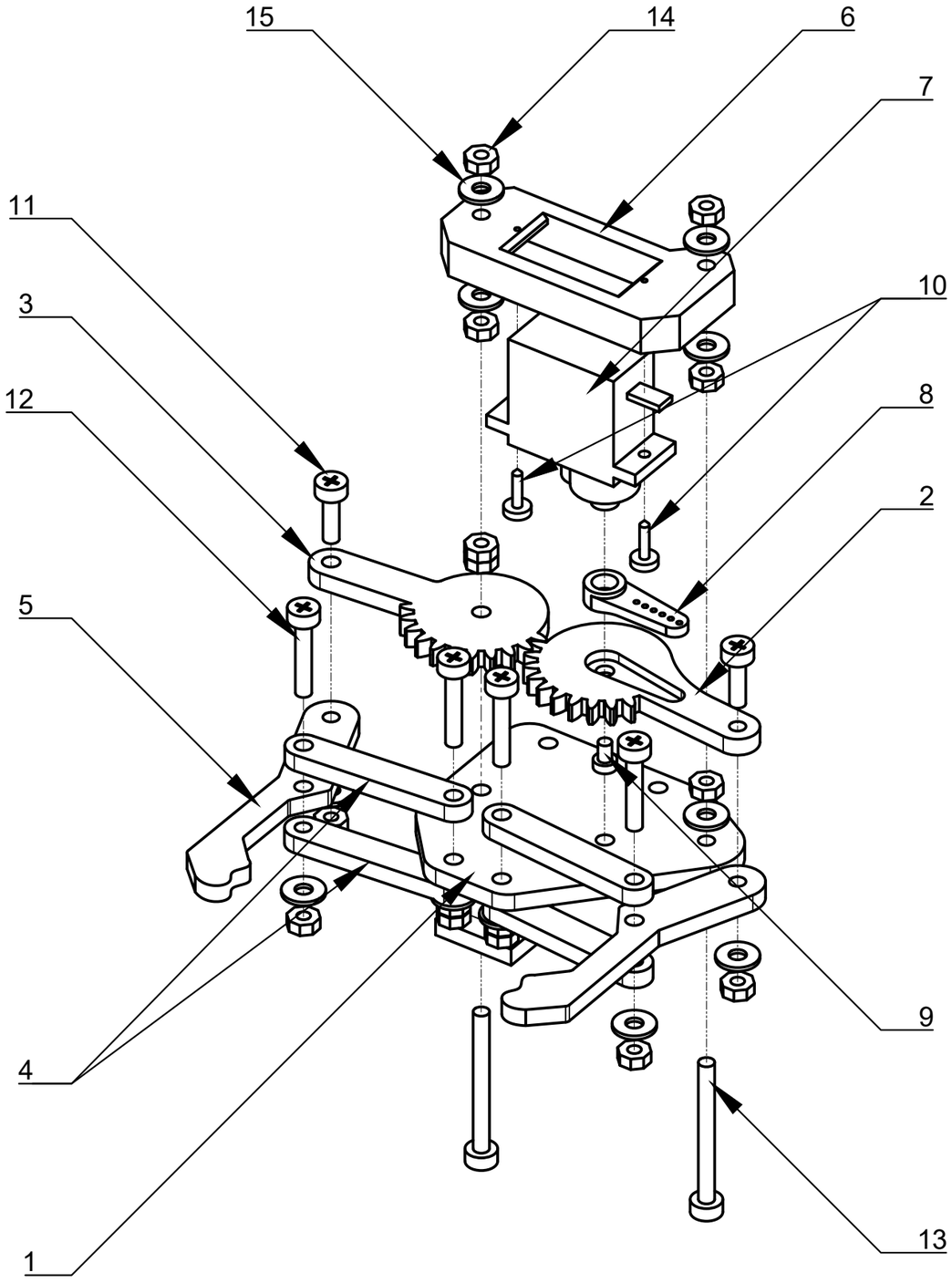
7	2	Rodamiento	Distribuidor haibin Store (aliexpress)	Mr63zz 3x6x2.5mm
6	2	Accesorio servo		Incluido con el servomotor
5	2	Tornillo 2		Incluido con el servomotor
4	4	Tornillo1		Incluido con el servomotor
3	2	Servomotor MG90S	Plano Nº 24, Fabricante: DM DIY MORE	SKU: 060004
2	1	Eslabón 2 pata	Plano Nº 14	
1	1	Eslabón 1 pata	Plano Nº 13	
Marca	Cantidad	Denominación	Nº plano, fabricante o distribuidor	Observaciones

PROYECTO	TÍTULO:		FECHA:	
RA-01	Pata, despiece		19/04/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: 1:1	UNIDADES: mm
Escuela Técnica Superior de Ingeniería del Diseño	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 7	





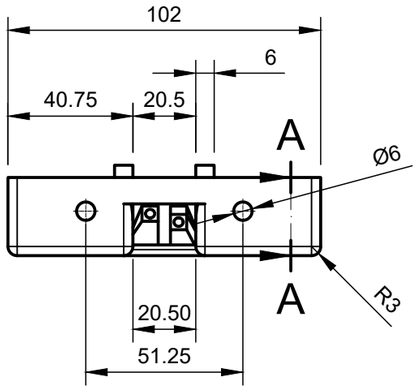
PROYECTO	TÍTULO:		FECHA:	
RA-01	Pinza, conjunto		25/04/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: 1:1	UNIDADES: mm
Escuela Técnica Superior de Ingeniería del Diseño	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 8	



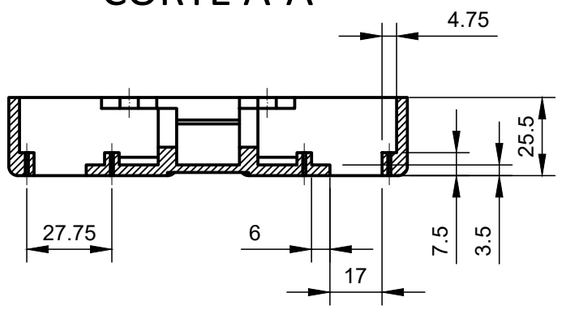
15	15	Tuerca M3	Fabricante: STANDERS	
14	11	Arandela M3	Fabricante: STANDERS	
13	2	Tornillo M3 L30	Fabricante: STANDERS	Cabeza cilíndrica
12	4	Tornillo M3 L16	Fabricante: STANDERS	Cabeza cilíndrica
11	2	Tornillo M3 L10	Fabricante: STANDERS	Cabeza cilíndrica
10	2	Tornillo madera M2 L8		Incluido con el servomotor
9	1	Tornillo M2,5 L4		Incluido con el servomotor
8	1	Accesorio servo		Incluido con el servomotor
7	1	Servomotor MG90S	Plano Nº 24, Fabricante: DM DIY MORE	SKU: 060004
6	1	Soporte Servo	Plano Nº 23	
5	2	Pieza pinza 4	Plano Nº 22	
4	4	Pieza pinza 3	Plano Nº 21	
3	1	Pieza pinza 2	Plano Nº 20	
2	1	Pieza pinza 1	Plano Nº 19	
1	1	Base pinza	Plano Nº 18	
Marca	Cantidad	Denominación	Nº plano, fabricante o distribuidor	Observaciones

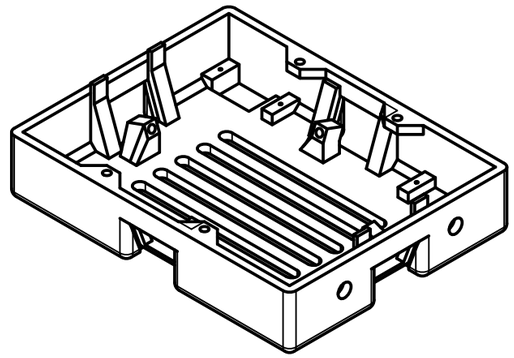
E

PROYECTO	TÍTULO:		FECHA:	
RA-01	Pinza, despiece		22/04/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA Escuela Técnica Superior de Ingeniería del Diseño	AUTOR: Ernest Folgado Brisa	FIRMA: Ernest FB	ESCALA: 1:1	UNIDADES: mm
	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 9	

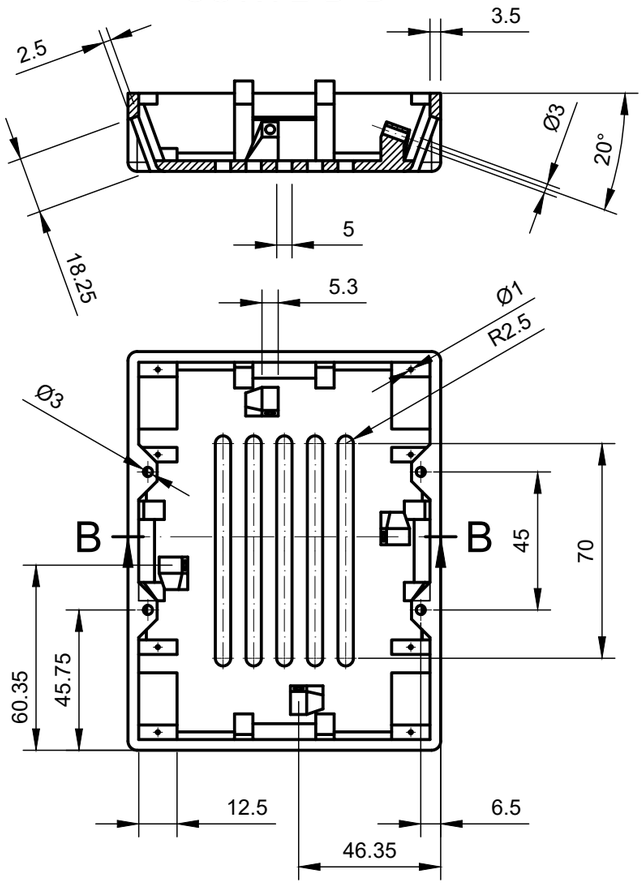


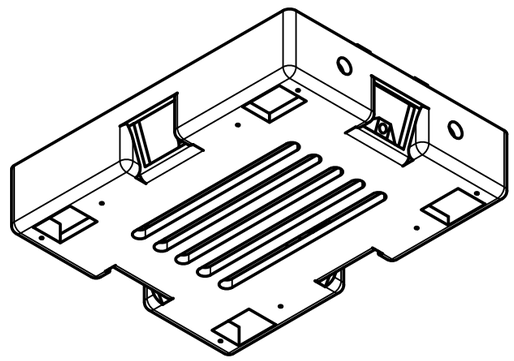
CORTE A-A



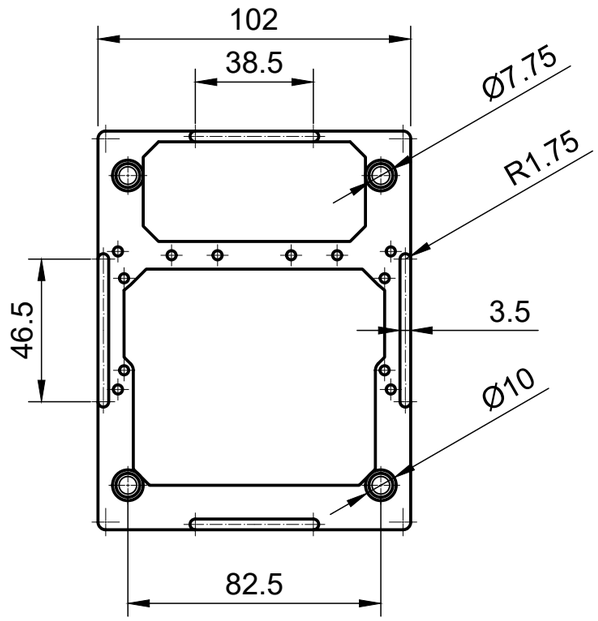


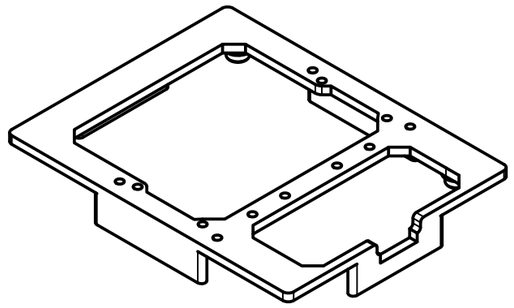
CORTE B-B

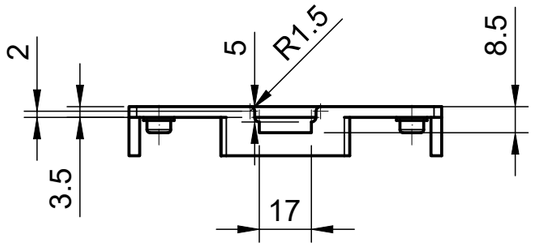


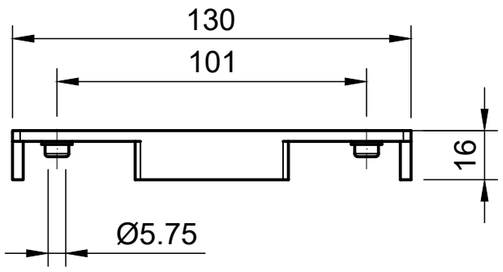


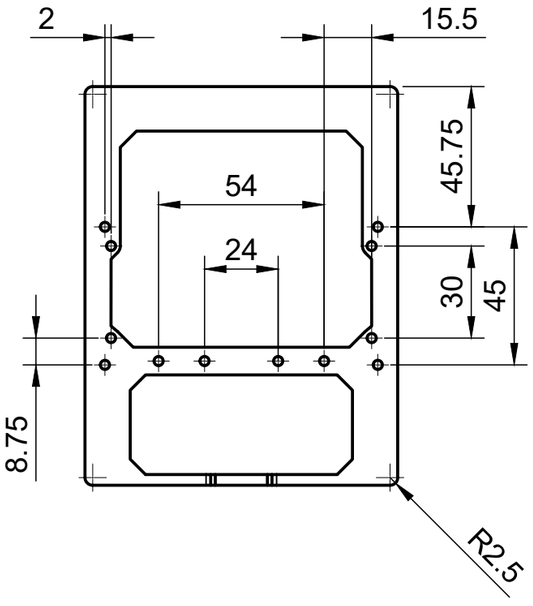
PROYECTO: RA-01	TÍTULO: Pieza inferior		FECHA: 05/04/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA Escuela Técnica Superior de Ingeniería del Diseño	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: 1:2	UNIDADES: mm
	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 10	

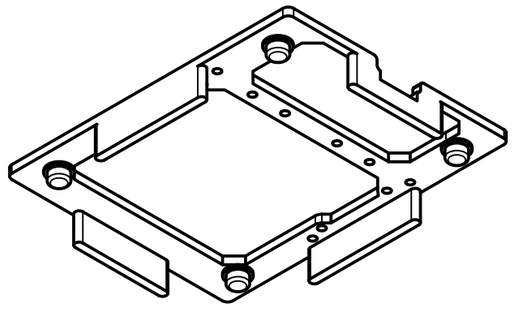




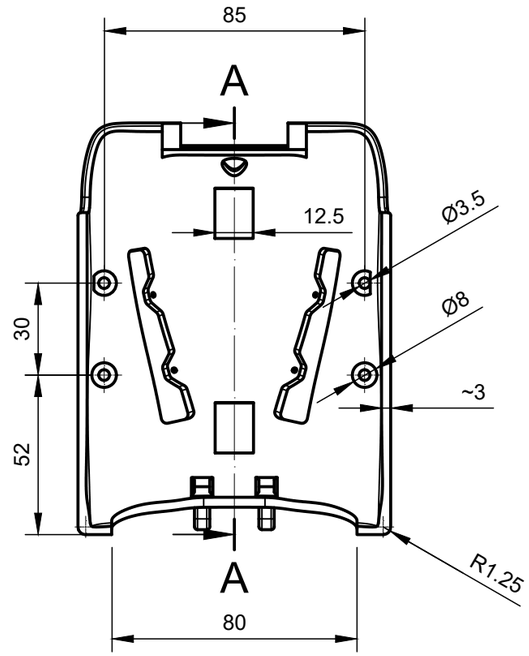




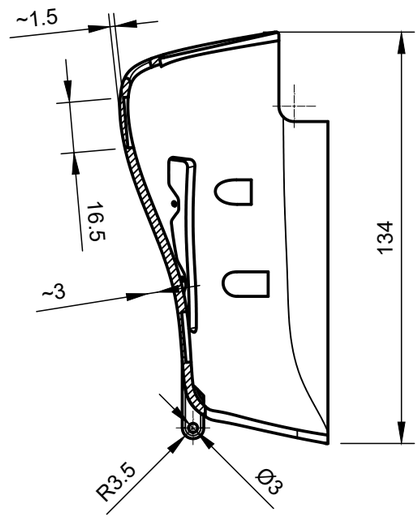


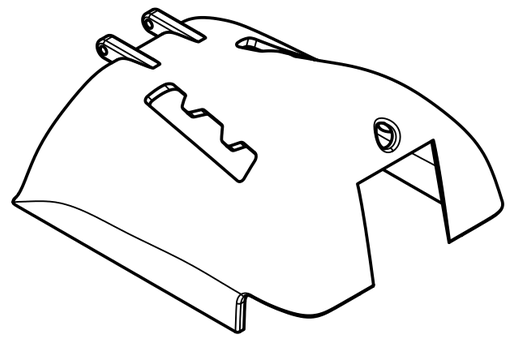


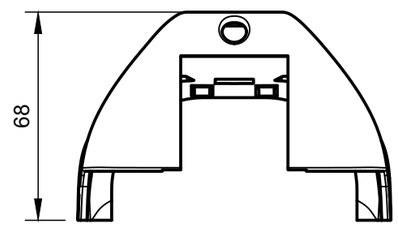
PROYECTO: RA-01	TÍTULO: Pieza intermedia		FECHA: 05/04/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA Escuela Técnica Superior de Ingeniería del Diseño	AUTOR: Ernest Folgado Brisa	FIRMA: Ernest FB	ESCALA: 1:2	UNIDADES: mm
	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 11	

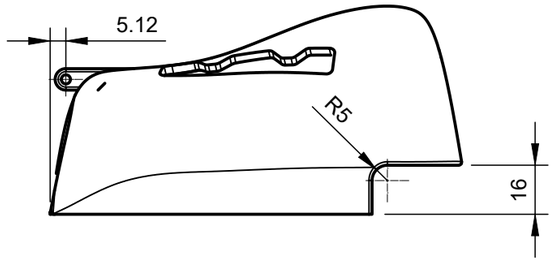


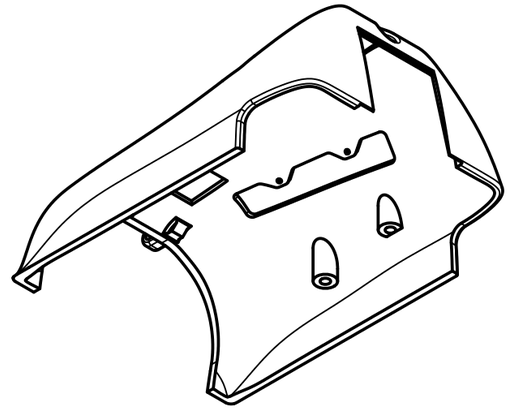
CORTE A-A

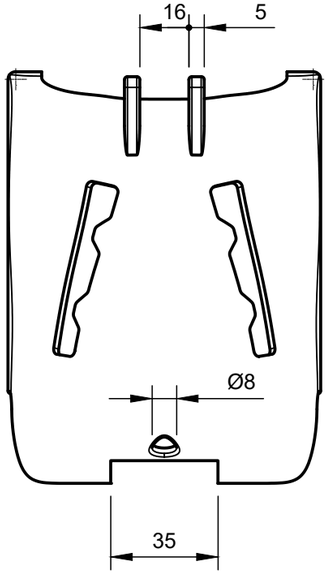




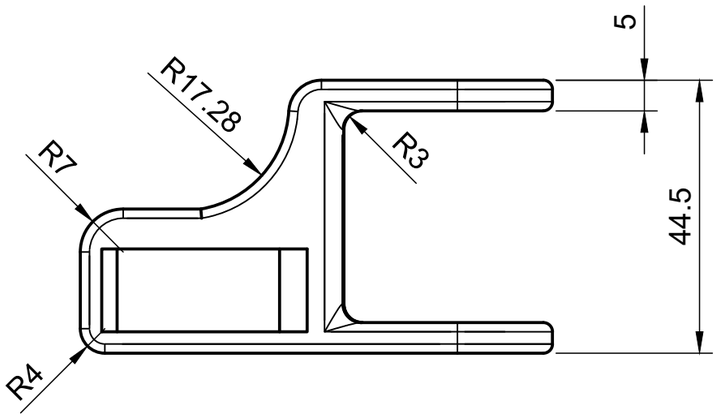


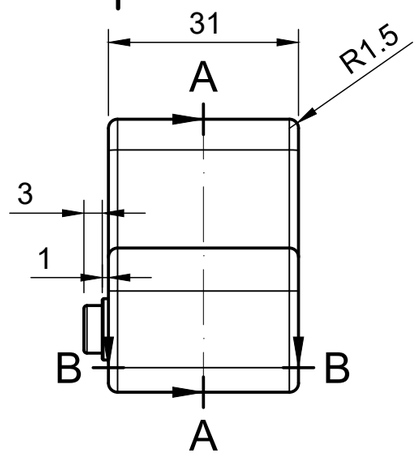




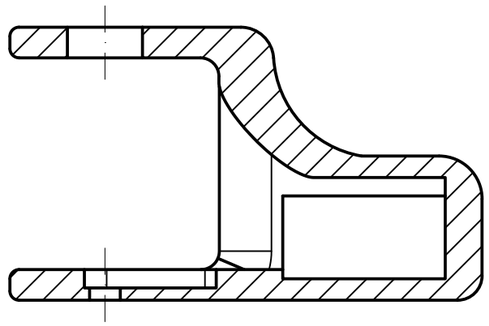


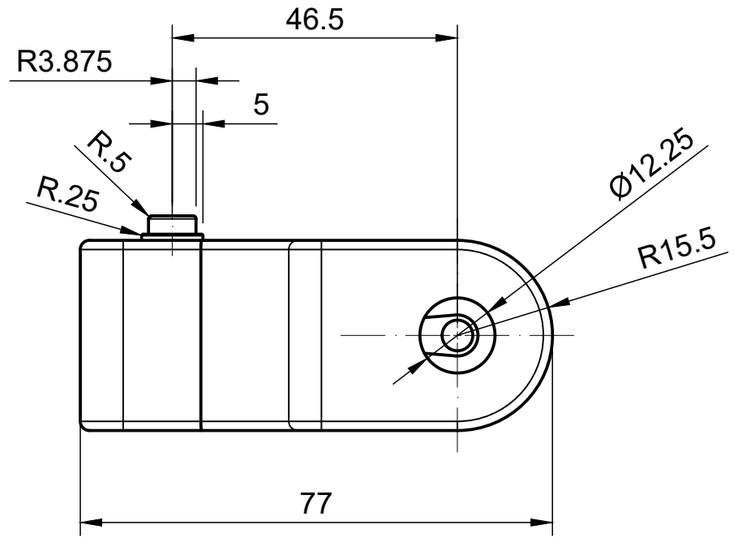
PROYECTO: RA-01	TÍTULO: Pieza superior		FECHA: 05/04/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA Escuela Técnica Superior de Ingeniería del Diseño	AUTOR: Ernest Folgado Brisa	FIRMA: Ernest FB	ESCALA: 1:2	UNIDADES: mm
	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 12	

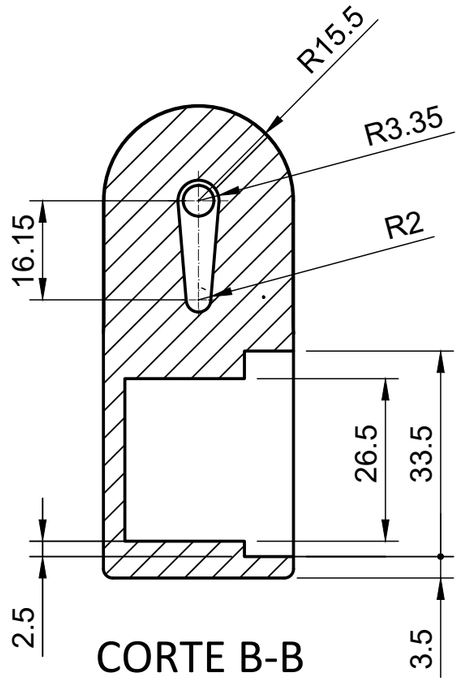




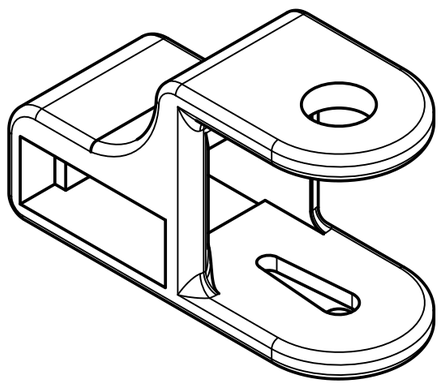
CORTE A-A





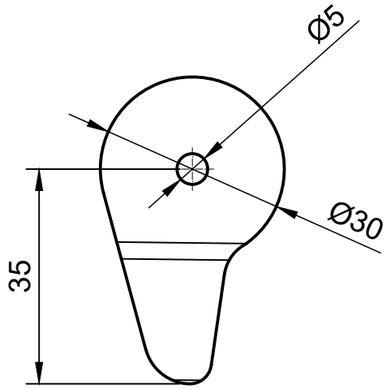


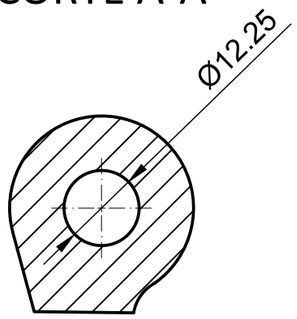
CORTE B-B

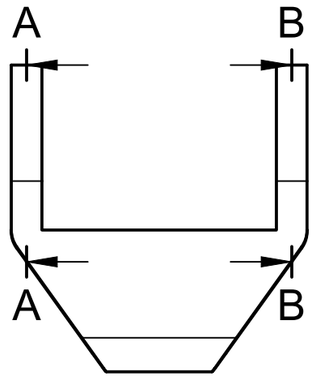


PROYECTO: RA-01	TÍTULO: Eslabón pata 1		FECHA: 10/04/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA Escuela Técnica Superior de Ingeniería del Diseño	AUTOR: Ernest Folgado Brisa	FIRMA: Ernest FB	ESCALA: 1:1	UNIDADES: mm
	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 13	

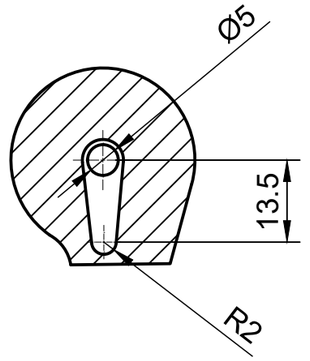
CORTE A-A

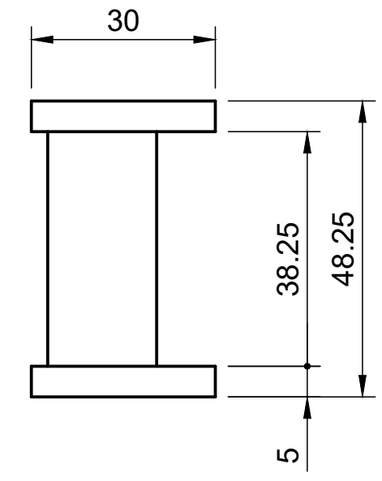


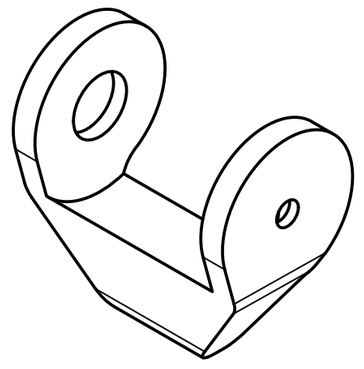




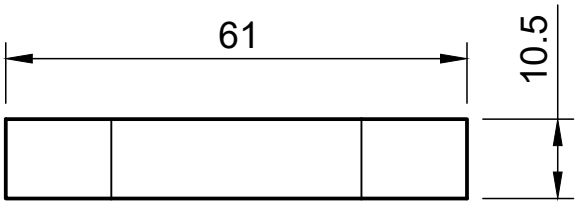
CORTE B-B

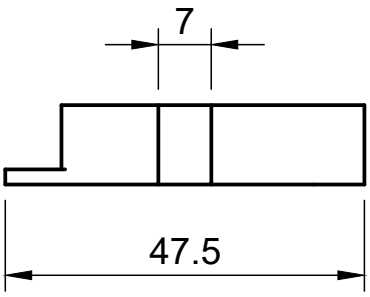


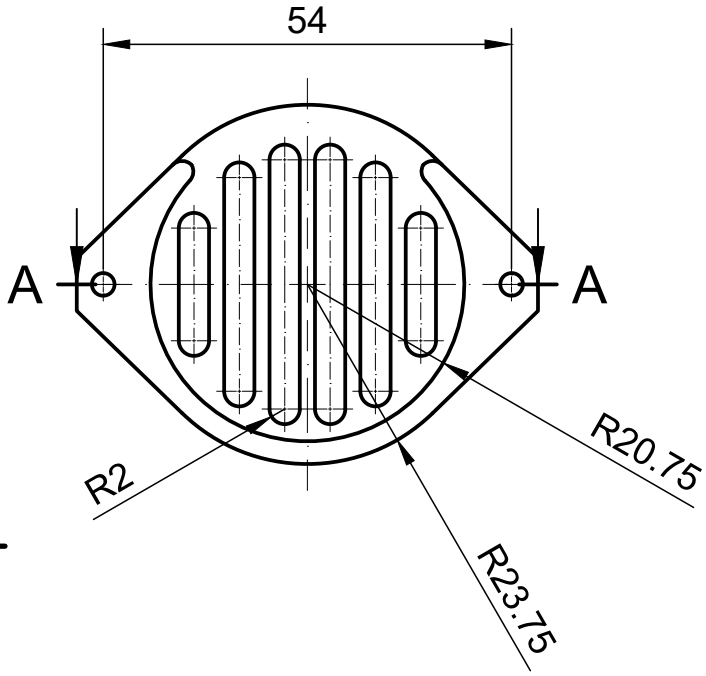


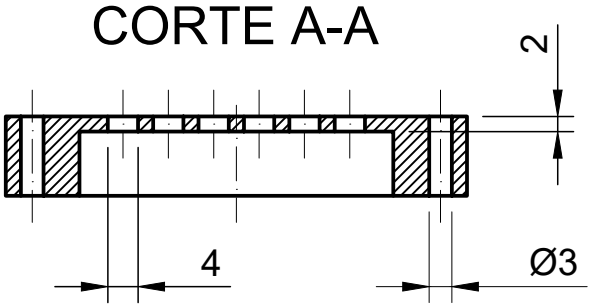


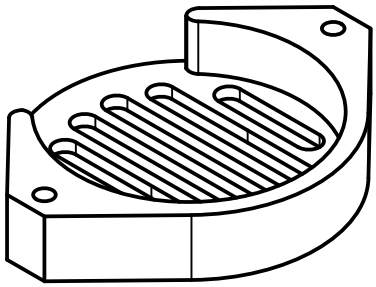
PROYECTO: RA-01	TÍTULO: Eslabón pata 2		FECHA: 10/04/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA Escuela Técnica Superior de Ingeniería del Diseño	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: 1:1	UNIDADES: mm
	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 14	



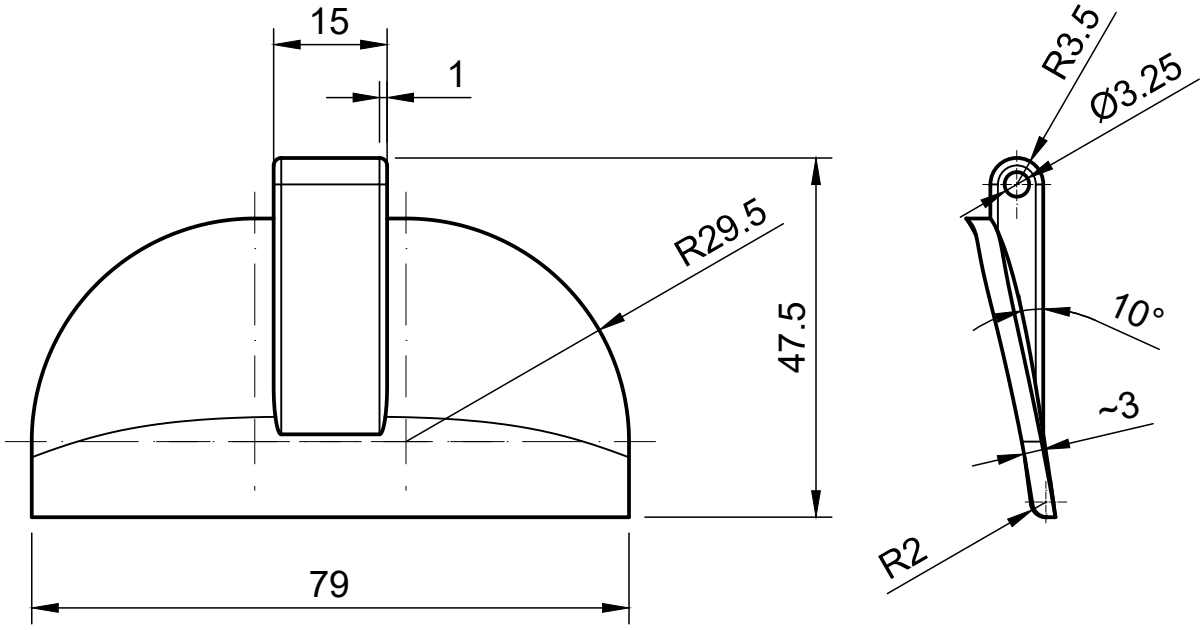


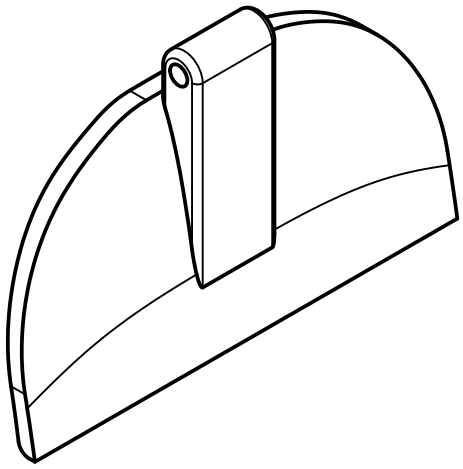




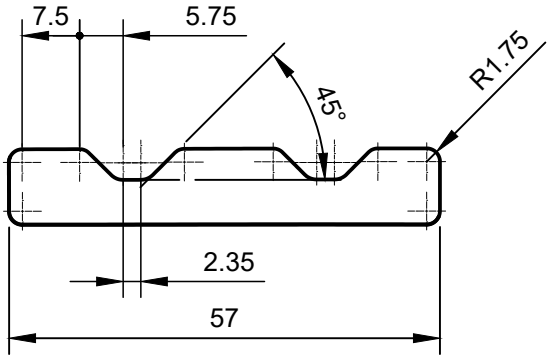


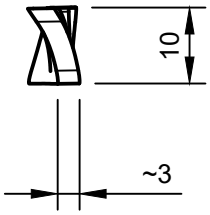
PROYECTO	TÍTULO:		FECHA:	
RA-01	Soporte altavoz		06/04/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: 1:1	UNIDADES: mm
Escuela Técnica Superior de Ingeniería del Diseño	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 15	

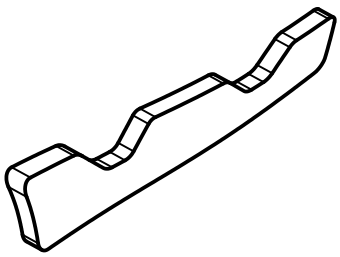




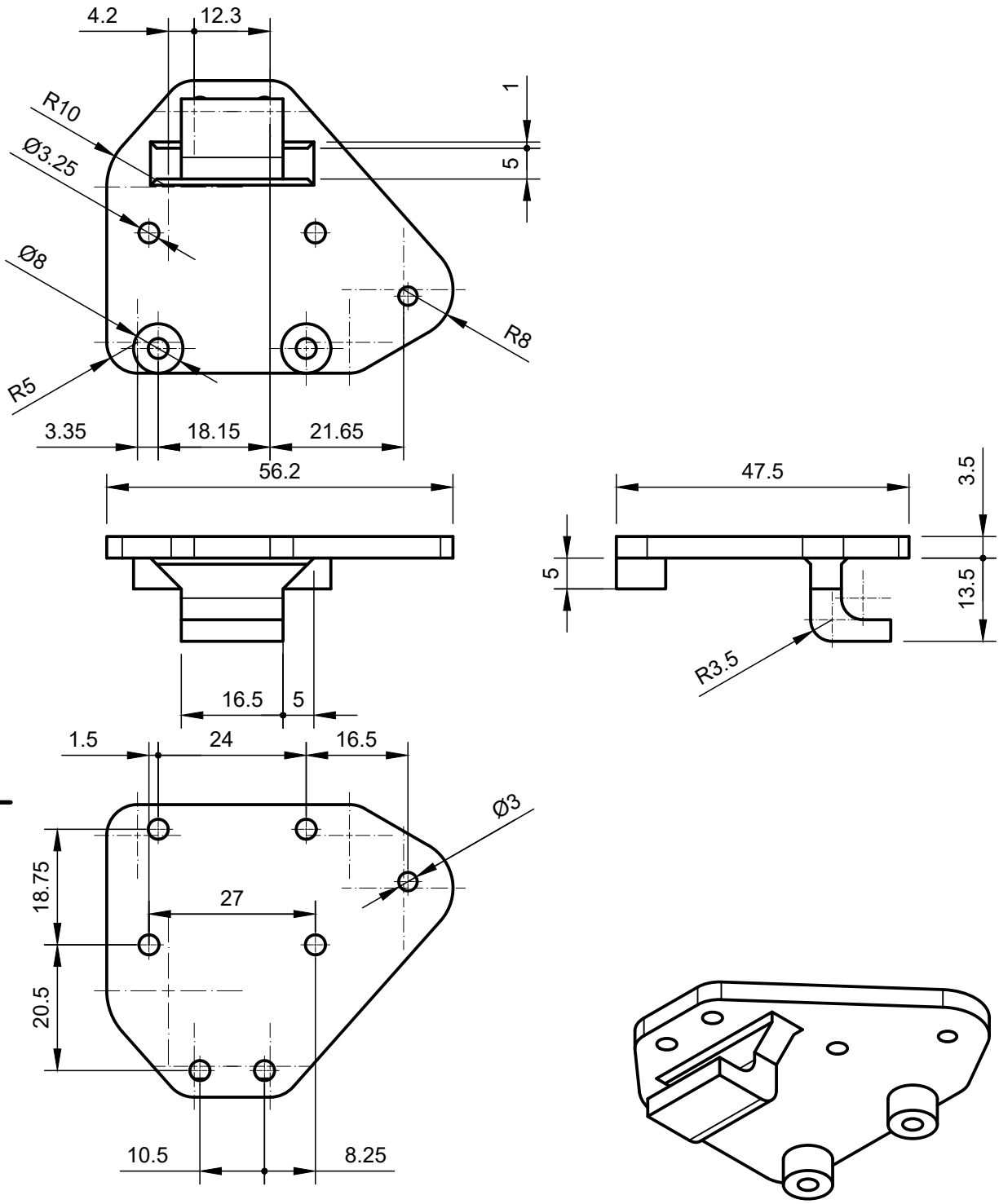
PROYECTO: RA-01	TÍTULO: Puerta		FECHA: 06/04/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA Escuela Técnica Superior de Ingeniería del Diseño	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: 1:1	UNIDADES: mm
	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 16	





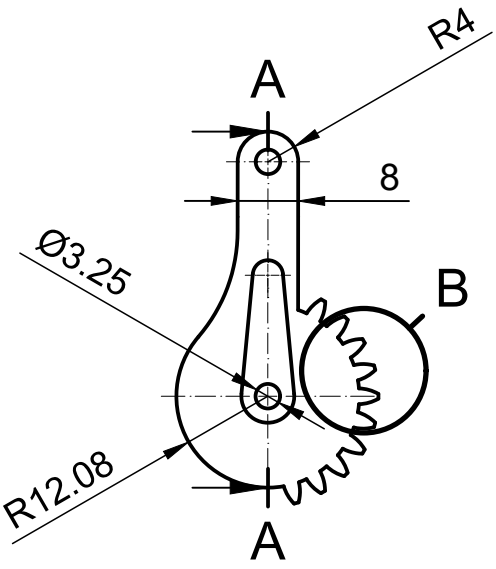


PROYECTO	TÍTULO:		FECHA:	
RA-01	Tapa luces		08/04/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: 1:1	UNIDADES: mm
Escuela Técnica Superior de Ingeniería del Diseño	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 17	

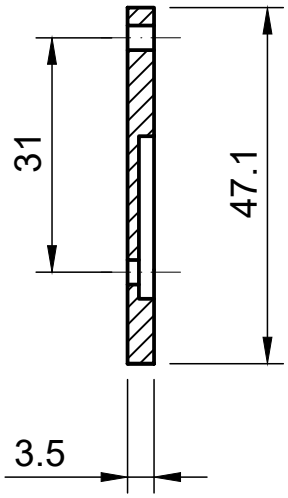


Los agujeros y filetess no acotados tienen un diámetro de 3mm y 10mm respectivamente.

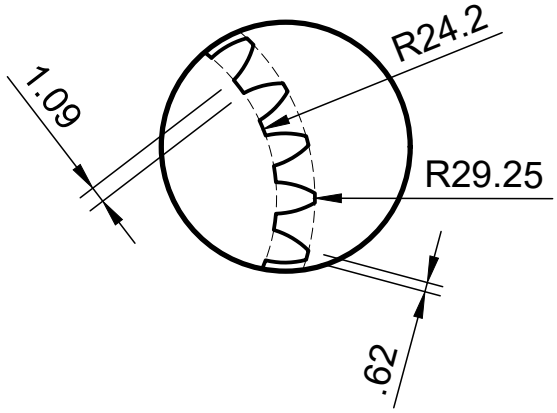
PROYECTO	TÍTULO:		FECHA:	
RA-01	Base pinza		19/04/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: 1:1	UNIDADES: mm
Escuela Técnica Superior de Ingeniería del Diseño	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 18	

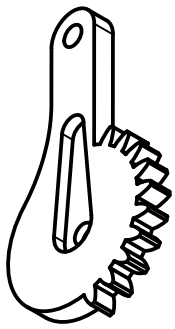


CORTE A-A



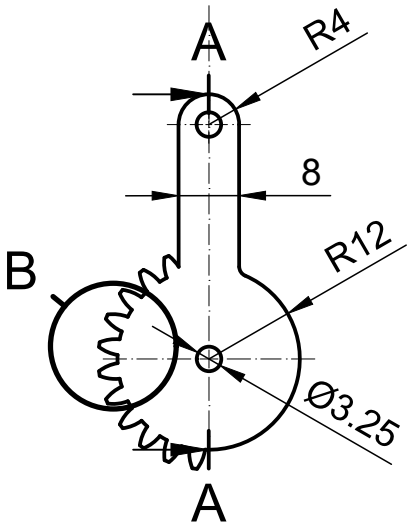
DETALLE B (2:1)

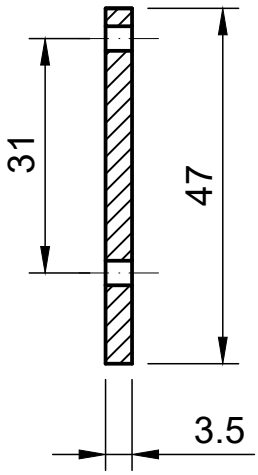




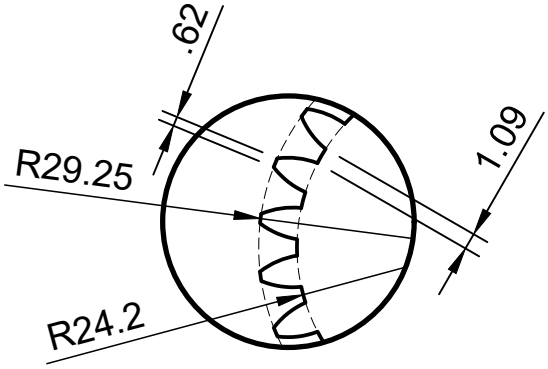
PROYECTO	TÍTULO:		FECHA:	
RA-01	Pieza pinza 1		10/04/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA	AUTOR: Ernest Folgado Brisa	FIRMA: Ernest FB	ESCALA: 1:1	UNIDADES: mm
Escuela Técnica Superior de Ingeniería del Diseño	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 19	

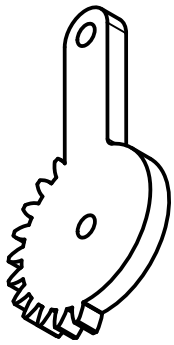
CORTE A-A



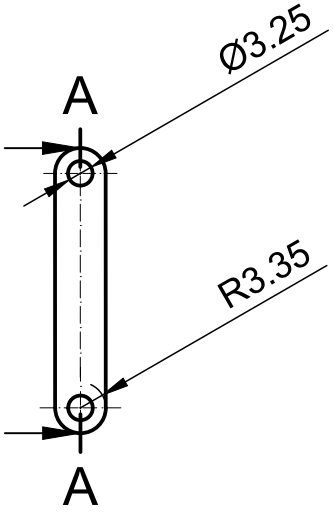


DETALLE B (2:1)

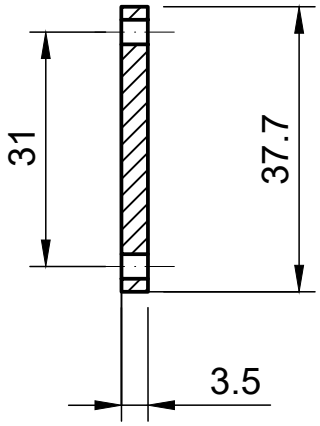




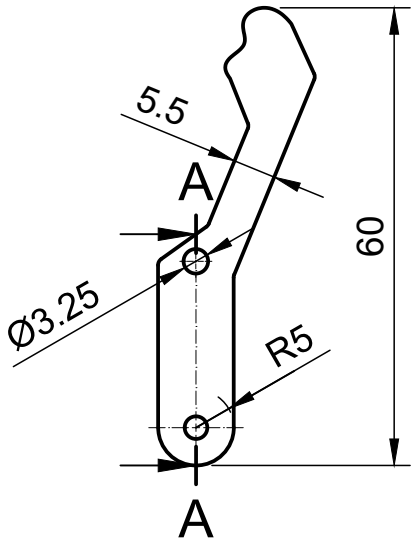
PROYECTO	TÍTULO:		FECHA:	
RA-01	Pieza pinza 2		10/04/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA	AUTOR: Ernest Folgado Brisa	FIRMA: Ernest FB	ESCALA: 1:1	UNIDADES: mm
Escuela Técnica Superior de Ingeniería del Diseño	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 20	



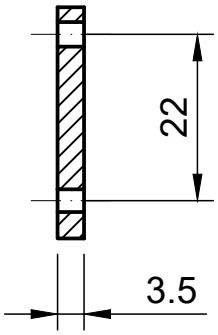
CORTE A-A

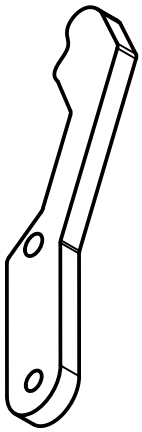


PROYECTO	TÍTULO:		FECHA:	
RA-01	Pieza pinza 3		10/04/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: 1:1	UNIDADES: mm
Escuela Técnica Superior de Ingeniería del Diseño	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 21	

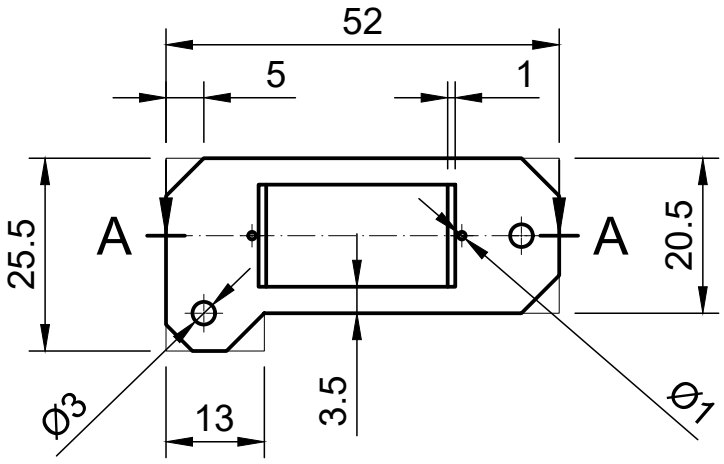


CORTE A-A

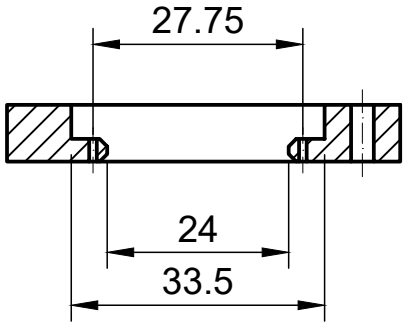


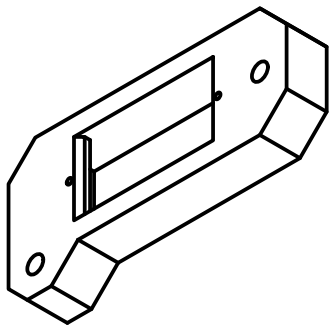


PROYECTO	TÍTULO:		FECHA:	
RA-01	Pieza pinza 4		10/04/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: 1:1	UNIDADES: mm
Escuela Técnica Superior de Ingeniería del Diseño	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 22	

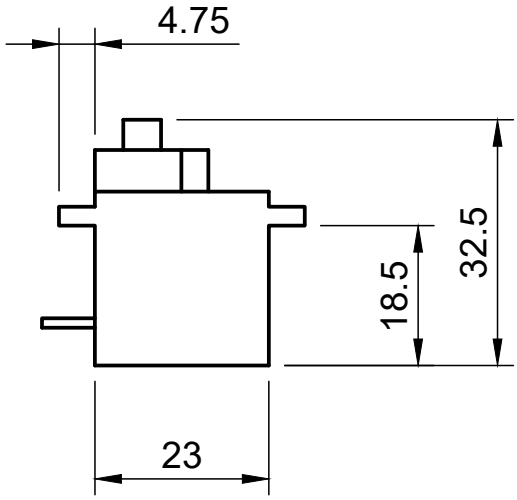


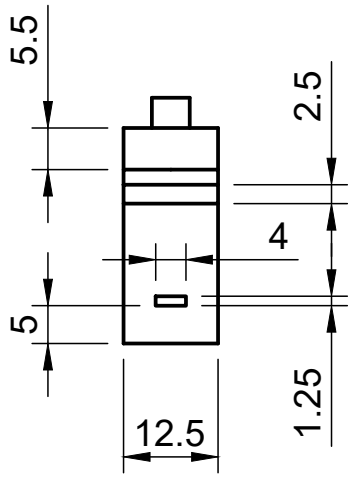
CORTE A-A

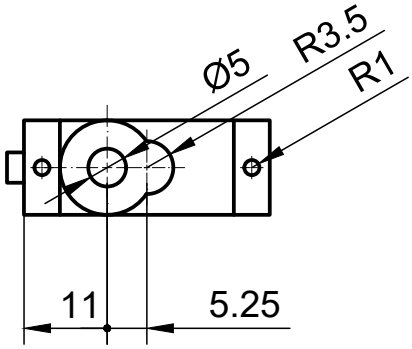


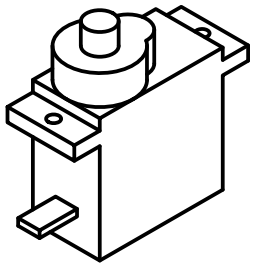


PROYECTO	TÍTULO:		FECHA:	
RA-01	Soporte servo		11/04/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA Escuela Técnica Superior de Ingeniería del Diseño	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: 1:1	UNIDADES: mm
	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 23	

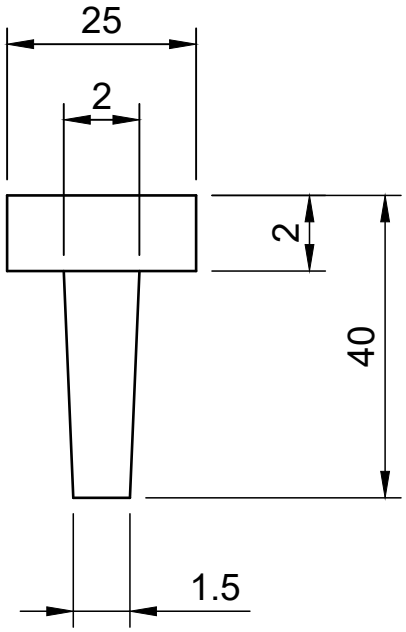


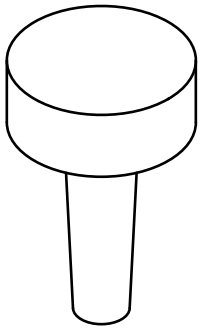




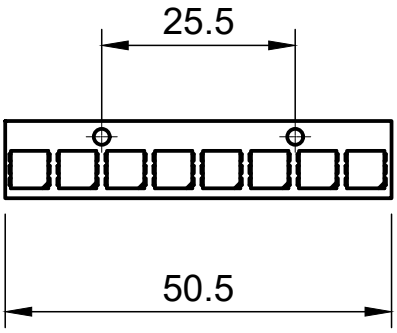


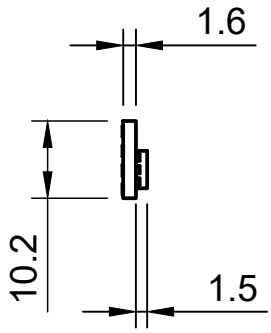
PROYECTO	TÍTULO:		FECHA:	
RA-01	Servomotor MG90S		11/04/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: 1:1	UNIDADES: mm
Escuela Técnica Superior de Ingeniería del Diseño	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 24	

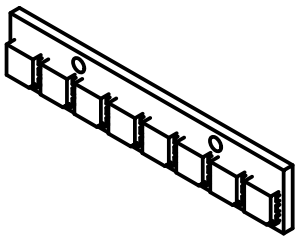




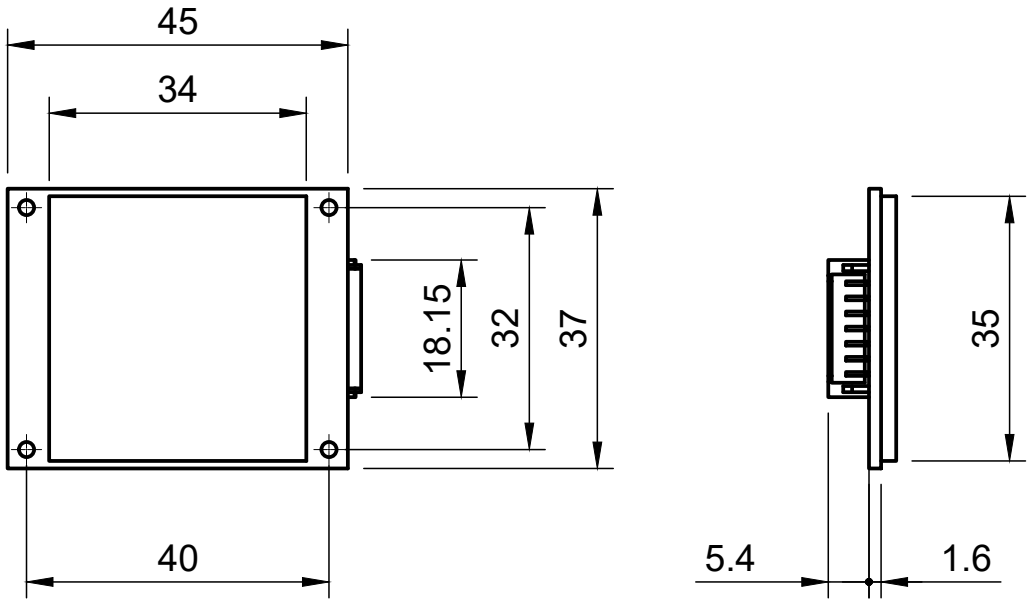
PROYECTO	TÍTULO:		FECHA:	
RA-01	Enganche		20/04/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: 5:1	UNIDADES: mm
Escuela Técnica Superior de Ingeniería del Diseño	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 25	

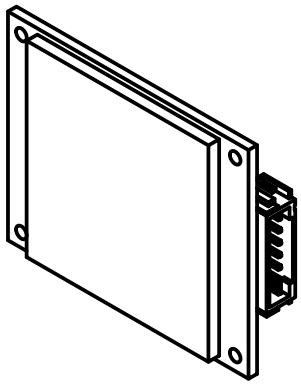




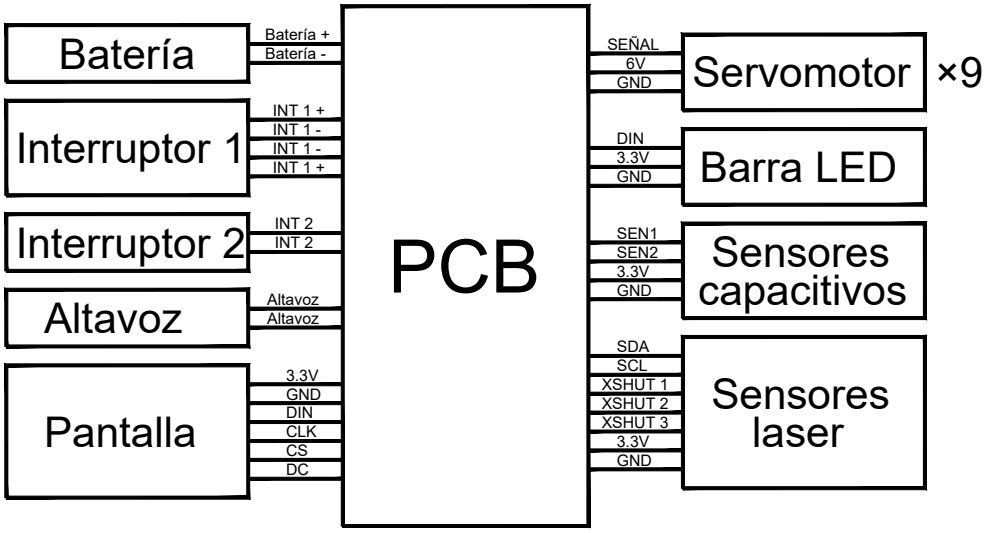


PROYECTO	TÍTULO:		FECHA:	
RA-01	Barra LED		14/05/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: 1:1	UNIDADES: mm
Escuela Técnica Superior de Ingeniería del Diseño	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 26	



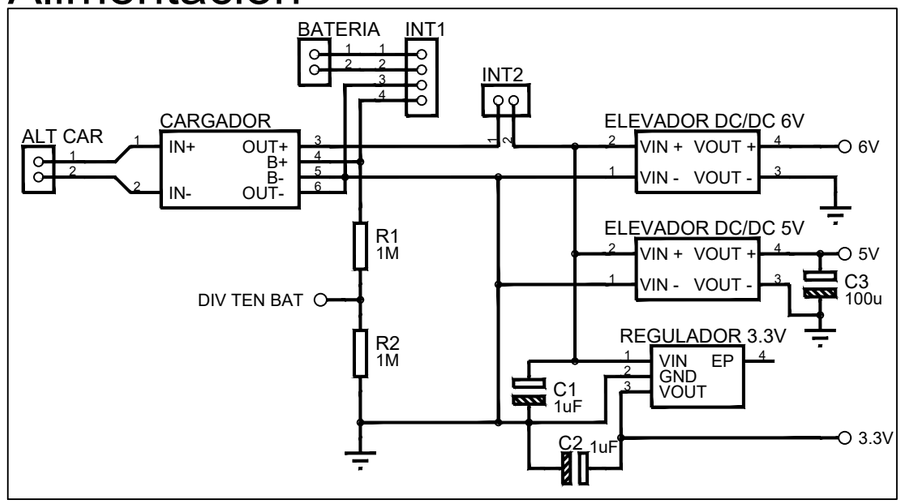


PROYECTO	TÍTULO:		FECHA:	
RA-01	Pantalla		14/05/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: 1:1	UNIDADES: mm
Escuela Técnica Superior de Ingeniería del Diseño	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 27	

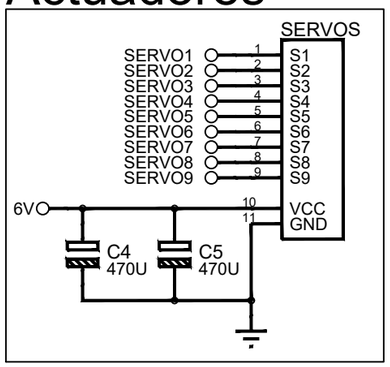


PROYECTO	TÍTULO:		FECHA:	
RA-01	Diagrama de conexiones		16/05/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA Escuela Técnica Superior de Ingeniería del Diseño	AUTOR: Ernest Folgado Brisa	FIRMA: Ernest FB	ESCALA: -	UNIDADES: -
	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 28	

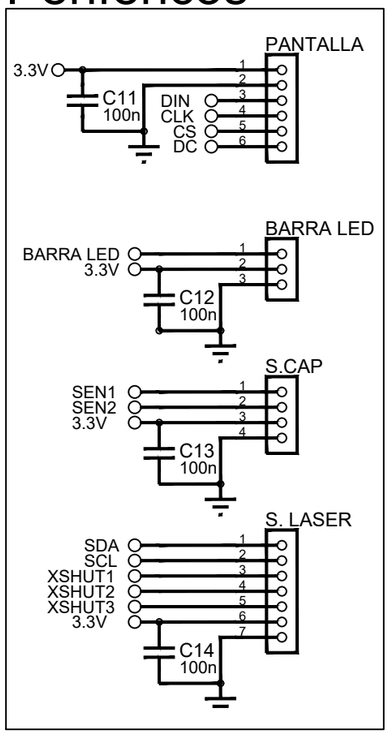
Alimentación



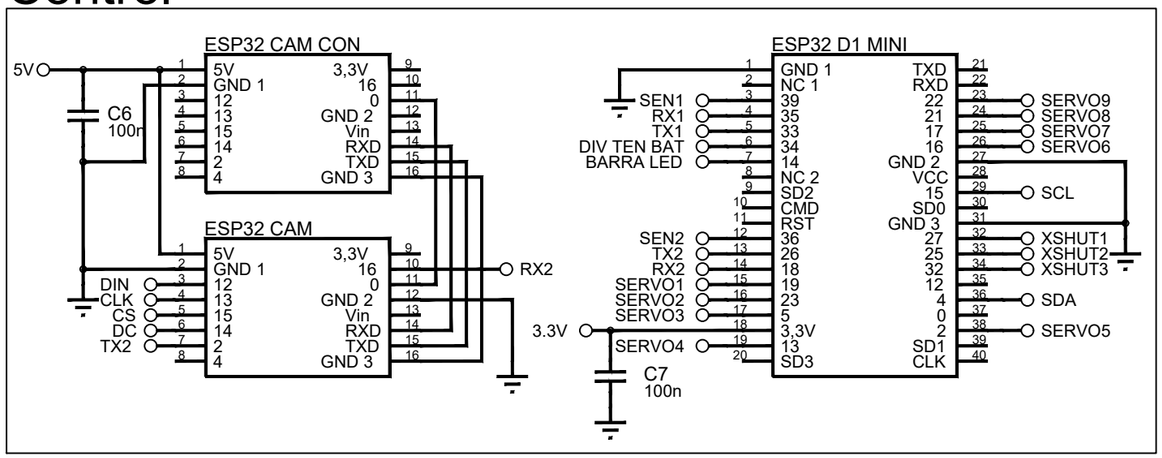
Actuadores



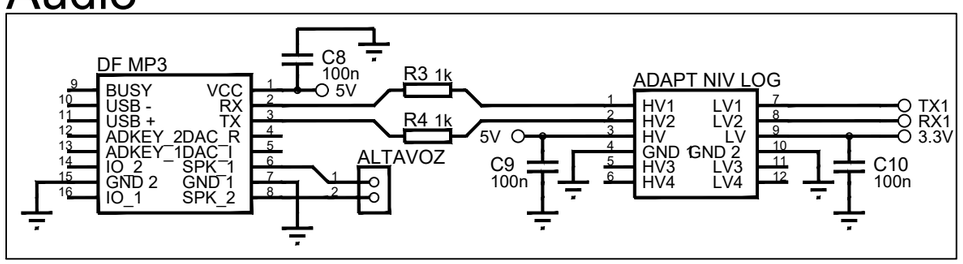
Periféricos



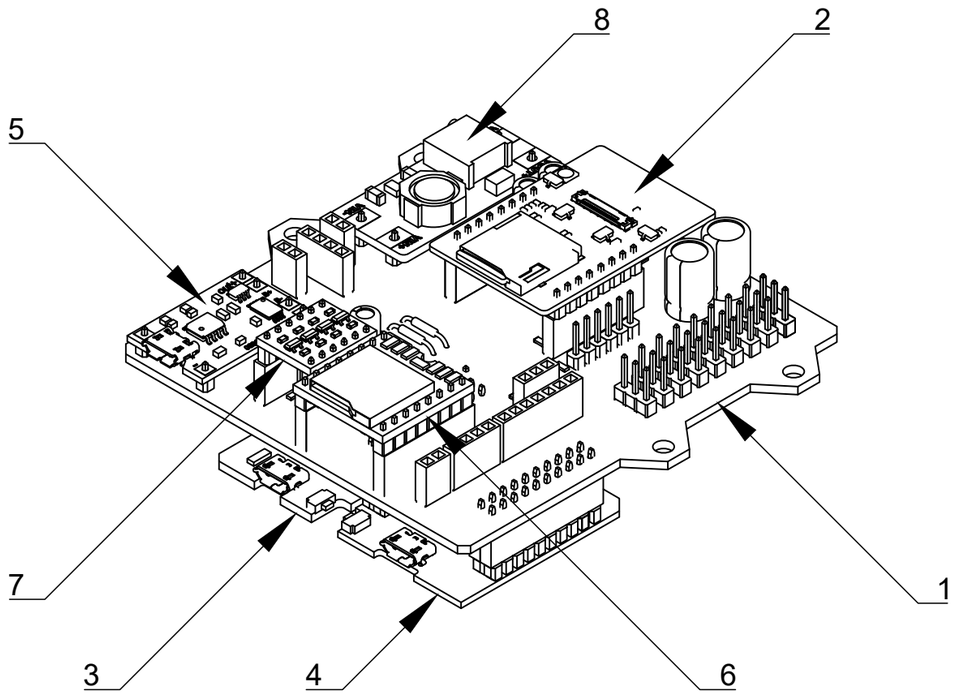
Control

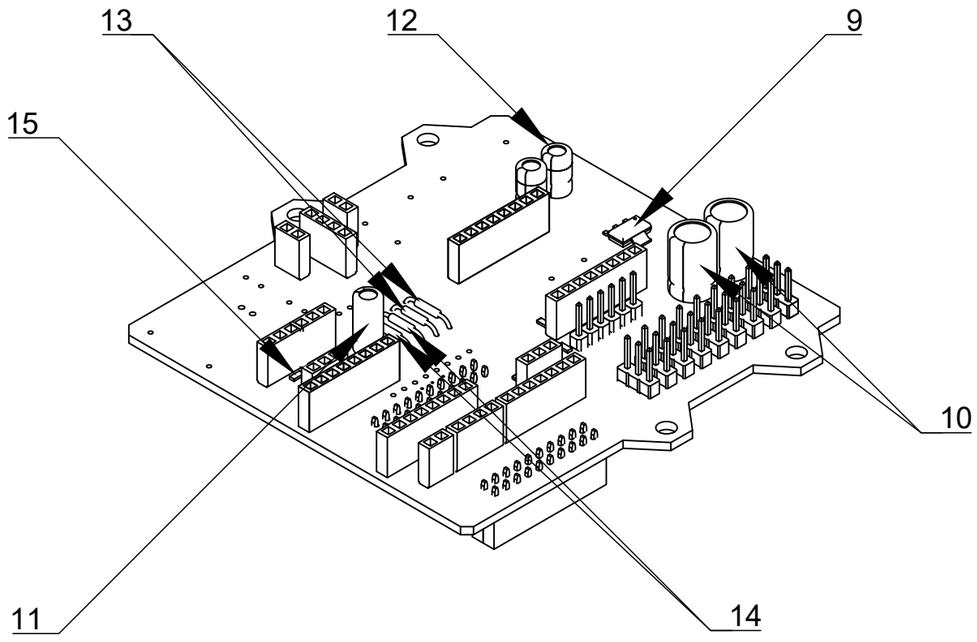


Audio



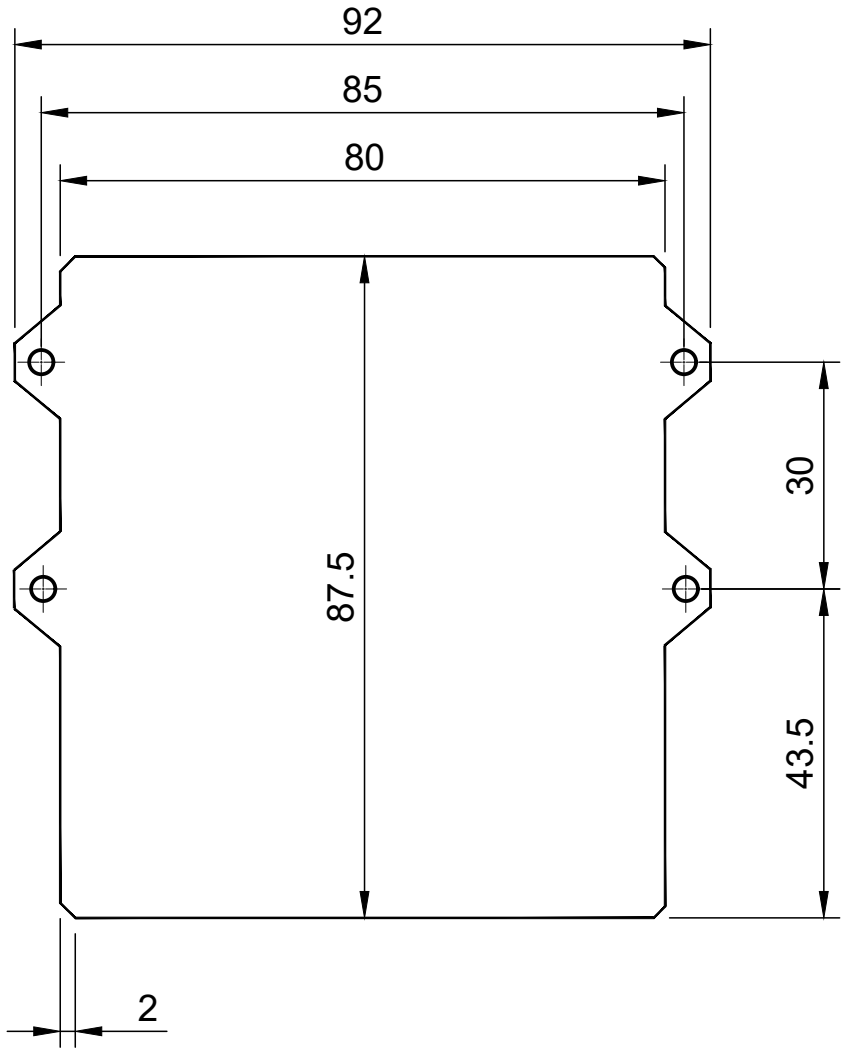
PROYECTO: RA-01	TÍTULO: Circuito electrónico PCB		FECHA: 14/05/2023
UNIVERSITAT POLITÈCNICA DE VALÈNCIA Escuela Técnica Superior de Ingeniería del Diseño	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: - / UNIDADES: -
	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 29



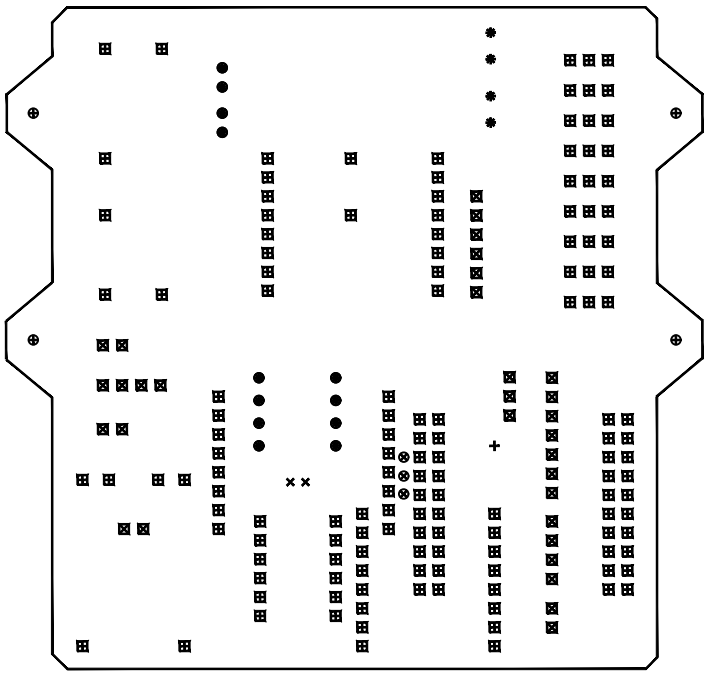


15	2	Resistencia 3 y 4		1kΩ, Norma E48 agujero pasante
14	2	Resistencia 1 y 2		1MΩ,Norma E48 agujero pasante
13	9	Cond. 6, 7, 8, 9, 10, 11, 12, 13 y 14		100nF, SMD, cerámico
12	2	Condensador 4 y 5		1µF, agujero pasante, electrolítico
11	1	Condensador 3		100µF, agujero pasante, electrolítico
10	2	Condensador 1 y 2		470µF, agujero pasante, electrolítico
9	1	Regulador 3.3V LDO MCP1755S	Distribuidor: electrón perdido	Referencia: 0583
8	2	Elevador DC/DC MT3608	Distribuidor: electrón perdido	Referencia:0364 El segundo se ubica debajo de la placa.
7	1	Convertidor de nivel lógico TE291	Fabricante: Movilideas (amazon)	ASIN: B08C4SJ7QG
6	1	DFPlayer Mini MP3	Fabricante DFRobot	SKU: DFR0299
5	1	Cargador LiPo micro USB TP4056	Distribuidor: Bricogeek	Referencia: BAT-034
4	1	ESP32 D1 mini	Fabricante AZDekivery (amazon)	EAN:4260581559441
3	1	Módulo de conversión serial USB-TTL	Fabricante: Diymore (amazon)	Incluido con el ESP32 cam
2	1	ESP32 cam	Fabricante: Diymore (amazon)	ASIN: B08X3GRK22
1	1	PCB	Plano Nº 31, 32, 33 y 34	
Marca	Cantidad	Denominación	Nº plano, fabricante o distribuidor	Observaciones

PROYECTO: RA-01	TÍTULO: Componentes PCB		FECHA: 16/05/2023
UNIVERSITAT POLITÈCNICA DE VALÈNCIA Escuela Técnica Superior de Ingeniería del Diseño	AUTOR: Ernest Folgado Brisa	FIRMA: Ernest FB	ESCALA: - / UNIDADES: -
	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 30

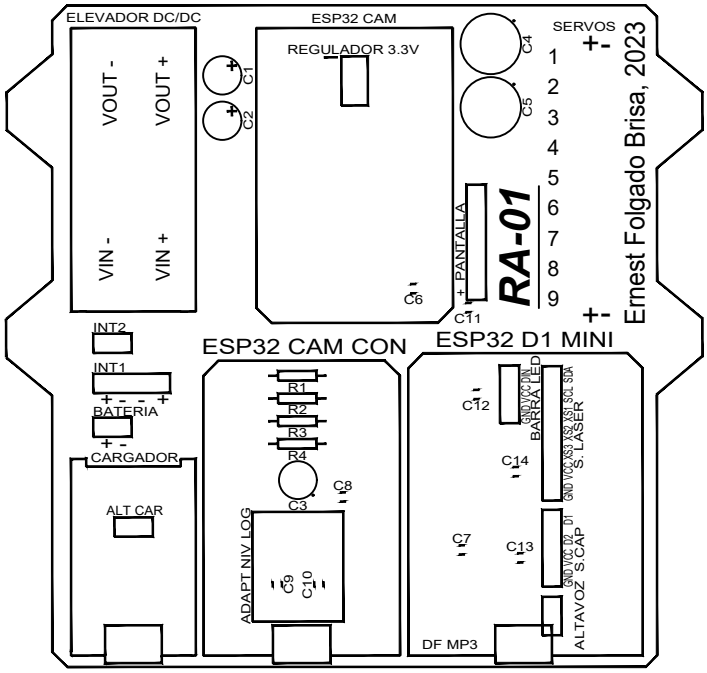


PROYECTO	TÍTULO:		FECHA:	
RA-01	Dimensiones PCB		14/05/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: 1:1	UNIDADES: mm
Escuela Técnica Superior de Ingeniería del Diseño	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 31	

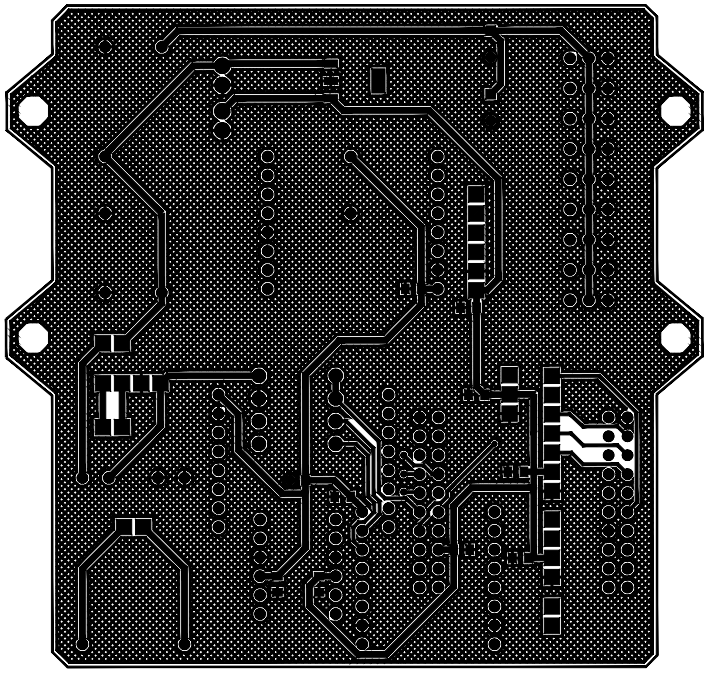


SYM	SIZE	PLATED	QTY
+	0.3mm	YES	1
×	0.75mm	YES	2
✱	0.85mm	YES	4
⊕	3.5mm	YES	4
⊗	15th	YES	3
◉	30th	YES	12
⊞	32th	YES	141
⊠	40th	YES	32

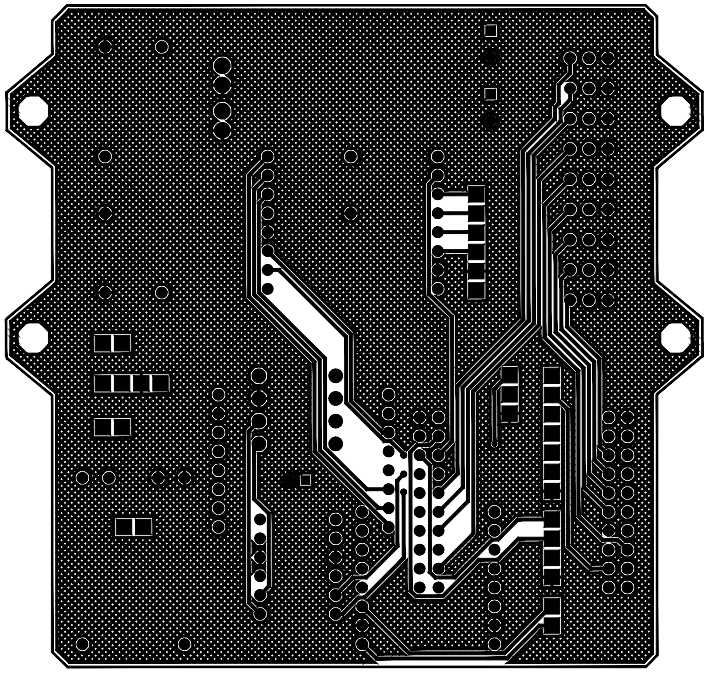
PROYECTO	TÍTULO:		FECHA:	
RA-01	Taladros PCB		19/05/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA	AUTOR: Ernest Folgado Brisa	FIRMA: Ernest FB	ESCALA: 1:1	UNIDADES: mm
Escuela Técnica Superior de Ingeniería del Diseño	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 32	



PROYECTO	TÍTULO:		FECHA:	
RA-01	Serigrafía PCB		14/05/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: 1:1	UNIDADES: mm
Escuela Técnica Superior de Ingeniería del Diseño	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 33	



PROYECTO	TÍTULO:		FECHA:	
RA-01	Capa cobre superior		14/05/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: 1:1	UNIDADES: mm
Escuela Técnica Superior de Ingeniería del Diseño	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 34	



PROYECTO	TÍTULO:		FECHA:	
RA-01	Capa cobre inferior		14/05/2023	
UNIVERSITAT POLITÈCNICA DE VALÈNCIA	AUTOR: Ernest Folgado Brisa	FIRMA:	ESCALA: 1:1	UNIDADES: mm
Escuela Técnica Superior de Ingeniería del Diseño	REVISADO POR: Javier Ibáñez Civera	FIRMA:	Nº DE PLANO: 35	

PLIEGO DE CONDICIONES

Diseño y desarrollo de un robot cuadrúpedo dedicado al tratamiento de niños con autismo.

Trabajo final del:	Grado en Ingeniería electrónica industrial y Automática
Realizado por:	Ernest Folgado Brisa
Tutorizado por:	Javier Ibáñez Civera
Curso académico:	2022/2023

Contenido pliego de condiciones

Pliego de condiciones

1. Objeto

La presente especificación técnica indica los pasos a seguir para llevar a cabo el montaje de una réplica del robot RA-01. Así mismo, también se describe como preparar el entorno de programación y subir el código correspondiente.

Cabe destacar que este documento se ha realizado para el robot RA-01 y puede no ser útil para futuras versiones del robot.

2. Materiales

2.1. Componentes electrónicos

Se requieren los siguientes componentes electrónicos para el montaje del dispositivo:

- Batería LiPo 6.000mAh (1)
- Interruptor Basculante (2)
- Cargador LiPo micro USB TP4056 (1)
- Elevador DC/DC MT3608 (2)
- Regulador 3.3V LDO MCP1755S (1)
- ESP 32 D1 mini (1)
- ESP 32 cam (1)
- Módulo conversión serial USB-TTL, para ESP 32 cam (1)
- Cámara OV2640 75mm (1)
- Módulo de reproductor de MP, DFPlayer Mini MP3 (1)
- Servomotor MG90S (9)
- Sensor laser, módulo GY-530 (4)
- Módulo interruptor táctil TTP223 (2)
- Barra LED RGB WS2812 8x5050 (2)
- Pantalla OLED RGB 1.5'' (1)
- Altavoz 8Ω, 2W y 4mm de diámetro (1)
- Convertidor de nivel lógico módulo TE291 (1)
- Condensador electrolítico agujero pasante 470µF (2)
- Condensador electrolítico agujero pasante 100µF (1)
- Condensador electrolítico agujero pasante 1µf (2)
- Condensador cerámico SMD 100nF (9)
- Resistencia agujero pasante 1MΩ (2)
- Resistencia agujero pasante 1kΩ (2)
- Conector JST hembra, 2 pines (1)
- Header macho 2,54mm (40x) (2)
- Header hembra 2,54mm (40x) (3)

Entre paréntesis se indica la cantidad necesaria de cada pieza.

2.2. PCB

La placa de circuito impresa debe de ser fabricada por una empresa dedicada a ello. Los archivos Gerber correspondientes se encuentran disponibles en la siguiente dirección web:

https://github.com/ErnestFB/RA-01

2.3. Piezas impresas 3D

Para el montaje del robot se requerirán múltiples piezas impresas en 3D, concretamente:

- Pieza inferior (1)
- Pieza intermedia (1)
- Pieza superior (1)
- Eslabón pata 1 (2)
- Eslabón pata 1, simétrico (2)
- Eslabón pata 2 (2)
- Eslabón pata 2, simétrico (2)
- Soporte altavoz (1)
- Puerta (1)
- Tapa luces (2)
- Enganche (4)
- Base pinza (1)
- Pieza pinza 1 (1)
- Pieza pinza 2 (1)
- Pieza pinza 3 (4)
- Pieza pinza 4 (2)

Entre paréntesis se indica la cantidad necesaria de cada pieza.

En caso de no disponer de los medios necesarios para fabricar las piezas, estas se deberán de encargar a una empresa que ofrezca el servicio.

Respecto la impresión de las piezas, se deberá utilizar impresión estereolitografía (resina) para las piezas “pieza inferior” y “pieza superior”.

Para la pieza “tapa luces” se deberá de utilizar impresión FDM con filamento translúcido.

En cuanto a el reto de piezas, el material utilizado no es importante, se recomienda utilizar modelado por deposición fundida.

2.4. Rodamientos

Par el montaje del robot se precisa de 8 rodamientos Mr128zz 8x12x3.5mm. Cuyas dimensiones son de 8mm de diámetro interior, 12mm de diámetro exterior y 3.5mm de altura.

3. Condiciones de ejecución

3.1. Soldadura de componentes

En primer lugar, se comprobará que se cuenta con la totalidad de componentes electrónicos necesarios para el montaje del dispositivo y se ajustarán los elevadores de tensión.

Para regular los elevadores de tensión se deberán de alimentar a una fuente de alimentación ajustada a 3.8V y se comprobará la tensión de salida utilizando un voltímetro o un osciloscopio. La tensión de salida varia en función a una resistencia variable. Se deberá de utilizar un destornillador plano pequeño para ello. Un regulador debe de ajustarse a 5V y otro a 6V.

Una vez reunidos los componentes necesarios y ajustados los reguladores se iniciará la soldadura de componentes de la PCB siguiendo el orden indicado:

1º Los componentes SMD (regulador 3.3V y condensadores 110nF).

2º Las resistencias.

3º Los headers macho.

4º Los headers hembra.

5º El regulador de tensión situado en la cara inferior (5V).

6º El otro regulador de tensión (6V) y el cargador LiPo.

7º Los condensadores de agujero pasante.

Cabe tener en cuenta los siguientes aspectos a la hora de soldar la PCB:

- En caso de duda, se deberá consultar la colocación de los componentes en el plano Nº 30, componentes PCB.
- Durante la soldadura de los distintos componentes se cortará el excedente de los terminales.
- Para la colocación de los elevadores y del cargador se deberán de utilizar headers macho.

Por otro lado, se deberán de soldar los componentes externos de la PCB a cables, que así mismo se soldarán a un header macho en el otro extremo.

Esto último no será necesario para los componentes como la batería o la pantalla, que ya incluyen cables con conectores adecuados. Siendo los componentes a los que si se les deban de soldar cables: los interruptores y los sensores de distancia y contacto.

Finalmente se deberán de encajar los componentes correspondientes en sus zócalos.

3.2. Montaje

Finalizado la soldadura de componentes se procederá al montaje del dispositivo.

En primer lugar, se montarán las patas, plano 7.

Cuando ya se hayan montado las 4 patas se procederá al ensamblaje de la parte inferior, donde se utilizará velcro adhesivo para fijar la batería, plano 3.

Después, se montará la pinza, plano 9, y la parte intermedia, plano 4.

Una vez estén llegado a este punto, se juntarán las partes inferior y intermedia, plano 2 y se montará la parte superior por separado, plano 5. Para colocar los enganches se utilizará pegamento para plásticos instantáneo y para la fijación de los sensores de proximidad y la pantalla se utilizará cinta adhesiva a doble cara.

Finalmente, se añadirá la parte superior, plano 2. Para ello se deberá de apartar los cables, evitando que estos se interpongan entre la parte superior y los tornillos donde se encaja. Además, se deberá de colocar con cuidado la cámara en su posición. Para que esta se sostenga en su sitio, se rodeará de material adhesivo.

4. Comprobación del correcto funcionamiento

4.1. Comprobación electrónica

Cuando se haya finalizado la soldadura de componentes se deberán de realizar las siguientes comprobaciones.

En primer lugar, una inspección visual, revisando los puntos de soldadura y que los condensadores electrolíticos estén correctamente polarizados.

A continuación, utilizando un “tester”, se comprobará que no haya ningún cortocircuito entre alimentación y masa.

Finalmente se conectarán todos los componentes, se alimentará el circuito, se subirá el código a los microcontroladores y se verificará que todo funciona correctamente.

4.2. Comprobación general final

Una vez montado el dispositivo, se revisará que todas las piezas se hayan encajado correctamente. Acto seguido se encenderá el robot y se probará su correcto funcionamiento.

Para ello se deberá de conectar al dispositivo desde un ordenador o un “smartphone” y se hará que este retransmita video, que reproduzca algún audio y que camine. También se tendrá que levantar para comprobar que los sensores de distancia funcionan correctamente y se acariciará para comprobar el funcionamiento de los sensores capacitivos.

PRESUPUESTO

Diseño y desarrollo de un robot cuadrúpedo dedicado al tratamiento de niños con autismo.

Trabajo final del:	Grado en Ingeniería electrónica industrial y Automática
Realizado por:	Ernest Folgado Brisa
Tutorizado por:	Javier Ibáñez Civera
Curso académico:	2022/2023

Contenido presupuesto

1. Precios elementales

Ref	ud.	Descripción	Precio (€)
Materiales			
m1	ud.	Placa de circuito impreso (PCB)	16,5
m2	ud.	Baterria LiPo 6.000 mAh	17,80
m3	ud.	Interruptor basculante	0,75
m4	ud.	Cargador LiPo micro USB TP4056	1,20
m5	ud.	Elevador DC/DC MT3608	1,25
m6	ud.	Regulador 3,3V LDO MCP1755S	1,30
m7	ud.	ESP 32 D1 mini	11,49
m8	ud.	ESP 32 cam (incluye módulo de conversión serial USB-TTL)	16,59
m9	ud.	Cámara OV2640 75mm	3,40
m10	ud.	Módulo reproductor de MP3, DFPlayer Mini MP3	2,95
m11	ud.	Servomotor MG90S (incluye los tornillos para su montaje)	4,50
m12	ud.	Rodamiento Mr128zz 8x12x3.5mm	0,31
m13	ud.	Sensor laser, módulo GY-530	1,97
m14	ud.	Mdulo interruptor tactil TTP223	0,60
m15	ud.	Barra LED RGB WS2812 8x5050	1,25
m16	ud.	Pantalla OLED RGB 1,5''	21,90
m17	ud.	Altavoz 8Ω, 2W y 4mm de diámetro	6,81
m18	ud.	Convertidor de nivel lógico módulo TE291	1,08
m19	ud.	Condensador electrolítico agujero pasante 470uF	0,50
m20	ud.	Condensador electrolítico agujero pasante 100uF	0,08
m21	ud.	Condensador electrolítico agujero pasante 1uf	0,08
m22	ud.	Condensador cerámico SMD 100nF	0,35
m23	ud.	Resistencia agujero pasante 1MΩ	0,30
m24	ud.	Resistencia agujero pasante 1kΩ	0,30
m25	ud.	Conector JST hembra, 2 pines	0,35
m26	ud.	Header macho 2,54mm (40x)	0,25
m27	ud.	Header hembra 2,54mm (40x)	0,50
m28	ud.	Pieza impresa, pieza inferior	10,41
m29	ud.	Pieza impresa, pieza intermedia	3,81
m31	ud.	Pieza impresa, pieza superior	8,47
m32	ud.	Pieza impresa, eslabón pata 1 (o su versión simétrica)	4,35
m33	ud.	Pieza impresa, eslabón pata 2 (o su versión simétrica)	2,06
m34	ud.	Pieza impresa, soporte altavoz	1,02
m35	ud.	Pieza impresa, puerta	1,36
m36	ud.	Pieza impresa, tapa luces	4,00
m37	ud.	Pieza impresa, enganche	0,25
m38	ud.	Pieza impresa, base pinza	0,84
m39	ud.	Pieza impresa, pinza 1	0,42
m40	ud.	Pieza impresa, pinza 2	0,38
m41	ud.	Pieza impresa, pinza 3	0,32
m42	ud.	Pieza impresa, pinza 4	0,35
m43	conjunto	Tornillos arandelas y tuercas necesarios para el montaje	3,00
m44	mes	Amortización Software Fusion 360	73,00
m45	mes	Amortización Software Autocad	247,00
m46	mes	Amortización software proteus (5620 € durante 5 años)	93,67

Ref	ud.	Descripción	Precio (€)
m47	mes	Amortización ordenador (900€ 1 año)	75,00
m48	mes	Gastos alquiler local	900,00
m49	mes	Gastos local (luz, calefacción, etc)	350,00
M.O.D.			
h1	h	Ingeniero electrónico	11,00
h2	h	Técnico	9,00

2. Tabla de precios descompuestos

2.1. Desarrollo de la PCB y la electrónica

Ref	ud.	Descripción	Precio (€)	Cantidad	Parcial (€)
d1	ud.	Desarrollo y diseño de la placa de la PCB mediante software. Y comporobación práctica del correcto funcionamiento de la misma.	1940,29	1	1940,29
Materiales					
m1	ud.	Placa de circuito impreso (PCB)	16,5	3	49,50
m2	ud.	Baterria LiPo 6.000 mAh	17,80	1	17,80
m4	ud.	Cargador LiPo micro USB TP4056	1,20	1	1,20
m5	ud.	Elevador DC/DC MT3608	1,25	2	2,50
m6	ud.	Regulador 3,3V LDO MCP1755S	1,30	2	2,60
m7	ud.	ESP 32 D1 mini	11,49	2	22,98
m8	ud.	ESP 32 cam	16,59	1	16,59
m9	ud.	Cámara OV2640 75mm	3,40	1	3,40
m10	ud.	Módulo DFPlayer Mini MP3	2,95	1	2,95
m11	ud.	Servomotor MG90S	4,50	9	40,50
m13	ud.	Sensor laser, módulo GY-530	1,97	5	9,87
m14	ud.	Mdulo interruptor tactil TTP223	0,60	2	1,20
m15	ud.	Barra LED RGB WS2812 8x5050	1,25	3	3,75
m16	ud.	Pantalla OLED RGB 1,5''	21,90	2	43,80
m17	ud.	Altavoz 8Ω, 2W y 4mm de diámetro	6,81	1	6,81
m18	ud.	Convertidor de nivel lógico TE291	1,08	1	1,08
m19	ud.	Condensador electrolítico 470uF	0,50	4	2,00
m20	ud.	Condensador electrolítico 100uF	0,08	1	0,08
m21	ud.	Condensador electrolítico 1uf	0,08	2	0,16
m22	ud.	Condensador cerámico SMD 100nF	0,35	27	9,45
m23	ud.	Resistencia agujero pasante 1MΩ	0,30	4	1,20
m24	ud.	Resistencia agujero pasante 1kΩ	0,30	4	1,20
m25	ud.	Conector JST hembra, 2 pines	0,35	2	0,70
m26	ud.	Header macho 2,54mm (40x)	0,25	5	1,25
m27	ud.	Header hembra 2,54mm (40x)	0,50	8	4,00
m46	mes	Amortización software proteus	93,6666667	2	187,33
M.O.D.					
h1	h	Ingeniero electrónico	11,00	80	880,00
h2	h	Técnico	9,00	50	450,00
Medios auxiliares					
	%	M. A. sobre costes directos	10%	1763,90	176,39

2.2. Desarrollo de las piezas 3D

Ref	ud.	Descripción	Precio (€)	Cantidad	Parcial (€)
d2	ud.	Desarrollo y diseño de las piezas 3D mediante software	1235,63	1	1235,63
Materiales					
m28	ud.	Pieza impresa, pieza inferior	10,41	1	10,41
m29	ud.	Pieza impresa, pieza intermedia	3,81	1	3,81
m31	ud.	Pieza impresa, pieza superior	8,47	1	8,47
m32	ud.	Pieza impresa, eslabón pata 1 (o su versión	4,35	4	17,40
m33	ud.	Pieza impresa, eslabón pata 2 (o su versión	2,06	4	8,24
m34	ud.	Pieza impresa, soporte altavoz	1,02	1	1,02
m35	ud.	Pieza impresa, puerta	1,36	1	1,36
m36	ud.	Pieza impresa, tapa luces	4,00	2	8,00
m37	ud.	Pieza impresa, enganche	0,25	4	1,00
m38	ud.	Pieza impresa, base pinza	0,84	1	0,84
m39	ud.	Pieza impresa, pinza 1	0,42	1	0,42
m40	ud.	Pieza impresa, pinza 2	0,38	1	0,38
m41	ud.	Pieza impresa, pinza 3	0,32	4	1,28
m42	ud.	Pieza impresa, pinza 4	0,35	2	0,70
m43	conjunto	Tornillos arandelas y tuercas necesarios pa	3,00	1	3,00
m44	mes	Amortización Software Fusion 360	73,00	1	73,00
M.O.D.					
h1	h	Ingeniero electrónico	11,00	90	990,00
Medios auxiliares					
	%	M. A. sobre costes directos	10%	1063,00	106,30

2.3. Desarrollo de software

Ref	ud.	Descripción	Precio (€)	Cantidad	Parcial (€)
d3	ud.	Desarrollo de los códigos necesarios para el funcionamiento del dispositivo	938,39	1	938,39
Materiales					
m7	ud.	ESP 32 D1 mini	11,49	1	11,49
m8	ud.	ESP 32 cam	16,59	1	16,59
h1	h	Ingeniero electrónico	11,00	75	825,00
Medios auxiliares					
	%	M. A. sobre costes directos	10%	853,08	85,31

2.4. Desarrollo de la documentación

Ref	ud.	Descripción	Precio (€)	Cantidad	Parcial (€)
d4	ud.	Elavoración de la memoria, planos pliego de condiciones y presupuesto.	3154,70	1	3154,70
Materiales					
m45	mes	Amortización Software Autocad	247,00	2	494,00
m44	mes	Amortización Software Fusion 360	73,00	1	73,00
M.O.D.					
h1	h	Ingeniero electrónico	11,00	250	2750,00
Medios auxiliares					
	%	M. A. sobre costes directos	10%	3317,00	331,70

2.5. Ensamblaje del dispositivo

Ref	ud.	Descripción	Precio (€)	Cantidad	Parcial (€)
d5	ud.	Costes amteriales y de montaje del robot RA-01	330,92	1	330,92
Materiales					
m1	ud.	Placa de circuito impreso (PCB)	16,5	1	16,50
m2	ud.	Baterria LiPo 6.000 mAh	17,80	1	17,80
m3	ud.	Interruptor basculante	0,75	2	1,50
m4	ud.	Cargador LiPo micro USB TP4056	1,20	1	1,20
m5	ud.	Elevador DC/DC MT3608	1,25	2	2,50
m6	ud.	Regulador 3,3V LDO MCP1755S	1,30	1	1,30
m7	ud.	ESP 32 D1 mini	11,49	1	11,49
m8	ud.	ESP 32 cam	16,59	1	16,59
m9	ud.	Cámara OV2640 75mm	3,40	1	3,40
m10	ud.	Módulo DFPlayer Mini MP3	2,95	1	2,95
m11	ud.	Servomotor MG90S	4,50	9	40,50
m12	ud.	Rodamiento Mr128zz 8x12x3.5mm	0,31	8	2,46
m13	ud.	Sensor laser, módulo GY-530	1,97	4	7,89
m14	ud.	Mdulo interruptor tactil TTP223	0,60	2	1,20
m15	ud.	Barra LED RGB WS2812 8x5050	1,25	2	2,50
m16	ud.	Pantalla OLED RGB 1,5''	21,90	1	21,90
m17	ud.	Altavoz 8Ω, 2W y 4mm de diámetro	6,81	1	6,81
m18	ud.	Convertidor de nivel lógico TE291	1,08	1	1,08
m19	ud.	Condensador electrolítico 470uF	0,50	2	1,00
m20	ud.	Condensador electrolítico 100uF	0,08	1	0,08
m21	ud.	Condensador electrolítico 1uf	0,08	2	0,16
m22	ud.	Condensador cerámico SMD 100nF	0,35	9	3,15
m23	ud.	Resistencia agujero pasante 1MΩ	0,30	2	0,60
m24	ud.	Resistencia agujero pasante 1kΩ	0,30	2	0,60
m25	ud.	Conector JST hembra, 2 pines	0,35	1	0,35
m26	ud.	Header macho 2,54mm (40x)	0,25	2	0,50
m27	ud.	Header hembra 2,54mm (40x)	0,50	3	1,50
m28	ud.	Pieza impresa, pieza inferior	10,41	1	10,41
m29	ud.	Pieza impresa, pieza intermedia	3,81	1	3,81
m31	ud.	Pieza impresa, pieza superior	8,47	1	8,47
m32	ud.	Pieza impresa, eslabón pata 1 (o su versión	4,35	4	17,40
m33	ud.	Pieza impresa, eslabón pata 2 (o su versión	2,06	4	8,24
m34	ud.	Pieza impresa, soporte altavoz	1,02	1	1,02
m35	ud.	Pieza impresa, puerta	1,36	1	1,36
m36	ud.	Pieza impresa, tapa luces	1,50	2	3,00
m37	ud.	Pieza impresa, enganche	0,25	4	1,00
m38	ud.	Pieza impresa, base pinza	0,84	1	0,84
m39	ud.	Pieza impresa, pinza 1	0,42	1	0,42
m40	ud.	Pieza impresa, pinza 2	0,38	1	0,38
m41	ud.	Pieza impresa, pinza 3	0,32	4	1,28
m42	ud.	Pieza impresa, pinza 4	0,35	2	0,70
m43	conjunto	Tornillos arandelas y tuercas	3,00	1	3,00
M.O.D.					
h2	h	Técnico	9,00	8	72,00

Ref	ud.	Descripción	Precio (€)	Cantidad	Parcial (€)
Medios auxiliares					
	%	M. A. sobre costes directos	10%	300,84	30,08

3. Estado de mediciones

Ref	ud.	Descripción	Cantidad
d1	ud.	Desarrollo y diseño de la placa de la PCB mediante software. Y comporobación práctica del correcto funcionamiento de la	1
d2	ud.	Desarrollo y diseño de las piezas 3D mediante software	1
d3	ud.	Desarrollo de los códigos necesarios para el funcionamiento del dispositivo	1
d4	ud.	Elavoración de la memoria, planos pliego de condiciones y presupuesto.	1
d5	ud.	Costes amteriales y de montaje del robot RA-01	1

4. Valoración

Ref	ud.	Descripción	Precio (€)	Cantidad	Parcial (€)
d1	ud.	Desarrollo y diseño de la placa de la PCB mediante software. Y comporobación práctica del correcto funcionamiento de la misma.	1940,29	1	1940,29
d2	ud.	Desarrollo y diseño de las piezas 3D mediante software	1235,63	1	1235,63
d3	ud.	Desarrollo de los códigos necesarios para el funcionamiento del dispositivo	938,39	1	938,39
d4	ud.	Elavoración de la memoria, planos pliego de condiciones y presupuesto.	3154,70	1	3154,70
d5	ud.	Costes amteriales y de montaje del robot RA-01	330,92	1	330,92
		Total presupuesto:			7599,93
	%	Beneficio sobre presupuesto final	13%		987,99
	%	Gastos personales	6%		4559,96
		Total presupuesto sin I.V.A.:			13147,88
	%	Impuesto sore el Valor Añadido	21%		2761,05
		Total presupuesto con I.V.A.:			15908,93

El presupuesto de diseño y desarrollo del prototipo de robot RA-01 asciende a:

QUINCE MIL NOVECIENTOS OCHO CON NOVENTA Y TRÉS EUROS (15908,93€)