UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

Diseño y fabricación de un brazo robótico de 6 GDL de bajo coste basado en Arduino

AUTOR: Fernando García Reig

TUTOR: Alicia Esparza Peidro

COTUTORA: Leopoldo Armesto Ángel

Curso Académico: 2019-20

AGRADECIMIENTOS

A mi familia por su apoyo continuo e incondicional y por preocuparse por mi futuro.

A mi tutor del proyecto, por su dedicación y apoyo continuo en todo este tiempo.

A mis amigos por apoyarme y motivarme con el proyecto.

Resumen

En este Trabajo de Fin de Grado se ha diseñado y fabricado un brazo robótico, tal y como se indica en el título de este. Para ello se ha comenzado por la selección de los componentes, en el que se ha optado por utilizar 6 servos, lo cual le da esos 6 grados de libertad al brazo robótico. Por otra parte, los motores (y el brazo) se controlan mediante dos joysticks. La placa y el lenguaje en el que se ha decidido programar es Arduino. En segundo lugar, se ha procedido a diseñar las partes que forman el brazo en el programa de diseño 3D, Solidworks. Para la fabricación de las piezas que forman el robot, se ha optado por fabricación en impresora 3D. El ensamblaje de estas partes con el resto de los componentes se ha hecho utilizando materiales de uso común y herramientas de alcance general. Se han realizado pruebas tanto con el software como con los componentes, y determinado también la elección o inclusión de nuevos componentes acordes a las necesidades, que han variado desde la idea inicial. En estas pruebas también se ha visto la necesidad de incluir una fuente de alimentación externa más potente para poder mover los 6 servos al mismo tiempo. La finalidad de este proyecto es crear un brazo robótico en su totalidad a partir de conocimientos básicos y semi-avanzados de la robótica, fabricación e impresión 3D, diseño de las partes en 3D y elección de componentes adecuados y comunes de uso extendido.

Palabras clave: robot, brazo, Arduino, servo.

Resum

En aquest Treball de Fi de Grau s'ha dissenyat i fabricat un braç robòtic, tal com s'indica en el títol d'aquest. Per això s'ha començat per la selecció dels components, en què s'ha optat per utilitzar 6 servos, la qual cosa li dóna aquests 6 graus de llibertat al braç robòtic. D'altra banda, els motors (i el braç) es controlen mitjançant dos joysticks. La placa i el llenguatge en el qual s'ha decidit programar és Arduino. En segon lloc, s'ha procedit a dissenyar les parts que formen el braç en el programa de disseny 3D, Solidworks. Per a la fabricació de les peces que formen el robot, s'ha optat per fabricació en impressora 3D. L'acoblament d'aquestes parts amb la resta dels components s'ha fet utilitzant materials d'ús comú i eines d'abast general. S'han realitzat proves tant amb el programari com amb els components, i determinat també l'elecció o inclusió de nous components acords a les necessitats, que han variat des de la idea inicial. En aquestes proves també s'ha vist la necessitat d'incloure una font d'alimentació externa més potent per poder moure als 6 servos a el mateix temps. La finalitat d'aquest projecte és crear un braç robòtic en la seva totalitat a partir de coneixements bàsics i semi-avançats de la robòtica, fabricació i impressió 3D, disseny de les parts en 3D i elecció de components adequats i comuns.

Paraules clau: robot, braç, Arduino, servo.

Abstract

In this Final Degree Project, a robotic arm has been designed and manufactured, as indicated in its title. For this purpose, the project has started with the selection of the components, choosing to use 6 servos, which gives these 6 degrees of freedom to the robotic arm. On the other hand, the motors (and the arm) are controlled by two joysticks. The plaque and the language in which it has been decided to program is Arduino. Secondly, we have proceeded to design the parts that form the arm in the 3D design program, Solidworks. To manufacture the parts that make up the robot, we have opted for manufacturing in 3D printer. The assembly of these parts with the rest of the components has been done being able to use materials of common use and tools of general scope. Tests have been carried out both with the software and with the components, and even determined the choice or inclusion of new components according to the needs, which have varied from the initial idea. These tests have also seen the need to include a more powerful external power supply to be able to move all 6 servos at once. The purpose of this project is to create a robotic arm in its entirety from basic and semi-advanced knowledge of robotics, 3D manufacturing and printing, design of 3D parts and selection of suitable and common components of extended use.

Key words: robot, arm, Arduino, servo.

ÍNDICE DE DOCUMENTOS

ÍNDICE

Página

Contenido TFG

Documento 1: *memoria*

I. MEMORIA

1. Objetivos del trabajo y justificación del proyecto

1.1 Objetivo

En este trabajo se tratarán una serie procedimientos y aspectos básicos para la creación de un brazo robótico. Se explicarán los pasos a seguir, el diseño del robot, programación, ensamblaje y puesta en funcionamiento.

El objetivo general de este proyecto es ser capaz, mediante los conceptos adquiridos a lo largo de la carrera de Ingeniería en Tecnologías Industriales y algunos otros de posterior adquisición, de poder diseñar, fabricar, ensamblar y programar desde cero un brazo robótico funcional de 6 GDL a partir de fundamentos básicos de la robótica, materiales comunes o de alcance general y en definitiva, procesos no muy complicados que la mayoría pueden realizar con la preparación necesaria.

En él se detalla cada decisión, hipótesis, y opciones que se han escogido a la hora de fabricar y diseñar el robot, para poder llegar al resultado de la forma más eficiente y económica posible.

1.2 Ámbito de aplicación

Este proyecto está dirigido a personas con formación técnica dentro del ámbito de la ingeniería, tanto en el diseño digital en 3D, conocimientos eléctricos y electrónicos, programación en C++ y Arduino, y otros conocimientos generales de ensamblaje y fabricación.

1.3 Motivación del proyecto

En mi primera vez visitando la universidad, en una sesión de puertas abiertas, en la cual nos mostraron un brazo robot para montaje en cadena de producción, me fascinó la complejidad y las posibilidades que aquel instrumento podía presentar. Desde entonces siempre me ha llamado la atención el mundo de la robótica y qué mejor manera de iniciarse en este ámbito que por lo primero que conocí de él. El brazo robótico dentro de este campo es un muy buen representante de los temas básicos y fundamentales que trata la robótica, ya que aporta muy buenas bases respecto a la cinemática, localización espacial y además la idea no es demasiado compleja comparada con otras aplicaciones móviles de la robótica, ya que adentrarse en un campo nuevo y enfrentarse de primeras a problemas muy avanzados normalmente resulta en fracaso y aprendizaje forzado y deficiente.

1.4 Análisis de los problemas a resolver

- ✓ Resolver la cinemática inversa del brazo robot. Como ya se detallará más extendidamente en el marco teórico, la localización espacial del robot es fundamental a la hora de poder establecer los movimientos, direcciones y giros que pueden tomar los sistemas de coordenadas del robot. Para ello se estudia de forma cinemática, tanto directa, como inversa, el caso del brazo robótico.
- ✓ Evitar que el par del motor sea superado por el par resistente. Inicialmente se diseña el brazo de una forma que no resulta efectiva, por lo que en la segunda hipótesis que se ha escogido, se han tomado las decisiones oportunas para el correcto funcionamiento de los servos, que también se detallará con mayor profundidad en la selección de componentes y el diseño.
- ✓ Seleccionar los componentes con mejor calidad-precio disponibles y más accesibles del mercado, para facilitar la velocidad de trabajo, compatibilidad de componentes, y en la mayor medida posible, el tema económico.
- ✓ Ser capaz de programar el brazo robot en el lenguaje C++ y Arduino, para poder moverlo con dos joysticks y ejecutar la cinemática inversa para realizar un programa demo, además del modo manual. Se escoge la opción de los joysticks, de otras posibles y también llamativas, como por ejemplo a través de una aplicación sencilla para móviles, potenciómetros u otros dispositivos, por elección personal.
- ✓ Diseño previo y puesta en marcha. El diseño de los distintos aspectos, tanto 3D de las piezas, como elección de los componentes, y programación y modos de funcionamiento, influirán a la hora de plantear el problema inicial o las bases de partida para llegar a la idea final.
- ✓ Remodelación del diseño y nueva distribución (de 5DL iniciales a 6GDL). En un principio se planteó el robot como un brazo de 5 GDL, pero posteriormente se adopta la decisión de realizar el control cinemático del brazo y por ello se decide remodelar y ampliar estos 5 grados de libertad a 6, para mayor facilidad de operación con las matrices de cinemática, directa e inversa.

2. Introducción a la robótica

Según la rae, un robot, palabra que tiene origen en el vocablo checo "*robota"* (servidumbre, trabajo forzado o esclavitud), es una "*máquina o ingenio electrónico programable que es capaz de manipular objetos y realizar diversas operaciones*" [1]. Podemos definir la robótica como una ciencia o técnica que se enfoca en el diseño, fabricación y utilización de robots. A su vez, esta recibe conocimientos de la rama de ingeniería electrónica, eléctrica, mecánica, biomédica e informática, por lo que es una ciencia que abarca diversos ámbitos de la tecnología actual.

La evolución de la robótica ha sido sorprendente, desde sus inicios con la civilización griega y egipcia, hasta hoy en día, que podemos crear robots muy complejos y fascinantes. Estos han sido algunos de los avances de la robótica a lo largo de los siglos [2][3]:

- En la época griega y egipcia se crearon diversas máquinas y autómatas, entre ellos un órgano de viento, una máquina de vapor, creada por Herón de Alejandría y algunos artilugios utilizados en estatuas religiosas para fascinar a los adoradores.
- En el año 1495, Da Vinci crea el diseño del caballero mecánico, un robot de aspecto humanoide capaz de hacer ciertas funciones similares a las humanas.
- En el siglo XVIII, Jacques de Vauncansos construyó músicos mecánicos de tamaño humano que ejecutaban piezas de música.
- En 1801, J. Jaquard invento su telar, que era una máquina programable para la urdimbre.
- En 1941, Isaac Asimov publica "Círculo vicioso", una obra de ciencia ficción en la que plasma las tres leyes de la robótica.
- En 1946, George C. Devol desarrolló un controlador capaz de registrar señales eléctricas utilizando medios magnéticos y reproducirlas para accionar una máquina mecánica.
- En 1948, gracias a George C. Devol, surge el primer brazo robótico, impulsando la robótica industrial.
- En 1951, se desarrollaron manipuladores de control remoto para manejar materiales radiactivos.
- En 1952, se diseñó una máquina prototipo de control numérico en el MIT.
- En 1968, se desarrolló un robot llamado Shakey, por el SRI, capaz de desplazarse y dotado de sensores y una cámara.
- En 1973, KUKA crea el primer robot industrial con 6 ejes de accionamiento electromagnético, el Famulus.

En la actualidad, los avances en la robótica son cada vez mayores, llegando incluso al manejo autónomo de vehículos, ejecución de procedimientos médicos complejos, transporte aéreo para miles de aplicaciones (drones), y la IA, capaz de discernir y solventar por su propio pie los problemas que le surgen. Y con estos avances vienen mejoras y comodidades para los seres humanos que jamás podríamos haber soñado, pero también hay un proceso de adaptación y sustitución de empleos al reemplazar a las personas por máquinas o autómatas (industrialización).

En el futuro, el campo de la IA se desarrollará cada vez más, al igual que las comunicaciones y redes, se mejorarán los robots móviles autónomos y habrá avances en la biotecnología cada vez más sorprendentes, como los nanorobots, entre otros avances.

3. Marco teórico

En este apartado del trabajo se detallarán las teorías e hipótesis utilizadas para el movimiento avanzado del brazo robótico en el espacio, siguiendo los ejes cartesianos, giros y desplazamientos en ellos. Estas teorías se fundamentan en tres aspectos fundamentales, la cinemática directa del robot, la inversa, y la de velocidades. Según [4], el control dinámico de robots, procura que la respuesta del robot sea lo más parecida posible a la referencia propuesta por el control cinemático. Se utilizarán fuentes de diversos autores para aproximar la teoría utilizada desde distintos enfoques, véase [5][6][7].

3.1 Localización espacial

La manipulación de objetos mediante el uso de robots implica conocer la posición y orientación de los elementos respecto al lugar que ocupa el robot. Para ello necesitamos determinar la posición de cualquier pieza en el espacio mediante el uso de herramientas matemáticas que nos permitan obtener relaciones espaciales entre los distintos objetos, de forma que el robot pueda posicionarse respecto al objeto de acuerdo con su programación o las órdenes del usuario, para finalmente interactuar con el objeto o no.

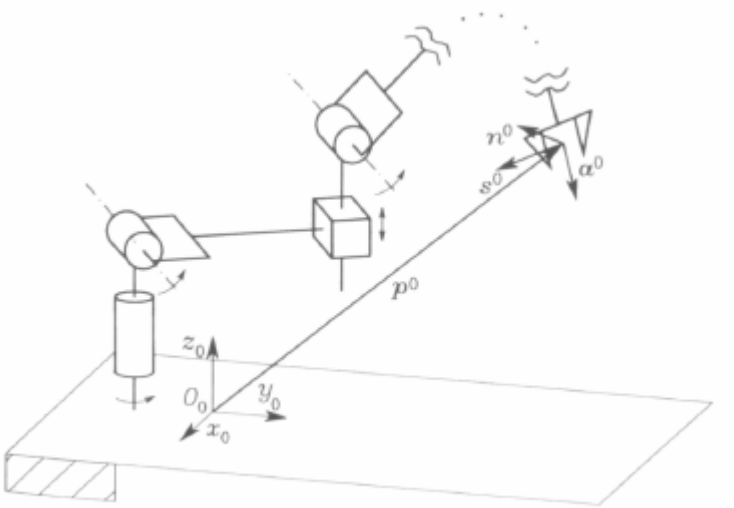


Fig 1: Representación ideal de un brazo robótico y el sistema de coordenadas. Fuente: [5]

3.2 Cinemática de posición

3.2.1 Cinemática directa

La cinemática consiste en estudiar el movimiento sin tomar en cuenta las fuerzas actuadoras. La cinemática del robot lo estudia respecto a un sistema de referencia fijo, a partir del cual podemos describir el movimiento espacial de este como una función del tiempo. De esta manera podemos extraer las relaciones de posición y orientación del **extremo** (o la pinza en este caso) del robot y los valores puntualizados de las coordenadas de cada **articulación**. Véase Fig. 2.

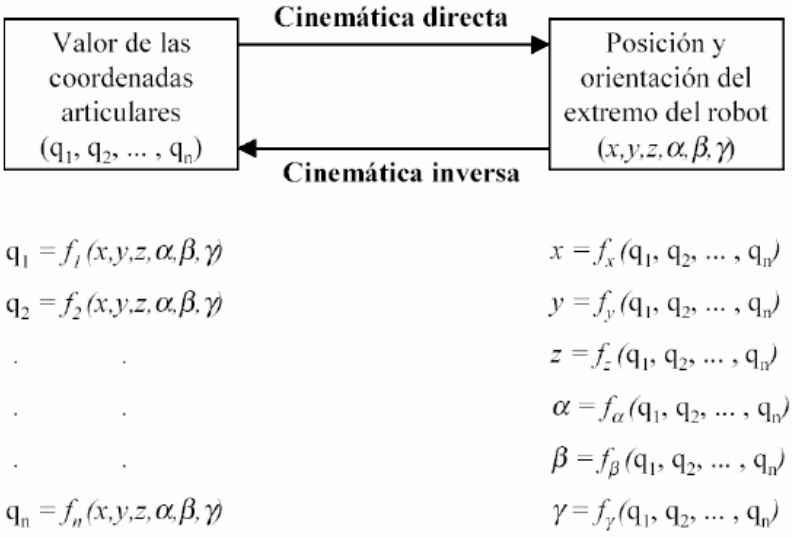


Fig. 2: Relaciones entre cinemática directa e inversa. Fuente [5].

Existen dos formas o soluciones de abordar el problema cinemático directo. La primera que trataremos es mediante un método geométrico. De esta manera se obtiene la posición y orientación del extremo del robot mediante relaciones geométricas y fórmulas trigonométricas, en función de las variables de las articulaciones. El problema de este método es que no es sistemático, y se limita a aplicaciones o robots con pocos grados de libertad. La segunda forma que es la que utilizaremos en este proyecto es un método basado en cambios de sistemas de referencia, lo cual permite resolver el modelo para distintas disposiciones y es independiente de las características geométricas del robot.

Este método utiliza las matrices de transformación homogénea (método de matrices Denavit-Hartemberg). Un robot se puede considerar como una cadena cinemática que está formada por eslabones (partes rígidas) unidos entre sí por articulaciones (ejes). Este método asocia a cada eslabón del robot un sistema de referencia solidario, comenzando por un sistema de referencia fijo en la base del robot que describe la posición de cada eslabón respecto a este. Para representar la posición de un eslabón respecto al origen, se utiliza la matriz de transformación homogénea T, compuesta por la multiplicación consecutiva de matrices ${}^{i-1}A_i$ que representan la posición y orientación relativa entre dos sistemas asociados por dos eslabones consecutivos. La matriz T total o parcial de la cadena cinemática la matriz homogénea que asocia las translaciones y rotaciones relativas entre el origen y sistema i, quedaría de la siguiente forma:

$$ {}^{0}T_i = {}^{0}A_i = {}^{0}A_1 \cdot {}^{1}A_2 \cdot {}^{2}A_3 \cdot \dots \cdot {}^{i-1}A_i $$

De lo siguiente se deduce un método para situar los sistemas de coordenadas asociados a cada eslabón y poder obtener la cadena cinemática del robot, denominado **Denavit-Hartemberg**. En este método cada sistema de coordenadas i, se define a partir del sistema de coordenadas anterior i-1 y los ejes de articulación como i e i+1. Permite pasar entre eslabones consecutivos con 4 transformaciones, que son rotación θ_i alrededor del eje Z_{i-1}, traslación di a lo largo del eje Z_{i-1}, traslación a_i a lo largo del eje X_i y rotación αi alrededor del eje X_i. En consecuencia, diferenciamos 4 parámetros principales para cada eslabón. La forma de obtenerlos es la siguiente:

- θ_i: Es el ángulo de xi-1 a xi medido sobre zi-1 (utilizando la regla de la

mano derecha).

- d_i: Es la distancia de xi-1 a xi medida a lo largo de zi-1
- a_i: Es la distancia de zi-1 a zi medida a lo largo de xi
- α_i: Es el ángulo de zi-1 a zi medida sobre xi (utilizando la regla de la

mano derecha).

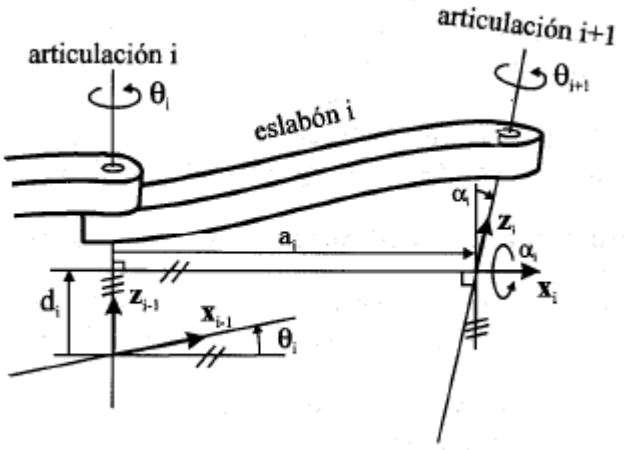


Fig. 3. Descripción gráfica de parámetros de D-H. Fuente: [5].

Una vez definidos los parámetros principales de DH, podemos definir la matriz ${}^{i-1}A_i$ de la siguiente manera, de forma que C son cosenos y S senos:

$$ {}^{i-1}_{i}A = \begin{bmatrix} C\theta i & -C\alpha i\, S\theta i & S\alpha i\, S\theta i & ai\, C\theta i \\ S\theta i & C\alpha i C\theta i & -S\alpha i C\theta i & ai\, S\theta i \\ 0 & S\alpha i & C\alpha i & di \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Existen 9 directrices o normas a seguir a la hora de determinar los componentes de nuestro sistema:

1. Identificar los eslabones y los ejes de las articulaciones trazando líneas imaginarias a lo largo de ellos. Los eslabones se numeran empezando por el 1 (primer eslabón móvil) y acabando por el n (último eslabón móvil). La base fija se numera como eslabón 0. Las articulaciones se numeran empezando por la 1 y acabando por la n.
2. El origen O_i debe situarse en la intersección de entre los ejes i+1 e i (z_{i-1} y z_i). Si no existe la intersección, identificar la perpendicular común entre ejes consecutivos y el origen O_i estará en la intersección del eje i+1 con la normal común entre los ejes i e i+1 (normal común entre z_{i-1} y z_i).
3. Colocar el eje Z_i sobre el eje de la articulación i+1.
4. Colocar el eje X_i sobre la perpendicular común, o si los ejes intersecan, sobre la normal al plano que forman los ejes Z_{i-1} y Z_i. El eje X_i debe ser perpendicular a los ejes Z_{i-1} y Z_i (se puede definir como el producto vectorial $Z_{i-1}\times Z_i$).

5. Colocar el eje Y_i completando un sistema de referencia dextrógiro (regla de la mano derecha).
6. Para i de 0 a n-1, situar el eje Z_i sobre el eje de la articulación i+1. Situar X_i con la normal común entre Z_{i-1} y Z_i (paso 4).
7. El primer sistema de coordenadas, asociado a la base del robot, SC_0 (con origen O_0), se puede situar en cualquier punto a lo largo del eje Z_0.
8. Para el resto de sistemas de coordenadas, debe cumplirse que el origen esté en la intersección entre Z_i y X_i. El eje Y_i, se define para que SC_i sea un sistema dextrógiro (paso 5).
9. El sistema de coordenadas SC_n (con origen O_n) asociado al último elemento podrá tener su origen en cualquier parte de éste, siempre que la dirección de Z_n sea equivalente a la de Z_{n-1}.

3.2.2 Cinemática inversa

La cinemática inversa determina las coordenadas articulares, o q_i, para la posición cartesiana del robot dada. Este problema puede tener varias soluciones o configuraciones redundantes. Por ello los métodos genéricos suelen resultar en cálculos de elevada duración y depende de la configuración de cada robot, con lo que se le añade otro componente más de dificultad. En el caso de tener 6 GDL, se suelen desacoplar los tres primeros ejes de los tres últimos, ya que con estos tres primeros se ajusta la posición y con los tres últimos la orientación para mayor facilidad de solución.

Para resolver la cinemática inversa también existen dos métodos, el geométrico y el de matrices homogéneas. De manera que se opta el segundo para resolver este proyecto como se ha argumentado anteriormente. Este método se basa en manipular las ecuaciones obtenidas a partir del modelo cinemático directo. De forma que obtenemos las q_i en función de los vectores n, o, a y p, los cuales determinan la posición y orientación que deseamos que tenga el punto extremo del brazo:

$$ {}^{0}_{n}\boldsymbol{T}(q_1,..,q_n) = \begin{bmatrix} \vec{n} & \vec{o} & \vec{a} & \vec{p} \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

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

Para llegar a la solución el problema se divide en dos partes como se comenta anteriormente. Primero resolver el problema cinemático inverso del brazo de las tres primeras articulaciones (q1, q2, q3) y después el de las tres articulaciones que llevan hasta el punto final o extremo (q4, q5, q6).

Una vez obtenida la expresión de T en función de las coordenadas articulares y supuesta una localización de destino para el extremo del robot definida por los vectores n, o, a y p, pasamos

a determinar las inversas de las matrices correspondientes. Puesto que ${}^{0}A_3$, que contiene la localización definida por los vectores anteriormente nombrados, se obtiene a partir de la multiplicación de las ${}^{i-1}A_i$ correspondientes (i = 1, 2 y 3), calculamos la inversa de cada una de estas matrices para poder extraer cada coordenada articular q_i de la siguiente manera:

$$\left({}^{0}\mathbf{A}_1\right)^{-1}{}^{0}\mathbf{T}_3={}^{1}\mathbf{A}_2\,{}^{2}\mathbf{A}_3=\begin{bmatrix} C_1 & S_1 & 0 & 0 \\ 0 & 0 & 1 & -l_1 \\ S_1 & -C_1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}=$$

$$=\begin{bmatrix} C_2 & 0 & -S_2 & 0 \\ S_2 & 0 & C_2 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & q_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}=\begin{bmatrix} C_2 & 0 & -S_2 & -S_2q_3 \\ S_2 & 0 & C_2 & C_2q_3 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$S_1p_x - C_1p_y = 0 \quad \Rightarrow$$

$$\tan(q_1)=\left(\frac{p_y}{p_x}\right) \quad \Rightarrow$$

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

$$\left({}^{1}\mathbf{A}_2\right)^{-1}\left({}^{0}\mathbf{A}_1\right)^{-1}\mathbf{T}={}^{2}\mathbf{A}_3=$$

$$=\begin{bmatrix} C_2 & S_2 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ -S_2 & C_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} C_1 & S_1 & 0 & 0 \\ 0 & 0 & 1 & -l_1 \\ S_1 & -C_1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & q_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$=\begin{bmatrix} C_2C_1 & C_2S_1 & S_2 & -l_1S_2 \\ -S_1 & C_1 & 0 & 0 \\ -S_2C_1 & -S_2S_1 & C_2 & -C_2l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & q_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$C_2C_1p_x + C_2S_1p_y + S_2p_z - l_1S_2 = 0 \quad \Rightarrow$$

$$C_2\left(C_1p_x + S_1p_y\right) + S_2\left(p_z - l_1\right) = 0 \quad \Rightarrow$$

$$q_2 = arctan\left(-\frac{(C_1p_x + S_1p_y)}{(p_z - l_1)}\right)$$

$$\left({}^{1}\mathbf{A}_{2}\right)^{-1}\left({}^{0}\mathbf{A}_{1}\right)^{-1}\mathbf{T}={}^{2}\mathbf{A}_{3}=$$

$$=\begin{bmatrix} C_2 & S_2 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ -S_2 & C_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} C_1 & S_1 & 0 & 0 \\ 0 & 0 & 1 & -l_1 \\ S_1 & -C_1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & q_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$=\begin{bmatrix} C_2C_1 & C_2S_1 & S_2 & -l_1S_2 \\ -S_1 & C_1 & 0 & 0 \\ -S_2C_1 & -S_2S_1 & C_2 & -C_2l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & q_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$-S_2C_1p_x-S_2S_1p_y+C_2p_z-C_2l_1=q_3 \quad \Rightarrow$$

$$C_2(p_z-l_1)-S_2(C_1p_x+S_1p_y)=q_3$$

De manera similar, empleando la matriz R, la cual utiliza ahora únicamente los vectores n, o y a de la matriz ${}^{i-1}A_i$, obtenemos las coordenadas articulares q_4, q_5 y q_6.

$${}^{0}\mathbf{R}_{6}=\begin{bmatrix}\mathbf{n} & \mathbf{o} & \mathbf{a}\end{bmatrix}={}^{0}\mathbf{R}_{3}\,{}^{3}\mathbf{R}_{6}$$

$${}^{0}\mathbf{R}_{3}={}^{0}\mathbf{A}_{1}\,{}^{1}\mathbf{A}_{2}\,{}^{2}\mathbf{A}_{3}$$

$${}^{3}\mathbf{R}_{6}=\left[r_{ij}\right]=\left({}^{0}\mathbf{R}_{3}\right)^{-1}{}^{0}\mathbf{R}_{6}=\left({}^{0}\mathbf{R}_{3}\right)^{T}\begin{bmatrix}\mathbf{n} & \mathbf{o} & \mathbf{a}\end{bmatrix}$$

$${}^{3}\mathbf{R}_{4}=\begin{bmatrix} C_4 & 0 & -S_4 \\ S_4 & 0 & C_4 \\ 0 & -1 & 0 \end{bmatrix}\quad {}^{4}\mathbf{R}_{5}=\begin{bmatrix} C_5 & 0 & S_5 \\ S_5 & 0 & -C_5 \\ 0 & 1 & 0 \end{bmatrix}\quad {}^{5}\mathbf{R}_{6}=\begin{bmatrix} C_6 & -S_6 & 0 \\ S_6 & C_6 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$ {}^{3}R_{6} = \begin{bmatrix} C_4C_5C_6 - S_4S_6 & -C_4C_5C_6 - S_4S_6 & C_4S_5 \\ S_4C_5C_6 + C_4S_6 & -S_4C_5S_6 - C_4C_6 & S_4S_5 \\ -S_5C_6 & S_5S_6 & C_5 \end{bmatrix} $$

$$ [r_{ij}] = \begin{bmatrix} C_4C_5C_6 - S_4S_6 & -C_4C_5C_6 - S_4S_6 & C_4S_5 \\ S_4C_5C_6 + C_4S_6 & -S_4C_5S_6 - C_4C_6 & S_4S_5 \\ -S_5C_6 & S_5S_6 & C_5 \end{bmatrix} $$

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

$$ q_5 = \arccos(r_{33}) $$

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

4. Selección de componentes

En el proceso de selección de componentes, se ha realizado una búsqueda y comparación exhaustiva entre las distintas opciones disponibles, en base a las características deseadas y a la finalidad del proyecto, manteniendo siempre los costes al mínimo posible, sin sacrificar rendimiento. Otros factores muy decisivos a la hora de la elección han sido los gastos de envío, plazos de entrega del proveedor y la fiabilidad de este, ya que influyen de manera directa en el resultado final y su agilidad. Además, en ocasiones se ha optado por opciones más comunes por tema de compatibilidad entre los componentes, pudiendo elegir un componente de mayor coste pero de mayor fiabilidad y compatibilidad con el resto de componentes que otro de menor coste pero menor fiabilidad.

4.1 Arduino + extensión para I/O

El chip microcontrolador, que se encarga de relacionar las entradas y salidas analógicas y digitales del robot, tanto como de recibir de forma directa la fuente de alimentación y actuar como núcleo central, es el componente más importante de todos. Es el encargado de enviar la información a los actuadores y sincronizarlos para ejecutar las tareas programadas. Para la aplicación que se va a llevar a cabo en concreto, en el mercado actual de microcontroladores, existen diversos modelos adecuados, pero hay dos muy extendidos que, por su popularidad, resultados, coste y, facilidad de programación y buena compatibilidad, son los que tendremos en cuenta en para este proyecto. Estos dos microcontroladores son Raspberry Pi y Arduino. A continuación, se explicará por qué se ha escogido el segundo para esta aplicación.

Por una parte, tenemos la Raspberry Pi, un microcontrolador enfocado más a la potencia de cálculo, por lo que el hardware que incluye tiene también mayor coste, generalmente. En contrapuesta, existe el Arduino, más dirigido a facilitar la conexión con sus entradas analógicas y digitales, gracias a su sencilla activación y desactivación por software, a pesar de ser menos potente en tema de cálculo. En cuanto al software, la rapidez de ejecución del Arduino es mayor que la de la Raspbery Pi, y esto como consecuencia, tiene una repercusión directa en los proyectos de electrónica, ya que implican reacción de los actuadores casi instantánea. Por esta razón se ha escogido utilizar como controlador un Arduino.

Dentro de esta rama, existen múltiples modelos de Arduino. Como se expone en la figura 4, extraída de [8], podemos ver los modelos actuales no retirados de Arduinos. Dentro de las opciones, descartamos las más complejas y por tanto costosas, tanto como las de mayor dimensión, ya que el proyecto actual no lo requiere y con un tamaño inferior podemos operar igualmente. Por tanto, nos quedamos con 3 microcontroladores, el Arduino Mini, el Micro y el Nano. Si cogemos la versión de 32 KB del Nano, los tres tiene la misma memoria flash. El Micro lo podemos encontrar por 18 €, el Mini por unos 15 €, y el Nano, el cual es de los más extendidos y por tanto más fácil de comprar o más accesible, lo podemos encontrar por 6-7€ en España. Puesto que no vamos a utilizar más de 14 entradas/salidas digitales, descartamos la Micro, ya que tiene 20, y no es necesario. Por otro lado, el Arduino Nano permite la conexión mediante USB, cosa que la facilita mucho, y la placa Mini, se conecta mediante tarjeta o cable FTDI. Por tanto, una vez expuestas las similitudes y diferencias, y valorando, tanto el acceso, envío de proveedores, precio, entradas y salidas, y conexión, se toma

la decisión de elegir la placa Arduino Nano [9]. Además añadiremos una placa de I/O para facilitar aún más las conexiones [10].

Name	Processor	Operating/Input Voltage	CPU Speed	Analog In/Out	Digital IO/PWM	EEPROM [kB]	SRAM [kB]	Flash [kB]	USB	UART
101	Intel® Curie	3.3 V/ 7-12V	32MHz	6/0	14/4	-	24	196	Regular	-
Gemma	ATtiny85	3.3 V / 4-16 V	8 MHz	1/0	3/2	0.5	0.5	8	Micro	0
LilyPad	ATmega168V ATmega328P	2.7-5.5 V / 2.7-5.5 V	8MHz	6/0	14/6	0.512	1	16	-	-
LilyPad SimpleSnap	ATmega328P	2.7-5.5 V / 2.7-5.5 V	8 MHz	4/0	9/4	1	2	32	-	-
LilyPad USB	ATmega32U4	3.3 V / 3.8-5 V	8 MHz	4/0	9/4	1	2.5	32	Micro	-
Mega 2560	ATmega2560	5 V / 7-12 V	16 MHz	16/0	54/15	4	8	256	Regular	4
Micro	ATmega32U4	5 V / 7-12 V	16 MHz	12/0	20/7	1	2.5	32	Micro	1
MKR1000	SAMD21 Cortex-M0+	3.3 V/ 5V	48MHz	7/1	8/4	-	32	256	Micro	1
Pro	ATmega168 ATmega328P	3.3 V / 3.35-12 V 5 V / 5-12 V	8 MHz 16 MHz	6/0	14/6	0.512 1	1 2	16 32	-	1
Pro Mini	ATmega328P	3.3 V / 3.35-12 V 5 V / 5-12 V	8 MHz 16 MHz	6/0	14/6	1	2	32	-	1
Uno	ATmega328P	5 V / 7-12 V	16 MHz	6/0	14/6	1	2	32	Regular	1
Zero	ATSAMD21G18	3.3 V / 7-12 V	48 MHz	6/1	14/10	-	32	256	2 Micro	2
Due	ATSAM3X8E	3.3 V / 7-12 V	84 MHz	12/2	54/12	-	96	512	2 Micro	4
Esplora	ATmega32U4	5 V / 7-12 V	16 MHz	-	-	1	2.5	32	Micro	-
Ethernet	ATmega328P	5 V / 7-12 V	16 MHz	6/0	14/4	1	2	32	Regular	-
Leonardo	ATmega32U4	5 V / 7-12 V	16 MHz	12/0	20/7	1	2.5	32	Micro	1
Mega ADK	ATmega2560	5 V / 7-12 V	16 MHz	16/0	54/15	4	8	256	Regular	4
Mini	ATmega328P	5 V / 7-9 V	16 MHz	8/0	14/6	1	2	32	-	-
Nano	ATmega168 ATmega328P	5 V / 7-9 V	16 MHz	8/0	14/6	0.512 1	1 2	16 32	Mini	1
Yún	ATmega32U4 AR9331 Linux	5 V	16 MHz 400MHz	12/0	20/7	1	2.5 16MB	32 64MB	Micro	1
Arduino Robot	ATmega32u4	5 V	16 MHz	6/0	20/6	1 KB (ATmega32u4)/ 512 Kbit (I2C)	2.5 KB (ATmega32u4)	32 KB (ATmega32u4) of which 4 KB used by bootloader	1	1
MKRZero	SAMD21 Cortex-M0+ 32bit low power ARM MCU	3.3 V	48 MHz	7 (ADC 8/10/12 bit)/1 (DAC 10 bit)	22/12	No	32 KB	256 KB	1	1

Fig. 4. Tabla de especificaciones y comparativa entre los distintos modelos de Arduino.

Fuente: https://www.arduino.cc/en/main/products

4.2 Joysticks

Como parte del control del robot, se ha optado, por elección personal, por utilizar dos joysticks analógicos [10] de dos ejes (X e Y) y un interruptor incorporado en cada joystick que actúa de switch digital, y permite desbloquear más opciones a la hora de programar. La intención de estos joysticks es similar a la de un potenciómetro. En un potenciómetro, se transmiten los giros analógicos del actuador que lleva (en este caso con un tamaño de bits de 10), de forma que si esta en su posición inicial o mínima, transmite un 0 en forma digital o 0 V (GND, véase fig.6) en la entrada analógica, y si esta en su posición mas alta, transmite un 1023 o 5 V (Vcc) en su entrada analógica (5 V en este modelo). De la misma forma, el joystick actúa como si tuviese dos potenciómetros integrados, uno para cada eje, de forma que el eje vertical o X, da valores digitales de 0 a 1023 (VRx) y el eje Y (VRy), exactamente igual. La diferencia es que un joystick reposa o tiene su estado de equilibrio al que vuelve de forma natural, en el medio de esta escala, alrededor de 490 – 520 (medio ideal de 512), dependiendo de su exactitud, y no se queda fijo en el lugar que lo soltamos, como sí ocurre con un potenciómetro corriente. Este modelo de joystick también cuenta con una tercera configuración, la cual es un switch (SW) o interruptor, que permite dar por entrada digital un valor de low o 0 cuando se lo está presionando, o un 1 o high, cuando esta en su posición normal.

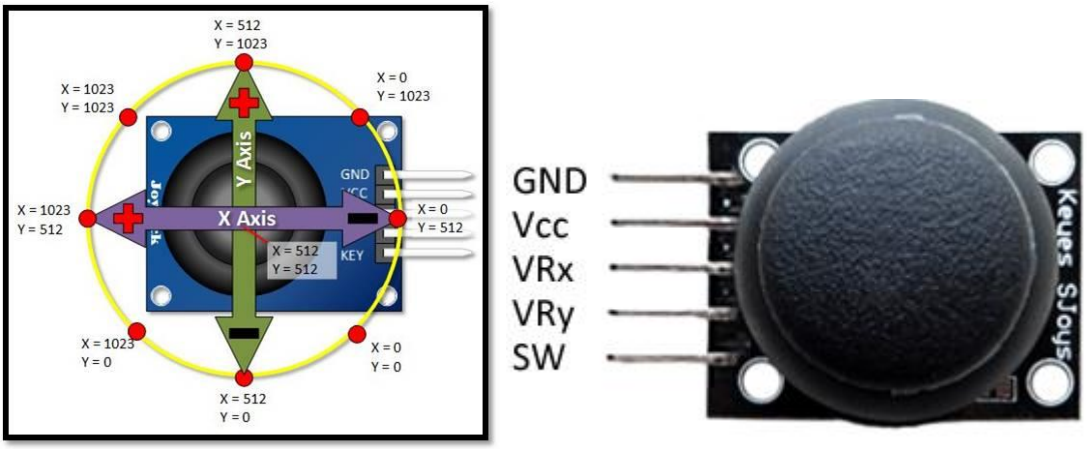


Fig. 5 y 6. Joystick, disposición de los ejes. Conexiones a los pines del joystick.

Fuentes: http://cursoarduino.proserquisa.com/2016/10/13/tutorial-25-modulo-joystick-ky-023/ y https://www.luisllamas.es/arduino-joystick/

4.3 Servos

Los servos son los principales actuadores del brazo robot. Son motores encargados de reproducir las ordenes enviadas por el Arduino. El servo, al contrario de otros motores, no tiene un rango de movimiento de 360º continuo. Está limitado de 0 a 180º únicamente. Además, su precisión es máxima, ya que controla la posición en todo momento mediante un potenciómetro, y envía al microcontrolador su posición en cada instante, y se asegura de esta mediante un sistema de

control. A diferencia de los motores de paso a paso, un servomotor, al ser más preciso suele tener un coste mayor, pero dado el proyecto, se decide no sacrificar la precisión ya que se requiere una actuación precisa e instantánea. Además, a la hora de programar el servo es mucho más fácil ya que el motor actúa situándose en los grados que se le indican mediante programa de forma instantánea (20 ms) con un margen de error de 1 o 2 grados como mucho. Dada la aplicación, se han escogido los servos más comunes y de alcance general (para facilitar tiempos de entrega y disposición o recambio) para construir el brazo robótico. Estos son los servos SG90, MG90S, y por último el de mayor par el MG996R. A continuación, se detallan cada uno de los servos:

- Micro servo SG90 [11]. Es el servo más pequeño. Está compuesto por engranajes de plástico, por lo que su resistencia es menor que la de los otros modelos. Su función en el brazo robótico es la de apertura y cierre de la pinza y también se encarga de un giro de la pinza. Ambos movimientos permiten fácilmente al servo operar. Su peso es de 10,6 g incluyendo los cables y tiene un par de 1,2 kg/cm a 4,8 V. Su velocidad de giro a 4,8 V es de 0,12 s / 60º.

- Micro servo MG90S [12]. Este servo es casi idéntico al SG90, solo que supone una mejora en varios aspectos. Para empezar los engranajes de este modelo son metálicos, lo que aumenta su resistencia. Es solo un poco más grande, pero sigue siendo considerado un servo pequeño. Su peso es de 20,4 g. Su velocidad de giro es prácticamente la misma, sin embargo, su par es de 2 Kg /cm a 4,8 V. Este servo se encarga de dos giros de la muñeca.

- Motor servo MG996R [13]. Por último, tenemos el motor más potente, compuesto totalmente por engranajes de metal. Estos motores son de dimensiones mayores y se consideran servos de tamaño medio. En este proyecto se usan para el giro de las partes principal del brazo (excluyendo la muñeca). Puesto que tienen que aguantar casi todo el peso del brazo como par, deben ser mucho más potentes. Su par es de 11 kg/cm y su velocidad de giro de 0,17 s / 60º a 4,8 V

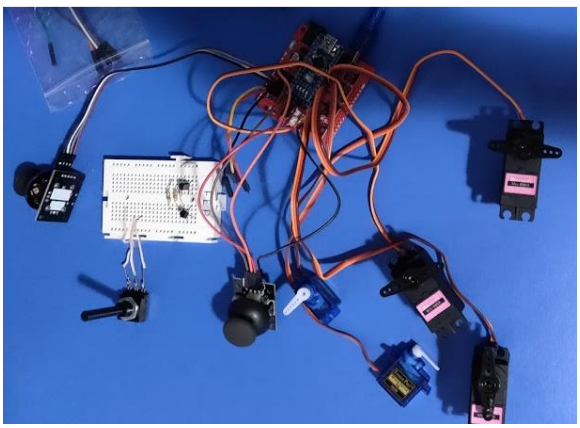


Fig. 6. Prueba inicial de funcionamiento de los servos. Fuente: propia.

4.4 Ruedas locas

Se utilizan 3 ruedas de libre dirección y giro [14], como soporte de la base del robot, para aguantar la fuerza de tracción que sufre el motor inferior y central, ya que va incrustado en la base de madera, y evitar que vuelque el brazo ante movimientos rápidos. Además, la carga también se distribuye entre las ruedas de forma que el peso que recibe el motor central es menor.

Fig. 7. Ruedas locas en la base del robot. Fuente: propia.

4.5 Pulsador KY-004

Es un pulsador o switch, que se utiliza como otra variable más para poder definir ciertos estados. El pulsador [15] tiene dos posiciones una arriba (HIGH) o permanente, y otra abajo (LOW) que se da únicamente cuando se pulsa y solo mientras se pulsa, de manera que cuando se levanta el dedo vuelve a su posición de arriba, aunque puede actuar como interruptor de estado programándolo con un contador, de forma que cambie de estado en números pares o impares, por ejemplo. Esto permite condicionar el código para activar un modo u otro. Dependiendo de la configuración que se utilice puede dar un valor de 1 o 0 para sus respectivos puntos HIGH y LOW.

4.6 Alimentación

La alimentación es una parte muy importante del proyecto. Es necesario que suministre suficiente potencia como para mover todos los servos utilizados y el Arduino, sin que haya mucha variación en el suministro. En el tema de la alimentación para este tipo de aplicaciones se pueden tener diversas configuraciones. Inicialmente se escogió una batería portátil de 9V conectada por la entrada, pero con las pruebas no se conseguía abastecer a todos los servos, al igual que si se

alimentaba directamente con el USB conectado a un ordenador. Por ello se decidió optar por una fuente de alimentación estable que tuviera más potencia y se seleccionó un banco de alimentación que pudiese tener dos baterías de alta capacidad e intensidad. Es el caso del ESP8266 ESP32 Shield Dual de baterías de litio 18650 de 5 V/3A y 3V/1A [16] y 2 Baterías 18650 de Sony [17]. Este tipo de bancos de batería es muy común cuando se trata de una aplicación que requiere mayor potencia, ya que, si no es necesario, un banco de carga portátil para móviles mediante USB suele ser suficiente o incluso como se ha comentado anteriormente, el puerto USB de un ordenador puede bastar. Estas baterías de litio de tipo 18650, son unas baterías recargables de capacidad nominal de 2600 mAh cada una, con una corriente de descarga continua de 25 A y una máxima de 35 A, tal y como se especifica en [18]. Aunque en la medida de lo posible es mejor no abusar e intentar demandar corriente con todos los servos al mismo tiempo ya que se producen picos de intensidad que pueden llegar a ser no deseables (siempre dentro del límite de seguridad, por supuesto).

Fig. 8. Banco de alimentación y 2 baterías de litio 18650. Fuente: propia.

5. Diseño y ensamblaje del brazo robot

En este apartado se describirán los criterios y la evolución que ha habido entorno al diseño de las partes fijas del robot, al igual que el ensamblaje con los demás componentes, ya que todo acaba integrado en uno mismo. Se han utilizado en la medida de lo posible elementos de uso común para abaratar los costes y simplificar el diseño final. La aproximación a la solución final se ha hecho desde el punto de perspectiva de una persona inexperta y novata en el tema de diseño, por lo que los pasos son progresivos y escalonados, en vez de previamente tener todo ajustado y diseñado para poder luego fabricar todo a medida según las predicciones, ya que el número de variables a tener en cuenta es muy alto y como se ha dicho no había experiencia previa con este tipo de proyectos. Todo el diseño de las partes del robot se ha hecho en el programa Solidworks 3D 2018, con licencia de estudiante cedida por la UPV. Los planos finales se adjuntan en la correspondiente parte.

Se explicará también el proceso del cambio y la decisión de convertir el brazo robot de 5 GDL, en uno de 6 grados de libertad. Todo este apartado estará basado en la ya previamente mencionada selección de componentes (apartado 4).

5.1 Diseño inicial

El primer factor decisivo que se ha tenido en cuenta para comenzar la fase de diseño de las partes fijas ha sido la elección del material, ya que este influye tanto económicamente, como en su peso, su resistencia y su capacidad de adaptación posterior, ya que si hay algún error a la hora de fabricar las partes, se requerirá un posterior trabajo de cambio o arreglo para que case con los demás componentes, y depende del material que ello sea menos o más costoso y tedioso, llegando a tener que fabricar de nuevo el componente en el peor de los casos, con lo que el precio se incrementaría, y se quiere evitar a toda costa esto ya que el proyecto es autofinanciado.

La primera idea que se ha barajado ha sido utilizar como método de fabricación el corte por láser de planchas de aluminio. El aluminio es un material metálico, muy ligero, con capacidades resistentes excelentes para esta aplicación. Su uso en la fabricación de robots está muy extendido, ya que además es un material relativamente maleable y flexible en determinadas condiciones, por lo que su adaptación a los diferentes escenarios es muy valorada. Se escoge este material como primera opción sin tener en cuenta el peso de este, simplemente se opta como primera opción pro las características ya mencionadas, pero se descartará por su elevado coste para el proyecto actual. El corte por láser de planchas de aluminio como método de fabricación, consiste en emplear un haz de luz concentrada con la suficiente energía sobre la superficie de trabajo, que permite un corte limpio y muy preciso incluso en materiales metálicos como es el caso del aluminio, por lo que se considera un método de fabricación excelente. Como se ha explicado anteriormente se descartó finalmente este método por temas económicos, pero a continuación se detallará por qué.

Para su fabricación, se contactó a una empresa Valenciana llamada Tecnología Ribarroja [19], la cual utiliza presentaba la posibilidad de usar tanto chorro de agua a presión como corte por laser, por lo que las posibilidades de elección de material eran varias, pero se escoge el aluminio por ser el metal más ligero y barato, como ya se ha mencionado. Esta empresa permite el corte en planchas de aluminio a medida según los planos que el cliente le envíe, por lo que no había limitaciones a la hora del diseño. No obstante, una vez contactada la empresa, se aproximó un presupuesto, pero

los precios aproximados se iban bastante del presupuesto esperado, ya que casi alcanzaban la mitad del presupuesto inicial esperado (unos 80+ euros) por lo que se rechaza esta opción como fabricación de las partes del robot.

Fig. 9. Corte por laser de planchas de aluminio. Fuente: [19]

La segunda idea que se baraja, ya que se quiere mantener la robustez de las piezas y mantener el peso bajo, es el corte por láser de madera. En concreto, un tipo de madera llamado fibropanel de densidad media o DM. Este material es un tipo de madera reconstruida, que se obtiene como descomposición de residuos de madera blanda o dura en fibras de madera, combinadas con cera y aglutinante de resina. Este método forma paneles mediante la aplicación de alta temperatura y presión, por lo que es un material, muy resistente, en cuanto a fuerzas es multidireccional, por lo que no solo trabaja en un eje y permite movimientos más complejos, y es muy denso, pero al mismo tiempo ligero, por lo que su valor a la hora de esta aplicación es muy alto. Para ello, se contacta con la empresa local Archicercle [20], un estudio especializado en arquitectura, diseño y fabricación digital, que, entre otros materiales con los que trabaja, está el DM. Ya que su peso y consistencia pueden ser variables dependiendo de los métodos de fabricación y del tipo en concreto de DM con el que se trabaje, se decide no calcular previamente los momentos que va a soportar el brazo robot, y experimentar posteriormente con los resultados.

Fig. 10 Primero, cortadora laser; segundo plancha con piezas recortadas. Fuente: propia.

Una vez tomada la decisión de qué material usar, se pasa a trabajar en el diseño. Todas las piezas diseñadas se recortan a laser sobre el DM. Se comienza decidiendo cuál será la base del robot. En vez de fabricarla se opta por una opción mucho más económica (debido a su tamaño), y ya fabricada, de un material con el suficiente peso, resistencia y dimensión para evitar que el momento

del brazo robot lo haga volcar. Para empezar, su forma debe ser redonda, ya que, al contrario de una forma cuadrangular, una forma esférica permite la misma distribución de esfuerzos en todas las direcciones, sin crear puntos de concentración o de desequilibrio. Por lo que una vez elegida cuál debe ser su forma y consistencia, se piensa en la madera como material, por su peso suficiente, sus características resistentes y su posible adaptación a la futura configuración. Para este propósito se escoge como base un plato de pulpo gallego invertido, de 29,5 cm de diámetro, de forma que queda la superficie plana cara arriba, el cual se puede obtener en cualquier tienda de barrio de todo a cien.

Fig. 11. Plato de pulpo gallego circular invertido. Fuente: propia.

Tal y como se puede apreciar en la fig. 11, se decide perforar una cavidad en la madera para alojar el motor encargado del giro principal del robot, de forma que el eje del motor queda centrado con el eje del plato. Esto se hace utilizando un taladro y otras herramientas para trabajar con madera, todo de forma manual. Para fijar el motor a la base, además de haber hecho la cavidad lo suficientemente ajustada para que el motor no requiera nada más para su sujeción, se añade una base de pegamento en el fondo de la cavidad para contrarrestar los momentos de tracción que sufre el motor y así evitar que se mueva en la dirección perpendicular a la base. Posteriormente se diseña la parte que va encima de este primer motor, una superficie plana redonda en la que se apoyará todo el brazo, y para ello se decide poner unos apoyos cercanos al motor para poder resistir estos momentos que al ser una plataforma giratoria pueden tener diferentes direcciones, por lo que se escogen unas “ruedas locas”, que son básicamente apoyos giratorios que permiten el deslizamiento libre de la plataforma giratoria pero soportan el peso y los momentos, que se trasladan a la base y no al motor, reduciendo así también el par que tiene que vencer el motor. Las ruedas locas se atornillan al plato de pulpo con tornillos en los puntos marcados con lápiz, de manera que mantengan una distancia equivalente al eje del motor. Inicialmente se decide poner 4, pero después se reducen a 3, ya que impiden el movimiento en algunas direcciones de la base, por razones que se explicaran posteriormente.

El siguiente paso como ya se ha dicho, es diseñar la plataforma giratoria, para ello simplemente se mide la distancia entre las ruedas de forma que cubra la trayectoria que tienen y que quede soportada la plataforma por ellas. Encima de la plataforma se diseñan dos placas verticales con

reborde superior semicircular, el cual es concéntrico al eje del segundo motor. Estas dos placas se atornillan a la plataforma giratoria, con 4 escuadras por cada placa. Los tornillos utilizados en las escuadras impiden el movimiento en algunas direcciones al chocar con las ruedas locas, por lo que al reducir a 3 el numero de ruedas, el problema se soluciona (se elimina la que da problemas), manteniendo suficientemente adecuado el apoyo de la plataforma giratoria con 3 puntos (mínimo posible). A este subconjunto se le denominará posteriormente como base del brazo. Una vez diseñada la esta parte(s) del brazo, se diseña un segmento de madera que une el segundo con el tercer motor, y su paralelo (sin unión de estos), el cual simplemente lleva un agujero lo suficientemente pequeño para poder unirlo a las placas anteriores, mediante un clip, como se aprecia en la fig. 12. A esta parte se le denominará posteriormente como codo del brazo.

Fig. 12. Uniones de la base y primeras partes del brazo. Fuente: propia.

La siguiente parte que se diseña es la pieza/s que alojan el primer giro de la pinza, así como el motor encargado del giro del codo del brazo. Como se puede apreciar en la fig. 13, el motor se aloja en la unión con el eje concéntrico a la parte semicircular del codo.

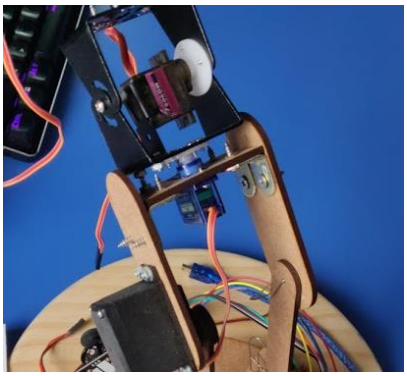


Fig. 13. Pre pinza. Fuente: propia.

Para la parte trasversal donde se aloja el motor del primer giro de la pinza, simplemente se diseña con una cavidad para alojar a este y un punto de sujeción, de forma que el centro de la pieza cuadra con el eje del motor, y se une a las partes próximas con 2 escuadras por cada lado, como se puede apreciar en la fig. 13.

Para la siguiente parte, en la que ya podemos considerar que comienza la pinza, se decide utilizar un producto ya fabricado, llamado pan and tilt [21], ya que requiere de mucha precisión, giros y tamaño pequeño, por lo que con el método actual de diseño y fabricación no se puede conseguir, debido a que la unión entre piezas se hace mediante escuadras y tornillos, lo cual ya ocupa y pesa lo suficiente y en la pinza es donde se genera el mayor momento. Este dispositivo aloja otro motor de tamaño inferior a los previos, y es el encargado del giro en el eje z de la pinza. Este motor, el cual es un SG90 (azul), tiene engranajes de plástico, como ya se ha mencionado anteriormente, y en un principio aguanta perfectamente el momento, pero con el paso del tiempo se desgasta el motor y los engranajes de plástico, y deja de funcionar o no consigue mover ya la pinza, por lo que se decide sustituir posteriormente por un MG90S (gris, véase fig. 13), que tiene engranajes metálicos y sí consigue moverlo continuamente.

A continuación, se diseña la pinza, eligiendo modelos clásicos de engranajes para pinzas, de forma que las patas de la pinza se muevan de forma horizontal en su recorrido para asegurar que el lugar de contacto es una superficie y no un punto. Para el motor encargado del cierre de la pinza se diseña una pequeña plataforma por debajo de la pinza para soportarlo y se encaja por el agujero con el engranaje unido al motor mediante pegamento. La pinza se acopla al dispositivo pan and tilt con dos escuadras por arriba y dos por abajo, como se puede apreciar en la fig. 14. Todas las uniones de los engranajes y patas de la pinza se realizan con clips también.

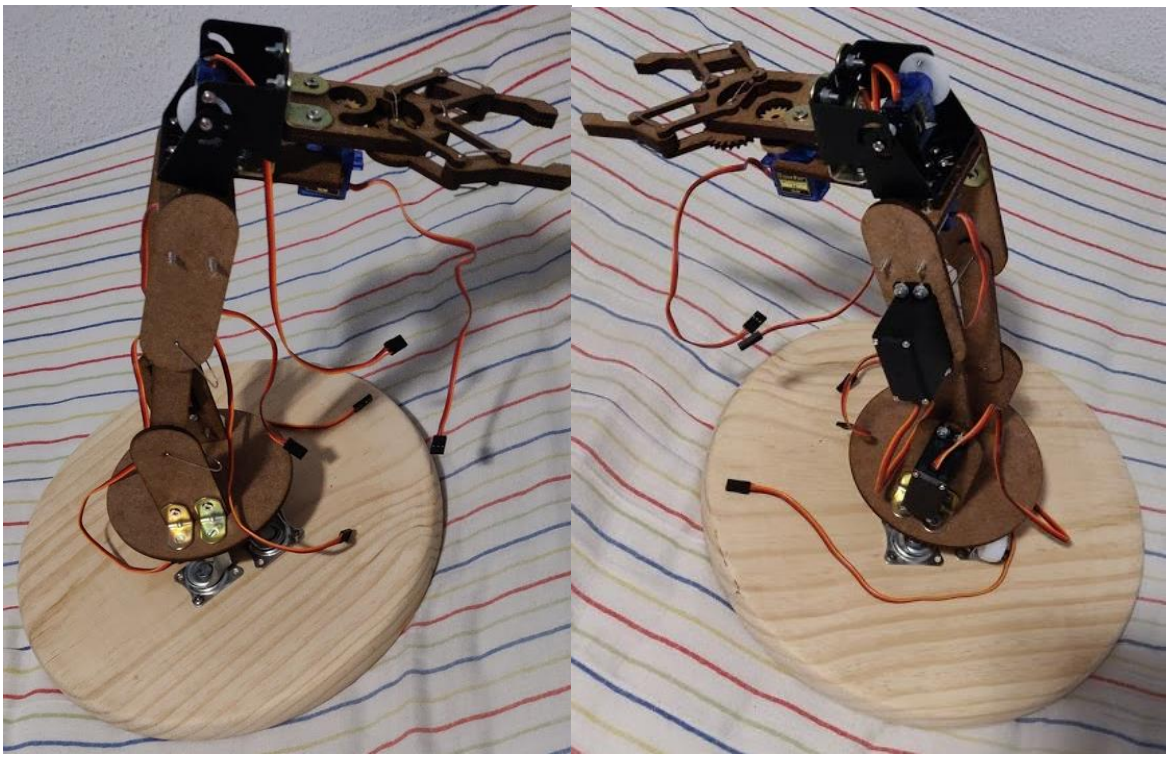


Fig. 14. Diseño del robot en madera. Fuente: propia.

Con el cambio de motor de la pinza de un SG90 a un MG90S, se comprueba mediante pruebas si consigue vencerlo, pero aún le cuesta en algunas ocasiones, por lo que se decide cambiar el material y proceso de fabricación de la pinza a impresión por 3D, lo cual se detallará posteriormente en el diseño final.

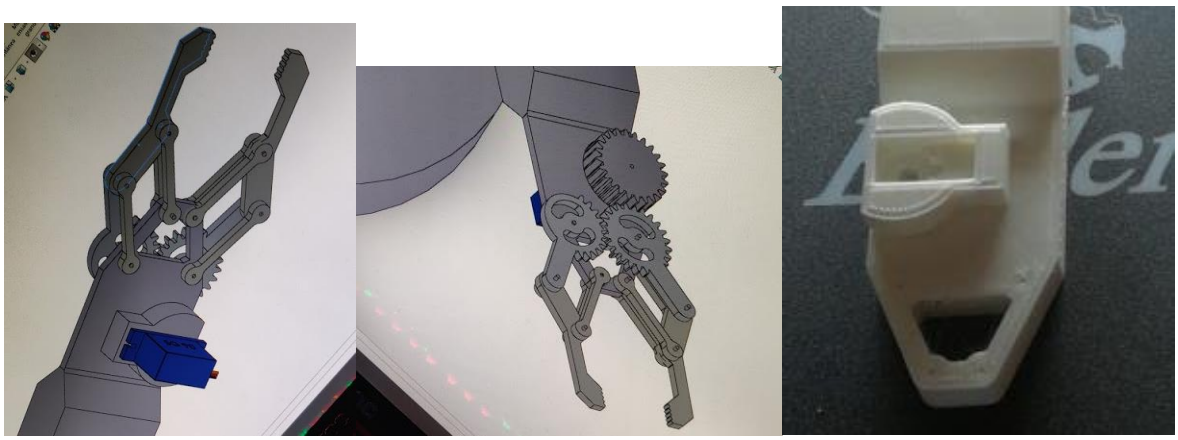


Fig. 15. Nuevo diseño de la pinza en 3D. Fuente: propia.

El motor ahora aguanta bien, pero el problema se traslada al segundo motor (al tener la pinza mayor longitud), situado en la plataforma giratoria, el cuál en movimientos rápidos del brazo o ascendentes rápidos, no aguanta, o pierde velocidad. Además, llegados a esta etapa, se decide añadir un grado de dificultad más al proyecto, y hacer un control cinemático del robot, tal y como se detalla en el marco teórico. Por estas razones se decide remodelar completamente el robot en Solidworks y hacer nuevamente el diseño de las partes fijas, pero esta vez teniendo en cuenta que utilizaremos como proceso la impresión 3D.

5.2 Diseño final

Debido a esta decisión de incluir un grado de libertad más, para poder realizar un mejor control cinemático o menos complejo que, si tuviera 5, el robot pasa a tener 6 grados de libertad. Con este grado de libertad nuevo se incluye un nuevo y segundo giro en la pinza, de manera que se aumentará el peso que soporta, con lo que ello conlleva.

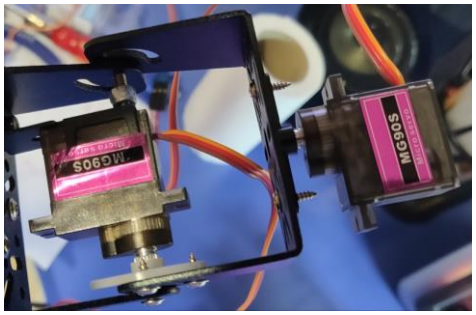


Fig. 16. Giro añadido de la pinza. Fuente: propia.

Se escoge como proceso de fabricación la impresión 3D, ya que el material que utiliza, ABS, es muy ligero, resistente al mismo tiempo y muy versátil a la hora de fabricar piezas ya que prácticamente cualquier cosa que se pueda modelar en 3D se puede imprimir. Inicialmente no se escoge este método de fabricación porque se quería hacer un brazo robot más robusto, ya que bajo ciertas deformaciones o tensiones el ABS al estar colocado por capas puede romperse o abrirse, pero finalmente, debido al presupuesto se decide tomar ese camino. Para ello no se contacta ninguna empresa como se ha hecho anteriormente, sino que se recurre a un amigo que dispone de una impresora 3D en su casa, lo cual facilita mucho todo el proceso.

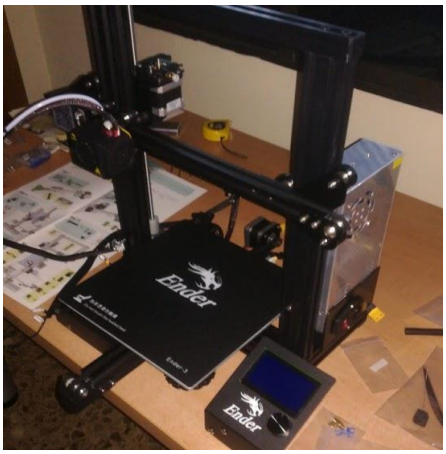


Fig. 17. Impresora con la que se ha realizado el proyecto. Fuente: propia.

Primero se comienza esbozando los cambios respecto al modelo anterior con 5 GDL. Por otro lado, ahora que disponemos de otro material, las uniones entre partes son mucho más eficientes y los diseños pueden adoptar soluciones más simples que si se quiere unir con escuadras y tornillos. La base del robot, es decir, el plato de pulpo de madera, junto con las ruedas y la cavidad, se decide que se mantenga como parte del diseño final, ya que es muy estable y no supone cambio alguno para el modelo final. La plataforma giratoria, que anteriormente tenia atornilladas las dos placas con escuadras, ahora lleva unidas estas dos placas, ya que puede fabricar de esta manera, con dos triángulos a cada lado con un ángulo de 45 grados para asegurar una suficiente sujeción. Ahora como podemos observar en la figura 18, se ha hecho una cavidad en la placa contraria a la que ya tenía en el diseño inicial de madera, para poder incluir un segundo motor que funciona exactamente igual que su opuesto. Esto se hace para poder resistir el par que ofrece el brazo ya que estos dos motores lo soportan en su totalidad, asegurando así un buen movimiento. Su funcionamiento es el mismo, pero con el ángulo invertido, que se tendrá en cuenta a la hora de programar.

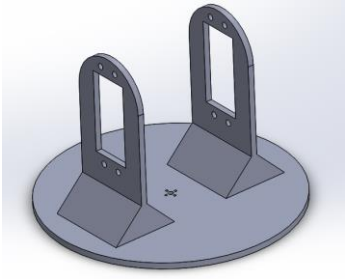


Fig. 18. Nueva disposición de la plataforma giratoria. Fuente: propia.

La siguiente pieza que se cambia de diseño es el codo del brazo, el cual anteriormente estaba formado por dos segmentos separados entre sí. Ahora se adopta una forma mucho más estable, la forma de H, de manera que ambos segmentos quedan unidos por medio de un apoyo rectangular que favorece y estabiliza el movimiento del robot en comparación al caso anterior. El codo se une a los dos motores anteriores mediante tornillos, que fijan la pieza al movimiento del motor, de forma que transmite directamente el par del motor. Se reduce la longitud del codo para reducir así también el par resistente que recibe el motor.

Ahora la parte denominada pre pinza, pasa a ser una pieza completa, en vez de ser 3 piezas, ya que no es necesaria la unión de las planchas mediante escuadras si se fabrica en 3D. El diseño es muy similar, aunque se ajustan las medidas para reducir el par también ya que el espacio que se necesitaba anteriormente para las escuadras ahora ya no es necesario. Además se unen las superficies perpendiculares mediante chaflanes de 45 grados asegurando así una mayor estabilidad.

Siguiendo el criterio de reducir los momentos, se decide cambiar por completo el diseño de la pinza, ya que ahora que se cuenta con un grado de libertad más, se tiene que añadir un motor nuevo, como se aprecia en la fig. 16. Por razones de tiempo a la hora de realizar un diseño muy eficiente para esta parte, la cual es excepcionalmente importante dentro del robot, y después de haber hecho varios diseños pero no acabar de cuadrar ninguno que ofreciera una solución al problema, y puesto que las aplicaciones que va a tener el brazo son reducidas o limitadas a objetos de poco peso y dimensiones pequeñas, se decide escoger un diseño ya existente [22], que se comprueba y funciona a la perfección, ya que el autor lo ha podido revisar y actualizar desde su primer diseño. Acorde a esta nueva pinza, se diseña una montura para el motor que soporta la estructura mencionada, de manera que queden el servo del giro de la pinza y el de la pinza en sí, lo más compactos posible, asegurando su estabilidad y simetría de manera que el motor que soporta la pinza se desplaza ligeramente a la izquierda para que el eje de la pinza cuadre con el eje de rotación del servo anterior.

Las técnicas de unión que se han utilizado a la hora de ensamblar el robot con el diseño final han sido simplemente tornillos atornillados en la estructura 3D en sí, ya que los agujeros se han diseñado de forma que quede muy ajustado para que al pasar el tornillo por el agujero se forme ya el surco que permita su fijación de manera permanente. En algunos casos, si el agujero es demasiado pequeño para el tornillo, por temas de inexactitud a la hora de fabricarlo o cualquier variable que hace que el diseño 3D sea diferente al real, se ha usado una aguja calentada con un mechero para poder hacer el agujero ligeramente más holgado.

En la siguiente hoja se puede apreciar el diseño final, una vez montado, fabricado y ensamblado todas las partes del brazo robot (véase fig. 19).

En los anexos se adjuntan los esquemas de montaje que se han seguido para las partes electrónicas.

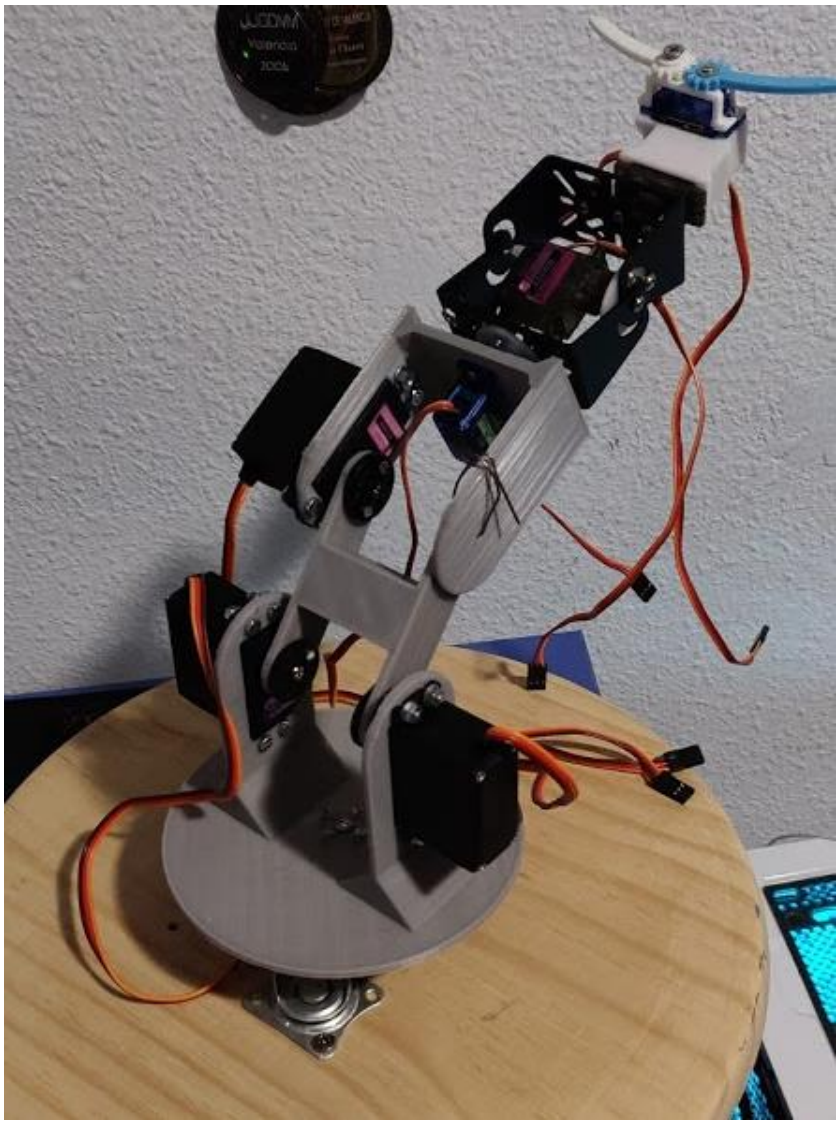


Fig. 19. Diseño final del brazo robot. Fuente: propia.

6. Elaboración del código

El código del robot es el programa que se escribe, en este caso en el lenguaje C++/Arduino, en la placa base, de forma que se puedan asignar tareas o determinadas funciones para los diferentes procedimientos del robot. Para escribir el código en el Arduino Nano, utilizaremos el software de Arduino IDE, versión 1.8.9 [23]. Además, se ha utilizado también el Dev C++, ya que como se comentará más adelante, ha habido problemas y este programa ha ayudado a solucionarlos o al menos a localizarlos e intentar solucionarlos.

En la realización de este código se han tenido en cuenta que la velocidad de giro de los servos no sea demasiado rápida para evitar problemas de sobreintensidades o demasiada demanda en ciertos momentos, la individualización de los movimientos por el mismo motivo, de manera que solo un servo se mueva cada vez y no todos al mismo tiempo ya que demandaría mucha potencia a la fuente.

La programación del robot ha sido la parte más difícil y compleja del proyecto estimando más 200 horas, entre el programa que se ha desarrollado con éxito y los intentos y pruebas de avanzar más en el trabajo, que se han llevado a cabo. Además, ha habido mucho aprendizaje posterior a los conocimientos adquiridos en la carrera para poder crear correctamente los programas que se buscaban. Por otra parte, el problema principal, además del tiempo final disponible, ha sido a la hora de declarar una gran cantidad de matrices en el Arduino ya que ha dado problemas al tratar de visualizar los resultados por pantalla, y no se ha podido comprobar si el resultado final de la programación era correcto, por este motivo o por otros posibles más complejos, que se escapan del alcance de este proyecto. Debido a esto y por el tiempo restante previo a la entrega, ya que una solución podría haber sido remodelar el brazo y aumentar las especificaciones de los componentes electrónicos (lo cual supondría mucho más tiempo añadido al proyecto, entre entrega de proveedores, pruebas, programación, etc.), no se ha podido llevar a cabo la cinemática de inversa, la cual permite determinar cambios de posición y orientación del sistema de coordenadas final de la pinza, como se detalla en el marco teórico, ni la de velocidades, que permite al robot moverse en tiempo real mediante las ordenes de los joysticks en las tres direcciones de los ejes y sus giros, que era la intención final. Por esa razón no se ha incluido el código de esos modos, pero si se ha desarrollado y se han empleado incontables horas en intentar hacer que funcionara, sin éxito lamentablemente.

A continuación, se detalla la metodología o el procedimiento que se ha llevado a cabo para diseñar el código del brazo robot.

6.1 Modos de funcionamiento

En un principio, se había diseñado el robot para que tuviese tres posibles modos de funcionamiento, modo manual, modo de cinemática inversa, y modo de cinemática de velocidades, los cuales no se han podido tener en un mismo programa por problemas de compatibilidad y declaración de matrices con el Arduino, entre otros como se ha detallado anteriormente. La idea inicial era a partir de interruptores o switches condicionar el modo de funcionamiento de manera que al activar uno u otro, se pudiese cambiar entre los distintos modos del brazo, de manera que el usuario pudiera elegir cuál es con el que quiere operar en cada momento.

El modo manual permite mover con cada eje de cada joystick un motor, como se ha explicado en la selección de componentes. Además, como cada joystick presenta un switch o interruptor, además de los 4 ejes disponibles se desbloquean hasta 8 en total, dependiendo del estado de los interruptores incorporados en los joysticks, de forma que se utilizan 6 ejes para los 6 grados de libertad del robot, y uno extra para poder cerrar y abrir la pinza.

En el siguiente apartado, se explica de forma detallada y esquematizada, el procedimiento que se sigue en el código a la hora de manejar el brazo de forma manual, por parte del usuario. El código en sí se incluye en los anexos al final de este documento.

6.1.1 Modo manual

Arduino IDE, el software de programación del brazo robot, es una derivación del lenguaje C++, ya que se basa en este pero que además utiliza librerías personalizadas y, otras funciones y clases integradas para facilitar la comunicación entre entradas y salidas (I/O) propias de este chip, como por ejemplo los servos. Por ello se incluye siempre que se vaya a trabajar con estos la librería Servo.h, ya que se utilizan las funciones y demás variables que ya se inicializan en esta librería.

Haciendo uso de esta librería, primero definimos como objetos de la clase "Servo" los 8 motores que utilizamos en este brazo robot. Esto se hace porque la librería Servo.h que definíamos previamente, está compuesta por la clase Servo y para utilizar las funciones (read, write, etc.) que incluye esta librería se necesita crear un objeto que haga uso de estas. A continuación, se declaran las variables que utilizaremos en el código constantemente: las posiciones de los puertos analógicos a los que están conectados los joysticks, las posiciones de los puertos digitales a los que van conectados los dos switches de los joysticks y el añadido pulsador y las variables en las que se guardarán los datos leídos y los que se usarán para escribir (pulsador, joystickVal, angle y swVal).

Arduino tiene una forma muy específica de estructurar su código, de forma que siempre debe haber dos funciones principales, el setup, que se usa como inicialización, y el loop, que es el programa que está constantemente utilizando, al cual se le añade un delay al final para dar un pequeño intervalo a que los objetos físicos (servos) no vayan por detrás de los datos digitales.

En el setup del modo manual, se asignan los servos creados como objetos, numerados del 1 al 8 y se les asigna el pin digital correspondiente al que están conectados. Por otra parte, para poder utilizar los switches o interruptores, se activan unas resistencias llamadas PULL UP, que se conectan entre 5V y 0V, de forma que se fuerza al interruptor a dar el valor HIGH o 1, cuando esta abierto, y 0 cuando esta presionado (véase fig. 20). Se puede configurar para que dé los valores contrarios, utilizando Resistencia PULL DOWN en vez de PULL UP. En el caso de los joysticks, se usan las resistencias internas del Arduino, y en el caso del pulsador externo, el componentes ya lleva integrada una pequeña resistencia. Además, dentro del setup se inicia la pantalla de visualización que utiliza Arduino IDE, llamada Serial, el cual se inicia a una velocidad de 9600 baud en este caso.

En el loop, tenemos por un lado la rutina que se le ha asignado al brazo cuando se presiona el pulsador, cuyo objetivo es posicionar los servos en la posición inicial, lo cual sirve para dejarlos como estaban antes de apagar el sistema, para que cuando tenga que iniciar la próxima vez no tengan que moverse; y por otro la rutina principal, con la que se mueven los servos con los joysticks.

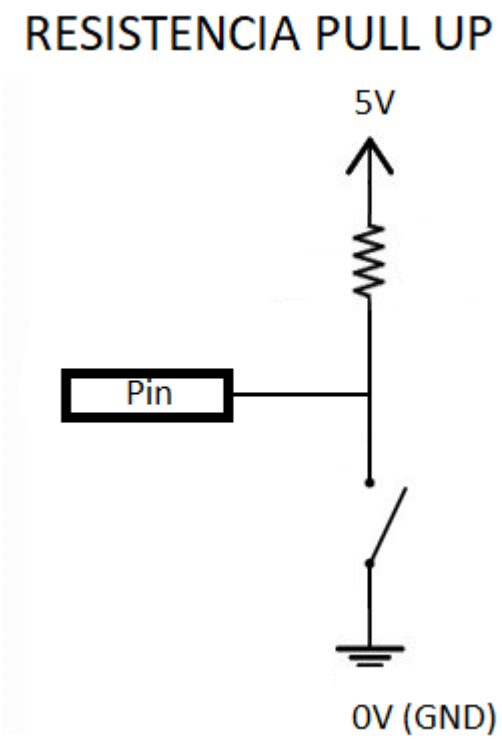


Fig. 20. Resistencia PULL UP. Fuente: propia.

La primera consiste en un condicionante, que a su vez encapsula dos condicionantes para cada servo (en total 8), uno que comprueba si el ángulo es superior al punto medio (93 grados) en el cual comienzan los servos al enchufarse por primera vez y otro que comprueba si el ángulo actual es inferior al punto medio, y devuelve al servo a su posición inicial o punto medio.

La segunda rutina, encargada de mover los servos con los joysticks sigue el siguiente patrón. Para los 4 primeros motores, utiliza los 4 ejes "básicos", es decir que no dependen de los interruptores de los 2 joysticks. Para cada motor de este tipo, primero lee el valor del joystick del eje asignado al motor y el ángulo actual del motor y guarda los valores para saber si se ha desplazado hacia arriba (1023) o hacia abajo (0) el joystick y así mover el servo hacia 180 o 0 grados, y lo hace de manera que suma o resta 4 grados según el caso, añadiendo un delay de 25 ms posteriormente para permitir que el servo llegue físicamente al lugar deseado antes de seguir. A excepción del servo 1 que, por la construcción física en el robot, su rango le impide realizar el barrido de 180 grados, y se mueve entre 40 y 100 grados, el resto de motores utilizan el rango entero.

Después tenemos los servos 4, 5 y 6, que se utilizan pulsando un joystick y con los ejes del joystick contrario, de forma que con uno pulsamos y con el otro desplazamos los servos (un eje para cada servo respectivamente). La dinámica es idéntica a los 4 primeros motores.

Y por último tenemos los servos 7 y 8, que son los servos en configuración doble, para mayor par. Estos servos actúan como uno mismo, por lo que en cuanto a grados de libertad se cuenta como uno. El servo predominante es el 7 y se mueve de 0 a 180, mientras que el 8 se mueve de 180 a 0, pero al estar enfrentados, es decir, tener sentidos de giro opuestos, ambos servos se mueven al unísono, de forma que el ángulo actual o en cada momento, al cual llamaremos "angle", se pasa al

servo 7, mientras que para el 8 se le pasa “180-angle”. Al final del loop, se incluye otro delay de 20 ms, por seguridad, para que los servos lleguen a su posición.

7. Conclusiones y posibles mejoras

Este proyecto, ha supuesto un gran aprendizaje al igual que motivación y progreso personal en cuanto al campo de la robótica, ya que supone un gran inicio. Se ha iniciado en este mundo con un proyecto que es muy representativo, como es el brazo robótico. El nivel de conocimientos en el que se encontraba al principio es muy inferior al final, por lo que se ha realizado un trabajo exhaustivo y muy completo. Se estima que el proyecto en total ha supuesto más de 500 horas de trabajo de las 300 esperadas. Por una parte, el aprendizaje inicial, la familiarización con los componentes, que hasta antes de realizar el proyecto eran nuevos para el autor, las pruebas realizadas con los distintos componentes y test de compatibilidad, los plazos de espera que han impedido en ocasiones avanzar ya que es un tiempo inevitable si ya se ha llegado a un punto en el que existe una barrera o impedimento que no se puede superar sin el nuevo componente, y por tanto esto ha hecho que el tiempo efectivo de trabajo estuviera muy espaciado o no continuado. Por otra parte, está el tiempo que ha supuesto programar, reaprender el lenguaje de programación, avanzar más en conceptos nuevos hasta ahora, necesarios para este proyecto, todo el tema de la cinemática de robots, que en la carrera se ha dado muy escasamente, de nuevo compatibilidad entre software y hardware, y el diseño continuo, con ajustes y cambios a medida que se iba avanzando en el proyecto.

Por estas razones, se puede considerar como un proyecto complejo y además completo por la cantidad de áreas que se tratan e integran en este trabajo. A parte del tema académico, ha sido un proyecto muy divertido de realizar y se seguirá trabajando en este tipo de proyectos en el tiempo libre a posteriori, ya que han despertado un gran interés.

En cuanto a objetivos, el proyecto comenzó como una idea semi difusa, en la que se pretendía manejar un brazo robot. Posteriormente se decidió que el manejo se haría mediante joysticks y eso influiría en gran medida en el diseño. Una vez completado el diseño “final”, en el que se incluye el modo manual, se decidió expandir los límites del trabajo, e incluir la cinemática directa, inversa y de velocidades. Desafortunadamente por los problemas comentados ya en la memoria no se pudo implementar en el Arduino, por lo que se ha tenido que descartar el incluirlo en esta memoria, pero se seguirá trabajando para hacerla funcionar en un futuro con más tiempo y presupuesto, ya que es una propuesta muy interesante que me gustaría llevar a cabo personalmente.

Como posibles mejoras o cambios en el brazo robot se enumeran los siguientes:

- Posible fabricación del pan and tilt en 3D para poder expandir el rango de movimientos del motor 1 a 180 grados, además de intentar buscar una opción de menor peso y dimensiones, aunque sea difícil.
- Añadir en el código de alguna forma, una función que permite a los servos dejar de comparar su posición actual con el potenciómetro que se modifica digitalmente, ya que existe una vibración o ruido que hace que el servo esté constantemente trabajando si no llega a su posición exacta en algunas ocasiones. Esto no es perjudicial, pero es un punto a tener en cuenta si se quiere hacer de forma perfecta.

Y como conclusión final, decir que se debe estar muy orgulloso del resultado y de poder haber realizado este proyecto y formar parte de él, y con muchas ganas y expectativas de futuro en el campo de la robótica.

8. Bibliografía

[1] Real Academia Española, «Diccionario de la lengua española. Edición del tricentenario», *Real Academia Española (RAE)*, 2017. [En línea]. Disponible en: https://dle.rae.es/?id=WYRlhzm. [Accedido: 22-abr-2019].

[2] S. et al Martín, «Historia de la robótica», *Actas Urol Esp*, vol. 31, n.º 2, pp. 69-76, 2007.

[3] Juan Bautista, «Evolución de la robótica - Monografias.com», 2018. [En línea]. Disponible en: https://www.monografias.com/trabajos107/evolucion-robotica/evolucion-robotica.shtml. [Accedido: 23-abr-2019].

[4] Á. V. Fernández, G. En, y Á. V. Fernández, «Trabajo Práctico : Control de Robots Móviles Control de Robots Móviles».

[5] D. de I. de S. y Automática y (DISA) UPV, «Cinemática directa».

[6] D. de I. de S. y Automática y (DISA) UPV, «Cinemática inversa».

[7] J. Legarreta, R. Martinez, y O. UPV/EHU, «MODELADO GEOMÉTRICO Y CINEMÁTICO DEL ROBOT».

[8] «Arduino - Compare». [En línea]. Disponible en: https://www.arduino.cc/en/products.compare. [Accedido: 03-feb-2020].

[9] «Arduino Nano v3.0 (compatible) – Robótica Fácil». [En línea]. Disponible en: https://roboticafacil.es/prod/arduino-nano3/. [Accedido: 03-feb-2020].

[10] «I/O Extension Shield para Arduino Nano – Robótica Fácil». [En línea]. Disponible en: https://roboticafacil.es/prod/arduino-nano-io-extension-shield/. [Accedido: 03-feb-2020].

[11] «Micro servo SG90 – Robótica Fácil». [En línea]. Disponible en: https://roboticafacil.es/prod/servo-sg90/. [Accedido: 01-may-2019].

[12] «ARCELI P MG90S Metal Geared Micro Servo para RC Coche Barco Avión Helicóptero T-Rex Trex450 x 2 PCS: Amazon.es: Electrónica». [En línea]. Disponible en: https://www.amazon.es/ARCELI-MG90S-Geared-Helicóptero-Trex450/dp/B07MY2SWMN/ref=pd_lpo_sbs_21_t_0?_encoding=UTF8&psc=1&refRID=YETBF3MBFBM2M55JH83P. [Accedido: 18-sep-2019].

[13] «Sedensy MG996R Motor Servos Digital Metal Gear para Futaba JR Coche RC Helicóptero, Show, mg996r: Amazon.es: Amazon.es». [En línea]. Disponible en: https://www.amazon.es/Sedensy-MG996R-Servos-Digital-Helicóptero/dp/B07P2WM5Q3/ref=sr_1_14?adgrpid=57805000524&gclid=Cj0KCQjwh6XmBRDRARIsAKNInDFoRrx1BKaBm5jp-krReb6YAH1Z4XwP-8JadHYJtVX2npNex3Zv6VcaAhKFEALw_wcB&hvadid=275292382983&hvdev=c&hvlocphy=1005. [Accedido: 01-may-2019].

[14] «Rueda Loca – Robótica Fácil». [En línea]. Disponible en: https://roboticafacil.es/prod/rueda-loca/. [Accedido: 02-may-2019].

[15] «Pulsador KY-004 – Robótica Fácil». [En línea]. Disponible en: https://roboticafacil.es/prod/pulsador-ky-004/. [Accedido: 10-feb-2020].

[16] «ESP8266 ESP32 Dual de la batería de litio 18650 escudo V8 5 V/3A 3 V/1A Banco móvil de la energía de la batería módulo de carga Micro USB para Arduino en Piezas y accesorios de

instrumentos de Herramientas en AliExpress.com | Alibaba Group». [En línea]. Disponible en: https://es.aliexpress.com/item/ESP8266-ESP32-Dual-de-la-bater-a-de-litio-18650-escudo-V8-5-V-3A-3/32967104569.html?spm=a2g0s.9042311.0.0.27424c4dN2pwfD. [Accedido: 01-may-2019].

[17] «Batería VTC5A 18650 2.600 mAh 35 Amperios - Baterias para todo Reguero Baterias». [En línea]. Disponible en: https://www.reguerobaterias.es/p90035303_bateria-vtc5a-18650-2-600-mah-35-amperios.html. [Accedido: 18-sep-2019].

[18] «Model Number US18650VTC5A Cell Type Cylindrical Cell Name US18650VTC5A Sony Code 49934720», 2015.

[19] «Tecnologia Ribarroja SL». [En línea]. Disponible en: http://www.cortelaser.com/index-1.htm. [Accedido: 04-feb-2020].

[20] «Archicercle Estudio Creativo. Arquitectura + diseño + Fablab». [En línea]. Disponible en: https://archicercle.com/. [Accedido: 04-feb-2020].

[21] «Kit Pan/Tilt: Amazon.es: Electrónica». [En línea]. Disponible en: https://www.amazon.es/gp/product/B00L7ZLF1K/ref=ppx_yo_dt_b_asin_title_o08_s00?ie=UTF8&psc=1. [Accedido: 03-feb-2020].

[22] «Simple Robot Gripper Servo Mount by hmikelson - Thingiverse». [En línea]. Disponible en: https://www.thingiverse.com/thing:3247600. [Accedido: 07-feb-2020].

[23] «Arduino - Software». [En línea]. Disponible en: https://www.arduino.cc/en/Main/Software. [Accedido: 05-may-2019].

Documento 2: *presupuesto*

II. Presupuesto

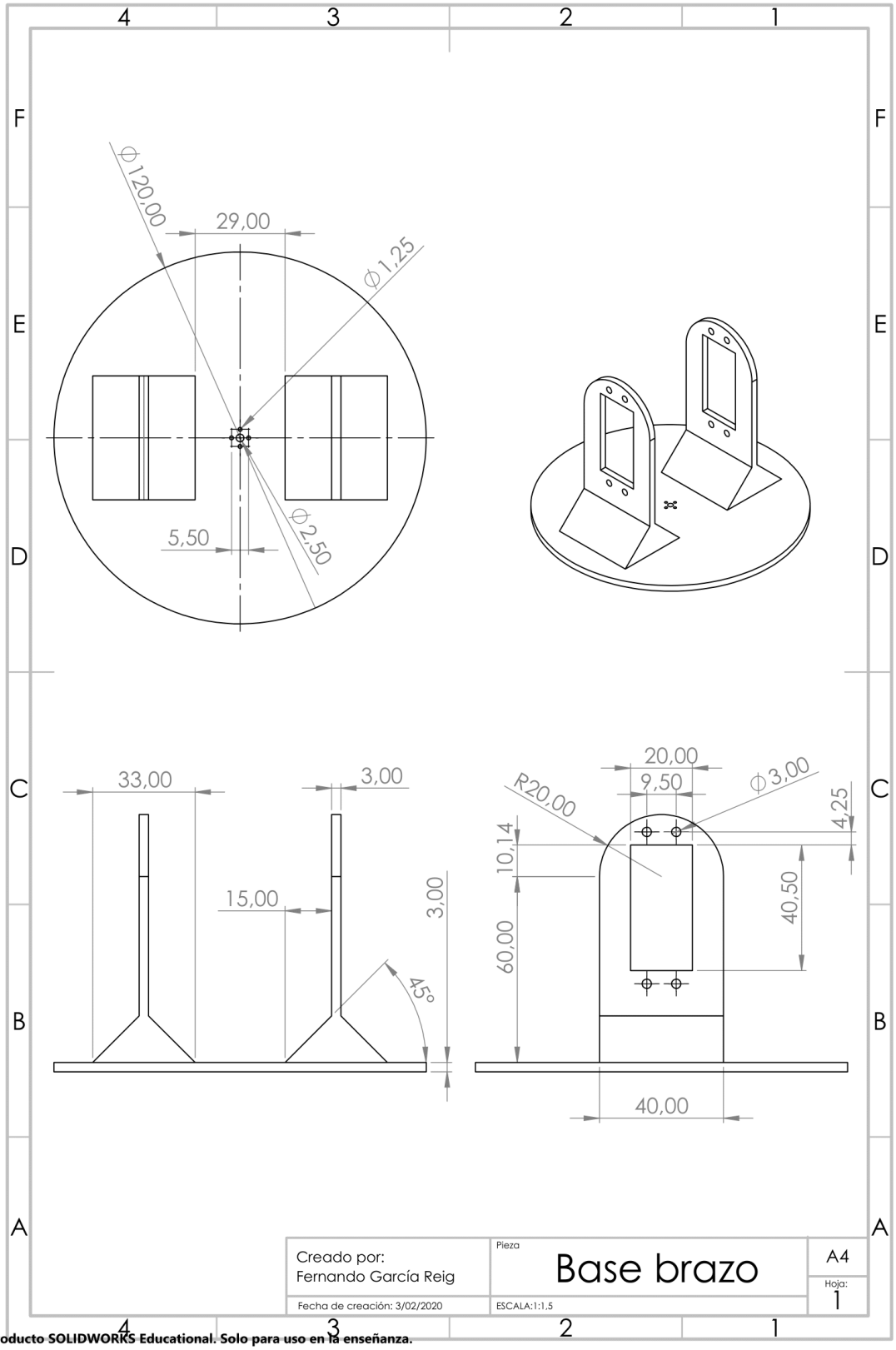
En este proyecto se ha invertido más de lo que se muestra en el cuadro de precios ya que se han querido hacer pruebas iniciales extras con distintos componentes hasta encontrar el propósito u objetivo final del proyecto, ya que cuando se inició no era claro, pero en el cuadro solo se reflejan los relevantes o los que se han tenido en cuenta al redactar esta memoria y los que han influido en el precio final del proyecto en sí.

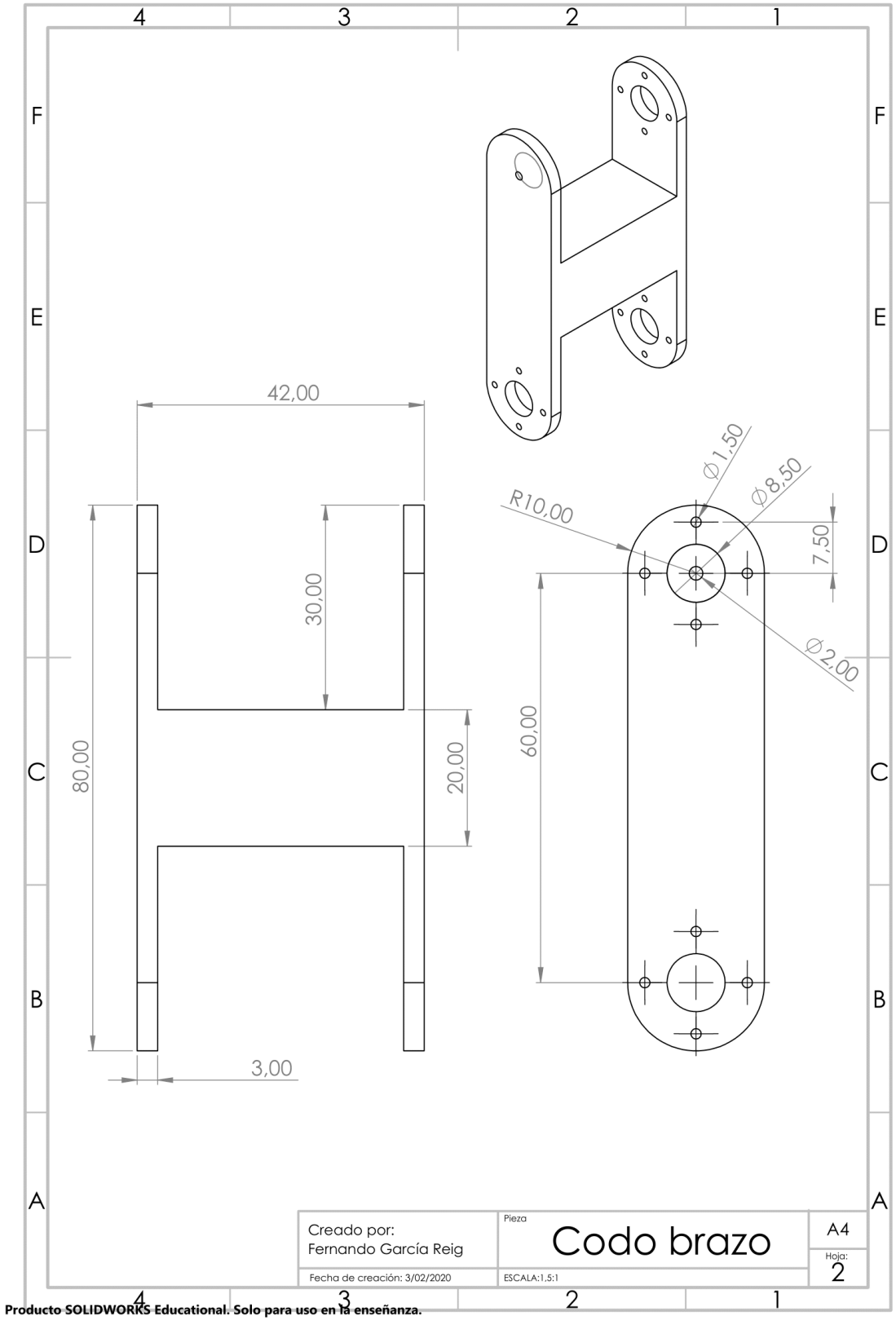
1. Cuadro de precios

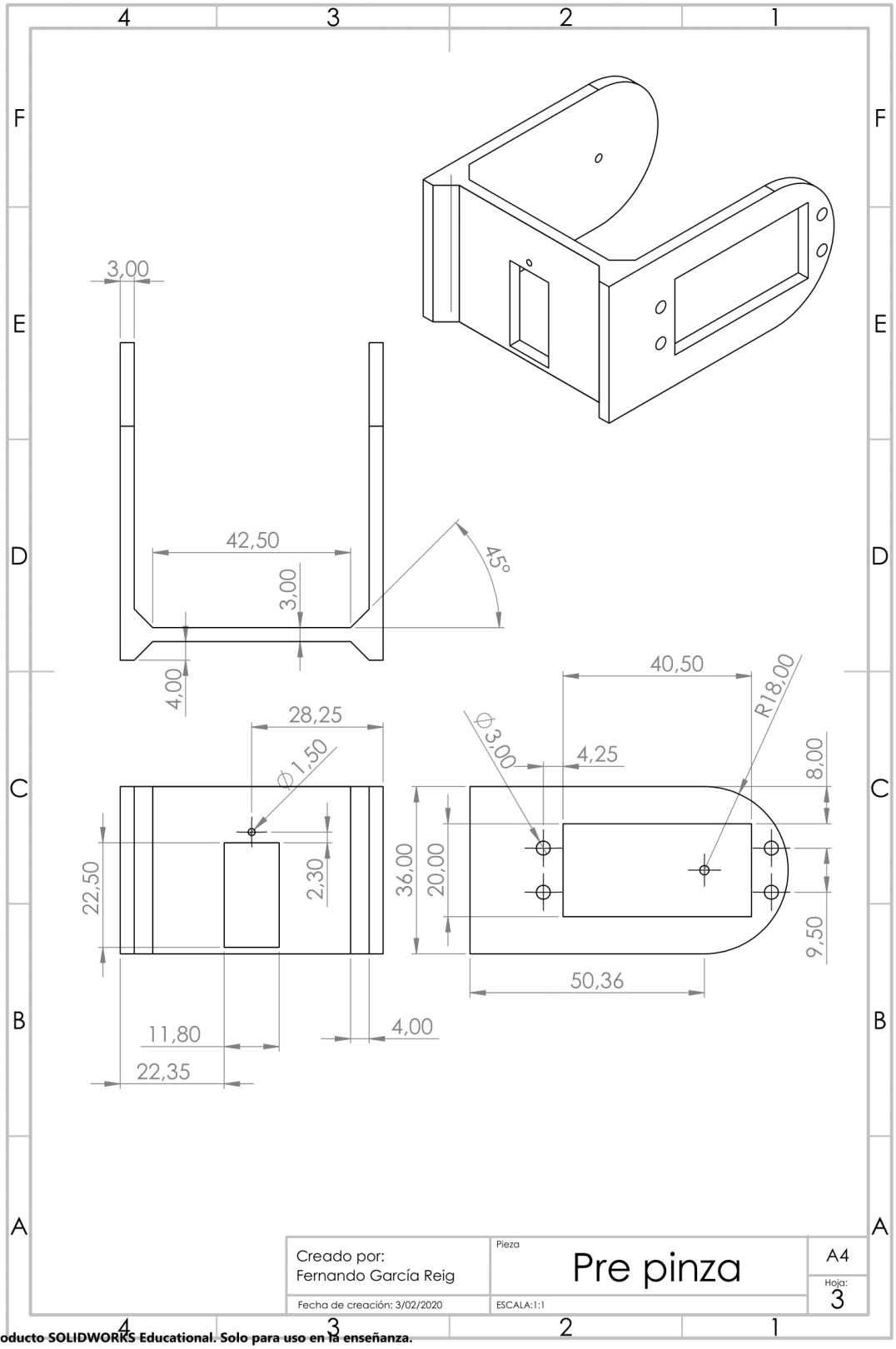
Ref	Ud.	Descripción (gastos de envío 0 si no se indica lo contrario)	Precio unit	Cant.	Parcial
m1	ud.	Arduino Nano v3	5,50 €	1	5,50 €
m2	ud.	I/O Extension Shield para Arduino Nano	4,50 €	1	4,50 €
m3	ud.	Batería dual de litio ESP8266 ESP32 18650 mAh 5V/2,2A 3V/1A	4,44 €	2	8,88 €
m4	ud.	gastos de envío del powerbank	1,56 €	1	1,56 €
m5	ud.	Multímetro	13,99 €	1	13,99 €
m6	ud.	Micro servo SG90	2,00 €	3	6,00 €
m7	ud.	Plato de madera circular para pulpo	4,80 €	1	4,80 €
m8	ud.	Motor servo Dilwee MG996R	9,99 €	3	29,97 €
m9	ud.	Batería VTC5A 18650 2.600 mAh 35 Amperios	5,50 €	2	11,00 €
m10	ud.	gastos de envío de la batería	4,72 €	1	4,72 €
m11	ud.	Cable mini-USB 30 cm	2,00 €	1	2,00 €
m12	ud.	Kit Pan/Tilt Amazon	7,02 €	1	7,02 €
m13	ud.	gastos de envío kit pan/tilt	4,95 €	1	4,95 €
m14	ud.	Micro servo MG90s	9,99 €	1	9,99 €
m15	ud.	Motor servo TeOhk MG996R	17,94 €	2	35,88 €
m16	ud.	Cables DuPont H-H 20 cm (pack de 10)	2,00 €	1	2,00 €
m17	ud.	Cables DuPont M-H 10 cm (pack de 10)	1,00 €	1	1,00 €
m18	ud.	Rueda loca	1,50 €	4	6,00 €
m19	ud.	TOOHUI 5Pcs Cable extensión servo 150 mm	6,39 €	1	6,39 €
m20	ud.	TOOHUI 5Pcs Cable extensión servo 300 mm	6,99 €	1	6,99 €
m21	%	Tornillos, cables, resistencias y demás materiales secundarios (aprox 15%)	25,95 €	1	25,95 €
		TOTAL			**199,09 €**

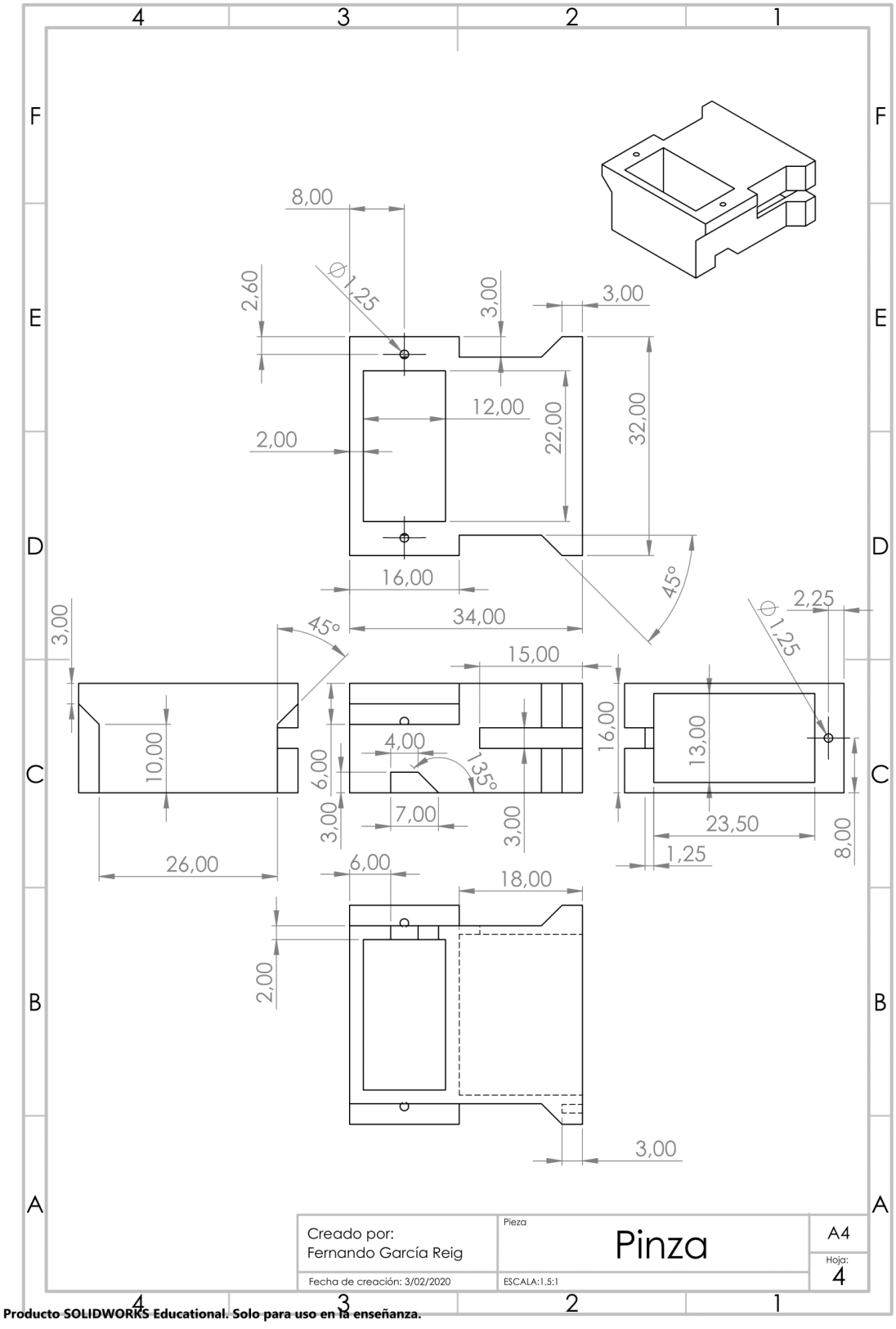
Documento 3: *planos*

III. Planos


4
3
2
1
F
E
D
C
B
A
Ø120,00
29,00
Ø1,25
5,50
Ø2,50
33,00
3,00
15,00
3,00
45°
20,00
9,50
R20,00
Ø3,00
4,25
10,14
40,50
60,00
40,00
Creado por:
Fernando García Reig
Pieza
Base brazo
A4
Hoja:
1
Fecha de creación: 3/02/2020
ESCALA:1:1,5
Producto SOLIDWORKS Educational. Solo para uso en la enseñanza.


4
3
2
1
F
E
D
C
B
A
42,00
30,00
80,00
20,00
3,00
R10,00
Ø1,50
Ø8,50
7,50
Ø2,00
60,00
Creado por:
Fernando García Reig
Pieza
Codo brazo
A4
Hoja:
2
Fecha de creación: 3/02/2020
ESCALA:1,5:1
Producto SOLIDWORKS Educational. Solo para uso en la enseñanza.


3,00
42,50
3,00
45°
4,00
28,25
Ø1,50
22,50
2,30
11,80
22,35
4,00
40,50
R18,00
Ø3,00
4,25
8,00
36,00
20,00
9,50
50,36
Creado por:
Fernando García Reig
Pieza
Pre pinza
A4
Hoja:
3
Fecha de creación: 3/02/2020
ESCALA:1:1
Producto SOLIDWORKS Educational. Solo para uso en la enseñanza.



Documento 4: *anexos*

IV. Anexos

1. Esquemas de montaje

Los esquemas de montaje se han diseñado con un programa llamado Fritzing, que permite diseñar conexiones entre componentes electrónicos de forma esquemática.

1.1 Servos

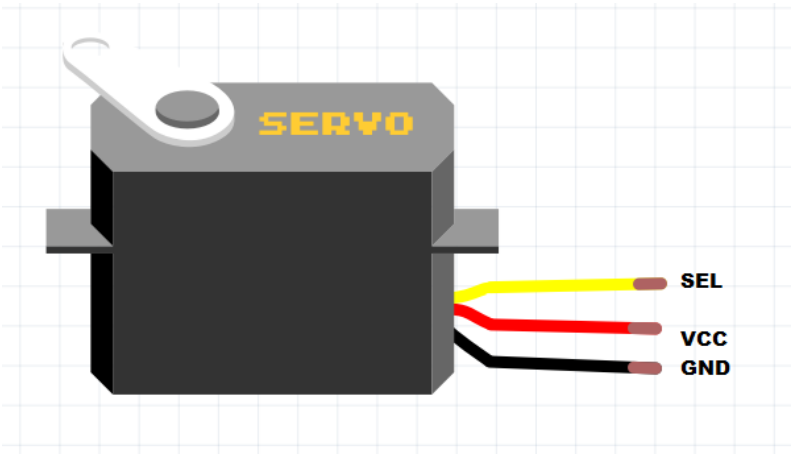


Fig. 21. Montaje servos. Fuente: propia.

1.2 Joysticks

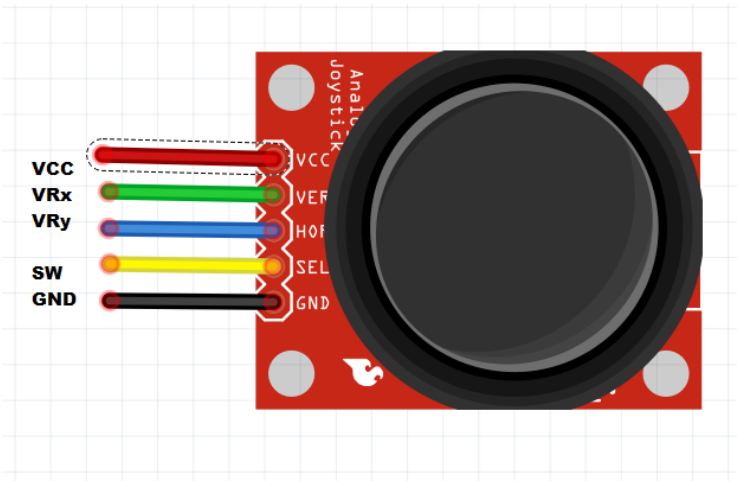


Fig. 22. Montaje joysticks. Fuente: propia.

1.3 Pulsador KY-004

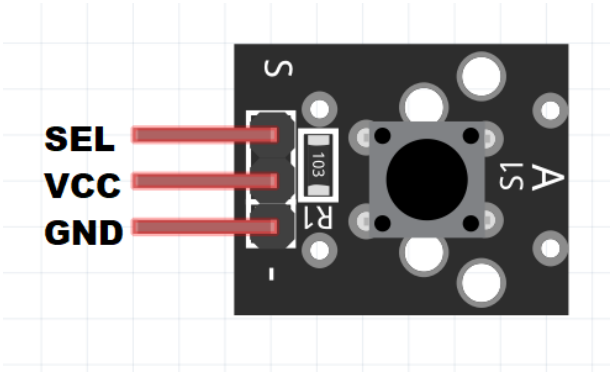


Fig. 23. Montaje pulsador. Fuente: propia.

1.4 Montaje general

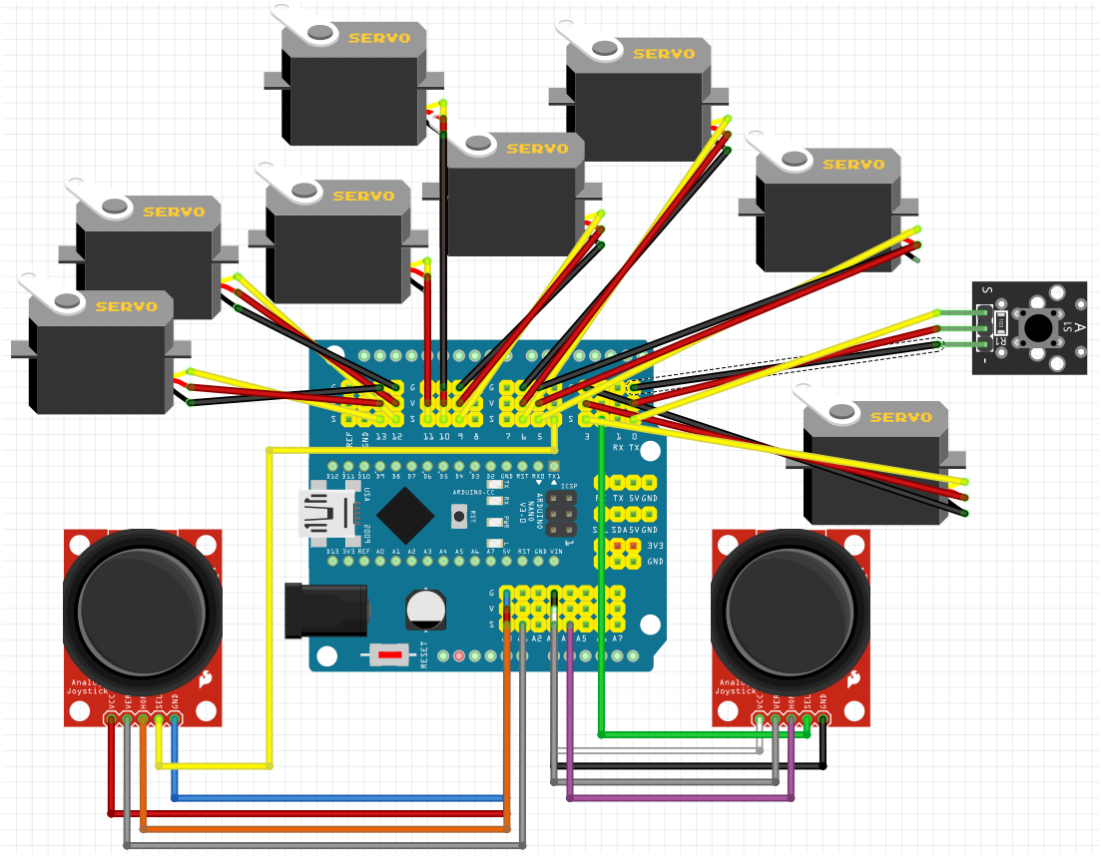


Fig. 24. Montaje general. Fuente: propia.

2. Código

2.1 Modo manual

```
#include <Servo.h>

Servo servo1;
Servo servo2;
Servo servo3;
Servo servo4;
Servo servo5;
Servo servo6;
Servo servo7;
Servo servo8;

int joyX=0;
int joyY=1;
int joyX1=3;
int joyY1=4;

int sw=4;
int sw1=2;
int sw2=0;

int pulsador;

int joystickVal;
int angle;
int swVal;

void setup()
```

```
{
 servo1.attach(3);
 servo2.attach(5);
 servo3.attach(6);
 servo4.attach(9);
 servo5.attach(10);
 servo6.attach(11);

 servo7.attach(12);
 servo8.attach(13);

 // activa resistencias pull-up en pin pulsador
 pinMode(sw, INPUT_PULLUP);
 pinMode(sw1, INPUT_PULLUP);
 pinMode(sw2, INPUT_PULLUP);

 Serial.begin(9600);
}

void loop()
{
pulsador=digitalRead(sw2);
//de normal está en posición 1 y mientras pulsas pasa a estar en posición 0

//vuelta a la posición base de 90 grados en cada servo

if(pulsador==0)
{
```

```
//servo 1
angle=servo1.read();
while(angle<=93)
{
  angle=servo1.read();
  angle=angle+1;
  servo1.write(angle);
  delay(20);
  Serial.println(angle);
}
while(angle>=93)
{
  angle=servo1.read();
  angle=angle-1;
  servo1.write(angle);
  delay(20);
  Serial.println(angle);
}

//servo 2
angle=servo2.read();
while(angle<=93)
{
  angle=servo2.read();
  angle=angle+1;
  servo2.write(angle);
  delay(20);
  Serial.println(angle);
}
while(angle>=93)
```

```
 {
   angle=servo2.read();
   angle=angle-1;
   servo2.write(angle);
   delay(20);
   Serial.println(angle);
 }

//servo 3
 angle=servo3.read();
 while(angle<=93)
 {
   angle=servo3.read();
   angle=angle+1;
   servo3.write(angle);
   delay(20);
   Serial.println(angle);
 }
 while(angle>=93)
 {
   angle=servo3.read();
   angle=angle-1;
   servo3.write(angle);
   delay(20);
   Serial.println(angle);
 }

 //servo 4
 angle=servo4.read();
 while(angle<=93)
```

```
 {
  angle=servo4.read();
  angle=angle+1;
  servo4.write(angle);
  delay(20);
  Serial.println(angle);
 }
 while(angle>=93)
 {
  angle=servo4.read();
  angle=angle-1;
  servo4.write(angle);
  delay(20);
  Serial.println(angle);
 }


//servo 5
 angle=servo5.read();
 while(angle<=93)
 {
  angle=servo5.read();
  angle=angle+1;
  servo5.write(angle);
  delay(20);
  Serial.println(angle);
 }
 while(angle>=93)
 {
  angle=servo5.read();
```

```
   angle=angle-1;
   servo5.write(angle);
   delay(20);
   Serial.println(angle);
  }

//servo 6
  angle=servo6.read();
  while(angle<=93)
  {
   angle=servo6.read();
   angle=angle+1;
   servo6.write(angle);
   delay(20);
   Serial.println(angle);
  }
  while(angle>=93)
  {
   angle=servo6.read();
   angle=angle-1;
   servo6.write(angle);
   delay(20);
   Serial.println(angle);
  }

//servo 7+8 (doble)
  angle=servo7.read();
  while(angle<=93)
  {
   angle=servo7.read();
```

```
   angle=angle+1;
   servo7.write(angle);
   servo8.write(180-angle);
   delay(20);
   Serial.print(angle);
   Serial.print("+");
   Serial.println(180-angle);
  }
  while(angle>=93)
  {
   angle=servo7.read();
   angle=angle-1;
   servo7.write(angle);
   servo8.write(180-angle);
   delay(20);
   Serial.println(angle);
   Serial.print("+");
   Serial.println(180-angle);
  }

}

//motor 1 (503)

  joystickVal = analogRead(joyX);
  angle=servo1.read();
  swVal = digitalRead(sw1);

  while (joystickVal < 493 && angle>40 && swVal==1) {
   joystickVal = analogRead(joyX);
```

```
  angle=angle-4;
  servo1.write(angle);
  delay(25);

 Serial.print("Posición Joystick X Rojo = ");
 Serial.print(joystickVal);
 Serial.print(" / ");
 Serial.print("Posición servo 1 = ");
 Serial.print(angle);
 Serial.print("\n");

  }
 while (joystickVal > 513 && angle<100  && swVal==1) {
  joystickVal = analogRead(joyX);
  angle=angle+4;
  servo1.write(angle);
  delay(25);

 Serial.print("Posición Joystick X Rojo= ");
 Serial.print(joystickVal);
 Serial.print(" / ");
 Serial.print("Posición servo 1 = ");
 Serial.print(angle);
 Serial.print("\n");

  }

//motor 2 (521-522)
```

```
joystickVal = analogRead(joyY);
angle=servo2.read();
swVal = digitalRead(sw1);

while (joystickVal < 500 && angle>1 && swVal==1) {
 joystickVal = analogRead(joyY);
 angle=angle-4;
 servo2.write(angle);
 delay(25);

Serial.print("Posición Joystick Y Rojo = ");
Serial.print(joystickVal);
Serial.print(" / ");
Serial.print("Posición servo 2 = ");
Serial.print(angle);
Serial.print("\n");

 }
while (joystickVal > 540 && angle<179 && swVal==1) {
 joystickVal = analogRead(joyY);
 angle=angle+4;
 servo2.write(angle);
 delay(25);

Serial.print("Posición Joystick Y Rojo = ");
Serial.print(joystickVal);
Serial.print(" / ");
Serial.print("Posición servo 2 = ");
Serial.print(angle);
```

```
 Serial.print("\n");

  }

//motor 3 (501-502)

 joystickVal = analogRead(joyX1);
 angle=servo3.read();
 swVal = digitalRead(sw);

 while (joystickVal < 490 && angle>1 && swVal==1) {
  joystickVal = analogRead(joyX1);
  angle=angle-4;
  servo3.write(angle);
  delay(25);

 Serial.print("Posición Joystick X Morado = ");
 Serial.print(joystickVal);
 Serial.print(" / ");
 Serial.print("Posición servo 3 = ");
 Serial.print(angle);
 Serial.print("\n");

  }
 while (joystickVal > 520 && angle<179 && swVal==1) {
  joystickVal = analogRead(joyX1);
  angle=angle+4;
  servo3.write(angle);
  delay(25);
```

```
  Serial.print("Posición Joystick X Morado = ");
  Serial.print(joystickVal);
  Serial.print(" / ");
  Serial.print("Posición servo 3 = ");
  Serial.print(angle);
  Serial.print("\n");

   }

//motor 4 (497)

  joystickVal = analogRead(joyY1);
  angle=servo4.read();
  swVal = digitalRead(sw);

  while (joystickVal < 487 && angle>1 && swVal==1) {
   joystickVal = analogRead(joyY1);
   angle=angle-4;
   servo4.write(angle);
   delay(25);

  Serial.print("Posición Joystick Y Morado = ");
  Serial.print(joystickVal);
  Serial.print(" / ");
  Serial.print("Posición servo 4 = ");
  Serial.print(angle);
  Serial.print("\n");

   }
  while (joystickVal > 510 && angle<179 && swVal==1) {
```

```
  joystickVal = analogRead(joyY1);
  angle=angle+4;
  servo4.write(angle);
  delay(25);

 Serial.print("Posición Joystick Y Morado = ");
 Serial.print(joystickVal);
 Serial.print(" / ");
 Serial.print("Posición servo 4 = ");
 Serial.print(angle);
 Serial.print("\n");

  }

//motor 5 (sw & 501-502)

 joystickVal = analogRead(joyX1);
 angle=servo5.read();
 swVal = digitalRead(sw);

 while (joystickVal < 490 && angle>1 && swVal==0) {
  joystickVal = analogRead(joyX1);
  angle=angle-4;
  servo5.write(angle);
  delay(25);

 Serial.print("Posición Joystick X Morado = ");
 Serial.print(joystickVal);
 Serial.print(" / ");
 Serial.print("Posición servo 5 = ");
```

```
 Serial.print(angle);
 Serial.print(" / ");
 Serial.print("Posición switch sw = ");
 Serial.print(swVal);
 Serial.print("\n");

  }
 while (joystickVal > 520 && angle<179 && swVal==0) {
  joystickVal = analogRead(joyX1);
  angle=angle+4;
  servo5.write(angle);
  delay(25);

 Serial.print("Posición Joystick X Morado = ");
 Serial.print(joystickVal);
 Serial.print(" / ");
 Serial.print("Posición servo 5 = ");
 Serial.print(angle);
 Serial.print(" / ");
 Serial.print("Posición switch sw = ");
 Serial.print(swVal);
 Serial.print("\n");
  }

//motor 6 (sw & 497)

 joystickVal = analogRead(joyY1);
 angle=servo6.read();
 swVal=digitalRead(sw);
```

```
while (joystickVal < 487 && angle>1 && swVal==0) {
  joystickVal = analogRead(joyY1);
  angle=angle-4;
  servo6.write(angle);
  delay(25);

Serial.print("Posición Joystick Y Morado = ");
Serial.print(joystickVal);
Serial.print(" / ");
Serial.print("Posición servo 6 = ");
Serial.print(angle);
Serial.print(" / ");
Serial.print("Posición switch sw = ");
Serial.print(swVal);
Serial.print("\n");

  }
while (joystickVal > 510 && angle<179 && swVal==0) {
  joystickVal = analogRead(joyY1);
  angle=angle+4;
  servo6.write(angle);
  delay(25);

Serial.print("Posición Joystick Y Morado= ");
Serial.print(joystickVal);
Serial.print(" / ");
Serial.print("Posición servo 6 = ");
Serial.print(angle);
Serial.print(" / ");
Serial.print("Posición switch sw = ");
```

```
 Serial.print(swVal);
 Serial.print("\n");

 }

//motor doble, servos 7 y 8 (sw1 & 503)

 joystickVal = analogRead(joyX);
 angle=servo7.read();
 swVal=digitalRead(sw1);

 while (joystickVal < 493 && angle>1 && swVal==0) {
  joystickVal = analogRead(joyX);
  angle=angle-4;
  servo7.write(angle);
  servo8.write(180-angle);
  delay(25);

 Serial.print("Posición Joystick X Rojo = ");
 Serial.print(joystickVal);
 Serial.print(" / ");
 Serial.print("Posición servo 7 y  8 = ");
 Serial.print(angle);
 Serial.print(" - ");
 Serial.print(180-angle);
 Serial.print(" / ");
 Serial.print("Posición switch sw1= ");
 Serial.print(swVal);
 Serial.print("\n");
```

```
  }
 while (joystickVal > 513 && angle<179  && swVal==0) {
  joystickVal = analogRead(joyX);
  angle=angle+4;
  servo7.write(angle);
  servo8.write(180-angle);
  delay(25);

 Serial.print("Posición Joystick X Rojo = ");
 Serial.print(joystickVal);
 Serial.print(" / ");
 Serial.print("Posición servo 7 y 8 = ");
 Serial.print(angle);
 Serial.print(" - ");
 Serial.print(180-angle);
 Serial.print(" / ");
 Serial.print("Posición switch sw1= ");
 Serial.print(swVal);
 Serial.print("\n");

 }

//tasa refresco del void 20 ms

 delay(20);
}
```