



COMILLAS

UNIVERSIDAD PONTIFICIA

ICAI

**GRADO EN INGENIERÍA EN TECNOLOGÍAS
INDUSTRIALES**

PROYECTO FIN DE GRADO
**MODELADO Y CONTROL DE UN BRAZO
SCARA**

Autor: Erik Martin Vik Espárrago
Director: Juan Luis Zamora Macho
Director: José Porras Galán

MADRID

Julio 2019

AUTORIZACIÓN PARA LA DIGITALIZACIÓN, DEPÓSITO Y DIVULGACIÓN EN RED DE PROYECTOS FIN DE GRADO, FIN DE MÁSTER, TESIS O MEMORIAS DE BACHILLERATO

1º. Declaración de la autoría y acreditación de la misma.

El autor D. Erik M Vik Espárrago DECLARA ser el titular de los derechos de propiedad intelectual de la obra: Modelado y control de un brazo SCARA, que ésta es una obra original, y que ostenta la condición de autor en el sentido que otorga la Ley de Propiedad Intelectual.

2º. Objeto y fines de la cesión.

Con el fin de dar la máxima difusión a la obra citada a través del Repositorio institucional de la Universidad, el autor **CEDE** a la Universidad Pontificia Comillas, de forma gratuita y no exclusiva, por el máximo plazo legal y con ámbito universal, los derechos de digitalización, de archivo, de reproducción, de distribución y de comunicación pública, incluido el derecho de puesta a disposición electrónica, tal y como se describen en la Ley de Propiedad Intelectual. El derecho de transformación se cede a los únicos efectos de lo dispuesto en la letra a) del apartado siguiente.

3º. Condiciones de la cesión y acceso

Sin perjuicio de la titularidad de la obra, que sigue correspondiendo a su autor, la cesión de derechos contemplada en esta licencia habilita para:

- a) Transformarla con el fin de adaptarla a cualquier tecnología que permita incorporarla a internet y hacerla accesible; incorporar metadatos para realizar el registro de la obra e incorporar “marcas de agua” o cualquier otro sistema de seguridad o de protección.
- b) Reproducir la en un soporte digital para su incorporación a una base de datos electrónica, incluyendo el derecho de reproducir y almacenar la obra en servidores, a los efectos de garantizar su seguridad, conservación y preservar el formato.
- c) Comunicarla, por defecto, a través de un archivo institucional abierto, accesible de modo libre y gratuito a través de internet.
- d) Cualquier otra forma de acceso (restringido, embargado, cerrado) deberá solicitarse expresamente y obedecer a causas justificadas.
- e) Asignar por defecto a estos trabajos una licencia Creative Commons.
- f) Asignar por defecto a estos trabajos un HANDLE (URL *persistente*).

4º. Derechos del autor.

El autor, en tanto que titular de una obra tiene derecho a:

- a) Que la Universidad identifique claramente su nombre como autor de la misma
- b) Comunicar y dar publicidad a la obra en la versión que ceda y en otras posteriores a través de cualquier medio.
- c) Solicitar la retirada de la obra del repositorio por causa justificada.
- d) Recibir notificación fehaciente de cualquier reclamación que puedan formular terceras personas en relación con la obra y, en particular, de reclamaciones relativas a los derechos de propiedad intelectual sobre ella.

5º. Deberes del autor.

El autor se compromete a:

- a) Garantizar que el compromiso que adquiere mediante el presente escrito no infringe ningún derecho de terceros, ya sean de propiedad industrial, intelectual o cualquier otro.
- b) Garantizar que el contenido de las obras no atenta contra los derechos al honor, a la intimidad y a la imagen de terceros.
- c) Asumir toda reclamación o responsabilidad, incluyendo las indemnizaciones por daños, que pudieran ejercitarse contra la Universidad por terceros que vieran infringidos sus derechos e intereses a causa de la cesión.
- d) Asumir la responsabilidad en el caso de que las instituciones fueran condenadas por infracción de derechos derivada de las obras objeto de la cesión.

6º. Fines y funcionamiento del Repositorio Institucional.

La obra se pondrá a disposición de los usuarios para que hagan de ella un uso justo y respetuoso con los derechos del autor, según lo permitido por la legislación aplicable, y con fines de estudio, investigación, o cualquier otro fin lícito. Con dicha finalidad, la Universidad asume los siguientes deberes y se reserva las siguientes facultades:

- La Universidad informará a los usuarios del archivo sobre los usos permitidos, y no garantiza ni asume responsabilidad alguna por otras formas en que los usuarios hagan un uso posterior de las obras no conforme con la legislación vigente. El uso posterior, más allá de la copia privada, requerirá que se cite la fuente y se reconozca la autoría, que no se obtenga beneficio comercial, y que no se realicen obras derivadas.
- La Universidad no revisará el contenido de las obras, que en todo caso permanecerá bajo la responsabilidad exclusiva del autor y no estará obligada a ejercitar acciones legales en nombre del autor en el supuesto de infracciones a derechos de propiedad intelectual derivados del depósito y archivo de las obras. El autor renuncia a cualquier reclamación frente a la Universidad por las formas no ajustadas a la legislación vigente en que los usuarios hagan uso de las obras.
- La Universidad adoptará las medidas necesarias para la preservación de la obra en un futuro.
- La Universidad se reserva la facultad de retirar la obra, previa notificación al autor, en supuestos suficientemente justificados, o en caso de reclamaciones de terceros.

Madrid, a 2 de Julio de 2019

ACEPTA

Fdo 

Motivos para solicitar el acceso restringido, cerrado o embargado del trabajo en el Repositorio Institucional:

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título
Modelado y control de un brazo SCARA
en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el
curso académico 2018/2019 es de mi autoría, original e inédito y
no ha sido presentado con anterioridad a otros efectos. El Proyecto no es plagio de otro,
ni total ni parcialmente y la información que ha sido tomada
de otros documentos está debidamente referenciada.

Fdo.: Erik Martin Vik Espárrago

Fecha: 2/07/2019



Autorizada la entrega del proyecto
EL DIRECTOR DEL PROYECTO

Fdo: Juan Luis Zamora Macho Fecha: 2/07/2019



Fdo.: José Porras Galán Fecha: 2/07/2019





COMILLAS

UNIVERSIDAD PONTIFICIA

ICAI

**GRADO EN INGENIERÍA EN TECNOLOGÍAS
INDUSTRIALES**

TRABAJO FIN DE GRADO
**MODELADO Y CONTROL DE UN BRAZO
SCARA**

Autor: Erik Martin Vik Espárrago
Director: Juan Luis Zamora Macho
Director: José Porras Galán

MADRID

Julio 2019

MODELADO Y CONTROL DE UN BRAZO SCARA

RESUMEN

1. Introducción

Este proyecto trata del modelaje y control de un brazo SCARA, el cual se ha diseñado utilizando una herramienta de diseño 3D, y producido posteriormente con una impresora también 3D. El control se programa con un microcontrolador que se comunica con los distintos actuadores y sensores del sistema.

La principal motivación de este proyecto es modelar y montar un brazo SCARA funcional, puesto que este tipo de brazo robóticos son de gran importancia en la industria. Además, existe una motivación didáctica puesto que se utilizará la Raspberry 3b+ y servos inteligente (LX-16 A o SCS15) y su estudio facilitará su uso en futuros proyectos y su implementación en las aulas de trabajo.

2. Objetivos

El desarrollo de este proyecto comprende una serie de objetivos secundarios, sin los cuales no se podría alcanzar el objetivo principal del proyecto: el desarrollo completo de un brazo SCARA. Los objetivos secundarios propuestos son:

- Fabricar un brazo SCARA 2D y equiparlo con toda la electrónica requerida para su control
- Construir un modelo matemático del robot que permite simular de forma precisa su comportamiento
- Diseñar un sistema de control e implantarlo en el entorno Matlab/Simulink

3. Metodología

En primer lugar, se realizó una investigación previa del brazo SCARA. Tras obtener una idea mas clara de éste, se procedió a seleccionar las herramientas, y materiales que se utilizarían. Esto incluye una investigación intensiva del hardware (Servos inteligentes, Raspberry pi 3B+), donde se analiza sus conexiones, comunicaciones y su correcto funcionamiento. Por otro lado, se diseño la estructura del brazo y se construyo utilizando una impresora 3D.

A continuación , se realizan una serie de ensayos para identificar la planta que se va a controlar, debido a las dinámicas del servo no se puedo identificar una planta que se pudiese controlar. Por tanto, los ángulos que debían girar los servos, se les envía directamente, y son calculados a través de la cinemática inversa del brazo, lo cual permitiría colocar el extremo en la posición que se quisiera del espacio x-y.

Tras comprobar que los cálculos mencionados eran correctos y que el brazo alcanzaba la posición deseada, se procedió a crear trayectorias geométricas simples que el brazo debía seguir. Seguidamente, con el objetivo de seguir trayectorias mas complejas, se creo un programa que permitiera la vectorización de imágenes, para posteriormente crear las trayectorias que el brazo debería seguir.

Una vez se completo el software, se realizo un montaje final del brazo SCARA y se obtuvieron resultados.

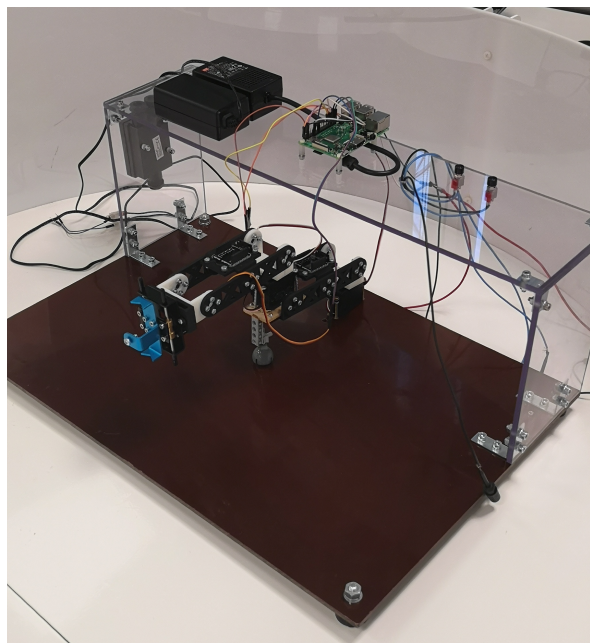


Figura 1 Brazo SCARA montado con toda la electrónica

4. Resultados

En la toma de resultados, debido a algunos problemas que han surgido durante el proyecto se ha tenido que cambiar los servos inteligentes por servos convencionales, y con estos se ha probado todo el software diseñado.

A continuación, se pueden ver los distintos trazados que son enviados a nuestro brazo SCARA. En rojo se ven las coordenadas enviadas, y en azul las trayectorias que dibujará el brazo.

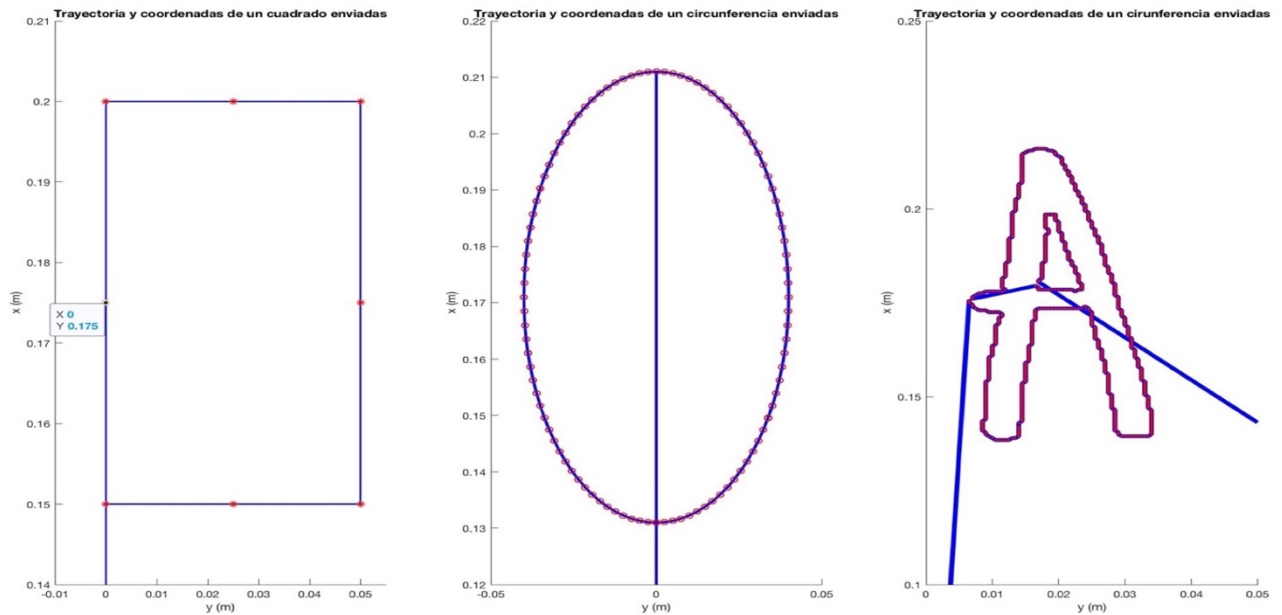


Figura 1 Trayectorias y coordenadas enviadas al brazo SCARA

A partir de estas coordenadas se puede obtener las posiciones angulares equivalentes, gracias a la cinemática inversa que se ha implementado. En la Figura 3 podemos ver las posiciones angulares equivalentes enviada a los servomotores.

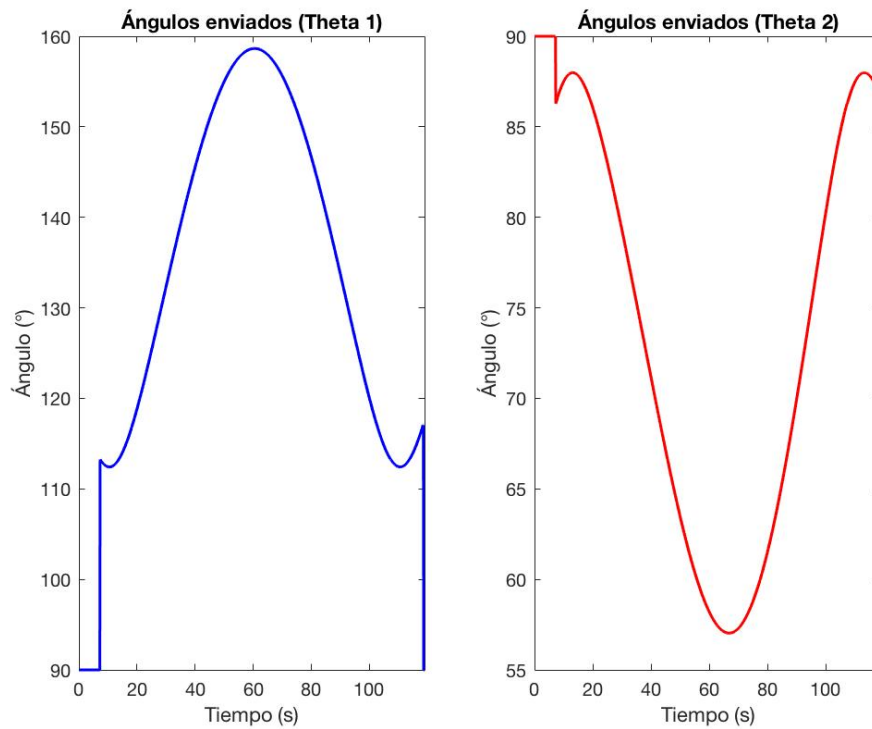


Figura 3 Posiciones angulares de una circunferencia enviadas a cada servo

Finalmente, estos ángulos son enviados a nuestro servo y somos capaces de obtener los resultados sobre el papel. Estos se pueden ver en la Figura 4. La trayectoria obtenida sobre el papel, se diferencia mucho a la que se usa de referencia. Esto se debe a la precisión de los servos que finalmente se han tenido que utilizar, y a las vibraciones que estos provocaban en el extremo del brazo. De izquierda a derecha se puede ver como estas vibraciones disminuyen, esto se consiguió aumentando el tiempo que tarda el brazo en dibujar la circunferencia.



Figura 4 Trayectorias dibujadas sobre el papel

5. Conclusiones

Se ha visto que la estructura diseñada funciona correctamente. Esta puede ser mejorada con un apoyo para el extremo del brazo, para que este sea mas robusto.

En cuanto a los servos inteligentes, disminuyen mucho el cableado , permiten conectar muchos servos sin utilizar muchos pines, facilitan obtener información sobre ellos, por ello se cree que pueden ser muy útiles para futuros proyectos. Pero debido a los fallos que se han obtenido, seria conveniente utilizar servos de otros fabricantes. Por un lado el fabricante lewansoul ha proporcionado un buslinker que ha dado muchos problemas y finalmente han dejado de funcionar. Y por otro lado , la empresa feetehe proporciona unos servos con apenas información sobre su protocolo de comunicación , y con un software para configurar el servo que no ha funcionado.

Para el control de un brazo SCARA sería mas interesante utilizar motores y encoder,, puesto las dinámicas de estos servos no permiten diseñar un control.

En cuanto al software implementado, se ha comprobado que este funciona. La máquina de estados, la cinemática inversa, el software de vectorización de imágenes, y las trayectorias de la circunferencia y cuadrado estaban bien diseñadas. Por su correcto funcionamiento, este software podría ser reutilizado en futuros proyectos.

DESIGN AND CONTROL OF A SCARA ROBOT

ABSTRACT

1. Introduction

The main objective of this project is the desing and control of a SCARA ROBOT, which has been designed using a 3D design tool, and then produced with a 3D printer. The control is programmed with a microcontroller that communicates with the different actuators and sensors of the system.

2. Objectives

For the development of the project we will have to complete a series of objectives. These are:

- Manufacture of a 2D SCARA robot, which will include all the electronics needed for its control
- Developing a mathematical model of the robot
- Design a control and implementing this one with Matlab/Simulink

3. Methodology

To complete the project successfully, a series of minor tasks must be carried out:

First, a preliminary investigation of the SCARA arm will be conducted. After the initial research, we selected the tools and materials that would be used. This includes an intensive investigation of the hardware (intelligent Servos, Raspberry pi 3B +). And an initial design of the structure of the arm, and the different material that could be used.

Then, a series of tests are carried out to identify the plant. Once we see thath this on is very simple , and that its study is not as interesting, we decided to do a direct control. The angles that the servos had to turn, are sent directly to them, and are calculated through the inverse kinematics of the arm, which would allow to place the robot in the desired position of the cartesian coordinates.

After checking that the mentioned calculations were correct and that the arm reached the desired position, we proceeded to create geometric trajectories that the arm had to follow. Then, with the aim of following more complex trajectories, a program was created that allowed the vectorization of images, to later create the trajectories that the arm should follow.

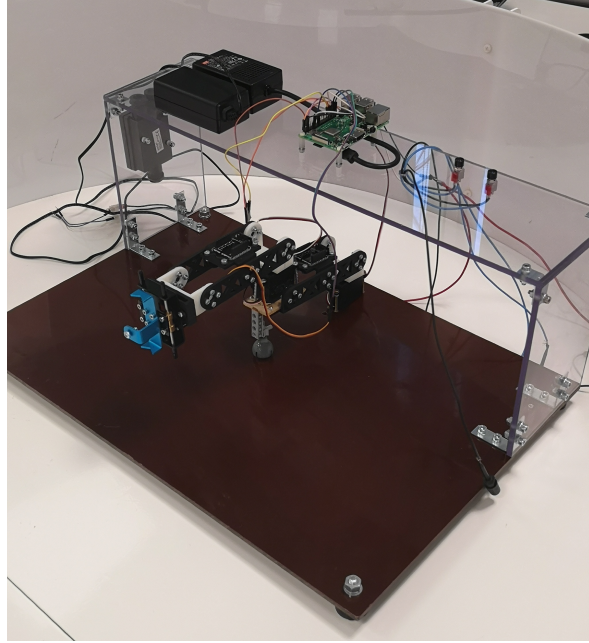


Figure 1. SCARA arm and all the electronic components that have been used

4. Results

When taking the results, due to some problems that arise during the project it has been necessary to change the smart servos for conventional servos, and with these we have tested the designed software.

In the Figure 1 you can see the different trajectories that are sent to out SCARA arm. In red you can see the coordinates that are sent, and in blue the trajectories that the arm will draw.

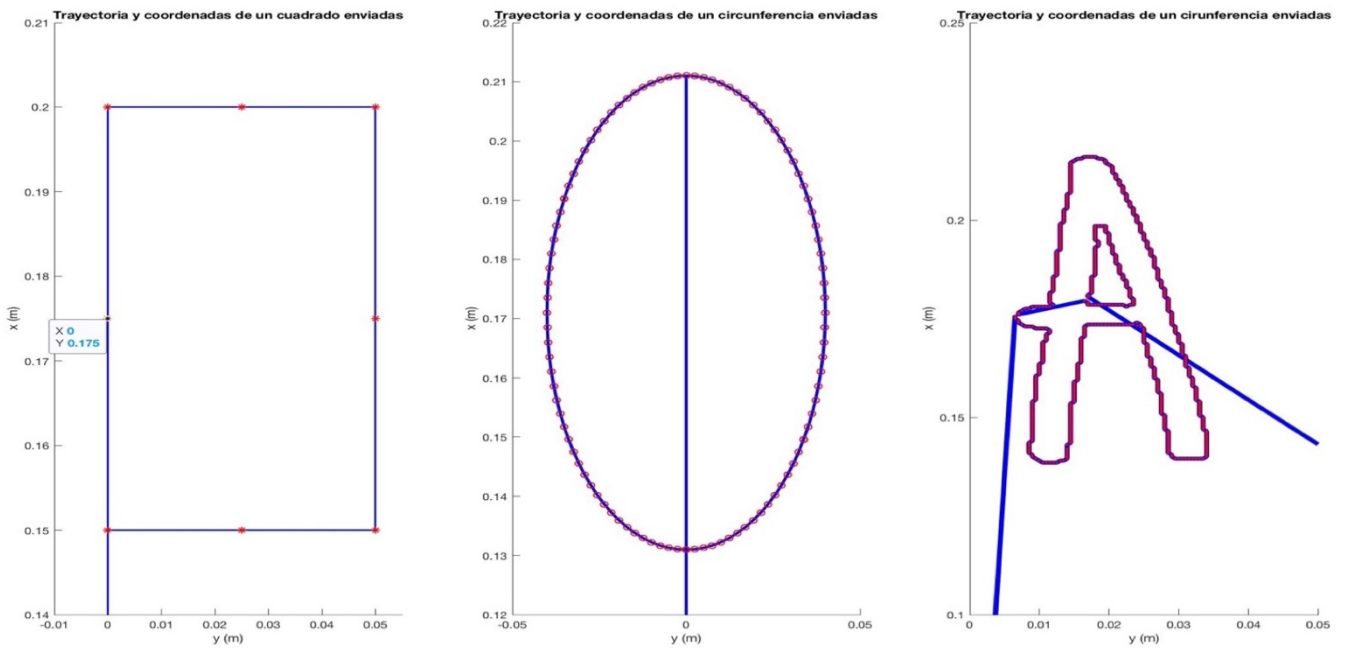


Figure 2. Trajectories and coordinates sent to the SCARA arm

Using these coordinates we can obtain the equivalent angular positions that must be sent to the servos. To do this we use the inverse kinematics that have been implemented. In the Figure 3, we can see the angular positions of a circumference

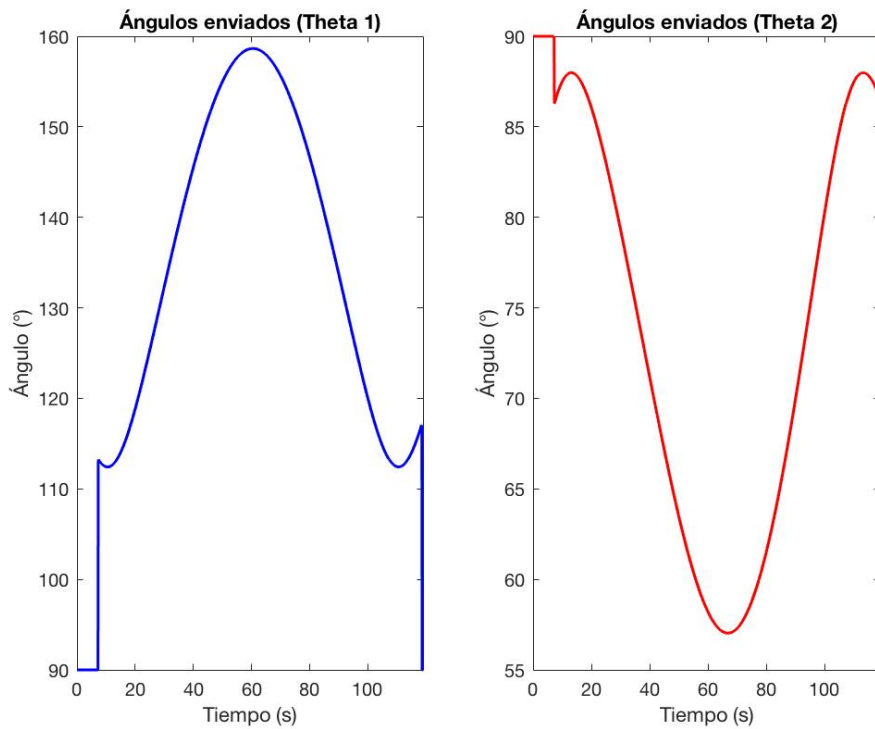


Figure 3 Angular positions of a circumference sent to the servo

Finally , these angle are sent to the servos and we are able to obtain the final results. These can be seen in the Figure 4. The trajectory obtained on paper is very different from the one used as a reference. This is due to the precision of the servos that finally had to be used, and the vibrations that they caused at the end of the arm .. From left to right you can see how these vibrations decrease, this was achieved by increasing the time it takes the arm in drawing the circumference.



Figure 4 Trajectories drawn on paper

5. Conclusions

It has been seen that the designed structure works correctly. This can be improved with a support for the end of the arm.

As for the intelligent servos, they reduce the wiring, they allow to connect many servos without using many pins, they facilitate obtaining information about them, for that reason it is believed that they can be very useful for future projects. But due to the problems that have been encountered, it would be convenient to use servos from other manufacturers. On the one hand the manufacturer lewansoul has provided a buslinker that has given many problems and have finally stopped working. And on the other hand, the company feetehe provides servos with little information about its communication protocol, and with software to configure the servo that has not worked. In spite of everything, for the control of a SCARA arm it would be more interesting to use motors, since the dynamics of these servos do not allow to design a control.

Regarding the software implemented, it has been proven that it worked. The state machine, the inverse kinematics, the vectorization software of images, and the trajectories of the circumference and square were well designed. Due to its correct functioning, this software could be reused in future projects.

PARTE I: Memoria descriptiva

Índices

Índice general

RESUMEN	- 7 -
ABSTRACT.....	- 12 -
PARTE I: MEMORIA DESCRIPTIVA.....	- 17 -
CAPÍTULO 1. INTRODUCCIÓN	- 24 -
1.1 Motivación	- 24 -
1.2 Objetivos	- 24 -
1.3 Metodología	- 24 -
1.4 Recursos a emplear	- 25 -
1.5 Estructura de la memoria	- 26 -
CAPÍTULO 2. ESTADO DEL ARTE	- 27 -
2.1 Historia	- 27 -
2.2 Modelo	- 28 -
2.3 Controles utilizados	- 29 -
2.3.1 Control PID	- 29 -
2.3.2 Control por par calculado (CPC)	- 30 -
2.3.3 Control predictivo	- 30 -
2.3.4 Control de redes neuronales	- 31 -
2.3.5 Aplicaciones	- 31 -
CAPÍTULO 3. HARDWARE.....	- 33 -
3.1 Raspberry PI 3B+	- 33 -
3.2 Servomotores inteligentes	- 33 -
3.2.1 Comunicación	- 34 -
3.2.2 Servo LX-16 A	- 34 -
3.2.3 Servo SCS15	- 36 -
3.3 Servo EMAX ES08AII Analog servo	- 37 -
3.4 Servo Futaba S3003	- 37 -
3.5 Diseño del brazo SCARA	- 37 -
3.5.1 Características del brazo SCARA	- 37 -
3.6 Montaje y área de dibujo	- 37 -
CAPÍTULO 4. SOFTWARE	- 39 -
4.1 Cinemática inversa	- 39 -
4.2 Trayectorias a seguir	- 41 -
4.2.1 Trazado de una circunferencia	- 41 -
4.2.2 Trazado de un cuadrado	- 41 -
4.2.3 Trazado de imágenes	- 41 -
4.3 Matlab/Simulink	- 43 -
4.3.1 Máquina de estados	- 43 -
4.3.2 Control de hardware	- 44 -
4.3.3 Monitorización	- 46 -
CAPÍTULO 5. RESULTADOS	- 47 -
5.1 Ensayos realizados sobre servo LX-16A	- 47 -
5.2 Identificación del modelo de la planta	- 48 -
5.3 Implementación y pruebas	- 49 -
CAPÍTULO 6. CONCLUSIONES, APLICACIONES Y FUTUROS PROYECTOS.....	- 53 -
6.1 Conclusiones	- 53 -
6.2 Futuros proyectos	- 53 -
REFERENCIAS.....	- 55 -
PARTE II: PRESUPUESTO	- 57 -
CAPÍTULO 1. MEDICIONES	- 58 -
1.1 Hardware	- 58 -
1.2 Software	- 58 -
1.3 Mano de obra directa	- 58 -
CAPÍTULO 2. PRECIO UNITARIO.....	- 59 -
2.1 Hardware	- 59 -
2.2 Software	- 59 -
2.3 Mano de obra directa	- 59 -
CAPÍTULO 3. SUMAS PARCIALES	- 60 -
3.1 Hardware	- 60 -
3.2 Software	- 60 -
3.3 Mano de obra directa	- 60 -

CAPÍTULO 4. SUMA TOTAL.....	- 61 -
PARTE III: ANEXOS	- 63 -
ANEXO A: LISTA DE PLANOS	- 64 -
ANEXO B: CÓDIGO PARA LA VECTORIZACIÓN DE IMÁGENES	- 70 -
ANEXO C: CÓDIGO UTILIZADO PARA EL CONTROL DE LOS SERVOMOTORES	- 72 -

Índice de figuras

<i>Figura 1 Brazo SCARA de 4 grados de libertad</i>	- 27 -
<i>Figura 2 Esquema del modelo de brazo SCARA 2D</i>	- 28 -
<i>Figura 3 Diagrama de bloques de un control PID para un brazo SCARA</i>	- 30 -
<i>Figura 4 Diagrama de bloques del control predictivo de un brazo SCARA</i>	- 30 -
<i>Figura 5 Raspberry PI 3B +</i>	- 33 -
<i>Figura 6 Comunicación servo-microcontrolador</i>	- 34 -
<i>Figura 7 Servomotor LX-16A</i>	- 34 -
<i>Figura 8 Esquema Buslinker</i>	- 35 -
<i>Figura 9 Servomotor SCS-15</i>	- 36 -
<i>Figura 10 Montaje del brazo y el resto de componentes</i>	- 38 -
<i>Figura 11 Esquema del modelo del brazo SCARA 2D</i>	- 39 -
<i>Figura 12 Procesamiento de imagen</i>	- 42 -
<i>Figura 13 Imagen formada por puntos en coordenadas cartesianas</i>	- 42 -
<i>Figura 14 Máquina de estados implementada</i>	- 43 -
<i>Figura 15 Diagramas usado para el control</i>	- 43 -
<i>Figura 16 Diagrama de bloques implementado para el control de sensores</i>	- 44 -
<i>Figura 17 Constructor de buses para el control del servo LX16A</i>	- 44 -
<i>Figura 18 Control del servo EMAX</i>	- 45 -
<i>Figura 19 Panel de monitorización</i>	- 46 -
<i>Figura 20 Senoidal de referencia vs senoidal seguida por el servomotor</i>	- 47 -
<i>Figura 21 Valores leídos del servo: Ángulo, Temperatura y Tensión</i>	- 48 -
<i>Figura 22 Gráfica de la referencia de velocidad y la velocidad del servomotor</i>	- 48 -
<i>Figura 23 Trayectorias y coordenadas a seguir por el brazo</i>	- 49 -
<i>Figura 24 Posiciones angulares de una circunferencia enviadas al brazo</i>	- 50 -
<i>Figura 25 Circunferencias dibujadas</i>	- 50 -

Índice de tablas

<i>Tabla 1 Tareas y recursos utilizados.....</i>	<i>- 25 -</i>
<i>Tabla 2. Paquete de instrucción LX-16 A.....</i>	<i>- 35 -</i>
<i>Tabla 3 Paquete de instrucción SCS-15.....</i>	<i>- 36 -</i>
<i>Tabla 4 Paquete recibido SCS-15</i>	<i>- 36 -</i>

Índice de ecuaciones

<i>Ecuación 1 Representación de estado no lineal de un brazo SCARA</i>	<i>- 29 -</i>
<i>Ecuación 2. Cálculo del checksum</i>	<i>- 35 -</i>
<i>Ecuación 3 Distancia al origen de coordenadas.....</i>	<i>- 39 -</i>
<i>Ecuación 4 Cálculo de la posición angular de la segunda articulación.....</i>	<i>- 40 -</i>
<i>Ecuación 5 Cálculo de la posición angular de la primera articulación</i>	<i>- 40 -</i>
<i>Ecuación 6 Trayectoria de una circunferencia</i>	<i>- 41 -</i>

Capítulo 1. Introducción

Este proyecto trata del modelaje y control de un brazo SCARA, el cual se ha diseñado utilizando una herramienta de diseño 3D, y producido posteriormente con una impresora también 3D. El control se programa con un microcontrolador que se comunica con los distintos actuadores y sensores del sistema.

1.1 Motivación

La principal motivación de este proyecto es modelar y montar un brazo SCARA funcional, puesto que este tipo de brazo robóticos son de gran importancia en la industria. Además, existe una motivación didáctica puesto que se utilizará la Raspberry 3b+ y servos inteligente (LX-16 A o SCS15), y su estudio facilitará su uso en futuros proyectos y su implementación en la docencia.

1.2 Objetivos

El desarrollo de este proyecto comprende una serie de objetivos secundarios, sin los cuales no se podría alcanzar el objetivo principal del proyecto: el desarrollo completo de un brazo SCARA. Los objetivos secundarios propuestos son:

- Fabricar un brazo SCARA 2D y equiparlo con toda la electrónica requerida para su control
- Construir un modelo matemático del robot que permite simular de forma precisa su comportamiento
- Diseñar un sistema de control e implantarlo en el entorno Matlab/Simulink

1.3 Metodología

Para finalizar el proyecto con éxito han de completarse todos los objetivos, para lo cual previamente hay que ejecutar una serie de tareas menores:

En primer lugar, realizar una investigación previa del brazo SCARA. Tras obtener una idea mas clara de éste, se procedió a seleccionar las herramientas, y materiales que se utilizarían. Esto incluye una investigación intensiva del hardware (Servos inteligentes, Raspberry pi 3B+), donde se analiza sus conexiones, comunicaciones y su correcto funcionamiento. Por otro lado, se planteó la forma de la estructura del brazo y el tipo de material que se emplearía.

A continuación, se realizan una serie de ensayos para identificar la planta que se va a controlar. Una vez identificada, y al ver que era demasiado simple, su estudio no resulto de interés, por lo que se decidió realizar un control directo. Los ángulos que debían girar los servos, se les envía directamente, y son calculados a través de la cinemática inversa del brazo, lo cual permitiría colocar el extremo en la posición que se quisiera del espacio x-y.

Tras comprobar que los cálculos mencionados eran correctos y que el brazo alcanzaba la posición deseada, se procedió a crear trayectorias geométricas simples que el brazo debía seguir. Seguidamente, con el objetivo de seguir trayectorias mas complejas, se creo un programa que

permitiera la vectorización de imágenes, para posteriormente crear las trayectorias que el brazo debería seguir.

En la *Tabla 1* podemos ver todas las tareas que se han ido siguiendo , en su correspondiente orden:

TAREAS Y RECURSOS UTILIZADOS		
NUMERO	TAREA	RECURSO
1	Estudio previo del brazo SCARA	
1.1	Puesta en marcha Raspberry PI 3B+	
1.2	Estudio y prueba de las comunicaciones del servo	Matlab/Simulink
1.3	Diseño e impresión de la estructura del brazo SCARA	Solid Edge 2019
1.4	Ensayos sobre el brazo SCARA	
2.	Desarrollo de software	
2.1	Cálculo de la cinemática inversa	
2.2	Desarrollo de máquina de estados	Matlab/Simulink
2.3	Implementación de la maquina de estados, configuración de los parámetros iniciales	Matlab/Simulink
2.4	Implementación de las trayectorias a seguir	Matlab/Simulink
2.5	Desarrollo del código de vectorización de imágenes	Matlab/Simulink
3.	Estudio de resultados	
3.1	Mejora de parámetros del sistema	Matlab/Simulink
4.	Elaboración del estudio económico	Microsoft office
5.	Redacción de la memoria	Microsoft office

Tabla 1 Tareas y recursos utilizados

1.4 Recursos a emplear

Hardware utilizado: como microcontrolador la Raspberry 3B+ , en la cual se implantaron los códigos de control. Para el movimiento del brazo se utilizaron servomotores inteligentes (LX-16A o SCS15) y para el control de la posición del lápiz se uso un servo analógico (EMAX ES08A II) . La estructura del brazo se diseñará con Solid Edge y se construyó a partir de una impresora 3D. En el diseño del software se utilizó el entorno de programación de Matlab/Simulink. Durante todo el proyecto se consultaron varias referencias bibliográficas, y la ficha de datos proporcionado por el fabricante de cada componente.

1.5 Estructura de la memoria

En este documento se expone el desarrollo del proyecto; se articula en 5 capítulos. En el primero se introduce el brazo SCARA, se habla del modelo y de los controles existentes y utilizados actualmente, y por último, se comenta la motivación de este proyecto y la metodología utilizada en el mismo. En el segundo capítulo, se explica el hardware utilizado, la estructura del brazo SCARA, el funcionamiento y las características de los servos inteligentes. En el tercer capítulo, se habla sobre el software implementado, lo que incluye la máquina de estados, el procesamiento de imágenes, la cinemática inversa implantada, y las funciones utilizadas para su funcionamiento. El cuarto capítulo se resume los ensayos realizados durante el proyecto, y se consignan los resultados. En el capítulo final, se habla sobre futuros proyectos en los que podría utilizarse todos los componentes estudiados.

Capítulo 2. Estado del arte

2.1 Historia

La aparición de los procesos automatizados, que hoy día constituyen la base fundamental de los procedimientos industriales, supuso un avance considerable en el desarrollo, pues gracias a ellos y sus múltiples ventajas: reducción de costes y tiempo de producción, entre otros se logra obtener una clara trazabilidad, estandarizar parámetros de negocio, y además facilitan la identificación de puntos de ineficiencia [1].

Existen muchos tipos de robots que permiten esta automatización de procesos. Uno de los más importantes son los brazos robóticos, que son, como su propio nombre indica, un brazo mecánico programable que permite movimientos muy parecidos a los de un brazo humano. Existen muchos tipos de brazos robóticos: entre otros, el robot cartesiano, robot cilíndrico, robot esférico, robot articulado y brazo SCARA, siendo este último el objetivo de este proyecto.

El primer brazo SCARA (Selective Compliant Articulated Robot Arm) fue creado en Japón, el año 1981 por un equipo de científicos liderado por Hiroshi Makino. Desde su aparición se han ido convirtiendo en objetos populares en la industria ya que pueden realizar multitud de funciones: soldar, atornillar, distribuir, etc. Sin embargo, su característica más importante es que permiten imitar el brazo humano, lo que les habilita a extenderse y retraerse en áreas confinadas, son muy apropiados también para realizar acciones de pick-and-place en un reducido tiempo [2]. Debido principalmente a su rapidez, efectividad y robustez resultan herramientas muy útiles.

Un robot SCARA consiste en un robot con 4 grados de libertad que se puede mover en cualquier coordenada X-Y-Z, dentro de su espacio de trabajo [3]. Como se puede ver en la Figura 2, tiene 4 articulaciones de las cuales 3 de ellas son de rotación (hombro, codo y muñeca) y una prismática [4].

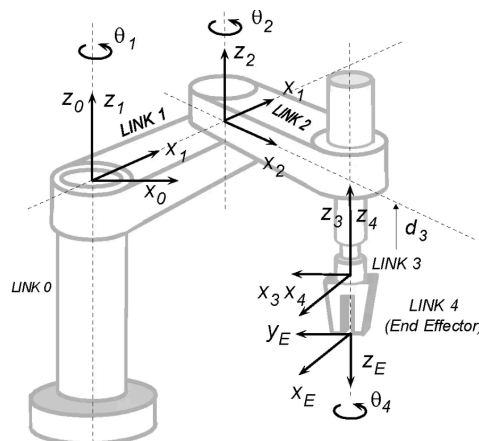


Figura 2 Brazo SCARA de 4 grados de libertad

2.2 Modelo

El modelo de un brazo SCARA de dos grados de libertad se puede ver en la Figura 3. Para la representación de estado, se utiliza como entradas los pares τ_1 y τ_2 , como variables de estado las velocidades angulares w_1 y w_2 , y las posiciones angulares θ_1 y θ_2 , y finalmente como salidas las coordenadas cartesianas x e y .

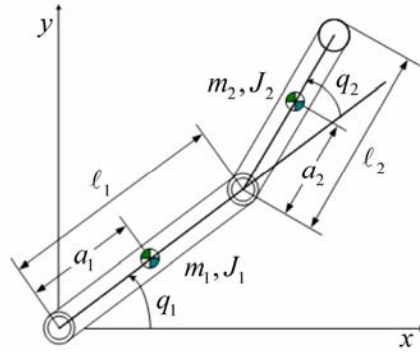


Figura 3 Esquema del modelo de brazo SCARA 2D

Además, para cada brazo SCARA se necesita una serie de parámetros, que serán específicos para cada modelo [5]:

- Longitud eslabón 1: L_1
- Longitud eslabón 2: L_2
- Distancia desde la articulación al CDG del eslabón 1: h_1
- Distancia desde la articulación al CDG del eslabón 2: h_2
- Masa eslabón 1: m_1
- Masa eslabón 2: m_2
- Momento de inercia respecto al CDG del eslabón 1: I_{g1}
- Momento de inercia respecto al CDG del eslabón 2: I_{g2}
- Momento de inercia respecto a la articulación del eslabón 1: I_1
- Momento de inercia respecto a la articulación del eslabón 2: I_2
- Coeficiente de fricción viscosa de la articulación 1: D_1
- Coeficiente de fricción viscosa de la articulación 2: D_2
- Relación de transformación de la transmisión del eslabón 1: n_1
- Relación de transformación de la transmisión del eslabón 2: n_2
- Resistencia eléctrica de los motores: R_m
- Momento de inercia de los motores: I_m
- Constante eléctrica del motor: K_e
- Constante mecánica del motor: K_t

Una vez definido los parámetros, entradas, salidas y variables de estado, se ha realizado los cálculos necesarios aplicando el método de Euler-Lagrange [6]. Finalmente, se obtiene la representación de estado lineal del robot:

$$\begin{bmatrix} \frac{d\omega_1}{dt} \\ \frac{d\omega_2}{dt} \\ \frac{d\theta_1}{dt} \\ \frac{d\theta_2}{dt} \end{bmatrix} = \begin{bmatrix} M^{-1}(\theta_1, \theta_2)N(\tau_1, \tau_2, \omega_1, \omega_2, \theta_1, \theta_2) \\ \omega_1 \\ \omega_2 \end{bmatrix}$$

Ecuación 1 Representación de estado no lineal de un brazo SCARA

Para obtener el desarrollo completo, y el procedimiento utilizado para obtener la representación de estado se puede consultar las fuente anteriormente mencionadas.

2.3 Controles utilizados

En este apartado se enumeran los controles mas utilizados en la actualidad para controlar el brazo SCARA.

2.3.1 Control PID

El control PID es el mas utilizado en la industria debido a su facilidad de diseño e implementación. También puede utilizarse para un brazo SCARA como se ve en la literatura [7]. Los resultados de este artículo indican que el principal problema de este control para ser aplicado en el brazo SCARA, es la planta, ya que se trata de un sistema no lineal y multivariable con acoplamientos significativos entre los dos grados de libertad. El PID esta orientado al control de sistema lineales y SISO (una entrada y una salida). Se ha visto en el artículo [8] que es posible realizar este control, pero cada articulación debe ser controlada con un PID individual. Esto lo podemos ver en la Figura 4:

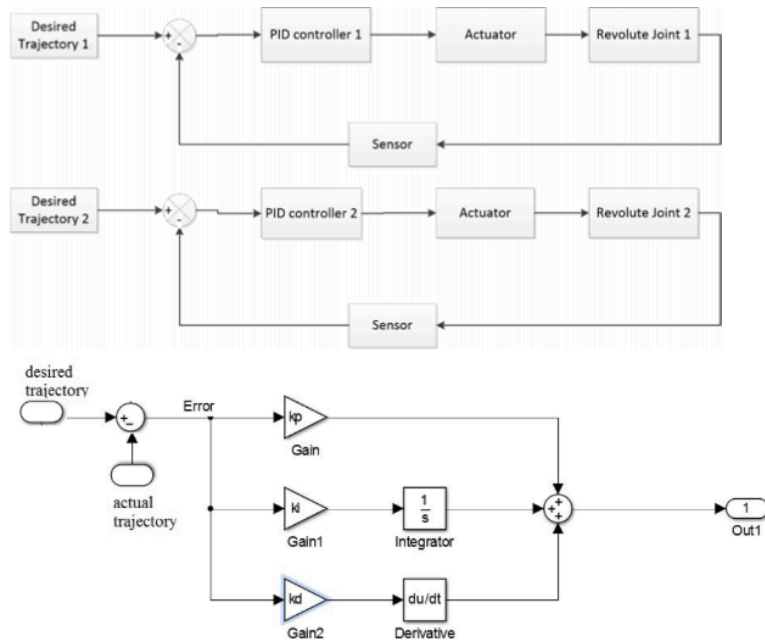


Figura 4 Diagrama de bloques de un control PID para un brazo SCARA

2.3.2 Control por par calculado (CPC)

El control por par calculado es el segundo tipo de control mas utilizado. En [7] se ve que los resultados son superiores a los del PID puesto que tiene en cuenta la planta. El problema que supone este tipo de control es se necesita conocer con exactitud los parámetros que caracterizan a la planta y estos pueden ser difícil de obtener con mucha precisión.

2.3.3 Control predictivo

Este tipo de control se caracteriza por tener en cuenta la planta además de intentar predecir el comportamiento futuro de está. Como se puede ver en la literatura [9] el control tiene unos resultados superiores al PID y al CPC en la mayoría de los casos. A continuación, se observa el esquema utilizado en el literatura anteriormente mencionada:

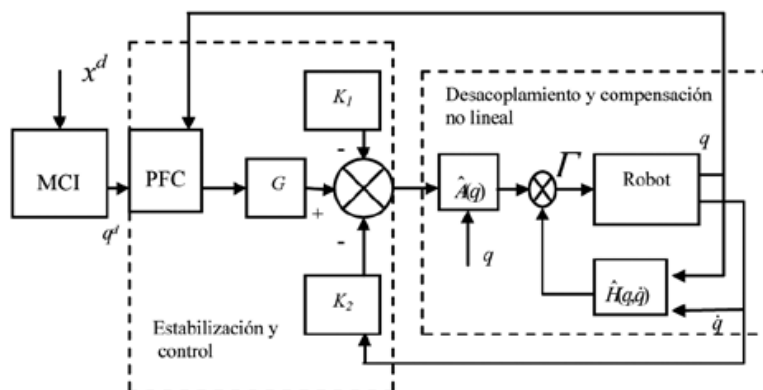


Figura 5 Diagrama de bloques del control predictivo de un brazo SCARA

2.3.4 Control de redes neuronales

Este control se ha utilizado para desarrollar el control de brazo SCARA como se puede ver en el artículo [10]. El control de redes neuronales funciona como un aproximador de funciones que se optimiza para obtener el mejor control. Este control es de los más nuevos, actualmente.

2.3.5 Aplicaciones

El robot SCARA es muy utilizado en aplicaciones industriales, especialmente aquellas que se realizan sobre un plano. Algunas de las aplicaciones más comunes son: dispensar, soldar, guiado, montaje de componentes electrónicos, y atornillar. En algunas de estas aplicaciones es necesario el guiado por visión, y por ello algunos brazos incluyen un software procesamiento de imágenes, que se ejecuta en un ordenador. Esto mejorará mucho la eficiencia de estos brazos [11].

Capítulo 3. Hardware

En este capítulo se trata el hardware utilizado, desde el microcontrolador hasta el diseño de la estructura del brazo.

3.1 Raspberry PI 3B+

La Raspberry PI 3B+ es un ordenador construido en una placa, la cual incluye el microprocesador, la memoria, inputs/outputs y otras funcionalidades [12]. Debido a sus buenas características, compatibilidad con Matlab/Simulink y su utilización en otros proyectos, se ha elegido esta placa como microcontrolador.



Figura 6 Raspberry PI 3B +

Se debe instalar el sistema que utiliza el microcontrolador en una tarjeta SD a través de Matlab. Además, se tiene que configurar una serie de parámetros para su correcto funcionamiento. Cabe destacar, que la Raspberry Pi 3B+ tiene la UART principal y la mini UART. La UART está conectada al bluetooth, por tanto para comunicaciones en serie, se utilizará la mini UART. Pero como por defecto los bloques de simulink utilizan la UART (/dev/ttyAMA0), se debe sustituir esta por la mini UART (/dev/ttyS0) [13].

Las Raspberry pi se ha utilizado para incorporar todo el hardware, y para ejecutar el software diseñado

3.2 Servomotores inteligentes

Para el desarrollo del proyecto es necesaria la utilización de servomotores que permitan la movilidad del brazo. Se ha optado por utilizar servos inteligentes. Estos permiten una conexión en daisy chain, de gran utilidad puesto que se reduce el cableado necesario para el montaje del brazo. Además, permiten obtener información sobre la posición, velocidad, temperatura y voltaje del servomotor. Se ha tenido acceso a dos tipos de servomotores inteligentes, y de ambos se ha estudiado su funcionamiento y protocolo de comunicación.

3.2.1 Comunicación

En ambos servos, el microcontrolador se comunica con el servomotor a través de la UART (Universal Asynchronous Receiver-Transmitter). Por un lado, el microcontrolador envía la información siguiendo un protocolo de comunicación full duplex, es decir, por un canal se envía información y por otro se recibe. Por otro lado, los servos se comunican entre si utilizando un protocolo de comunicación half dúplex, estos permite transmitir información en ambas direcciones, pero no a la vez [14]. Puesto que los protocolos de comunicaciones son diferentes para ambos servos, se utilizó un circuito lógico por el cual, la información que viene del microcontrolador, siguiendo el protocolo full duplex , se convierte a un protocolo half dúplex. En la Figura 7 podemos ver como quedaría la conexión final.

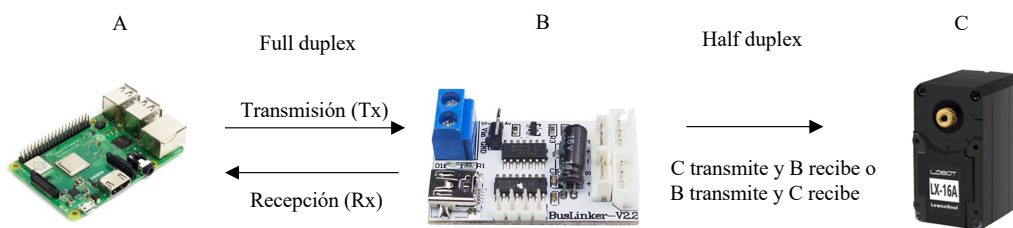


Figura 7 Comunicación servo-microcontrolador

3.2.2 Servo LX-16 A

El servomotor LX-16 A tiene un peso de 52g, una precisión de 0.24°, y permite ser alimentado entre [6-7.4]V, lo cual afecta al par proporcionado por el servo entre [17-19.5]Kg. En modo servo puede rotar de 0-240°.



Figura 8 Servomotor LX-16A

En el protocolo de comunicación del servo [15] se puede ver la estructura de los mensajes que se han de enviar. Estos tienen una forma definida, que se puede ver en la *Tabla 2*.

En primer lugar se encuentra el header este sirve para indicar al servo, que llega un mensaje nuevo, a continuación se incluye la identidad del servo (0-253), la longitud del mensaje, el tipo de instrucción a ejecutar (en el protocolo de comunicación se pueden ver todos los tipos), los parámetros correspondientes a la instrucción enviada, y finalmente se envía el checksum. El checksum se calcula

con la Ecuación 2, si la suma de los valores entre paréntesis es menor o igual a 255. En caso contrario, se debe coger los 8 bits mas bajos.

Header	Id	Length	Instrucción	Parameter	Checksum
0x55 0x55	ID(0-253)	Lenght	Cmd	Prm1...PrmN	Checksum

Tabla 2. Paquete de instrucción LX-16 A

$$checksum = \sim(id + length + cmd + prm1 + \dots + prmN)$$

Ecuación 2. Cálculo del checksum

Como se ha comentado, la transmisión y recepción del paquete se realiza a través de la mini UART, con un baudrate de 115200 bits/s (este parámetro se puede configurar si se necesita). Esta comunicación empleará un protocolo duplex, es decir, por un cable se transmitirá y por otro se recibirá. Estas señales irán conectadas al buslinker, que convertirá las señales en halfduplex, para luego, enviar esta información directamente a los servos. En la Figura 9 podemos ver el esquema del buslinker. Este componente lo proporciona el fabricante, su diseño no será necesario.

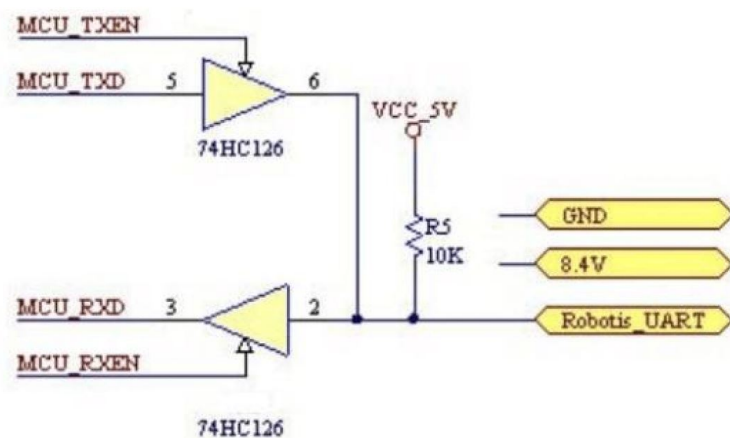


Figura 9 Esquema Buslinker

3.2.3 Servo SCS15

El servo SCS15 tiene unas características muy similares, un peso de , una precisión de 0.19°, y permite ser alimentado entre [6-9]V, con un par de 15 kg.cm. En modo servo puede rotar de 0-200°.



Figura 10 Servomotor SCS-15

El principal problema que se encontró con este servo fue la falta de información sobre su protocolo de comunicación, puesto que el fabricante no la proporcionaba. Para poder identificar los paquetes que se debían enviar, se descargó de github [16] una serie de librerías de arduino, en la cual se utilizaba este servo. Tras el estudio del código se pudo obtener una estructura del paquete a enviar. En la Tabla 3 podemos ver la estructura del paquete a enviar. Y en la Tabla 4 podemos ver la información que se recibe, se ve que el bus recibido informa si se ha producido un error.

Header	Id	Length	Instrucción	Parameter	Checksum
0xFF 0xFF	ID(0-253)	Length	Instr	Prm1...PrmN	Checksum

Tabla 3 Paquete de instrucción SCS-15

Header	Id	Length	Error	Instrucción	Parameter	Checksum
0xFF 0xFF	ID(0-253)	Length	Error	Instr	Prm1...PrmN	Checksum

Tabla 4 Paquete recibido SCS-15

Debido a la falta de un componente, este servo no se llegó a probar. El siguiente paso en su estudio consistiría en descargar el programa del fabricante, y configurar con este algunas de sus propiedades, en especial el baudrate, puesto que de fabrica esta en 1000000 bits/s, y la Raspberry Pi es incapaz de trabajar con este, por tanto sería necesario disminuirlo. Una vez configurado, convendría comprobar las distintas instrucciones que se pueden mandar [16].

3.3 Servo EMAX ES08AII Analog servo

Este servo motor va montado en una estructura de dibujo, la cual servirá para controlar la posición del lápiz. Su control es bastante sencillo puesto que solo se utilizan 2 posiciones, arriba (180°) o abajo (0°). Para su control se utiliza el bloque proporcionado por simulink para controlar este tipo de servos. Solo será necesario introducir el ángulo que se requiere.

3.4 Servo Futaba S3003

Este servo puede girar de (0-180°) a una velocidad de 260°/s. Se alimenta con una tensión entre 4.8V-6V, y se controla con un PWM de una frecuencia de 50Hz.

3.5 Diseño del brazo SCARA

En el siguiente apartado se describe el diseño de la estructura del brazo, además de los ensayos necesarios para obtener la planta a controlar.

3.5.1 Características del brazo SCARA

La estructura del brazo SCARA se ha diseñado con Solid Edge, y se imprimió en 3D. La estructura del brazo SCARA se ha diseñado con Solid Edge, y se ha impreso 3D. El brazo está formado por cinco piezas diferentes; se realizó de esta manera para que, en la eventualidad de que alguna de las piezas se rompiera, no fuera preciso imprimir todo el brazo, sino únicamente la pieza rota. Y puesto que se utilizan distintos tipos de servo, para cambiarlos solo será necesario sustituir una pieza.

En el Anexo A: Lista de planos se ven los planos del brazo. En primer lugar se observa la viga, esta pieza se podría modificar si se necesitase mayor o menor longitud. A continuación, hay tres piezas que sirven de conector: el conector servo-brazo, el conector brazo-giróscopo (esto se ha incluido para facilitar su utilización en futuros proyectos), y la conexión brazo-lápiz. Y finalmente, esta pieza Soporte servo LX16A, la cual sirve para conectar este servomotor al brazo. Para el servomotor SCS15 se utilizó una pieza que ya viene de fábrica.

La sujeción del servo SCS15 no se ha diseñado puesto que esta venía proporcionada por el fabricante. Para imprimir de forma 3D se han convertido los documentos .xsl a .stl con una tolerancia de 0.01mm y 3°. Se ha utilizado la impresora 3D Dimensión, de Stratasys del laboratorio de fabricación, y se usó plástico como material, y para eliminar el material de aporte se han bañado.

3.6 Montaje y área de dibujo

Para el montaje del brazo SCARA, se utilizó una tabla de 52cm×64 cm, además de todas las piezas impresas y los servo motores anteriormente descritos. En la se puede ver un primer diseño. Tras el montaje se estudió la zona de trabajo del espacio x-y, en la cual podría trabajar el brazo SCARA. Hay ciertas zonas que no son accesibles ya que podría dañar el brazo o simplemente era imposible su acceso.

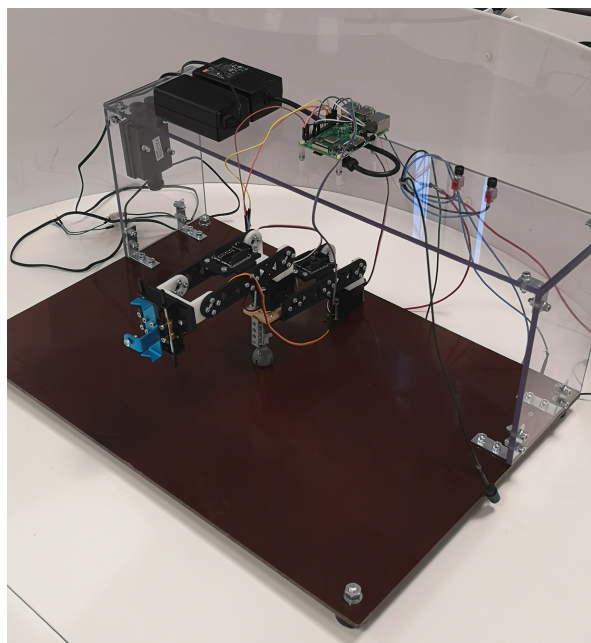
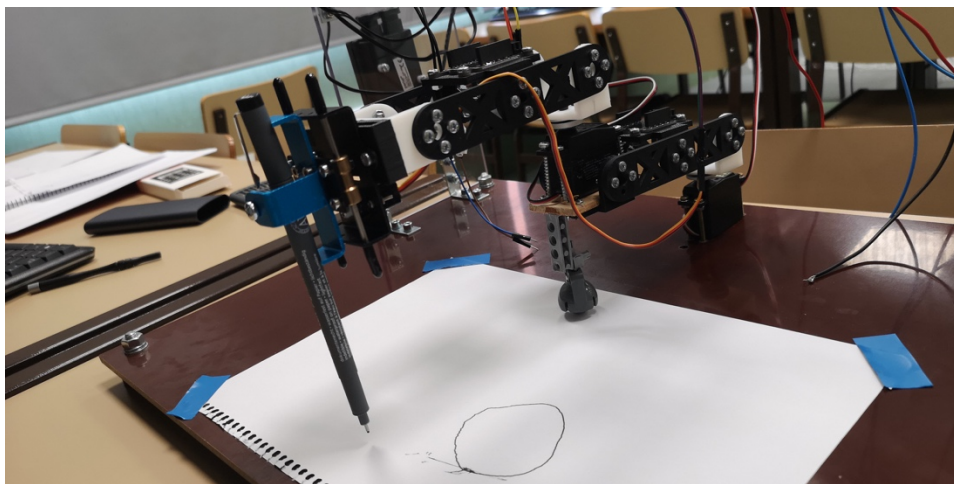


Figura 11 Montaje del brazo y el resto de componentes

Capítulo 4. Software

En este capítulo se expone el software implementado: la cinemática inversa, maquina de estados, el programa de vectorización de imagen , y finalmente se explica los bloques de simulink implementados.

4.1 Cinemática inversa

A la hora de controlar la trayectoria del extremo del robot en ejes cartesianos, es necesario transformar dichas coordenadas en las posiciones angulares equivalentes de ambas articulaciones. Estas transformaciones se realizan mediante la cinemática inversa del robot. En la Figura 12 se puede ver el brazo con longitud l_1 y antebrazo con longitud l_2 del robot , a partir de los puntos (x_0, y_0) se debe obtener el ángulo de θ_1 y θ_2 , y para ello se calcularan los ángulos auxiliares α, β .

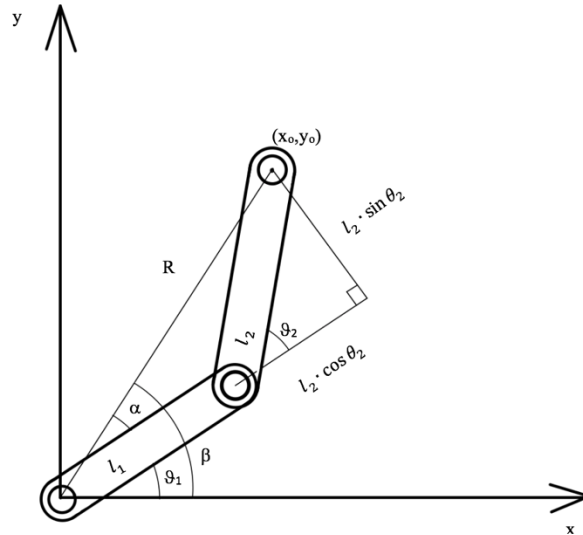


Figura 12 Esquema del modelo del brazo SCARA 2D

En primer lugar se obtiene la distancia desde el origen de coordenadas al punto objetivo (x_0, y_0) , esto se puede ver en la Figura 12.

$$R = \sqrt{x^2 + y^2}$$

Ecuación 3 Distancia al origen de coordenadas

Aplicando ahora el teorema del coseno , al triángulo formado por L_1, L_2 y R , se obtiene obtener el ángulo relativo que ha de girar L_2 .

$$x^2 + y^2 = l_1^2 + l_2^2 - 2 \cdot l_1 \cdot l_2 \cdot \cos(180 - \theta_2)$$

$$\theta_2 = \cos^{-1} \left(\frac{x^2 + y^2 - l_1^2 - l_2^2}{2 \cdot l_1 \cdot l_2} \right)$$

Ecuación 4 Cálculo de la posición angular de la segunda articulación

Una vez obtenido θ_2 , con simple trigonometría se obtiene α, β y finalmente θ_1

$$\alpha = \tan^{-1} \left(\frac{l_2 \cdot \sin \theta_2}{l_1 + l_2 \cdot \cos \theta_2} \right)$$

$$\beta = \tan^{-1} \left(\frac{y}{x} \right)$$

$$\theta_1 = \beta - \alpha$$

Ecuación 5 Cálculo de la posición angular de la primera articulación

Finalmente, con estas ecuaciones se lleva el extremo del brazo a la posición (x,y)se desee. Se introducirán estas ecuaciones en la siguiente secuencia en el código para determinar las posiciones angulares del brazo SCARA.

1. $R = \sqrt{x^2 + y^2}$
2. $\theta_2 = \cos^{-1} \left(\frac{x^2 + y^2 - l_1^2 - l_2^2}{2 \cdot l_1 \cdot l_2} \right)$
3. $\alpha = \tan^{-1} \left(\frac{l_2 \cdot \sin \theta_2}{l_1 + l_2 \cdot \cos \theta_2} \right)$
4. $\beta = \tan^{-1} \left(\frac{y}{x} \right)$
5. $\theta_1 = \beta - \alpha$

4.2 Trayectorias a seguir

En este apartado se hablará de las distintas trayectorias a seguir. Para cada trayectoria se obtendrá una serie de puntos, y a través de la cinemática inversa se obtienen las posiciones angulares equivalentes.

4.2.1 Trazado de una circunferencia

Para el trazado de la circunferencia, se han obtenido las ecuaciones que debe ir siguiendo el brazo SCARA. Estas ecuaciones, que se pueden ver a continuación, van dando coordenadas cartesianas, en función del tiempo. Estos puntos, pertenecen a la circunferencia que se quiere trazar. Donde (x_0, y_0) es el centro de la circunferencia, R el radio, y w la velocidad que tarda en trazar la circunferencia.

$$\begin{aligned}x &= x_0 + R \cdot \cos(w \cdot t) \\ y &= y_0 + R \cdot \sin(w \cdot t)\end{aligned}$$

Ecuación 6 Trayectoria de una circunferencia

4.2.2 Trazado de un cuadrado

El trazado de un cuadrado ha sido mas sencillo puesto que simplemente se ha tenido que dar 9 puntos del espacio, asegurándose que estos estén en el área de trabajo anteriormente descrito. El brazo irá de un punto a otro cada cierto tiempo, que se puede determinar en el código.

4.2.3 Trazado de imágenes

Se ha creado el código por el cual partiendo de una imagen se obtiene una serie de punto en coordenadas cartesianas, los cuales debe ir siguiendo el brazo. Para este desarrollo se ha seguido los pasos leídos en el artículo [17] y la función `pic2point` [18].

En la Figura 13 se ve el proceso que sigue la imagen antes de la vectorización de esta. En este primer proceso el objetivo será pasar de una imagen a color a una figura formada sólo por líneas negras y un fondo blanco.

En primer lugar se debe leer la imagen y guardarla como una matriz de píxeles. Tras esto se pasa imagen a color a una imagen en escala de grises, utilizando el comando `rgb2grey()`. A continuación, se pasa a una escala de blancos y negros, gracias al comando `im2bw()`, y finalmente con la función `edge()` y un bucle `for`, se obtiene los bordes de la imagen original.



Figura 13 Procesamiento de imagen

Finalmente, para el proceso de vectorización de la imagen se ha utilizado la función `pic2point()` [18], a la cual se le proporciona una imagen, la función detecta todos aquellos pixeles que están en blanco y negro, y los guarda como puntos en coordenadas cartesianas. Tras esto se realiza un cambio de escala, con el que se obtiene todos los puntos cartesianos que debe seguir el brazo para el área de trabajo. En la Figura 14 se puede ver la vectorización de la imagen procesada anteriormente, se observa como se trata muchos puntos en el espacio.

Tras obtener todos los puntos de la figura en ejes cartesianos, se procede a ordenar éstos para que el brazo pueda seguir una trayectoria. Se toma un punto inicial, y se calcula la distancia euclídea de este punto respecto a todo los demás, después se ordena todas estas distancias menor a mayor, seleccionado el punto mas cercano. Este punto se almacena en una matriz, y se repite el proceso anteriormente descrito, tomando ahora el punto almacenado como punto inicial. Finalmente, con este proceso se obtiene una trayectoria de puntos ordenada que debe seguir el brazo.

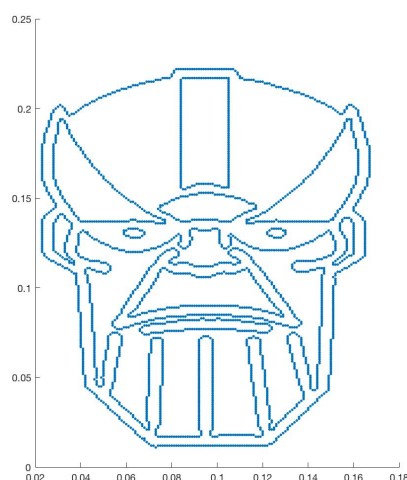


Figura 14 Imagen formada por puntos en coordenadas cartesianas

4.3 Matlab/Simulink

4.3.1 Máquina de estados

Para indicar al brazo SCARA que debe hacer en cada momento , hemos creado un maquina de estados sencilla, se puede ver en la . Como entradas hemos utilizado 2 pulsadores, uno ellos indicará si el brazo debe estar en su posición inicial o dibujando , y con el otro pulsador somos capaces de elegir que trayectoria queremos seguir, ya sea una circunferencia, un cuadrado o una imagen vectorizada. Los estados son posición inicial o dibujando, y en cada estado determinaremos la posición del lápiz (arriba o abajo).

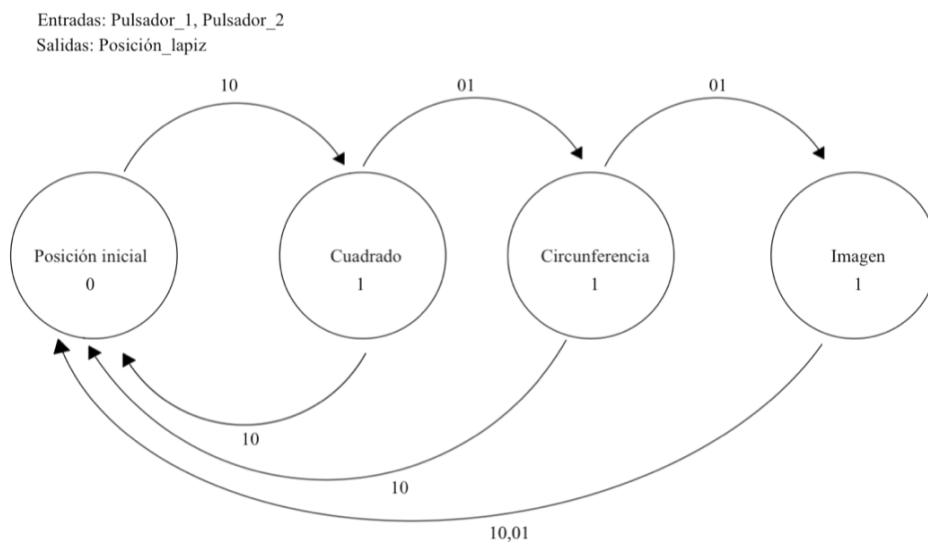


Figura 15 Máquina de estados implementada

En la Figura 16 se puede ver dos funciones con las que controlaremos la maquina de estado. En la función *State machine* tenemos incorporada la máquina de estados anteriormente descrita, y en esta se determinara las salidas. En la función *control update* se irán ejecutando las características de cada estado. En este bloque implementamos la cinemática inversa.

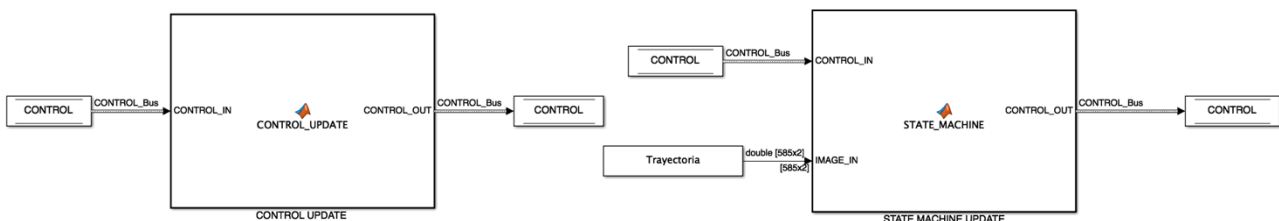


Figura 16 Diagramas usado para el control

4.3.2 Control de hardware

En este apartado se hablará de las señales que debe recibir los sensores, y los mensajes que se deben enviar a los distintos actuadores.

4.3.2.1 Sensores

Como sensores se utilizan 2 pulsadores. Estos determinaran el estado, y al trayectoria a seguir. En la Figura 17 podemos ver como se ha implementado estos en simulink.

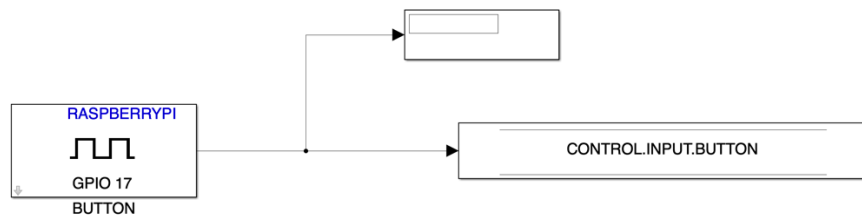


Figura 17 Diagrama de bloques implementado para el control de sensores

4.3.2.2 Actuadores

Como actuadores se emplean servos inteligentes para controlar la posición del extremo del brazo , unos servos analógico para controlar la posición del lápiz, y debido a los problemas que han aparecido también el extremo del brazo.

Para el control de los servos inteligentes se utiliza la función *Instrucción Escritura Posicion*. Esta construye los buses que se enviaran. A la función se le debe proporcionar el ángulo, el tiempo de recorrido, el comando, la longitud del bus y la cabecera de esta, y finalmente la función crea un bus con la estructura vista anteriormente en la

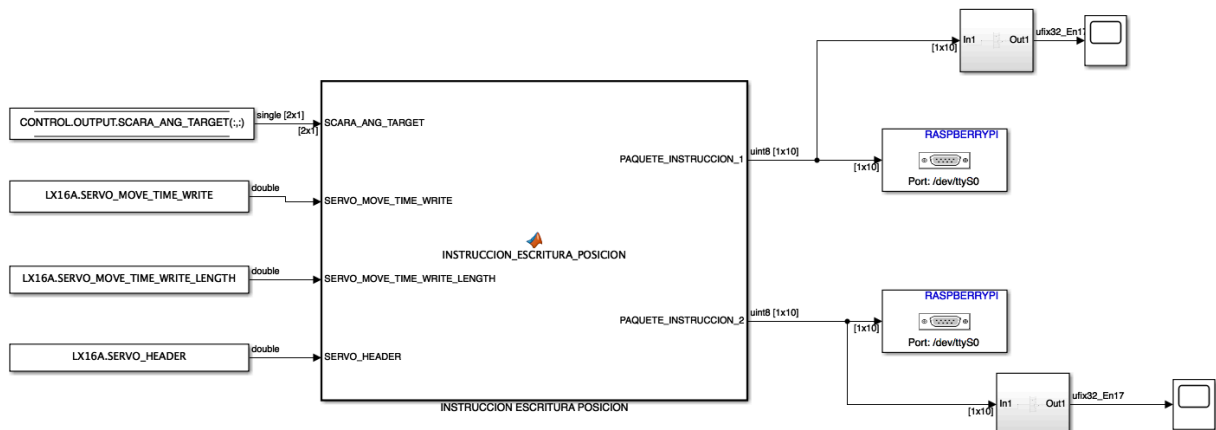


Figura 18 Constructor de buses para el control del servo LX16A

También se ha creado una función capaz de leer la posición en la que se encuentra el servo, esto será mas útil para futuros proyectos.

El control de los servos analógicos esta bastante sencillo, se utiliza el bloque *Standard Servo Write* proporcionado por Matlab. En la Figura 19 se puede ver el diagrama de bloque utilizado para controlar el servo analógico de la estructura de dibujo. En este caso solo se enviaran 2 posiciones 0° o 180°. El diagrama de bloques para el control de los otros servos será igual. Pero en este caso se enviaran las posiciones angulares que se vayan obteniendo a través de la cinemática inversa.

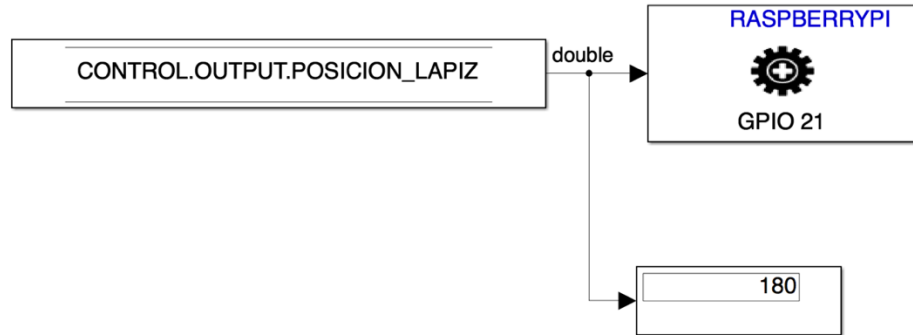


Figura 19 Control del servo EMAX

4.3.3 Monitorización

La monitorización como se ve en la Figura 20 se utilizará principalmente para depurar fallos, y obtener los resultados finales. En ella hacemos seguimiento de todas las variables que nos interesen, desde los ángulos que introducimos en los servos como los estados en los que se encuentra el sistema.

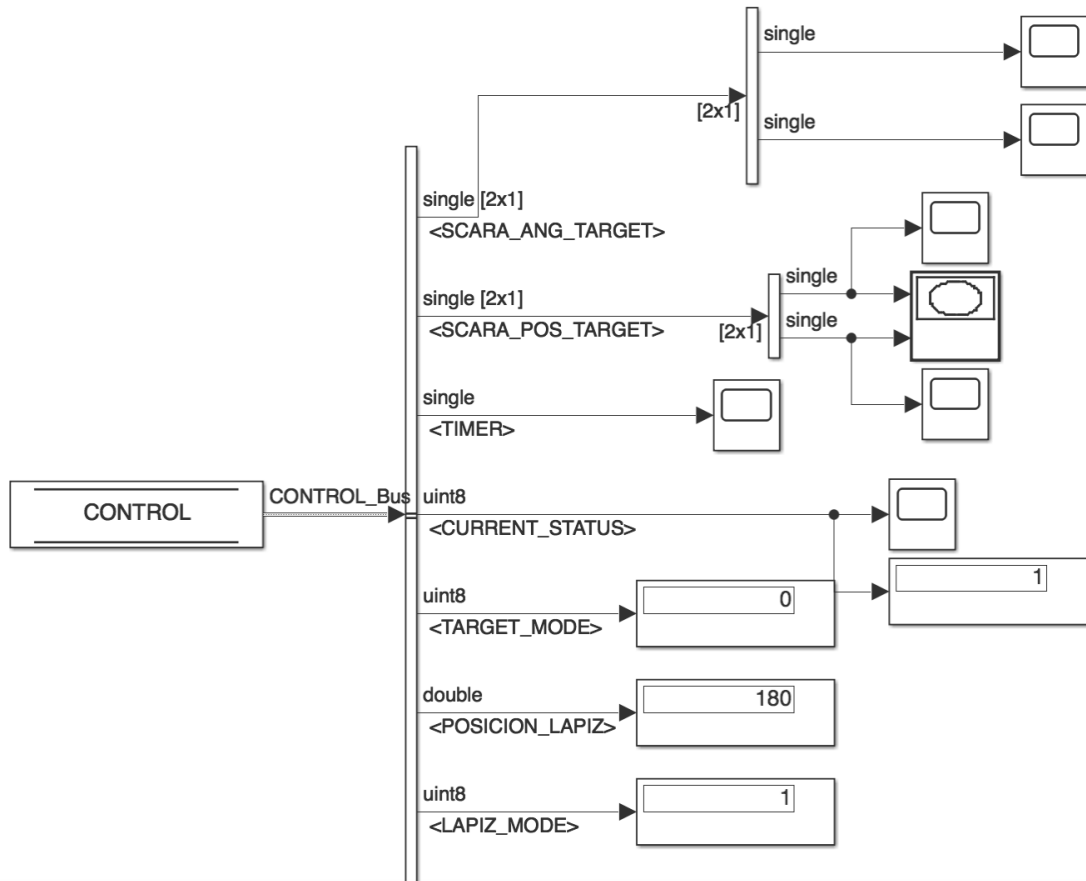


Figura 20 Panel de monitorización

Capítulo 5. Resultados

En este capítulo, se hablará de todas las pruebas que se han realizado durante el proyecto, y los resultados que se han obtenido en este.

5.1 Ensayos realizados sobre servo LX-16A

Se realizó una serie de pruebas sobre el servo para comprobar su correcto funcionamiento, y para familiarizarse con su protocolo de comunicación. El código de las funciones utilizadas se puede ver en el Anexo C: Código utilizado para el control de los servomotores.

En primer lugar, se envió paquetes sencillos con instrucciones simples. Tras esto se realizó en simulink un programa con el cual el servo sigue los cambios de ángulo de una senoidal, en la Figura 21 se puede ver por un lado la senoidal de referencia, y por otro lado todos los ángulos que son enviados al servo. Se ve que esta también forma una senoidal, pero de forma escalonada con un tiempo de muestro de 0.1 s.

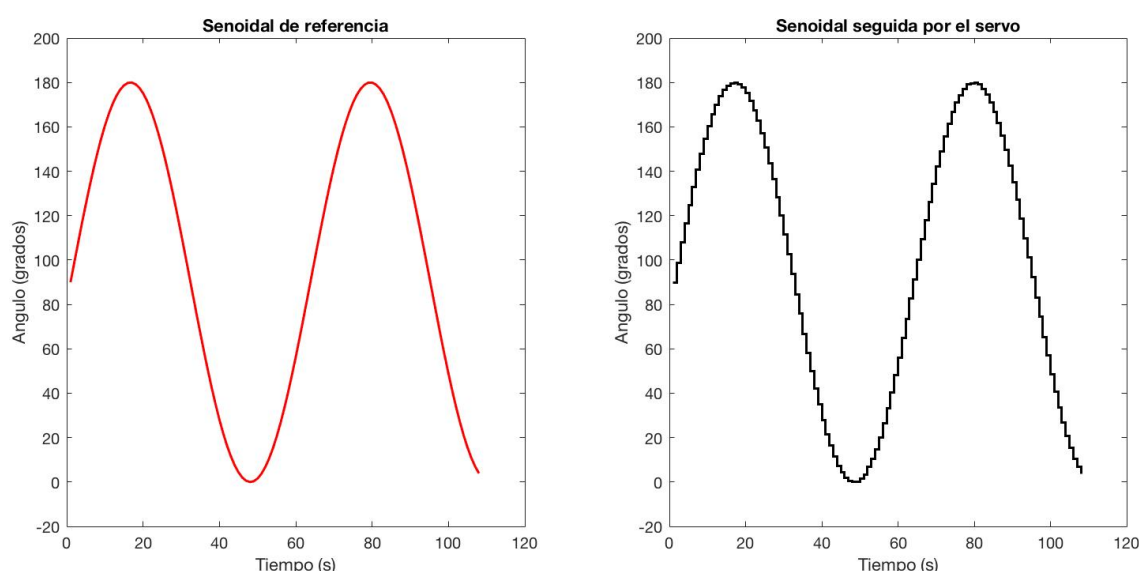


Figura 21 Senoidal de referencia vs senoidal seguida por el servomotor

Una vez comprobado el funcionamiento en modo servomotor, se procedió a realizar unas pruebas en modo motor. Para ello se utilizó como referencia una señal cuadrada de 200 a -200. Se comprobó que para valores positivos el servo gira en sentido horario, y para valores negativos en sentido antihorario.

Tras esto, se decidió crear un diagrama de bloques en simulink, con el cual se puede leer una serie de datos del servomotor: posición, temperatura y voltaje. En la Figura 22 se puede ver los distintos datos obtenidos.

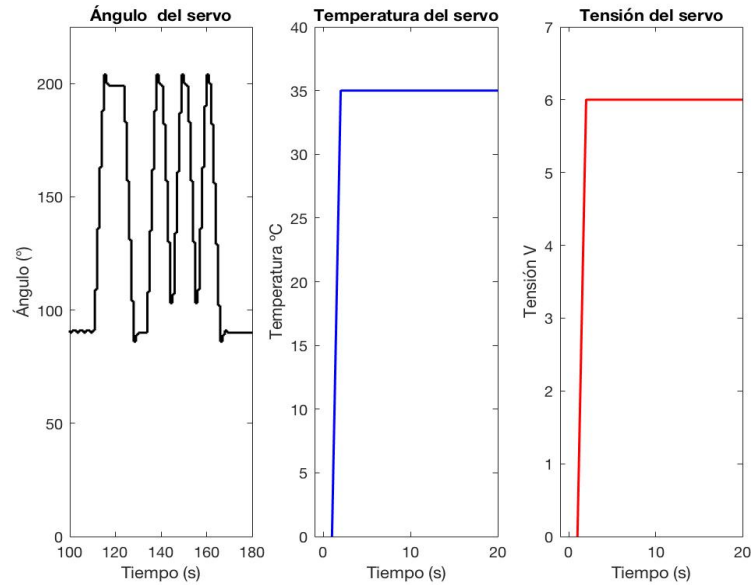


Figura 22 Valores leídos del servo: Ángulo, Temperatura y Tensión

5.2 Identificación del modelo de la planta

En este apartado se intentó identificar la planta del brazo SCARA, para ello se realizó un ensayo mediante simulink. Se introdujeron escalones de referencia en velocidad, y se observó la velocidad real del servo. Los cambios en la velocidad eran instantáneos con un pequeño retraso, debido a la resolución y la dinámica del servomotor. Por ello, la planta del modelo resultaría simplemente en un retraso y un integrador. Además, como los servomotores trabajan con un tiempo de muestro relativamente alto, no se podía apreciar el cambio en la velocidad del servo. Por tanto se optó por utilizar un control directo. En la Figura 23 se puede ver claramente que la velocidad del servo (en rojo), tiene un pequeño retraso respecto a la velocidad de referencia.

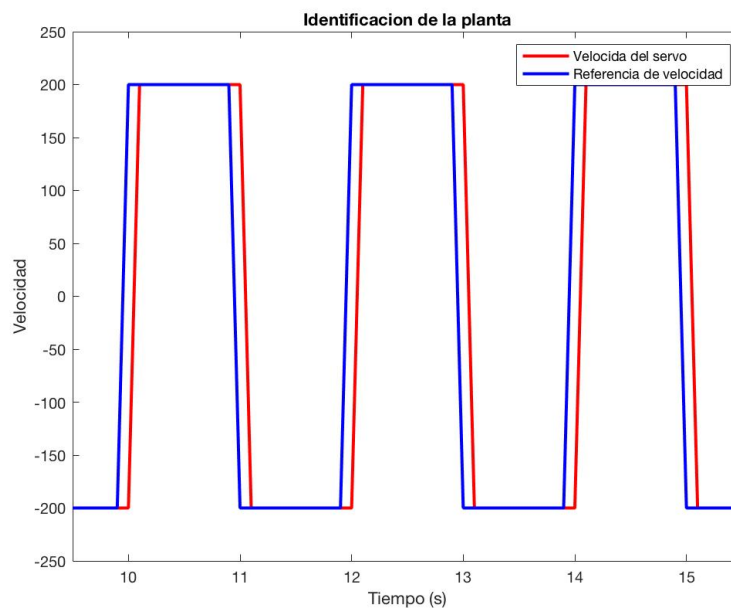


Figura 23 Gráfica de la referencia de velocidad y la velocidad del servomotor

5.3 Implementación y pruebas

Una vez realizada la maquina de estados, y comprobado su correcto funcionamiento se puede empezar a tomar resultados. Para ello se prueban los distintos trazados implementados: circunferencia, cuadrado e imagen.

Para cada trazado se han ido tomando distintas medidas. En primer lugar se comprueba que los puntos elegidos forman el trazado requerido, tras ello se compararan las medidas angulares obtenidas a través de la cinemática inversa con los ángulos leídos del servo, luego a partir de estos ángulos se obtiene de nuevo la figura enviada , y se comprueban sus diferencias. Y finalmente, se mira el trazado que ha realizado el brazo SCARA sobre el papel.

En primer lugar, se comprueba que las coordenadas cartesianas enviadas forman los trazados que se quieren. A continuación, se puede ver de izquierda a derecha, las coordenadas cartesianas de un cuadrado, circunferencia, y la imagen seleccionada. Se observa que todos estos puntos forman las figuras que deben. En la Figura 24 se pueden ver en azul las distintas trayectorias deberá seguir el brazo, y en rojo todos los puntos que se envían.

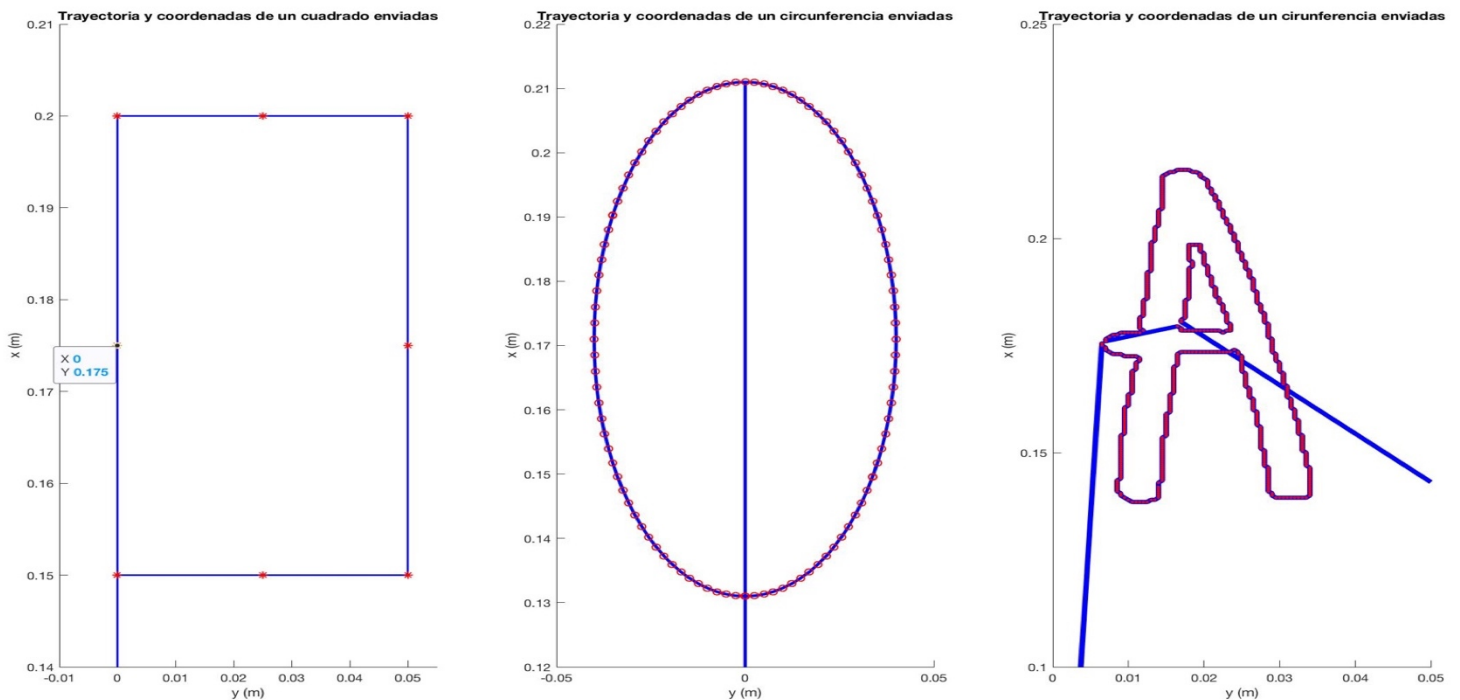


Figura 24 Trayectorias y coordenadas a seguir por el brazo

A continuación, en la se pueden ver las posiciones angulares equivalentes obtenidas para la trayectoria de una circunferencia.

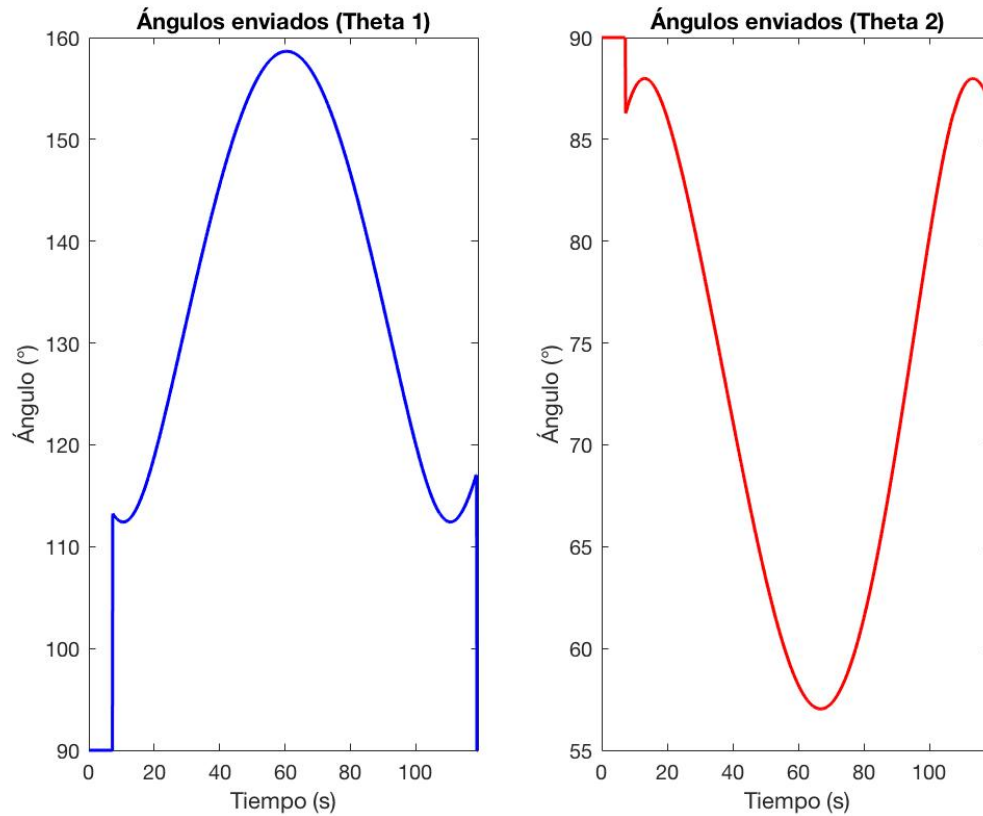


Figura 25 Posiciones angulares de una circunferencia enviadas al brazo

Al final del proyecto el buslinker la pieza encargada de pasar de full dúplex a half dúplex se estropeo. Debido a esto se tuvo que cambiar los servos inteligentes por unos servos convencionales. Esto eran de menor calidad y producían muchas vibraciones en el brazo. Se modifico un poco el software para enviar las posiciones angulares a los nuevos servos. En la Figura 26 se pueden ver los resultados reales obtenidos sobre el papel.

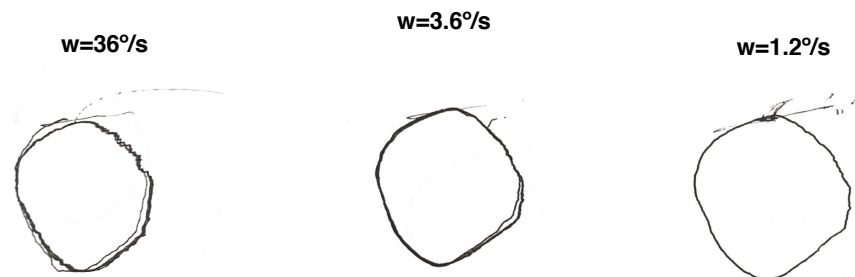


Figura 26 Circunferencias dibujadas

La trayectoria obtenida sobre el papel, se diferencia mucho a la que se usa de referencia. Esto se debe a la precisión de los servos que finalmente se han tenido que utilizar, y a las vibraciones que estos provocaban en el extremo del brazo. Estas se lograron disminuir ligeramente al aumentar el tiempo que tarda en realizar la circunferencia.

Capítulo 6. Conclusiones, aplicaciones y futuros proyectos

6.1 Conclusiones

Se ha visto que la estructura diseñada funciona correctamente. Esta puede ser mejorada con un apoyo para el extremo del brazo, para que este sea mas robusto.

En cuanto a los servos inteligentes, disminuyen mucho el cableado , permiten conectar muchos servos sin utilizar muchos pines, facilitan obtener información sobre ellos, por ello se cree que pueden ser muy útiles para futuros proyectos. Pero debido a los fallos que se han obtenido , seria conveniente utilizar servos de otros fabricantes. Por un lado el fabricante lewansoul ha proporcionado un buslinker que ha dado muchos problemas y finalmente han dejado de funcionar. Y por otro lado , la empresa feetehc proporciona unos servos con apenas información sobre su protocolo de comunicación , y con un software para configurar el servo que no ha funcionado.

[13]Para el control de un brazo SCARA sería mas interesante utilizar motores y encoder,, puesto las dinámicas de estos servos no permiten diseñar un control.

En cuanto al software implementado, se ha comprobado que este funcione. La máquina de estados, la cinemática inversa, el software de vectorización de imágenes, y las trayectorias de la circunferencia y cuadrado estaban bien diseñadas. Por su correcto funcionamiento, este software podría ser reutilizado en futuros proyectos.

6.2 Futuros proyectos

Como posible continuación de este proyecto, se podrían utilizar giróscopos para el diseño de un control mas preciso. Esto seria fácil puesto que la estructura ya esta diseñada e impresa. Seria posible realizar comunicaciones con el móvil para poder enviar imágenes al brazo. Otra alternativa, seria añadir una cámara que siga la imagen que dibuje un humano y a la vez el brazo SCARA vaya dibujando esta. También se le podría introducir mas grados de libertad que permitiese realizar funciones mas complejas.

En este proyecto se ha trabajado bastante con un hardware que aun no se había utilizado, principalmente los servos inteligentes, estos serán muy útil en futuros proyectos puesto que disminuyen el cableado y facilitan el control de sistemas mas complejos. Se podrían utilizar para muchos proyectos, y uno de los mas interesantes seria el desarrollo de un humanoide.

Referencias

- [1] «HEFLO,» [En línea]. Available: <https://www.heflo.com/es/blog/automatizacion-procesos/que-es-la-automatizacion-de-procesos/>. [Último acceso: 12 Abril 2019].
- [2] C. Tec, What is a SCARA robot?, pp. <https://cyan-tec.com/laser-systems/when-to-use-a-scara-robot>.
- [3] O. R. Lopez, «Diseño y construcción de un brazo manipulador SCARA,» Universidad de Michoacán de San Nicolás de Hidalgo, 2016.
- [4] Wandoo. [En línea]. Available: <http://perso.wanadoo.es/e/alimapp/scara/scara.htm>. [Último acceso: 25 Abril 2019].
- [5] J. L. Z. Macho, «Control por realimentación de estado».
- [6] T. Benjanarasuth, N. S. y N. N. , «Two-Degree-of-Freedom Simple Servo Adaptive Control for SCARA Robot,» King Mongkut's Institute of Technology Ladkrabang, Bangkok, 2010.
- [7] E. Muñoz, V. H. M. y C. A. G. , Evaluación del desempeño de diversos controladores avanzados aplicados a un robot manipulador tipo SCARA, vol. 29, Barranquilla, Colombia: Universidad del Norte, 2011, pp. 202-223.
- [8] B. Ibrahim y A. M. A. Z. , «Modeling and Control of SCARA manipulator,» University of Tun Hussein Onn Malaysia, 2014.
- [9] O. A. V. Albán, CONTROL PREDICTIVO DE UN ROBOT TIPO SCARA, vol. 14, Revista Chilena de Ingeniería, 2006, pp. 134-145.
- [10] M. A. Al-Khedher y M. S. A. , SCARA Robot Control using Neural Networks, Amman: Al-Balqa Applied University, 2012.
- [11] A. V. Pérez y M. A. P. Pulido. [En línea]. Available: <http://perso.wanadoo.es/e/alimapp/scara/index.htm>.
- [12] wikipedia. [En línea]. Available: https://es.wikipedia.org/wiki/Placa_computadora. [Último acceso: 16 Mayo 2019].
- [13] R. pi. [En línea]. Available: <https://www.raspberrypi.org/documentation/configuration/uart.md>. [Último acceso: 16 Mayo 2019].
- [14] S. Perkins. [En línea]. Available: <https://study.com/academy/lesson/simplex-half-duplex-duplex-communication-channels.html>. [Último acceso: 28 Mayo 2019].
- [15] L. Soul, «LewanSoul Bus Servo Communication Protocol,» [En línea]. Available: www.lewansoul.com.
- [16] L. Dauphin, «Github,» [En línea]. Available: <https://github.com/AversivePlusPlus/feetech>. [Último acceso: 4 Junio 2019].
- [17] J. L. M. Pérez, «DISEÑO E IMPLANTACIÓN DEL SISTEMA DE CONTROL DE UN ROBOT SCARA,» ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA (ICAI), Madrid, 2015.
- [18] Amir, «MathWorks,» [En línea]. Available: <https://es.mathworks.com/matlabcentral/fileexchange/54799-convert-image-pixels-to-xy-coordinates>. [Último acceso: 20 Junio 2019].
- [19] L. Soul, «LewanSoul Bus Servo Communication Protocol».
- [20] Raspberrypi. [En línea]. Available: <https://www.raspberrypi.org/documentation/configuration/uart.md>.
- [21] Lewansoul, «www.lewansoul.com,» [En línea]. Available: <http://www.lewansoul.com/product/detail-17.html>. [Último acceso: 24 Junio 2019].

[22] Geekforgeeks. [En línea]. Available: <https://www.geeksforgeeks.org/transmission-modes-computer-networks/>. [Último acceso: 16 Junio 2019].

PARTE II: Presupuesto

Capítulo 1. Mediciones

1.1 Hardware

Componente	Cantidad
Servomotor LX-16A	2
Servomotor SCS15	2
Raspberry Pi 3B +	1
Piezas impresas	8
FUENTE_1 de 6 V	1
FUENTE_2 de 5V	1
EMAX 800A II	1
Pulsadores, cableado, tornillos	-
Placa de policarbonato	1

1.2 Software

Componente	Unidades de uso
Solid Edge	1
Matlab/Simulink	1

1.3 Mano de obra directa

Componente	Horas
Programación	120
Horas de diseño 3D	20
Montaje	5
Corrección de errores y mejoras	50
Estudio de los servomototes	50

Capítulo 2. Precio unitario

2.1 Hardware

Componente	Precio/Unidad (€)
Servomotor LX-16A	15
Servomotor SCS15	30
Raspberry Pi 3B +	40
Piezas impresas	9
FUENTE_1 de 6 V	22
FUENTE_2 de 5V	24
EMAX 800A II	21,68
Pulsadores, cableado, tornillos	-
Estructura de soporte	118,3

2.2 Software

Componente	Precio (€)
Solid edge	0
Matlab/Simulink	800

2.3 Mano de obra directa

Tarea	Precio/Hora (€)
Programación	15
Horas de diseño 3D	15
Montaje	7
Corrección de errores y mejoras	15
Estudio de los servomototes	15

Capítulo 3. Sumas parciales

3.1 Hardware

Material	Cantidad	Precio unitario	Total
Servomotor LX-16A	2	31,02	62,04
Servomotor SCS15	2	23,05	46,1
Raspberry Pi 3B +	1	36,4	36,4
Piezas impresas	8	7,5	60
FUENTE_1 de 6 V	1	22	22
FUENTE_2 de 5V	1	24	24
EMAX 800A II	1	12,68	12,68
Pulsadores, cableado, tornillos	-	-	15
Estructura de soporte	1	118,3	118,3

3.2 Software

Material	Cantidad	Precio unitario	Total
Solid edge	1	0	0
Matlab/Simulink	1	800	800

3.3 Mano de obra directa

Material	Cantidad	Precio unitario	Total
Programación	80	15	1200
Horas de diseño 3d	35	15	525
Montaje	10	7	70
Corrección de errores y mejoras	50	15	750
Estudio de los servomotores	50	15	750

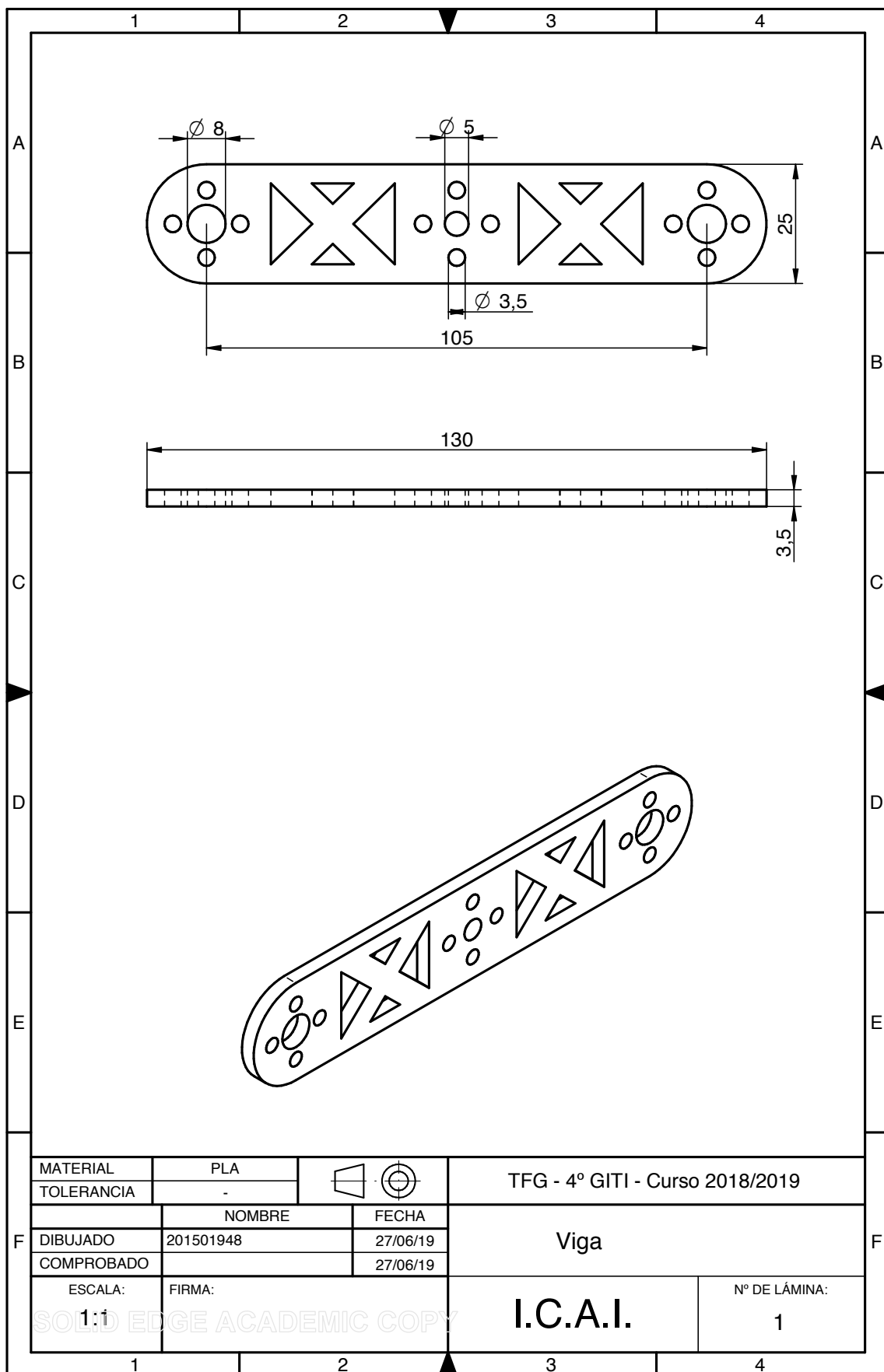
Capítulo 4. Suma total

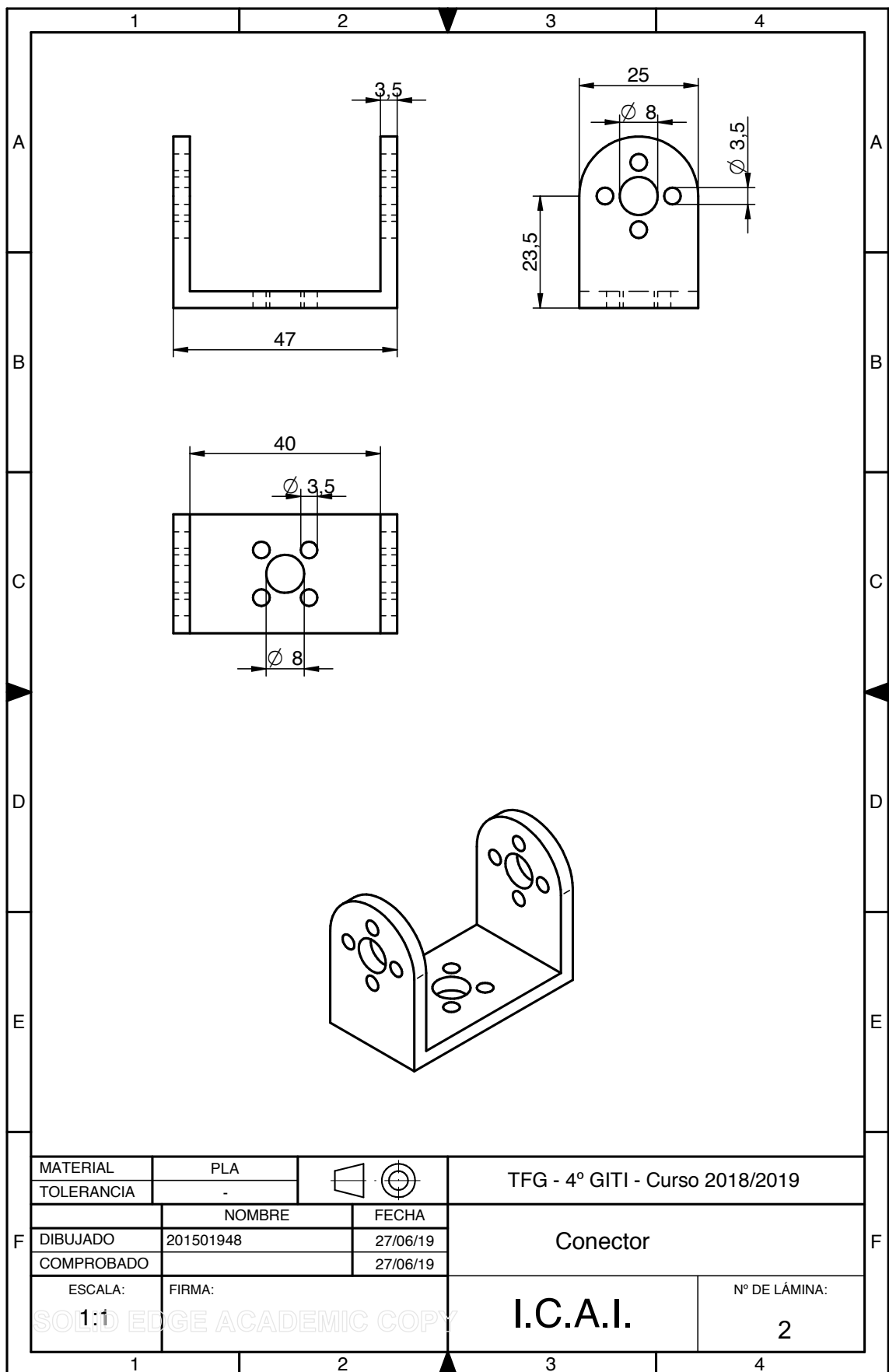
Suma total	
Apartado	Coste parcial (€)
Hardware	396,52
Software	800
Mano de obra directa	3295
Total	4491,52

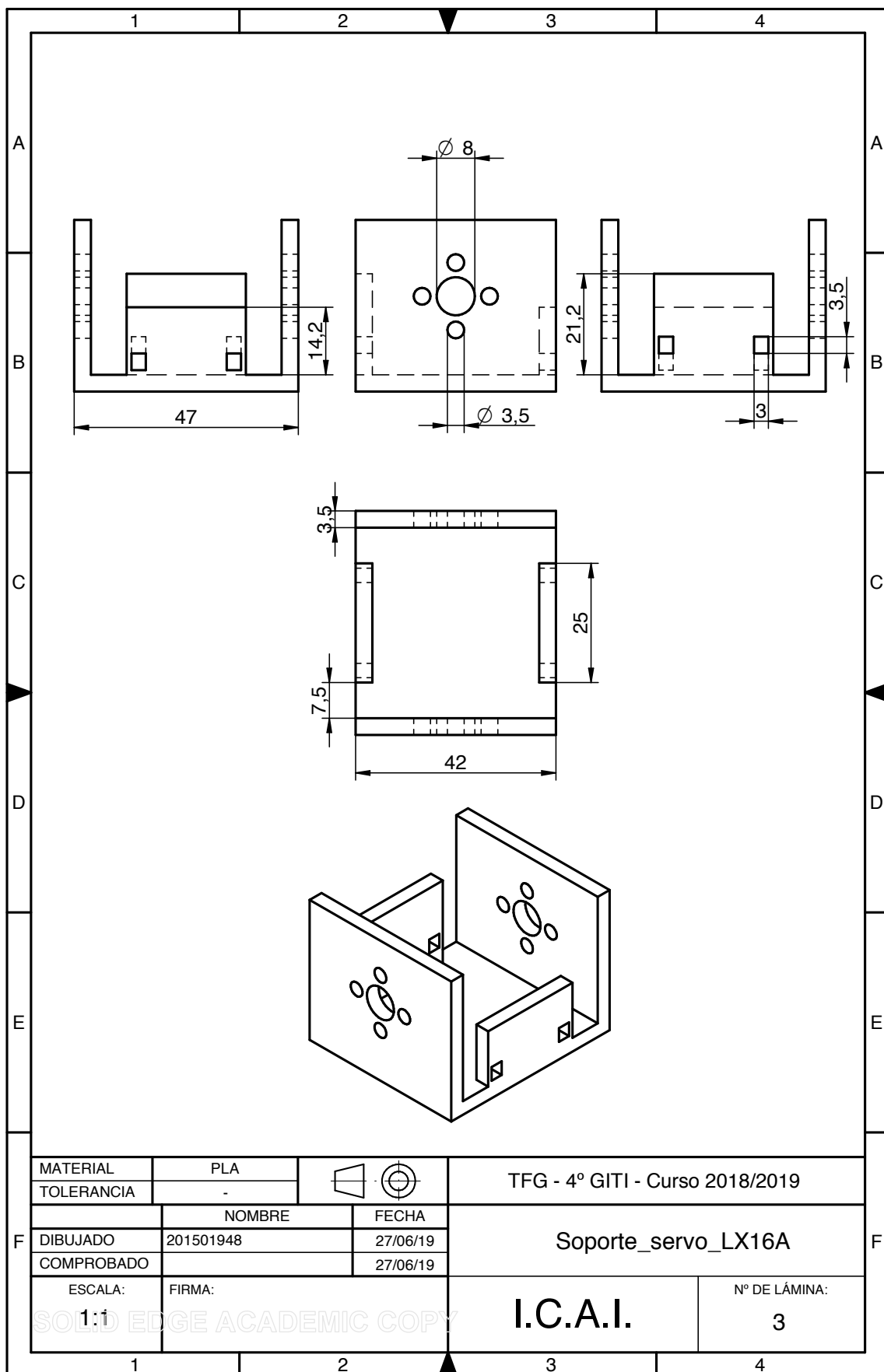
PARTE III: ANEXOS

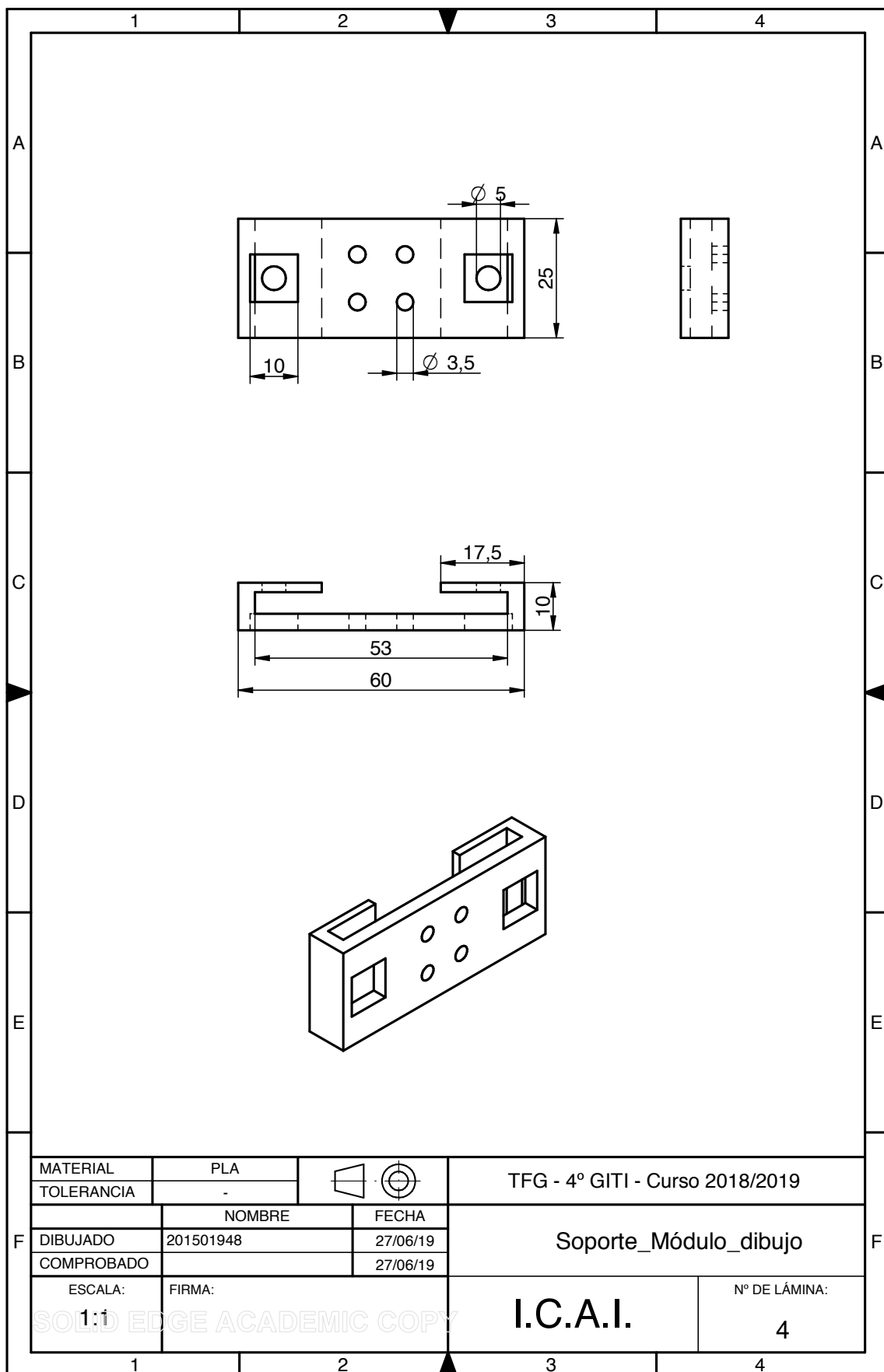
Anexo A: Lista de planos

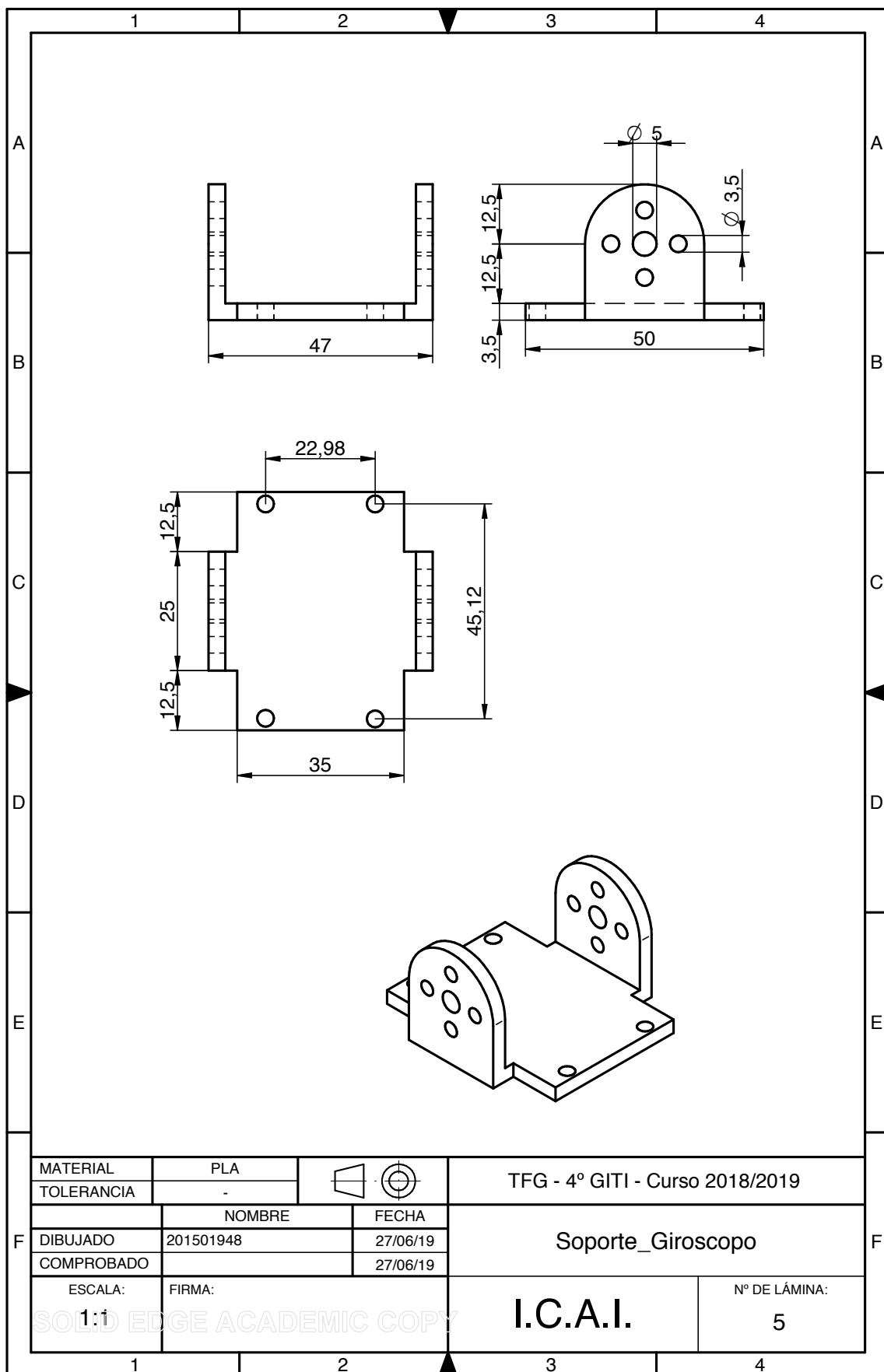
En primer lugar se mostraran los planos de cada una de las piezas de forma independiente, veremos en primer lugar el brazo, el conector y el encaje del servo . Y finalmente veremos el conjunto de todo el brazo ensamblado.











Anexo B: Código para la vectorización de imágenes

A continuación, se puede ver el código que se ha implementado para el procesamiento de imágenes.

```
% Leer imagen original
A = imread('.png');
figure(1);
subplot(3,2,1);
imshow(A);
title('Imagen original');

%Convertir a escala de grises
B=rgb2gray(A);
subplot(3,2,2);
imshow(B);
title('Imagen en escala de grises');

%Convertir a blanco y negro
C = im2bw(B,0.4);
subplot(3,2,3);
imshow(C);
title('Imagen en blanco y negro');

%Obtener los bordes de la imagen
D = edge(C, 'zerocross');
s=size(D);

for i=1:s(1)
    for j=1:s(2)
        if D(i,j)==1
            D(i,j)=0;
        elseif D(i,j)==0
            D(i,j)=1;
        end
    end
end
subplot(3,2,4);
imshow(D);
title('Imagen solo con bordes');

% Pasar de una imagen a puntos en el espacio
figure(2)
CoordinateMatrix = pic2points(D,0.5,1);
scatter(CoordinateMatrix(:,1), CoordinateMatrix(:,2),'.');

% Cambiar la escala de puntos
%     escala_1=(1/1.45)*10^-3;
%     escala_2=(1/1.45)*10^-3;
%     CoordinateMatrix_escalado(:,1)=escala_1*CoordinateMatrix(:,1);
%     CoordinateMatrix_escalado(:,2)=escala_2*CoordinateMatrix(:,2)+0.105;
%     scatter(CoordinateMatrix_escalado(:,1), CoordinateMatrix_escalado
(:,2),'.');

%Ordenar los puntos para obtener una trayectoria continua
```

```

Matriz=CoordinateMatrix_escala;
Trayectoria=Matriz(1,:);
tam=size(Matriz);

    for i=1:tam(1)

        if i>=tam(1)-1

            Trayectoria=[Trayectoria];

        else
            %Diferencia entre los puntos
            Diferencia=Matriz(1,:)'-Matriz';

            % Distancia euclidea de todos los puntos
            Distancia=sqrt(Diferencia(1,:).^2+Diferencia(2,:).^2);
            [aux1 , aux2]=sort(Distancia,'ascend');

            %Reordenando todo los puntos
            Trayectoria=[Trayectoria; Matriz(aux2(2),:)]';
            Matriz([2 aux2(2)],:)=Matriz([aux2(2) 2],:);
            Matriz= Matriz([2:end],:);
        end
    end

figure(3);
scatter(Trayectoria (:,1), Trayectoria (:,2),'.');

```

Anexo C: Código utilizado para el control de los servomotores

En este apartado, el código utilizado en las funciones de simulink para el control de los servomotores inteligentes.

1. Función para enviar el ángulo requerido

```
function [PAQUETE_INSTRUCCION_1 , PAQUETE_INSTRUCCION_2]=  
INSTRUCCION_ESCRITURA_POSICION (SCARA_ANG_TARGET, SERVO_MOVE_TIME_WRITE,  
SERVO_MOVE_TIME_WRITE_LENGTH , SERVO_HEADER)  
  
ID_1=1;  
ID_2=2;  
  
ANG_1=getang(SCARA_ANG_TARGET(1));  
ANG_2=getang(SCARA_ANG_TARGET(2));  
  
Time_1=double(typecast(uint16(0), 'uint8'));  
Time_2=double(typecast(uint16(0), 'uint8'));  
  
c_suma_1=double(1+SERVO_MOVE_TIME_WRITE+SERVO_MOVE_TIME_WRITE_LENGTH+ANG_1(1)+  
... ANG_1(2)+Time_1(1)+Time_1(2));  
  
checksum_1=getchecksum(c_suma_1);  
  
%%% poner el tiempo en que queramos que se mueva a la posicion  
c_suma_2=double(2+SERVO_MOVE_TIME_WRITE+SERVO_MOVE_TIME_WRITE_LENGTH+ANG_2(1)+  
... ANG_2(2)+Time_2(1)+Time_2(2));  
  
checksum_2=getchecksum(c_suma_2);  
  
PAQUETE_INSTRUCCION_1=uint8([SERVO_HEADER SERVO_HEADER ID_1...  
SERVO_MOVE_TIME_WRITE_LENGTH SERVO_MOVE_TIME_WRITE  ANG_1(1) ANG_1(2)  
Time_1(1) Time_1(2) checksum_1 ]);  
PAQUETE_INSTRUCCION_2=uint8([SERVO_HEADER SERVO_HEADER ID_2...  
SERVO_MOVE_TIME_WRITE_LENGTH SERVO_MOVE_TIME_WRITE  ANG_2(1) ANG_2(2)  
Time_2(1) Time_2(2) checksum_2 ]);
```

2. Función para el cálculo del checksum de un paquete de instrucción

```
function y = getchecksum( c_suma )

if (c_suma>=256)
    checksum=bitcmp(c_suma, 'uint16');
    h_b=floor(checksum/2^8);
    checksum = checksum - h_b*2^8;
else
    checksum=bitcmp(c_suma, 'uint8');
end

    y=checksum;
end
```

3. Función para obtener el ángulo leído del servo

```
function [dato_1,dato_2]= informacion(buffer_entrada)

buffer_lectura=buffer_entrada;

id=buffer_lectura(3);
cmd=buffer_lectura(5);
dato_1=0;
dato_2=0;

if id==1    %SERVO_POS_READ 28 5

    dato_1=double(0.24*(buffer_lectura(6)+buffer_lectura(7)*2^8));

elseif id==2
    dato_2=double(0.24*(buffer_lectura(6)+buffer_lectura(7)*2^8));
end

dato_1;
dato_2 ;
```