



COMILLAS

UNIVERSIDAD PONTIFICIA

ICAI

GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

TRABAJO FIN DE GRADO

**CONSTRUCCIÓN DE UN EQUIPO DE
EQUILIBRADO DE UNA ESFERA SOBRE
PLATAFORMA MÓVIL PARA DOCENCIA**

Autor: D. Enrique Alonso Álvarez

Director: Dr. Juan Luis Zamora Macho

MADRID

Agosto de 2019

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título
**Construcción de un equipo de equilibrado de una esfera sobre una plataforma
móvil para docencia**

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.: Enrique Alonso Álvarez

Fecha: 25/08/2019



Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Fdo.: Juan Luis Zamora Macho

Fecha: 22/08/2019



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. **Enrique Alonso Álvarez** DECLARA ser el titular de los derechos de propiedad intelectual de la obra: **Construcción de un equipo de equilibrado de una esfera sobre plataforma móvil para docencia**, 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) Reproducirla 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 25 de agosto de 2019,

ACEPTA

Fdo. Enrique Alonso Álvarez



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



GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

TRABAJO FIN DE GRADO

**CONSTRUCCIÓN DE UN EQUIPO DE
EQUILIBRADO DE UNA ESFERA SOBRE
PLATAFORMA MÓVIL PARA DOCENCIA**

Autor: D. Enrique Alonso Álvarez
Director: Dr. Juan Luis Zamora Macho

MADRID
Agosto de 2019

RESUMEN DEL PROYECTO

Hoy en día hay un gran interés, por parte de las empresas y de los particulares, en que las cosas se realicen de forma automática. La automatización de procesos tiene amplios beneficios como una reducción de los ciclos de producción, un aumento de la seguridad y de la efectividad y un abaratamiento de dichos procesos.

Es por ello por lo que se está poniendo mucho empeño en optimizar la automatización, la cual depende de muchos factores, entre los que destaca el sistema de control. Además, cada vez se trata de concienciar más a los alumnos sobre este tema, haciendo que profundicen en su estudio mediante ejemplos reales.

El proyecto “ball on plate” es un clásico problema de control el cual mezcla tanto una parte mecánica como una parte electrónica. Mecánica por el hecho de tener que diseñar y construir una estructura capaz de sostenerse y de rotar en ambos ejes X e Y, según dicten los servos empleados. Electrónica en tanto en cuanto se debe manejar esa estructura para controlar totalmente el movimiento de la esfera dentro de la pantalla resistiva. Para ello, se emplearán los sensores adecuados, todos ellos supervisados por un microprocesador que, en este caso, será la Raspberry Pi.

Podría decirse que el “ball on plate” es una actualización del famoso “ball and beam”, proyecto el cual está formado por un rectángulo estrecho sobre el que reposa una esfera. Dicha esfera está controlada por el movimiento de un solo servo que garantiza su estabilidad, como podemos observar en [1].



Figura 1: modelo “ball and beam” de Quanser [QUAN13]

En este caso, la esfera solo se mueve en un eje, por lo que tanto el diseño de su estructura como el control de la esfera son más asequibles que en el caso del “ball on plate”, donde la esfera se mueve en dos ejes y serán necesarios al menos dos servos para controlar su movimiento. Digo al menos dos

porque hay variantes del proyecto donde se utilizan más, para dotar a la bandeja de más estabilidad, como podemos ver en [2].

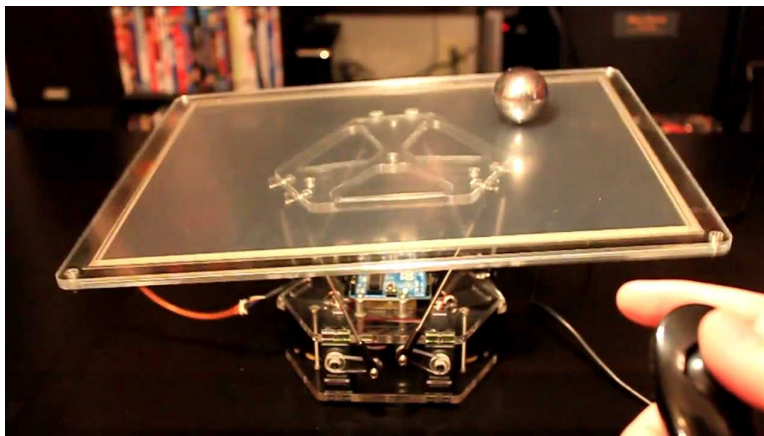


Figura 2: ball on plate con 3 servos [TYLE12]

La manera de conocer la posición de la esfera es mediante una pantalla resistiva, la cual capta las coordenadas de la esfera en ambos ejes. Sin embargo, hay otras múltiples opciones como puede ser el uso de una pantalla capacitiva (más práctica, pero más compleja y costosa) o bien mediante el procesamiento de imágenes. Esta última es realmente interesante, ya que se usa una cámara sobre la bandeja que capta el movimiento de la esfera, como podemos ver en [3]. Al usar la cámara, el uso de una pantalla resistiva o capacitiva es prescindible. De todas las opciones, la más efectiva es la elegida para este proyecto, con dos servos que mueven la bandeja en ambos ejes y utilizando una pantalla resistiva para captar la posición de la esfera.

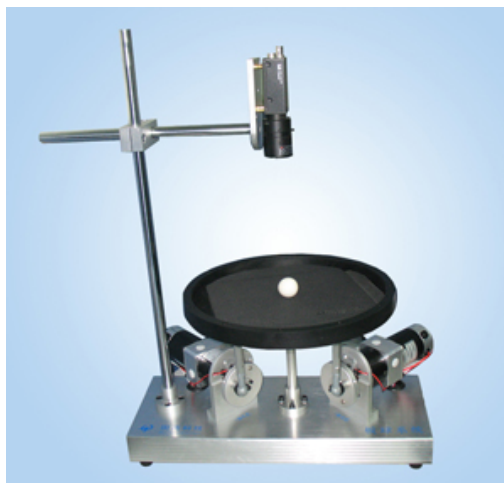


Figura 3: ball on plate por procesamiento de imágenes [SOLU99]

Para el funcionamiento de la pantalla, era preciso incluir un circuito de adaptación formado por resistencias, condensadores y transistores. Este circuito fue diseñado con la herramienta libre KiCad, que consiste en un software libre de diseño de PCB's. Una vez diseñado, se imprimió la pista del circuito en una placa de cobre y se soldaron los elementos necesarios.

Además, con respecto al resto de sensores, este proyecto usa una IMU para medir la posición angular de la bandeja en cada uno de los ejes y un conversor ADC para transformar las tensiones resultantes de la pantalla resistiva con el fin de que puedan llegar y ser usadas por la Raspberry Pi. La IMU es un elemento muy sensible que detecta cualquier variación, por muy leve que sea, por lo que se recomienda no mover mucho la bandeja de apoyo de la pantalla, ya que la IMU estará colocada en su reverso, como podemos observar en [4].

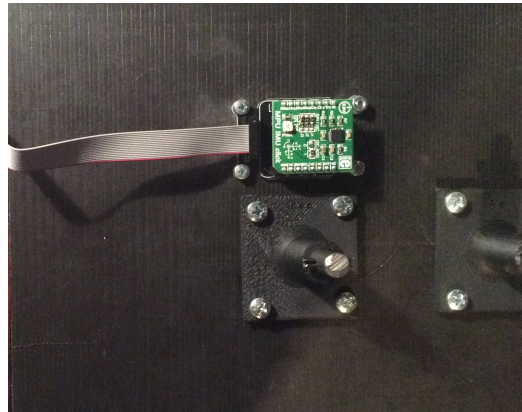


Figura 4: posición de la IMU en la bandeja de apoyo

Por otro lado, el diseño de la estructura del proyecto no fue tarea fácil, ya que la mayor parte de sus piezas fueron hechas por impresión 3D. La impresión 3D, debido al material que se usa para hacer las piezas, no es del todo precisa con las medidas, por lo que cada pieza tuvo que diseñarse e imprimirse varias veces. Mención especial a la bandeja que sostiene la pantalla que, al ser tan grande, salió deformada, por lo que tuve que hacer varios retoques en ella.

Una vez impresas las piezas, había varias formas de construir la estructura: mediante juntas universales, rótulas esféricas, ventosas, etc. Sin embargo, opté por la primera, que, a mi punto de vista, era la más práctica. Las juntas universales permiten el movimiento de la bandeja sobre la que se sostiene la pantalla resistiva de forma que, si los servos se mueven, la pantalla se mueve con ellos mediante una relación mecánica que veremos más adelante en el apartado del modelado. En ese apartado también veremos las ecuaciones de la esfera, las cuales se acaban simplificando si tenemos en cuenta una serie de consideraciones como que la esfera no desliza sobre la pantalla o que la bola

es simétrica y homogénea. Con las ecuaciones de los servos, las de la esfera y con las relaciones entre servos y pantalla, tenemos un sistema de 8 ecuaciones de estado. En este sistema, las variables de estado son las siguientes: la posición angular de la bandeja en ambos ejes, la velocidad angular de la bandeja en ambos ejes, la posición de la esfera en ambos ejes y la velocidad de la esfera en ambos ejes.

Durante la realización del proyecto surgieron varios problemas como la dificultad de impresión de ciertas piezas, en especial de la bandeja, que salió deforme y tuve que lijar su interior y aplicar calor para dar una nueva forma a las esquinas. Por ello, recomendaría volver a imprimirla o incluso sustituirla por una bandeja real con las medidas oportunas. Otro contratiempo fue la compra de las juntas universales, las cuales se demoraron mucho en llegar, por lo que no pude montar la estructura hasta el final.

La continuación de este proyecto consistiría en el diseño de un control para automatizar la plataforma. Este control permitiría mantener la bola en una posición fija o bien hacer que la bola siga una trayectoria definida dentro de la bandeja. Además, recomendaría la prueba del procesamiento de imágenes mediante el uso de una cámara o bien el cambio de la pantalla resistiva por una capacitiva.

ABSTRACT

Nowadays there is a great interest, on the part of companies and individuals, in things being done automatically. The automation of processes has extensive benefits such as a reduction in production cycles, an increase in safety and effectiveness and a reduction in the cost of these processes.

For this reason, great efforts are being made to optimise automation, which depends on many factors, including the control system. In addition, it is increasingly important to make students more aware of this subject, making them study it in greater depth by using real examples.

The "ball on plate" project is a classic control problem which mixes both a mechanical and an electronic part. Mechanical since it is necessary to design and build a structure capable of supporting and rotating on both X and Y axes, as dictated by the servos used. Electronic inasmuch as this structure must be managed to totally control the movement of the sphere within the resistive screen. Due to this, the appropriate sensors will be used, all of them supervised by a microprocessor that, in this case, will be the Raspberry Pi.

It could be said that the "ball on plate" is an update of the famous "ball and beam", project which is formed by a narrow rectangle on which rests a sphere. This sphere is controlled by the movement of a single servo that guarantees its stability, as we can see in [1].



Figure 1: model "ball and beam" by Quanser [QUAN13]

In this case, the sphere only moves in one axis, so both the design of its structure and the control of the sphere are more affordable than in the case of the "ball on plate", where the sphere moves in two axes and at least two servos will be necessary to control its movement. I say at least two because there are variants of the project where they are used more than two, to give the tray more stability, as we can see in [2].

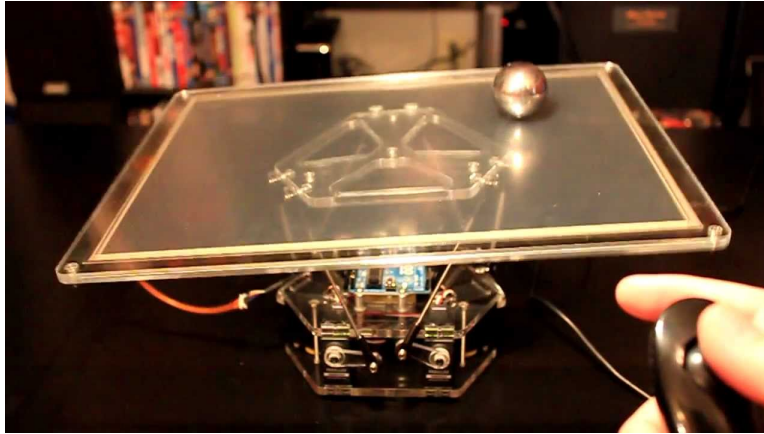


Figure 2: ball on plate with 3 servos [TYLE12]

The way to know the position of the sphere is by means of a resistive screen, which captures the coordinates of the sphere in both axes. However, there are other multiple options such as the use of a capacitive screen (more practical, but more complex and expensive) or through image processing. The latter is quite interesting, since a camera is used on the tray that captures the movement of the sphere, as we can see in [3]. When using the camera, the use of a resistive or capacitive screen is dispensable. Of all the options, the most effective is the one chosen for this project, with two servos moving the tray in both axes and using a resistive screen to capture the position of the sphere.

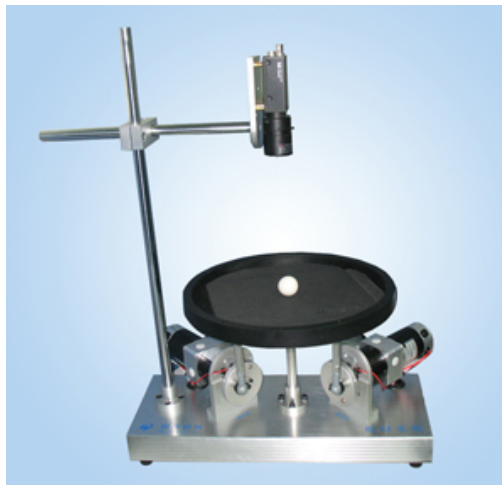


Figure 3: ball on plate for image processing [SOLU99]

For the operation of the screen, it was necessary to include an adaptation circuit consisting of resistors, capacitors and transistors. This circuit was designed with the free KiCad tool, which consists of a free

PCB's design software. Once designed, the circuit track was printed on a copper plate and the necessary elements were welded.

In addition, with respect to the rest of the sensors, this project uses an IMU to measure the angular position of the tray in each of the axes and an ADC converter to transform the resulting voltages of the resistive screen so that they can be reached and be used by the Raspberry Pi. The IMU is a very sensitive element that detects any variation, however slight, so it is recommended not to move the screen support tray much, since the IMU will be placed on its back, as we can see in [4].

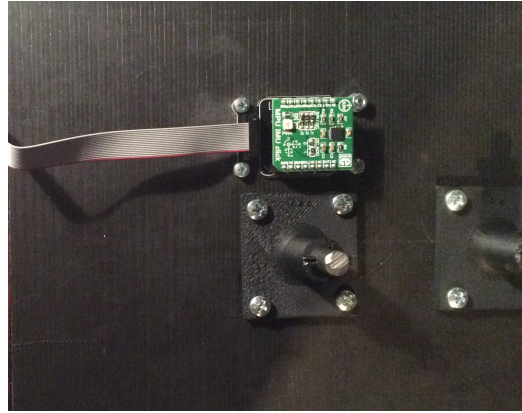


Figure 4: IMU position in the support tray

On the other hand, the design of the structure of the project was not an easy task, since most of its parts were made by 3D printing. The 3D printing, due to the material used to make the pieces, is not entirely accurate with the measurements, so each piece had to be repeated several times. Special mention to the tray that holds the screen that, being so big, came out deformed, so I had to make several retouches on it.

Once the pieces were printed, there were several ways to build the structure: by means of universal joints, spherical ball-and-socket joints, suction cups, etc. However, I opted for the first one, which, in my view, was the most practical. The universal joints allow the movement of the tray on which the resistive screen is held so that, if the servos move, the screen moves with them by means of a mechanical relation that we will see later in the section on modeling. In this section, we will also see the equations of the sphere, which are simplified if we take into account a series of considerations such as that the sphere does not slide on the screen or that the ball is symmetrical and homogeneous. With the equations of the servos, those of the sphere and with the relations between servos and screen, we have a system of 8 equations of state. In this system, the state variables are the following: the

angular position of the tray in both axes, the angular velocity of the tray in both axes, the position of the sphere in both axes and the velocity of the sphere in both axes.

During the realization of the project, several problems arose such as the difficulty of printing certain pieces, especially the tray, which came out deformed and I had to sand its interior and apply heat to give a new shape to the corners. Therefore, I would recommend printing it again or even replace it with a real tray with the appropriate measures. Another setback was the purchase of universal joints, which took a long time to arrive, so I could not assemble the structure until the end.

The next step of this project would consist of the design of a control to automate the platform. This control would allow to keep the ball in a fixed position or make the ball follow a defined path inside the tray. In addition, I would recommend testing image processing using a camera or changing the resistive screen to a capacitive one.

PALABRAS CLAVE:

Pantalla resistiva, servos, control, esfera, circuito de adaptación, Raspberry Pi, impresión 3D, junta universal, Simulink.

KEY WORDS:

Resistive screen, servos, control, sphere, adaptation circuit, Raspberry Pi, 3D printing, universal joint, Simulink.

DOCUMENTO 1:

MEMORIA

ÍNDICE:

I.	Memoria.....	19
1.1.	ESTADO DEL ARTE	20
1.1.1.	Estructura mecánica.....	20
1.1.2.	Posición de la esfera.....	22
1.2.	MOTIVACIÓN	25
1.3.	OBJETIVOS	26
1.4.	RECURSOS EMPLEADOS Y METODOLOGÍA DE TRABAJO	27
1.5.	DISEÑO Y MONTAJE	28
1.5.1.	Estructura	28
1.5.2.	Sensores	39
1.5.3.	Actuadores	41
1.6.	FUNCIONAMIENTO DE SENSORES Y OBTENCIÓN DE MEDIDAS	43
1.6.1.	Pantalla resistiva y circuito de adaptación	43
1.6.2.	Conversor ADC	48
1.6.3.	IMU (Inertial measurement unit)	54
1.7.	MODELADO DEL SISTEMA	58
1.8.	CONCLUSIONES Y TRABAJOS FUTUROS	62
1.9.	BIBLIOGRAFÍA	64
II.	Anexo A: Diagramas Simulink	67
III.	Anexo B: Código Matlab.....	73

ÍNDICE DE FIGURAS:

Figura 2: sistema ball and beam [MUKT13]	20
Figura 1: sistema ball on plate [DAVI16]	20
Figura 3: primera opción de diseño de estructura [MIAD08].....	21
Figura 4: segunda opción de diseño de estructura [TKUH18]	22
Figura 5: diseño de la estructura muy similar al original del proyecto [FILE16].....	22
Figura 6: diseño interno pantalla resistiva [JAVI16]	23
Figura 7: diseño interno pantalla capacitiva [OSCA09]	23
Figura 8: ball on plate con procesamiento de imagen [EDNA16].....	24
Figura 9: estructura construida con LEGO technic [PEDR15]	28
Figura 11: modelo con varillas 2 [DAVI17]	29
Figura 10: modelo con varillas 1 [FILE16]	29
Figura 12: estructura original empleada	30
Figura 13: tornillos para “aprisionar” al servo.....	31
Figura 15: vista frontal de la pieza	31
Figura 14: vista en perspectiva de la pieza.....	31
Figura 16: vista posición de los servos	32
Figura 18: vista lateral de la pieza	32
Figura 17: vista en perspectiva de la pieza.....	32
Figura 19: vista de tuerca + arandela	33
Figura 21: vista frontal de la pieza	34
Figura 20: vista en perspectiva de la pieza.....	34
Figura 23: vista aérea de la pieza	34
Figura 22: vista en perspectiva de la pieza.....	34
Figura 24: vista de los agujeros con sus respectivos tornillos	35
Figura 26: vista lateral de la pieza	36
Figura 25: vista en perspectiva de la pieza.....	36
Figura 28: vista aérea de la bandeja.....	36
Figura 27: vista en perspectiva de la bandeja	36
Figura 30: vista inferior de la bandeja	37
Figura 29: vista superior de la bandeja	37
Figura 31: vista bandeja + pantalla.....	38
Figura 32: unión junta y varillas	38
Figura 33: circuito auxiliar de la pantalla	39
Figura 34: localización del conversor ADC.....	40
Figura 35: divisores de tensión.....	41
Figura 36: localización de la IMU.....	41
Figura 37: estructura interna de la pantalla	43
Figura 38: lectura de la coordenada del eje Y	44
Figura 39: circuito de adaptación inicial.....	44

Figura 41: parte superior del circuito	45
Figura 40: diseño de la PCB con KiCad	45
Figura 42: parte inferior del circuito	46
Figura 43: coordenada eje X con la bola a mitad de la pantalla	46
Figura 44: coordenada eje X con la bola en el extremo derecho de la pantalla	46
Figura 45: coordenada eje Y con la bola antes de la mitad de la pantalla	47
Figura 46: coordenada eje Y con la bola en la parte de abajo.....	47
Figura 47: diseño interno del divisor de tensión	49
Figura 48: coordenada en X con divisor de tensión	49
Figura 49: conexión conversor ADC con divisor de tensión al circuito de adaptación	50
Figura 50: posición de la bola con el conversor ADC.....	50
Figura 51: scopes de los canales mostrando los resultados 1	51
Figura 52: scopes de los canales mostrando los resultados 2	51
Figura 53: conexión del Level Converter	52
Figura 54: nuevo diseño del circuito de adaptación.....	53
Figura 55: scopes de los canales mostrando los resultados 3	53
Figura 56: conexión entre IMU y Raspberry.....	54
Figura 57: scope de la IMU al girar la bandeja en el eje X repetidamente	55
Figura 58: scope de la IMU al girar la bandeja en el eje Y repetidamente	56
Figura 59: scope de la IMU al girar la bandeja en ambos ejes X e Y repetidamente	56
Figura 60: scopes útiles del diagrama de bloques de la IMU	56
Figura 61: sistema “ball and beam” [JOSE14]	58
Figura 62: estructura mecánica del sistema 1 [AWTA02]	60
Figura 63: estructura mecánica del sistema 2 [AWTA02]	60

I. Memoria

1.1. ESTADO DEL ARTE

El sistema de equilibrado de una esfera sobre una plataforma móvil, también llamado “ball on plate”, es uno de los sistemas más comunes para la prueba y aplicación de diversos sistemas de control. En él, controlamos la posición de la esfera actuando sobre una plataforma que la sostiene, la cual incluye una pantalla resistiva sobre la que reposa la esfera. De esta forma, y variando los ángulos en ambos ejes X e Y, conseguiremos que la esfera se mantenga fija en una posición e incluso que realice el recorrido que le pidamos a lo largo de dicha plataforma.

La plataforma estará controlada por dos servos, los cuales se encargan de cambiar los grados de esta en ambos ejes, como podemos ver en [1]. El uso de los servos es la mejor opción en este caso, aparte de ser la más barata.

Podría decirse que el sistema “ball on plate” es una actualización del famoso sistema “ball and beam”, donde una esfera reposa sobre una barra estrecha, como podemos ver en [2]. La posición de la bola en dicha barra es controlada por un único motor, el cual se encarga de gestionar la inclinación de la barra. A diferencia del “ball on plate”, el sistema “ball and beam” solo actúa sobre un eje, lo que lo hace más asequible que el primero.

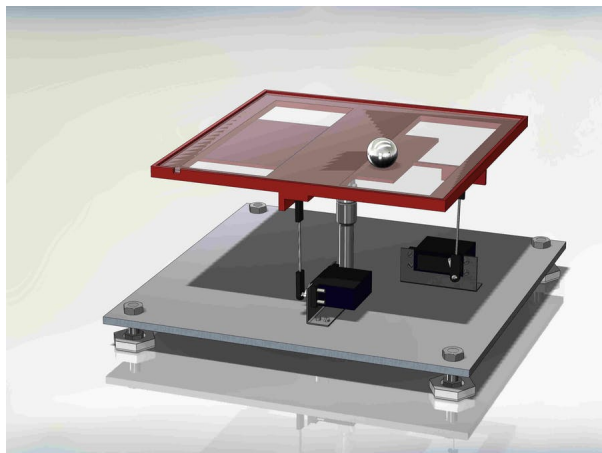


Figura 2: sistema ball on plate [DAVI16]

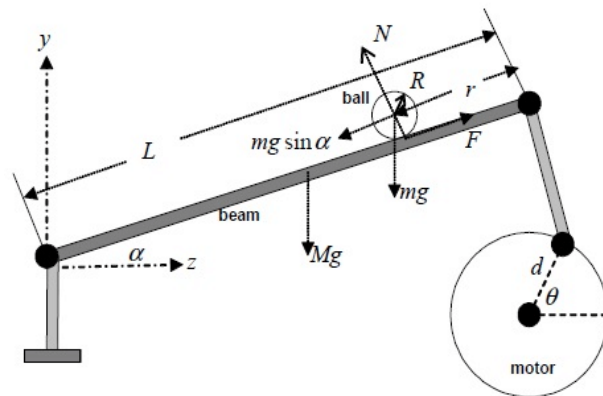


Figura 1: sistema ball and beam [MUKT13]

Para la realización de este proyecto, existen muchas alternativas con respecto a la estructura mecánica del sistema y a la forma de obtener la posición de la esfera.

1.1.1. Estructura mecánica

Como podemos ver en [1] y [2], el sistema “ball and beam” no solo es más asequible que el “ball on plate” en cuanto al control de la esfera, sino también en cuanto al diseño y construcción de la estructura mecánica.

De todas formas, dentro del “ball on plate” hay multitud de opciones a la hora de diseñar la estructura, bien por la forma de conectar los servos con la bandeja de soporte de la pantalla resistiva, bien por la forma de habilitar el movimiento de la bandeja con respecto al resto de la estructura.

Una opción es la que podemos ver en [3], donde la bandeja está suspendida en el aire solamente sostenida por apoyos laterales. En este caso, la bandeja no tiene un apoyo central, lo que hace que sea más fácil de montar, pero a la vez menos estable.

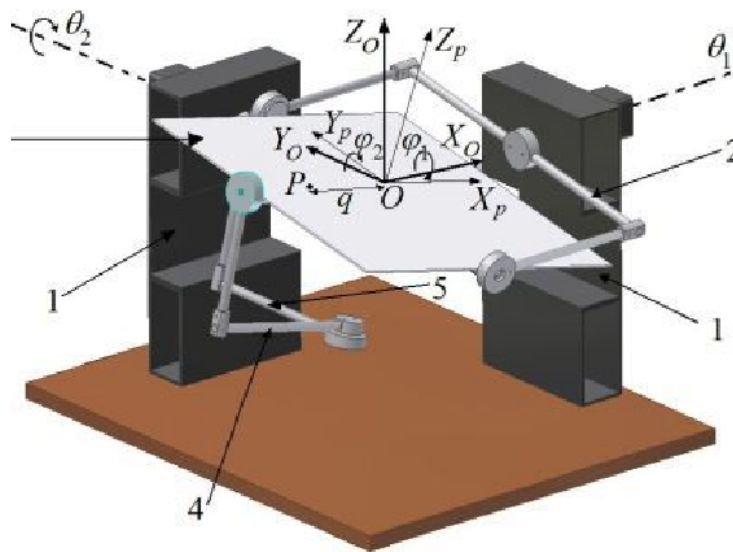


Figura 3: primera opción de diseño de estructura [MIAD08]

Además, como podemos ver, la unión de las barras metálicas con la pantalla resistiva se realiza mediante un pequeño apoyo en el reverso de la misma.

Otra opción es la que podemos ver en [4], el cual se trata de un “ball on plate” un tanto peculiar, ya que se usan 4 servos para actuar sobre una serie de barras metálicas que hacen la función de un trampolín. De esta forma, se controla la posición de la esfera, pero esta se encuentra botando de forma ininterrumpida.

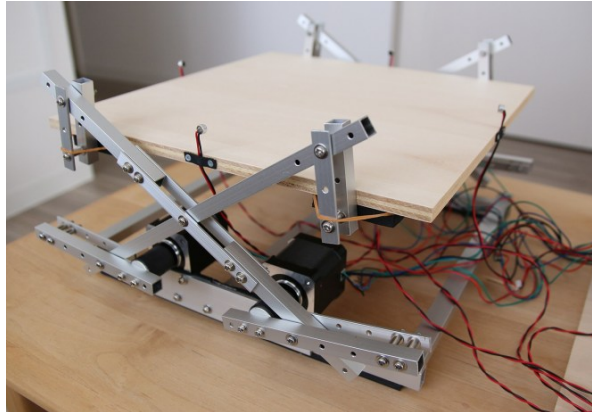


Figura 4: segunda opción de diseño de estructura [TKUH18]

Luego hay otras opciones más donde se usan rótulas esféricas para permitir el movimiento de la bandeja con respecto al resto de la estructura. Sin embargo, la opción más efectiva es la utilizada en este proyecto, que es muy parecida a la que podemos ver en [5], donde se emplean dos servos que controlan los dos brazos laterales. Además, consta de 3 juntas universales para dar libertad de movimiento a la bandeja, y la unión con la bandeja de arriba se hace mediante piezas por impresión 3D que se atornillan a ella.

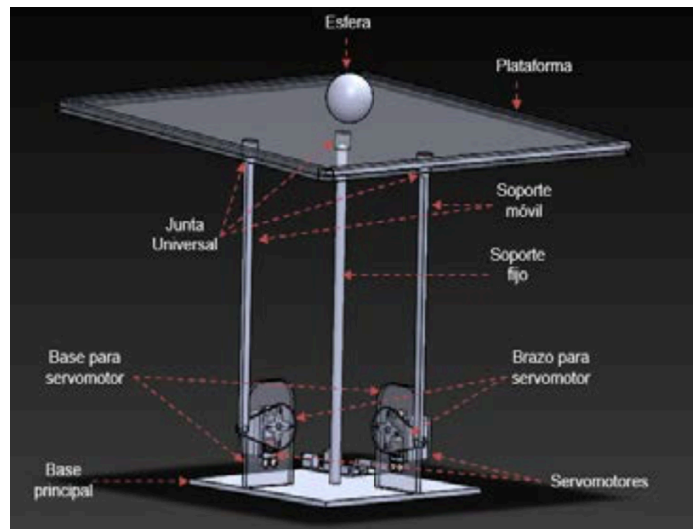


Figura 5: diseño de la estructura muy similar al original del proyecto [FILE16]

1.1.2. Posición de la esfera

Para determinar la posición de la esfera también hay muchas opciones distintas, entre las que destacan el uso de una pantalla, ya sea resistiva o capacitiva y el uso de una cámara para procesar imágenes.

En primer lugar, la pantalla resistiva detecta la pulsación de un objeto que tenga un mínimo peso, por lo que son muy sensibles [JAVI16]. Sin embargo, estas son más lentas que las capacitivas, aunque menos costosas y más resistentes. Estas están formadas por varias capas, como podemos observar en [6].

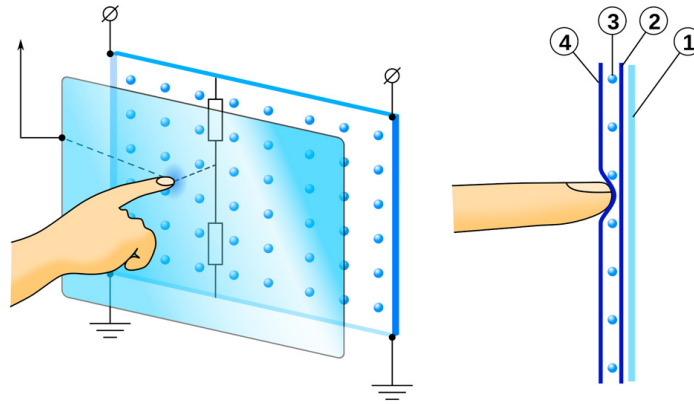


Figura 6: diseño interno pantalla resistiva [JAVI16]

Por otro lado, la pantalla capacitiva está basada en sensores capacitivos, como podemos observar en [7]. Generalmente, tienen mejor respuesta que la pantalla resistiva, aunque son más caras y menos resistentes [OSCA09]. Además, permiten que se pulse la pantalla varias veces a la vez, cosa que no ocurre en el caso de la resistiva, donde solo es posible pulsar una vez al mismo tiempo.

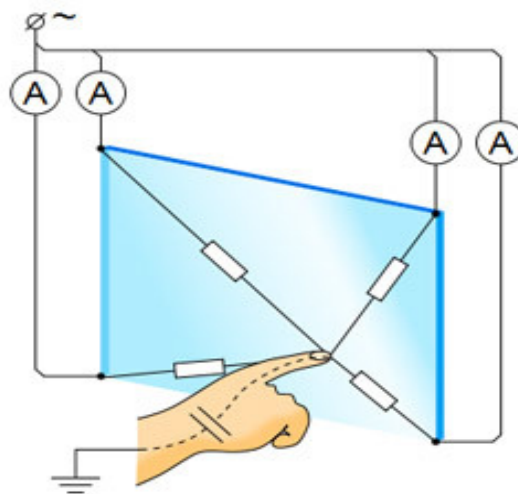


Figura 7: diseño interno pantalla capacitiva [OSCA09]

Por último, nos encontramos con el procesamiento de imagen. Para esta opción, se utiliza una cámara colocada por encima de la bandeja donde se sitúa la esfera, la cual mide la posición e incluso la

velocidad de esta, para que después sean usadas por el control implantado. La cámara se maneja desde Simulink, donde tendremos nuestros diagramas de bloques con lo que queramos que haga. Es cierto que esta es la opción más cara, pero también es más compleja que las demás, por lo que puede suponer un reto.

Como he dicho antes, la cámara está suspendida por encima de la bandeja mediante un apoyo lateral ajeno a la estructura del sistema, como podemos ver en [8].

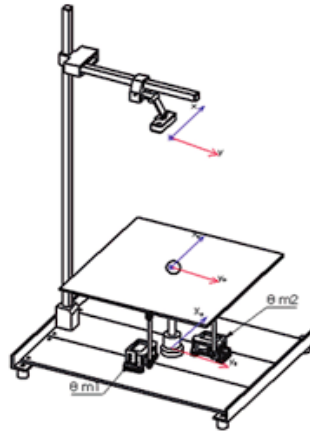


Figura 8: ball on plate con procesamiento de imagen [EDNA16]

1.2. MOTIVACIÓN

La motivación de este proyecto es el uso que se le va a dar en el futuro. Se trata de un proyecto que será utilizado en docencia por la propia universidad. Por ello, se busca el desarrollo de uno de los problemas clásicos de control mediante el uso de la Raspberry Pi, el cual no está todavía muy expandido. Este proyecto tiene un gran interés, ya que implica conocimientos de las principales asignaturas impartidas en ICAI relacionada con la electrónica y los controles. Además, tiene una parte mecánica correspondiente con el diseño y la construcción de la estructura empleada.

Por este motivo, considero que en un futuro será una herramienta muy útil con la que enseñar a los alumnos de dichas asignaturas.

1.3. OBJETIVOS

Los objetivos del proyecto podemos dividirlos en 2 bloques principales más un bloque extra que podrá cumplirse depende del tiempo del que dispongamos.

El primer bloque está relacionado con el diseño y construcción de la estructura sobre la que se apoya la pantalla resistiva empleada. Los objetivos son los siguientes:

1. Diseño con SolidEdge de todas las piezas que conforman la estructura, dando las medidas oportunas a las mismas.
2. Impresión 3D de las piezas diseñadas. Para ello, se utilizará la impresora del laboratorio de fabricación de ICAI.
3. Montaje de la estructura con las piezas impresas y con el resto de elementos, los cuales son juntas universales y varillas de acero.

El otro bloque principal corresponde con la implementación de los sensores y actuadores en el sistema. Los objetivos son los siguientes:

1. Diseño del circuito de adaptación de la pantalla resistiva con la herramienta KiCad.
2. Implementación de los sensores y actuadores necesarios, es decir, del conversor ADC, de la IMU, de la pantalla resistiva y de los dos servos empleados.
3. Combinación de la parte mecánica y del hardware del sistema. Modelado del mismo.

Por último, está el bloque extra de objetivos, que es donde entraría el diseño del control. Este bloque dependerá del tiempo del que se disponga, aunque la idea principal es centrarse en los dos bloques principales.

1.4. RECURSOS EMPLEADOS Y METODOLOGÍA DE TRABAJO

Con respecto a la parte mecánica, usaré: piezas de plástico impresas en 3D, varillas de acero de 8 mm de diámetro de distinta longitud, juntas universales para el encaje de dichas varillas, tornillos y arandelas para unir algunas piezas entre sí.

Después, en cuanto a sensores y actuadores, el sistema constará de una pantalla resistiva junto con su correspondiente circuito de adaptación, mediante la cual se sabrá la posición de la esfera en todo momento. La pantalla, con ayuda de un Level Converter, devolverá dos tensiones correspondientes a cada eje, que pasarán al conversor ADC, el cual se encargará de transformar esas señales de analógico a digital y de pasarlas a la Raspberry Pi. Por último, se utilizará una IMU para saber los ángulos de inclinación de la plataforma de arriba en ambos ejes y dos servos para controlar el movimiento de la plataforma en esos ejes.

Como he dicho antes, la información de la pantalla resistiva llegará a la Raspberry, la cual se encargará, con ayuda de Simulink, de mover los servos en función de la posición de la esfera. Además, usaré tanto Matlab como Simulink para hacer el modelado del sistema física y la posible implantación de un futuro control.

1.5. DISEÑO Y MONTAJE

En esta parte de la memoria voy a entrar en profundidad para explicar la construcción del sistema, el cual podemos separar en estructura y sensores.

1.5.1. Estructura

Al iniciar el proyecto, tenía como referencia el trabajo de un compañero de ICAI de hace unos años [PEDR15], cuya idea era similar a la mía salvo que la estructura donde se sustentaban la pantalla resistiva y la bola estaban construidas con piezas LEGO technic, como podemos observar en [9]. En mi opinión, utilizar estas piezas no era del todo práctico, debido a la complejidad del montaje y a la fragilidad del mismo. Además, no utilizaba servos como tal sino motores lineales, los cuales daban a la estructura la capacidad de moverse según se requiriese. El microprocesador empleado era un Arduino, mientras que en mi caso utilizo una Raspberry Pi, el cual es más completo y actual, a mi entender. Además, al ser para docencia, este trabajo es un ejemplo ideal para enseñar al alumnado acerca de la Raspberry Pi y su uso.

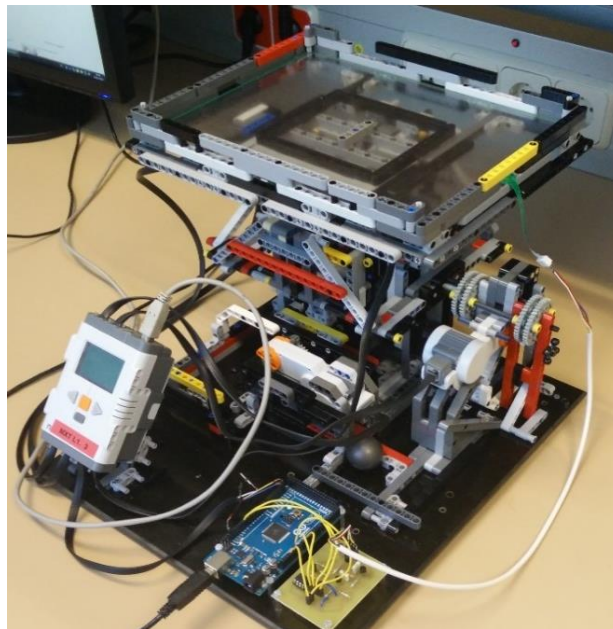


Figura 9: estructura construida con LEGO technic [PEDR15]

En [9], el autor dice que la elevación del eje de giro era prácticamente despreciable y que la transmisión de ángulo del eje de motor y la plataforma era lineal (con una ganancia que depende de los engranajes del sistema) [PEDR15], siendo esto último bastante parecido a lo que ocurre en mi sistema, como podremos comprobar más adelante.

Por tanto, al no convencerme el tipo de material de [9], decidí buscar otras opciones. Por ello, investigué sobre trabajos ya realizados relacionados sobre el tema “*ball on plate*”, por lo que me puede nutrir de ideas que podía aplicar a la construcción de mi plataforma, como bien se puede observar en [10] y [11]. Lo cierto es que al final seguí un poco mi propio instinto y desarrollé una estructura donde iba a emplear juntas universales que permitiesen el movimiento de la pantalla resistiva al moverse los servos, algo parecido a [10], pero con un formato un tanto más complejo en cuanto a la sujeción de los servos.

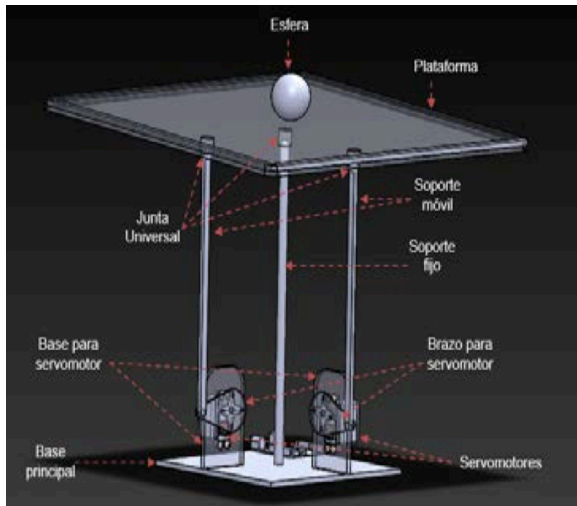


Figura 11: modelo con varillas 1 [FILE16]

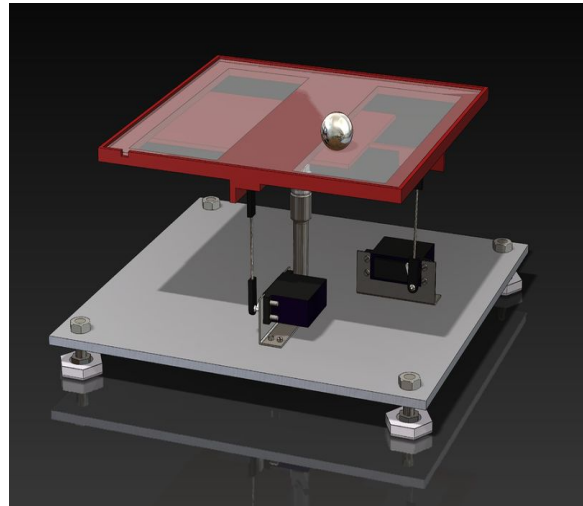


Figura 10: modelo con varillas 2 [DAVI17]

Finalmente, mi estructura se compone mayoritariamente de piezas construidas por impresión 3D, junto con una placa de metacrilato sobre la que se sustenta el sistema y otros elementos como juntas universales, varillas metálicas, tuercas y tornillos. Esto podemos verlo en [12], a partir de la cual voy a ir desgranando cada parte de la estructura elegida, entre las que podemos encontrar los brazos laterales (donde están situados los servos, los cuales mueven la plataforma en los ejes X e Y), el brazo central (que sirve de soporte para que no se desestabilice el sistema) y la bandeja sobre la que se apoya la pantalla resistiva, de grandes dimensiones.

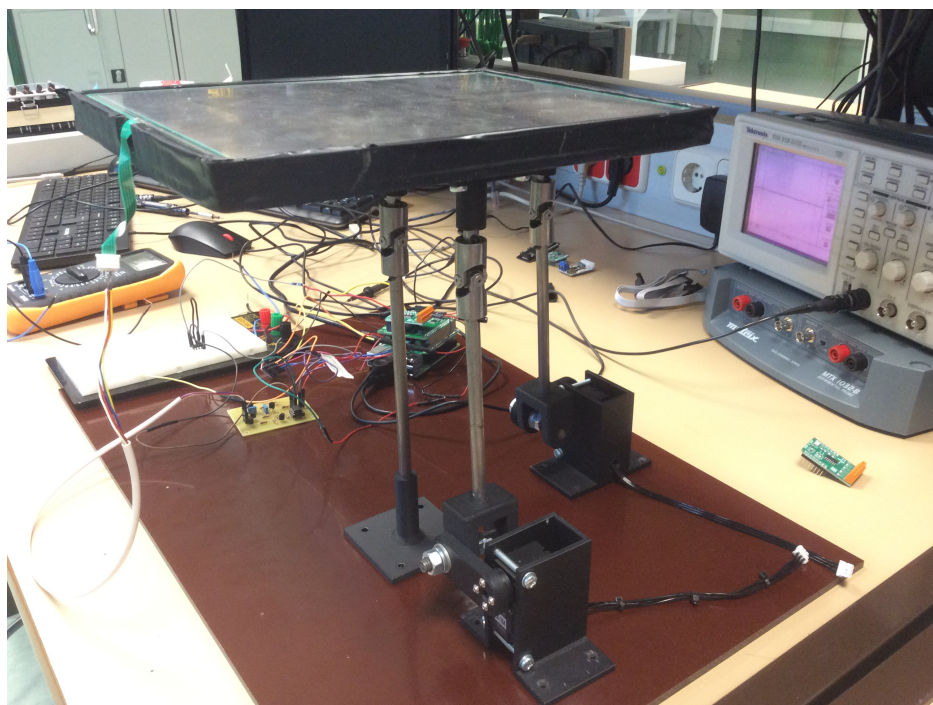


Figura 12: estructura original empleada

En cuanto a la impresión de las piezas, usé la impresora Stratasys del laboratorio de impresión 3D de ICAI con unas tolerancias de 1 mm y de 3 grados. Esta impresora utiliza plástico como material y tarda entre 4 y 7 horas en construir la pieza, según el tamaño de la misma. El proceso consiste en ir disparando filamentos de plástico a alta temperatura sobre un soporte hasta que se completa la figura a imprimir. Después, se deja reposar metiéndolo en ácido y se le quita el soporte.

Dependiendo del tamaño de las piezas se deben utilizar unas impresoras u otras. Por fortuna, no tuvimos problema gracias al excelente equipo de impresoras del laboratorio de impresión de ICAI, lo cual nos permitió trabajar de manera rápida y eficaz.

Como he dicho antes, mi estructura tiene varias partes, por lo que voy a empezar hablando de los brazos laterales, los cuales constan de una primera pieza que ejerce de apoyo para los servos. Como queremos fijarlos y que no deslicen en ningún momento, utilizamos unos tornillos largos para hacer presión sobre las paredes laterales de la pieza, como podemos ver en [13]. El modelo de diseño STL de este apoyo lo podemos ver en [14] y [15], el cual tuve que diseñar con extremo cuidado para no privar de libertad al servo. Así mismo, tuve que perforar la parte trasera de la pieza para dejar un hueco por el que saliese el conexionado de los servos y tuve que hacer cuatro agujeros en la base para los tornillos que hacen que esta pieza quedase anclada a la base de metacrilato del sistema.



Figura 13: tornillos para “aprisionar” al servo



Figura 15: vista en perspectiva de la pieza



Figura 14: vista frontal de la pieza

Tendremos dos piezas de estas, una para cada eje, puesto que utilizaremos dos servos para el control de ambos ejes X e Y. Los servos estarán colocados un poco por encima de la mitad de la pieza, como se puede observar en [16]. A cada uno de los servos le he unido, atornillándola, la siguiente pieza de

la que voy a hablar, cuyo formato STL se puede ver en [17] y [18]. Esta pieza conecta el servo con una varilla metálica secundaria roscada que tiene cada brazo lateral. Esta varilla secundaria unirá esta y otra pieza de la que voy a hablar más adelante (se puede ver en [20] y [21]).



Figura 16: vista posición de los servos

Se puede observar en [16] como el servo está bien sujeto debido a que las paredes de la pieza están ligeramente dobladas. Esto no supondrá ningún peligro ya que se trata de un material suficientemente resistente (plástico) como para aguantar este tipo de deformaciones elásticas.

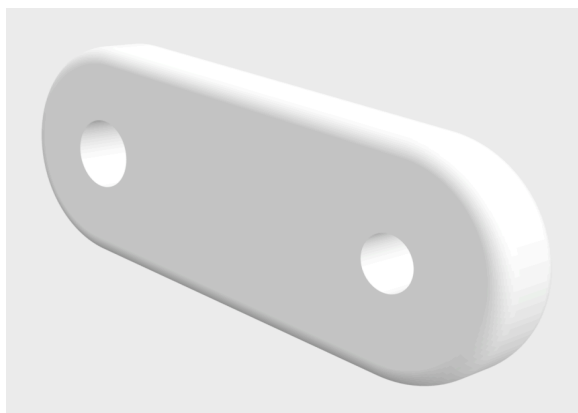


Figura 18: vista en perspectiva de la pieza



Figura 17: vista lateral de la pieza

Como podemos ver en [18], el agujero de la derecha es de menor diámetro que el de la izquierda. Esto se debe a que en la derecha es donde irá la unión al servo, mientras que en la izquierda va la varilla metálica, la cual es de 8 mm de diámetro. En el diseño en SolidEdge puse un diámetro de 7,9 mm, de forma que al encajarla esta estuviese bien anclada. Además, como no quería que hubiese deslizamiento entre la pieza y la varilla metálica roscada, coloqué a presión una tuerca con su correspondiente arandela que impedian dicho deslizamiento. Esto podemos verlo en [19].

También cabe destacar que tuve que rediseñar el modelo de CAD de la pieza varias veces cambiando el grosor de la misma debido al esfuerzo que iba a soportar durante el funcionamiento del sistema (recordar que la varillas son de acero, por lo que su peso es notable, al contrario que las varillas de aluminio). Además, achaflanar sus bordes también le daba mayor resistencia a la pieza.



Figura 19: vista de tuerca + arandela

Como he dicho antes, la varilla secundaria une las piezas de [18] y la de [20] y [21]. Esta última pieza es donde se colocan las varillas principales de los brazos laterales. Ocurrirá lo mismo que antes, donde al no querer que haya deslizamiento entre varilla y pieza, haré el diámetro de la misma de 7,9 mm, siendo la varilla de 8 mm. Esto podemos verlo en su formato STL de [20] y [21]. El diseño de esta pieza ha sido en vista al peso que iba a soportar, de ahí que haya puesto un doble agujero para un mayor soporte de la varilla secundaria roscada. Además, esto le ha dado a la pieza una mayor estabilidad.

Añadir también que sobre dichas varillas principales de cada brazo lateral irán colocadas sendas juntas universales que permiten el movimiento de la bandeja superior y, por ello, de la pantalla resistiva. Al principio estuve dudando si poner eso o unas rótulas que diesen de la misma forma libertad de

movimiento a la bandeja, pero debido a la complejidad de su diseño para la impresión en 3D, me decanté por las juntas universales, aunque fuesen menos económicas.

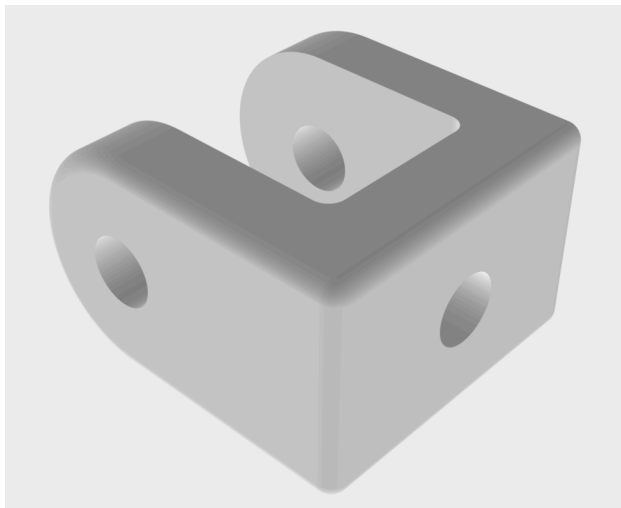


Figura 21: vista en perspectiva de la pieza



Figura 20: vista frontal de la pieza

Por tanto, sobre el extremo inferior de la junta universal irá la varilla principal, mientras que sobre el otro extremo irá un trozo de varilla más pequeña que conectará con la última pieza de los brazos laterales. Esta pieza no es más que un conector con la bandeja de arriba donde va apoyada la pantalla resistiva. Por ello, al diseño en STL, que podemos ver en [22] y [23], tuve que hacerle 4 penetraciones por donde pasan los tornillos que unen la pieza con la bandeja. Esto podemos verlo en [24].

La base de esta pieza no será muy grande (menos de la mitad de la bandeja) pues queremos que su apoyo controle la dirección del eje. De lo contrario, ocuparía mucho (más de la mitad) y, al moverse el servo, se movería prácticamente toda la bandeja, no solo el eje.

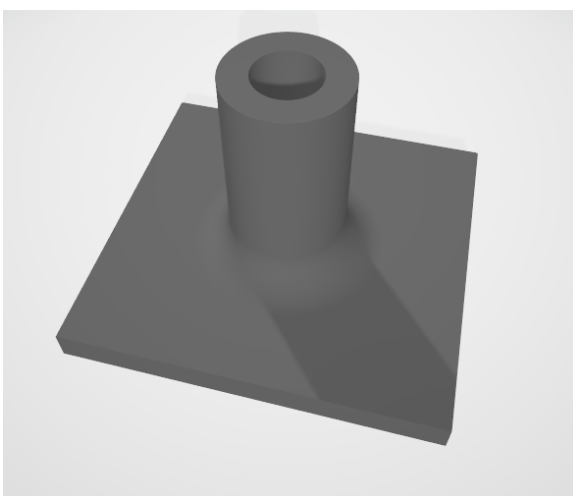


Figura 23: vista en perspectiva de la pieza



Figura 22: vista aérea de la pieza

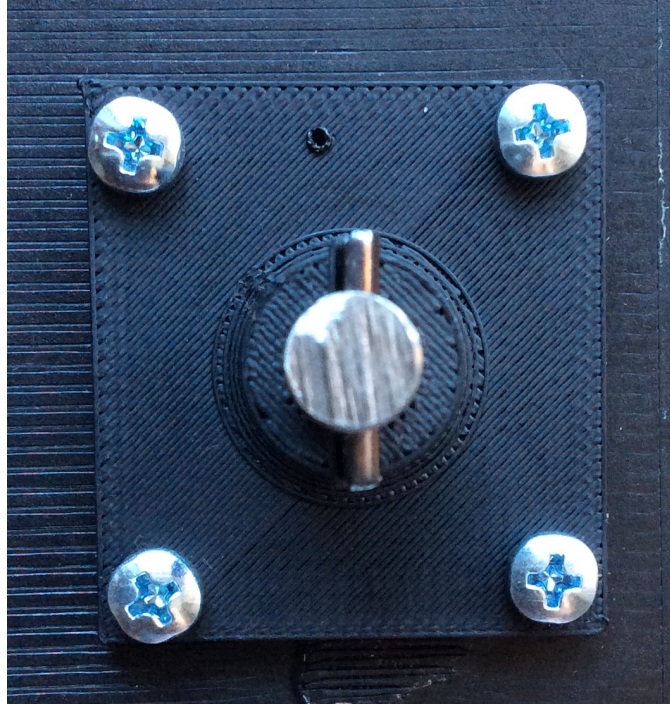


Figura 24: vista de los agujeros con sus respectivos tornillos

Cabe destacar que esta última pieza también la encontramos en la siguiente parte de la estructura, el brazo central, el cual se encarga de dar apoyo al sistema. Esta parte también constará de dos varillas metálicas (una pequeña y otra más grande, como en los brazos laterales) y de una junta universal mediante la que se unen dichas varillas. Por último, tiene una pieza en la parte de abajo que se encarga de unir la varilla central grande con la base del sistema (la placa de metacrilato), como podemos ver en la versión STL de [25] y [26]. Esta pieza es muy parecida a [22], puesto que también es necesario realizar cuatro agujeros mediante los cuales pasan los tornillos. Sin embargo, la base de la misma es mayor que en el caso anterior, ya que el esfuerzo al que se va a someter también será mayor.

En un principio pensé en prescindir de esta parte de la estructura, para hacer el trabajo menos costoso. Sin embargo, me di cuenta que, solo con los brazos laterales, el sistema se volvía un tanto inestable.

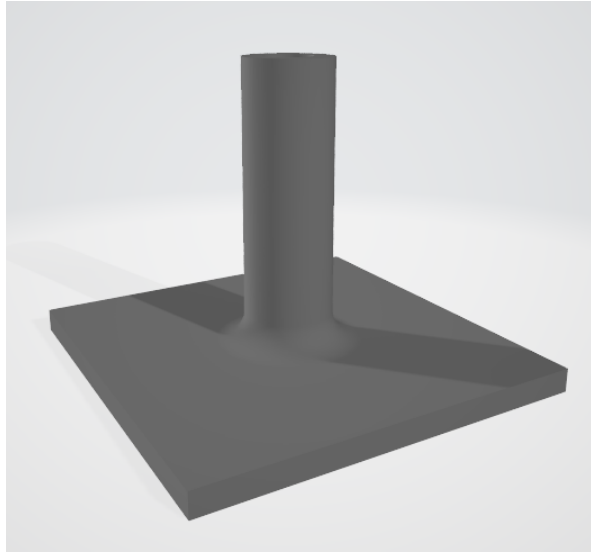


Figura 26: vista en perspectiva de la pieza

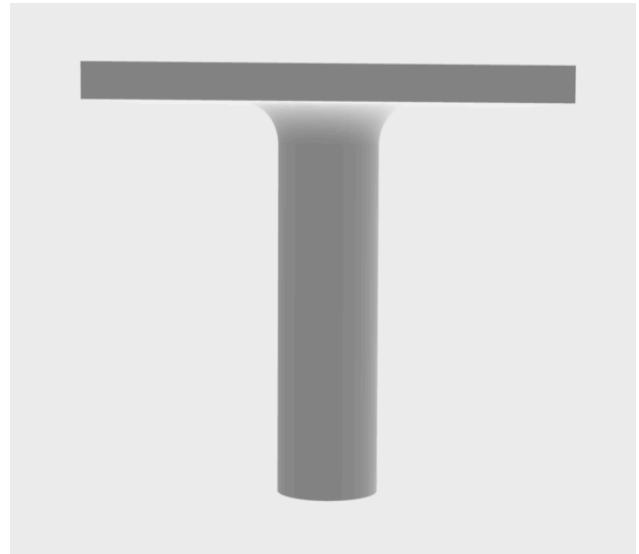


Figura 25: vista lateral de la pieza

La última parte de la estructura se trata de la bandeja, también impresa en 3D, sobre la que se sustenta la pantalla resistiva. Esta, al ser de gran tamaño, tardó bastante más en imprimirse que el resto y salió deformada, por lo que tuve que hacer unos retoques. Además, debido a un fallo en las medidas de diseño, tuve que lijar las paredes interiores con el objetivo de aumentar su superficie interior. Al lijarla había que tener mucho cuidado de que las paredes quedasen totalmente simétricas para que al poner la bandeja esta no estuviese en desnivel, lo que haría que la bola se desplazase de la posición inicial, cosa que no queríamos. Para facilitar esto, en el diseño con SolidEdge le puse unos topes (en concreto, 6) sobre los que se iba a apoyar la bandeja. Esto podemos ver su versión STL en [27] y [28].

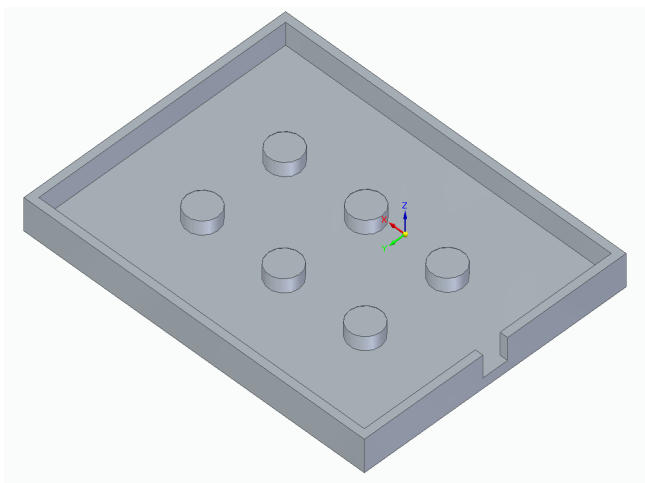


Figura 28: vista en perspectiva de la bandeja

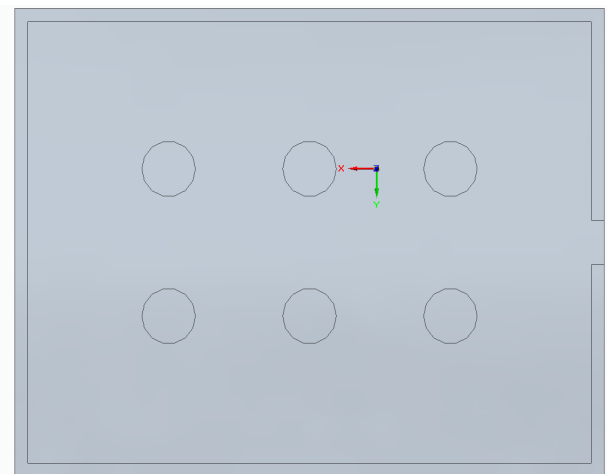


Figura 27: vista aérea de la bandeja

La bandeja está anclada a los brazos laterales y al brazo central mediante la pieza de [22] y [23] y una serie de tornillos y arandelas. Esto podemos verlo en [29] y [30], donde se muestra la parte inferior y superior de la bandeja.

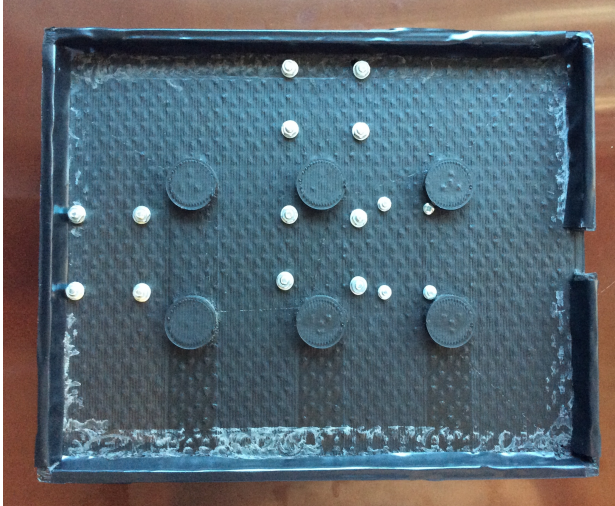


Figura 30: vista superior de la bandeja

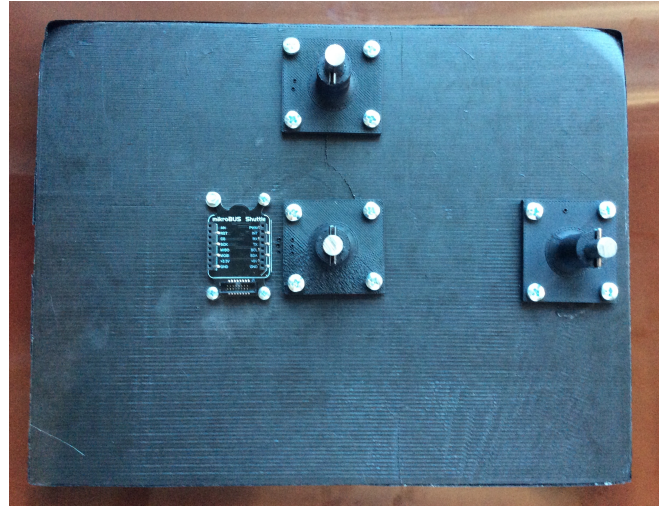


Figura 29: vista inferior de la bandeja

Como podemos ver en [29], los bordes interiores están desgastados. Esto es debido a lo que he comentado antes de que tuve que lijarlos para ampliar la superficie interior, al haber habido un fallo a la hora del diseño. Además, el modelo de la bandeja con la pantalla incorporada lo podemos ver en [31].

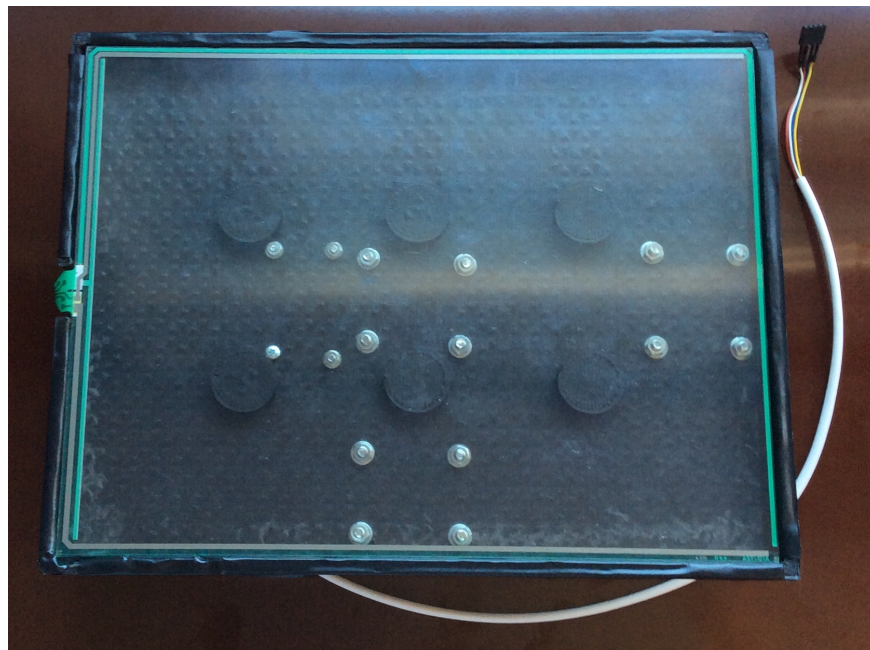


Figura 31: vista bandeja + pantalla

Otra observación es el hueco que he dejado en la parte izquierda de la pieza de [31] para que pueda salir el conexionado de la pantalla.

Para cerrar el apartado de la estructura, me gustaría comentar unos breves aspectos de los elementos utilizados, más allá de las propias piezas. Por tanto, y como he comentado anteriormente, hemos empleado juntas universales, las cuales eran de diámetro 8 mm, al igual que las varillas metálicas. La unión de las juntas y las varillas podemos verla en [32]. Como ambos eran del mismo diámetro, encajaban, pero bailaban un poco, cosa que yo no quería que pasase. Por tanto, para mantener la junta totalmente fija a las varillas, usé el pegamento fuerte LOCTITE.

Además, para fijar los brazos a la bandeja al principio pensé en una especie de ventosas, las cuales se compraron y se probaron, pero, antes la falta de consistencia, decidí no utilizarlas y optar por la pieza [22] y [23]. Esto es debido a que las ventosas eran demasiado pequeñas como para mantener pegado la bandeja y los 3 brazos.



Figura 32: unión junta y varillas

Por último, como podemos ver en [32], para encajar las juntas puse unos toques en las varillas perforándolas y poniendo pasadores. Había dos pasadores por cada junta universal. Esto evitaba que las varillas privasen de movimiento a la junta.

1.5.2. Sensores

A parte de los servos, los cuales son actuadores, nuestro sistema tiene una serie de sensores los cuales debemos montar, modificar y poner en lugares específicos de nuestra estructura. Estos sensores son: la pantalla resistiva, el conversor ADC y la IMU. En este apartado, no voy a explicar su funcionamiento ni como obtenemos las medidas con ellos, sino simplemente como los he montado y donde los he colocado en nuestro sistema.

Por un lado, la pantalla resistiva irá en la parte de arriba de la estructura, apoyada sobre la bandeja que hemos podido ver antes en [31]. Además, esta pantalla tiene un circuito auxiliar formado por resistencias, transistores, condensadores y dos amplificadores operacionales *buffer*, que hace que esta funcione, el cual podemos ver en [33], donde falta uno de los *buffer*, el cual estará situado en una placa protoboard. Este circuito estará localizado en la parte de abajo, es decir, en la base de metacrilato.

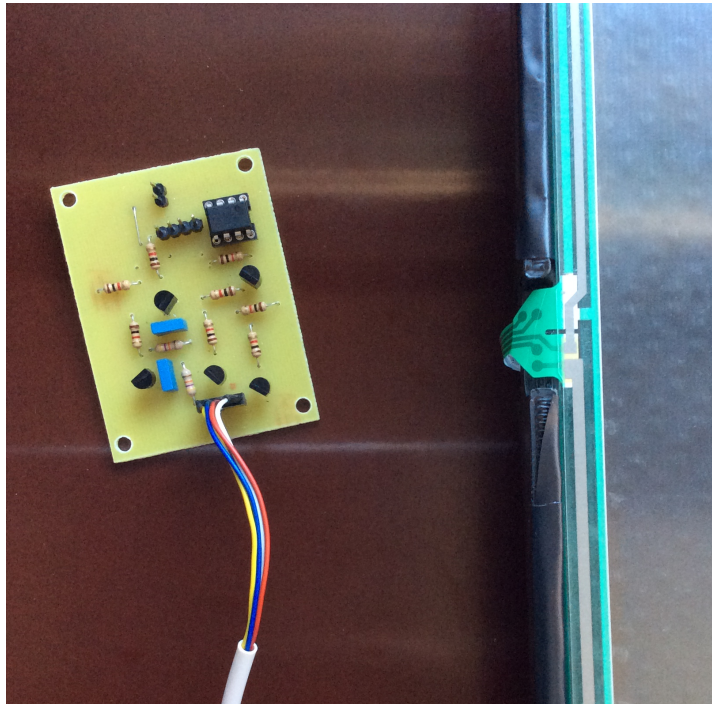


Figura 33: circuito auxiliar de la pantalla

Este circuito, como veremos en los próximos apartados, tuvo que ser diseñado y luego tuve que aplicar ese diseño a la realidad, creando las pistas adecuadas y soldando los elementos necesarios para su correcto funcionamiento. Los 2 pines juntos son las alimentaciones de 5V y tierra, mientras que los 4 pines juntos son los DRIVES y las salidas, cuyos datos se tomarán de la forma que explicaré en el siguiente apartado.

El siguiente sensor empleado es el conversor ADC, el cual está conectado a la Raspberry Pi a través de una placa *click shield*, como podemos observar en [34]. Además, a él irán las salidas del circuito auxiliar visto en [33], para que este las convierta a digital y la información pueda procesarse en la Raspberry. Para poder ajustarlo a mi trabajo, tuve que cambiar el Power Select de 3,3V a 5V. Con el sistema montado y al hacer las pruebas pertinentes, vimos como el conversor ADC saturaba, ya que este estaba limitado a 2 voltios por defecto. Para remediarlo, pusimos un par de divisores de tensión con el fin de que dejase de saturar, como podemos ver en [35]. Estos están compuestos de resistencias soldadas a cables macho-macho. Esto lo explicaré en el próximo apartado cuando hable de su funcionamiento.

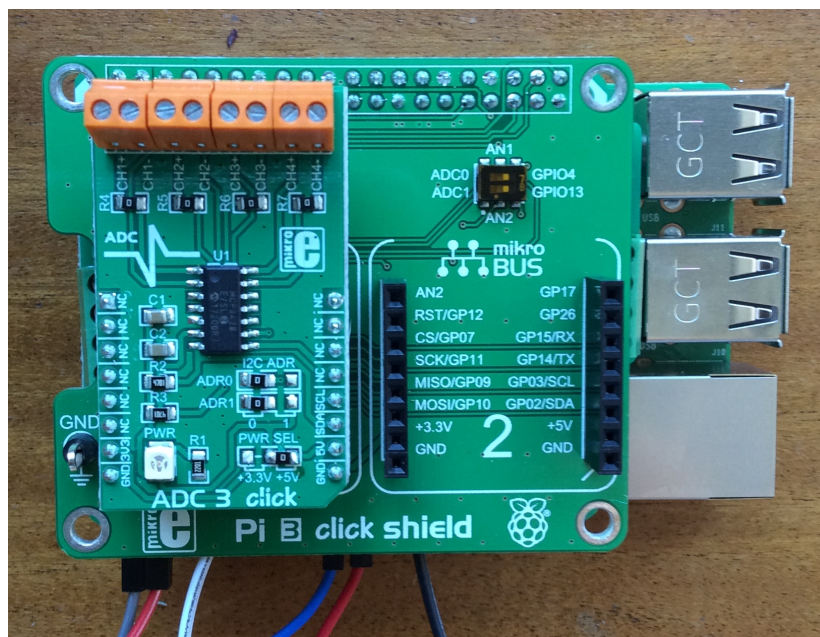


Figura 34: localización del conversor ADC

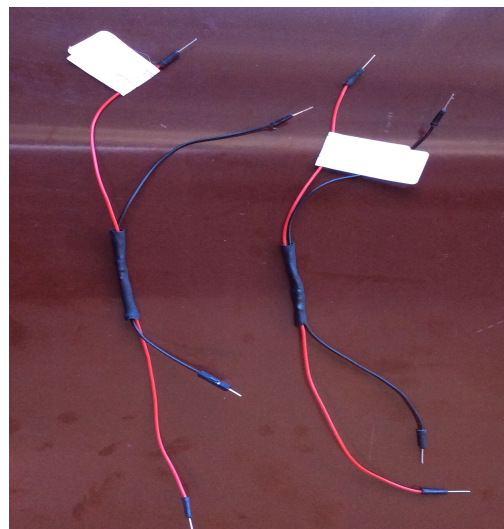


Figura 35: divisores de tensión

Por último, tenemos la IMU, que está ubicada en la parte trasera de la bandeja sobre la que se apoya la pantalla resistiva, como podemos observar en [36]. Esta está anclada a la bandeja mediante 4 tornillos que la mantienen totalmente fija.

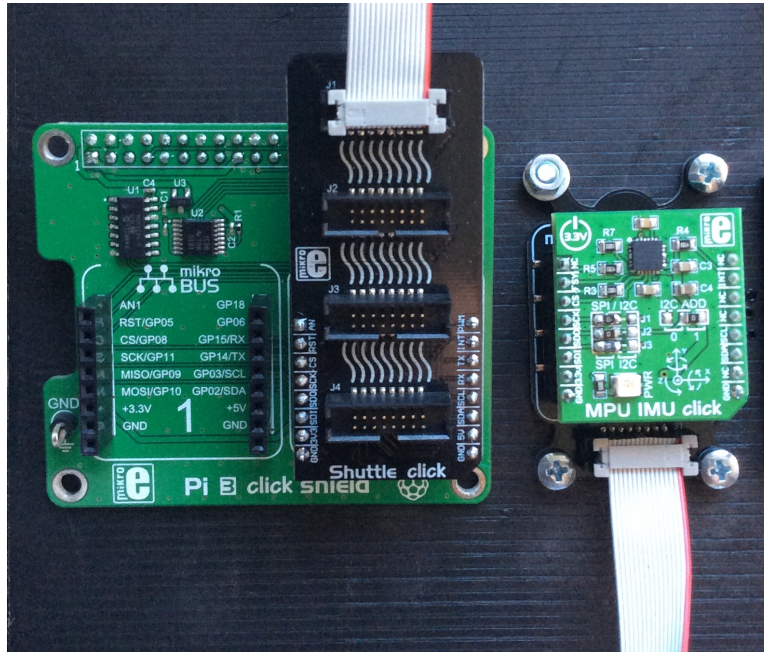


Figura 36: localización de la IMU

La IMU, que sirve para saber en todo momento la inclinación de la pantalla resistiva en ambos ejes X e Y, puede ir en cualquier parte de la bandeja, pero lo óptimo es ponerla en la parte central. Además, la IMU va conectada a la Raspberry, en concreto al puerto 2 del *click shield*.

Por último, debemos quitar las micro resistencias de I2C y soldarlas de nuevo en SPI, que será la forma en la que la IMU se comunicará con la Raspberry Pi.

1.5.3. Actuadores

Como actuadores, inicialmente empleamos dos servos LX-16A, también llamados “servos inteligentes”. Estos servos son los encargados de mover la estructura construida. En cuanto a su posición, como hemos comentado anteriormente, los servos estaban colocados en las piezas [14] y [15]. Como queremos que no se desplacen verticalmente, pusimos dos tornillos para apretar las paredes de dicha pieza, como podemos observar en [13].

Estuvimos intentando un tiempo hacer funcionar los servos inteligentes, pero no pudimos, por lo que cambiamos a unos servos más sencillos, los Futaba S3003. Para alimentarlos, usamos una fuente de alimentación de 6V.

1.6. FUNCIONAMIENTO DE SENSORES Y OBTENCIÓN DE MEDIDAS

Como he comentado anteriormente, vamos a utilizar 3 sensores en nuestro sistema. En este apartado vamos a explicar la forma de actuar de cada uno ellos y cómo podemos obtener la información que pasará a la Raspberry para un futuro control.

1.6.1. Pantalla resistiva y circuito de adaptación

Este primer sensor consta de una pantalla resistiva junto con un circuito de adaptación que le permite captar la información que le llega. A la hora de elegir el tipo de pantalla lo cierto es que tuve ciertas dudas ya que había otra opción disponible, la pantalla capacitiva.

Por una parte, las pantallas resistivas son paneles que detectan la pulsación de un objeto que tenga un mínimo peso. Son más lentas que las otras, pero menos costosas y más resistentes. Por otro lado, las pantallas capacitivas no necesitan de un objeto con peso, sino simplemente necesitan que este tenga capacidad. Su manejo es más complejo puesto que son más precisas y rápidas, pero son menos resistentes y más costosas. Finalmente, me decanté por el uso de una pantalla resistiva, ya que iba a emplear una bola de acero como objeto a poner sobre la pantalla.

Por tanto, mi pantalla resistiva consta de dos láminas rígidas transparentes, formando una estructura “sándwich”, con una capa resistiva en sus caras internas, como podemos observar en [37].

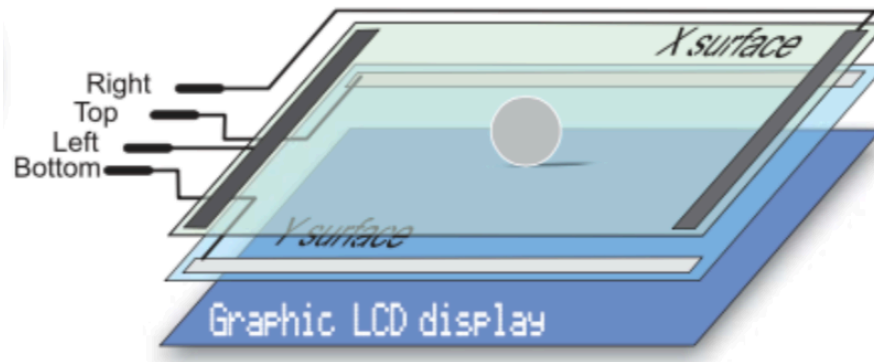


Figura 37: estructura interna de la pantalla

Las resistencias de estas capas normalmente no son mayores de 1KOhm. El proceso para leer las coordenadas de cada eje de la pantalla se hace por separado, primero el eje X y luego el Y, o viceversa. Por ello, cuando ponemos un objeto en un punto de la pantalla hay un divisor de tensión entre la resistencia de lado a lado y la resistencia del punto de presión. Este divisor de tensión nos dará la tensión que estamos buscando, que corresponderá con la coordenada del objeto en dicho eje.

Para la lectura del eje X, se alimenta con cierta tensión positiva un lado de la capa correspondiente y el otro se pone a 0V. Para medir se usan los dos lados de la otra capa, los cuales irán conectados al convertidor ADC, como veremos más adelante. Para la lectura del eje Y ocurre lo mismo, pero cambiando las capas. Este procedimiento podemos verlo en [38], donde podemos observar como para medir la coordenada Y, se ponen a Vcc y a tierra los lados de la capa Y. De la misma forma, para medir la coordenada X, se utilizaría la capa X para poner tensiones. Lo normal es medir primero la coordenada X y luego la Y, pero debe hacerse de forma simultánea y periódica.

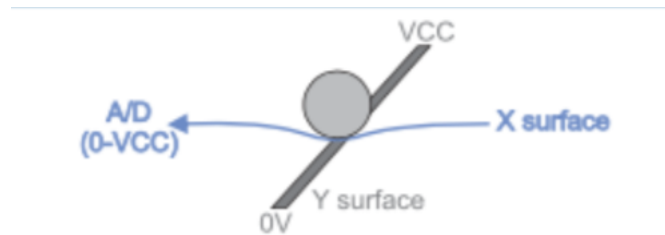


Figura 38: lectura de la coordenada del eje Y

La información de las coordenadas pasa luego al microcontrolador utilizado, en este caso la Raspberry Pi, a través del circuito de adaptación de la pantalla que podemos ver en [39] y del conversor ADC, como veremos más adelante.

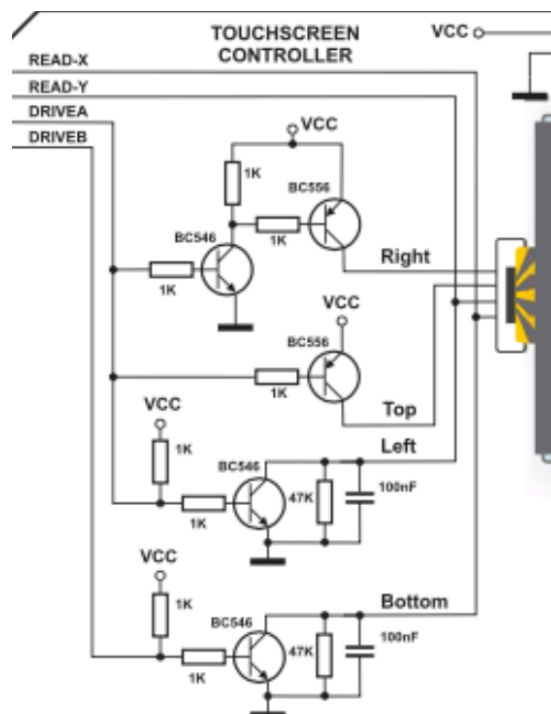


Figura 39: circuito de adaptación inicial

La idea es ir mandando información a la Raspberry de las coordenadas de la bola cada 20 ms. En primer lugar, 10 ms para medir la coordenada del eje X. Para ello, pondremos 5V en el DRIVEA y 0V en el DRIVEB y medimos. Instantáneamente después, cambiamos el DRIVEB a 5V y el DRIVEA a 0V y medimos la coordenada del eje Y. Esto último serían otros 10 ms. Se repetirá este proceso de manera periódica e indefinida mientras la bola permanezca en la pantalla resistiva, ya que esta información es la que utilizará la Raspberry para ir moviendo los servos según se requiera.

Para mostrar por pantalla la tensión en cada eje, utilizaremos scopes en el Simulink correspondiente. Como he dicho en el apartado anterior, tuve que realizar el diseño de la PCB del circuito de adaptación, como podemos ver en [40]. Para ello, utilice la herramienta libre KiCad, para después fabricar la placa con sus pistas correspondientes y soldar los elementos necesarios, obteniendo el circuito de [41] y [42], a falta del *buffer* del DRIVEB.

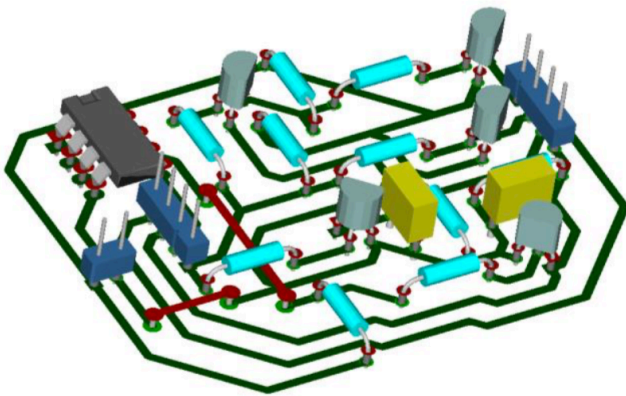


Figura 41: diseño de la PCB con KiCad

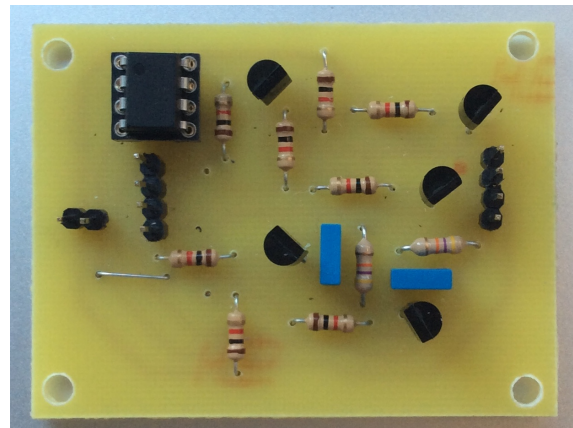


Figura 40: parte superior del circuito

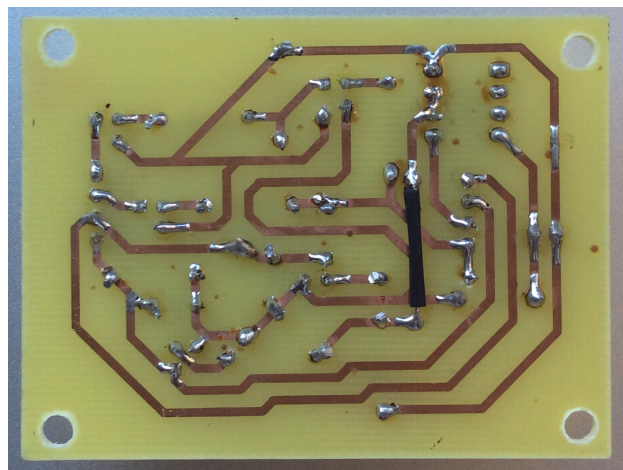


Figura 42: parte inferior del circuito

Para realizar las pruebas de funcionamiento de la pantalla por separado, basta con conectar la pantalla al circuito de adaptación y dar tensión al circuito con la Raspberry Pi. Por ejemplo, para medir la coordenada del eje X de [43], pongo en el DRIVEA 5V y en el DRIVEB 0V, por lo que obtengo, para la bola en esa posición, 2,57 V. Si muevo la bola hacia la derecha como en [44], obtengo 4,59 V.

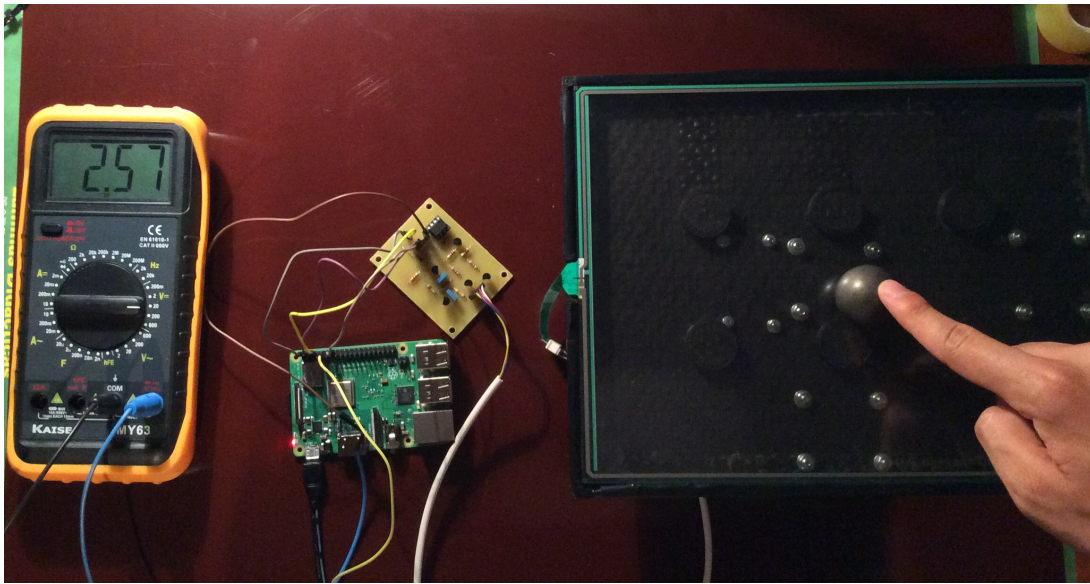


Figura 43: coordenada eje X con la bola a mitad de la pantalla

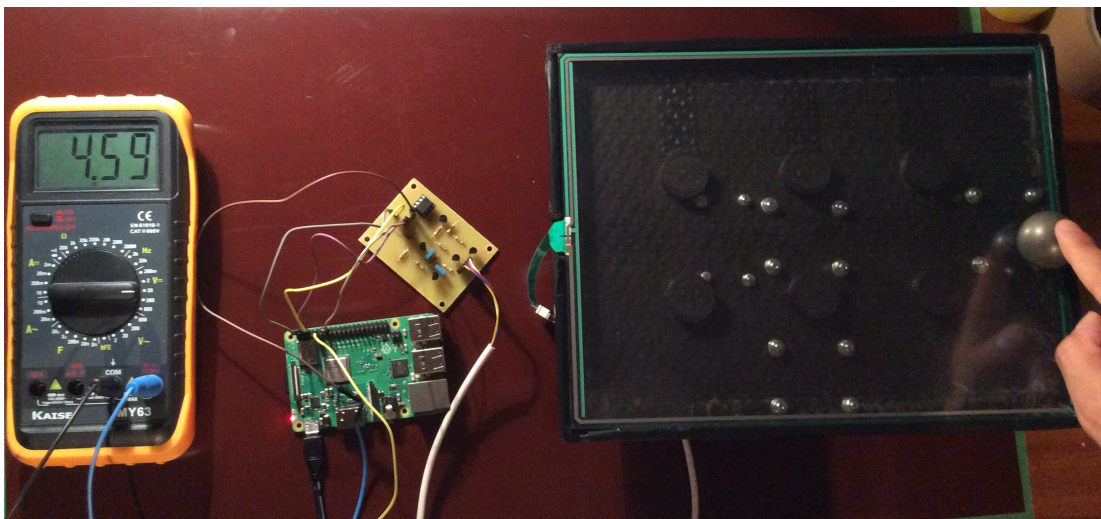


Figura 44: coordenada eje X con la bola en el extremo derecho de la pantalla

Como podemos observar, la tensión del eje X aumenta conforme nos vamos desplazando hacia la derecha.

Ahora, hacemos lo mismo para el eje Y cambiando las cosas pertinentes para su lectura. Si pongo la bola en el medio, obtendremos 2,05V, como aparece en [45], mientras que, si muevo la bola hacia la derecha, obtenemos 4,02V, como podemos ver en [46].

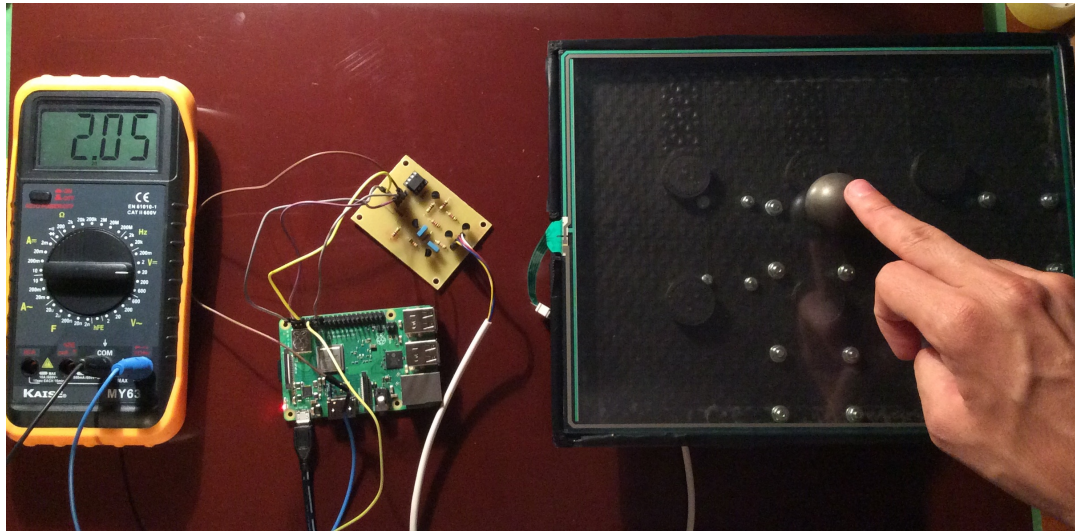


Figura 45: coordenada eje Y con la bola antes de la mitad de la pantalla

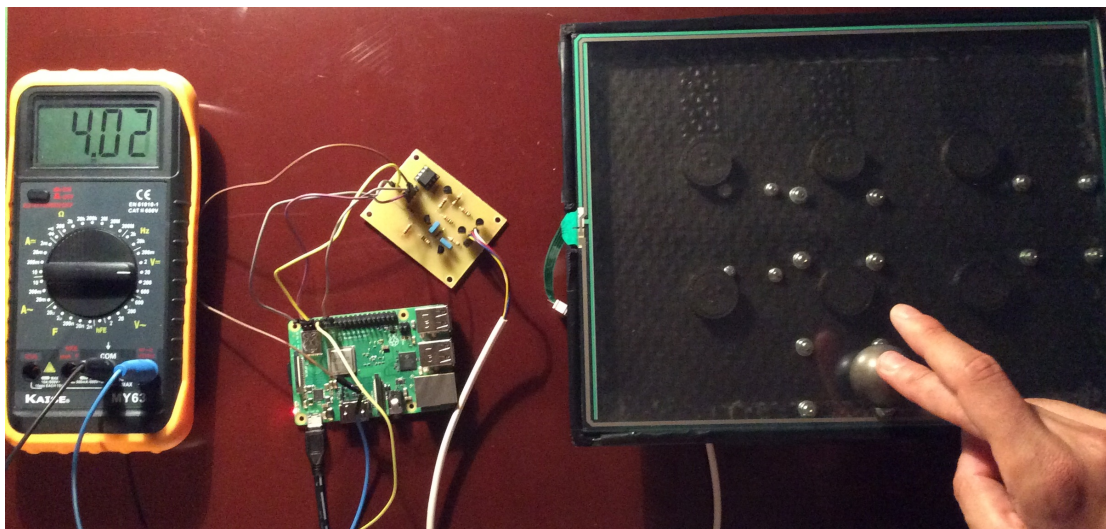


Figura 46: coordenada eje Y con la bola en la parte de abajo

Como podemos observar, la tensión del eje Y aumenta conforme nos vamos desplazando hacia abajo. Estas coordenadas de cada eje, dadas en forma de tensión, no pueden pasar directamente a la Raspberry, ya que son datos analógicos. Por ello, necesitamos un conversor ADC que lo convierta a

digital para que la Raspberry pueda recibir y procesar esa información. Esto me lleva a la explicación del segundo sensor empleado.

1.6.2. Conversor ADC

Como he dicho antes, los conversores ADC transforman las señales analógicas en digitales para que puedan ser procesadas por los microprocesadores correspondientes. Hay distintos tipos de conversores en función de su funcionamiento: conversor flash, el cual es muy rápido, pero sin embargo es demasiado grande y presenta un consumo excesivo. El segundo es el conversor por aproximaciones sucesivas, el cual es más compacto y apenas consume, pero es de velocidad moderada. Por último, se encuentra el conversor de rampa doble, el cual es muy preciso y muy lineal, pero puede llegar a ser muy lento, por lo que no interesa del todo.

Por tanto, la información que sale del circuito de adaptación pasa al conversor ADC. Sin embargo, este conversor satura a 2V según las limitaciones de diseño, por lo que no podremos llegar a los 5V que da la pantalla. Para subsanar este problema, diseñamos un divisor de tensión de forma que multiplicamos la tensión de salida por $\frac{2}{5}$. Para ello, tenemos dos opciones, utilizar resistencias relativamente pequeñas cercanas a 1-2 KOhm o utilizar resistencias grandes de 10-20KOhm. Para ello, debemos ver si la resistencia de salida de la pantalla afecta al divisor de tensión, de donde sacamos la siguiente ecuación:

$$3,8 = 4,6 * \frac{5}{5 + R_o}$$

De esta forma, tenemos que la resistencia de salida del circuito de adaptación + pantalla es de 1,05 KOhm, por lo que debemos coger para el divisor de tensión resistencias del orden de 20KOhm. Por tanto, nos quedará el divisor de tensión de [47] con resistencias de 20 y 30 KOhm y con la ganancia deseada.

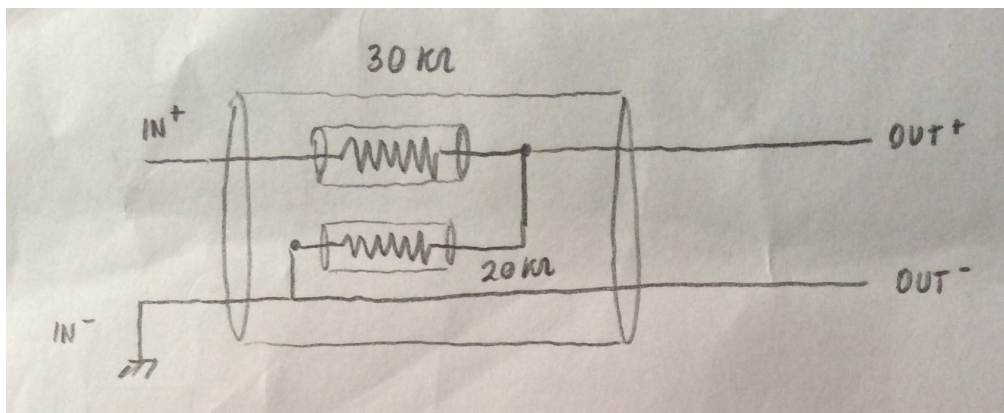


Figura 47: diseño interno del divisor de tensión

Las entradas IN vienen del circuito de adaptación y las salidas OUT van al conversor ADC, el cual estará conectado a la Raspberry Pi. Además, las resistencias deben ir cubiertas de material termorretráctil, para evitar el contacto entre ellas.

Ahora probamos el mismo caso que [43], pero con el divisor de tensión, obteniendo una tensión en el eje X de 0,98 V, como podemos observar en [48].

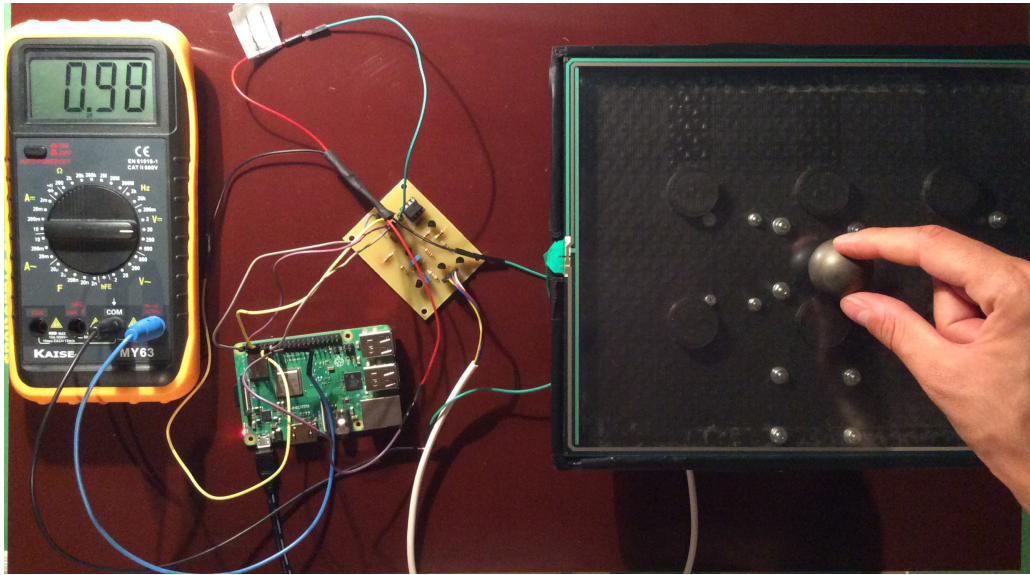


Figura 48: coordenada en X con divisor de tensión

Como podemos observar comparando ambos casos de [43] y [48] es que se cumple la ganancia propuesta para el divisor de tensión, pasando de 2,57 a 0,98 V para la misma posición de la bola en el eje X.

Una vez configurado el divisor de tensión, conectamos el conversor ADC y ya podremos obtener los valores de tensión en los scopes que hay en el diagrama de bloques correspondiente al conversor. Al conectarlo, nos quedará de la forma que aparece en [49].

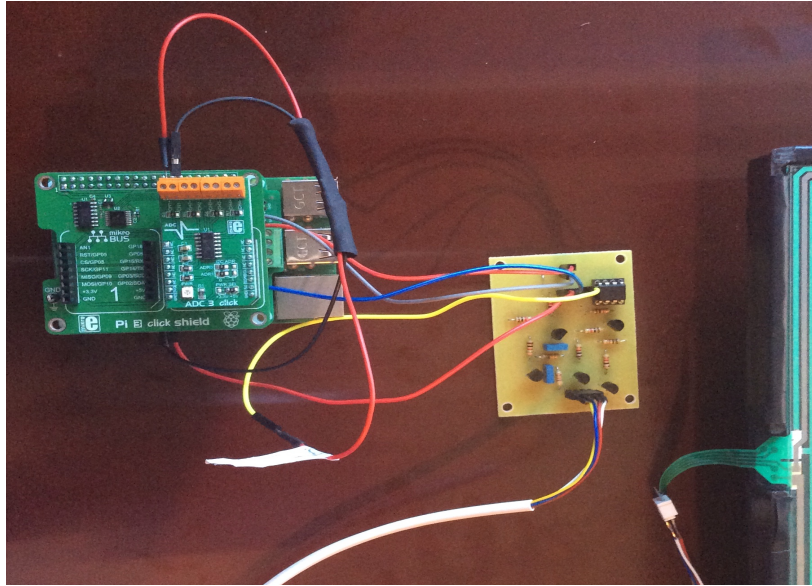


Figura 49: conexión conversor ADC con divisor de tensión al circuito de adaptación

Ahora que lo tenemos conectado, vamos a hacer las pruebas de funcionamiento, poniendo la esfera en la misma posición que en [48]. Hay que tener en cuenta que ahora solo tenemos conectado el eje X, por lo que el eje Y marcará 0V. Los resultados podemos verlos en [50] y [51].

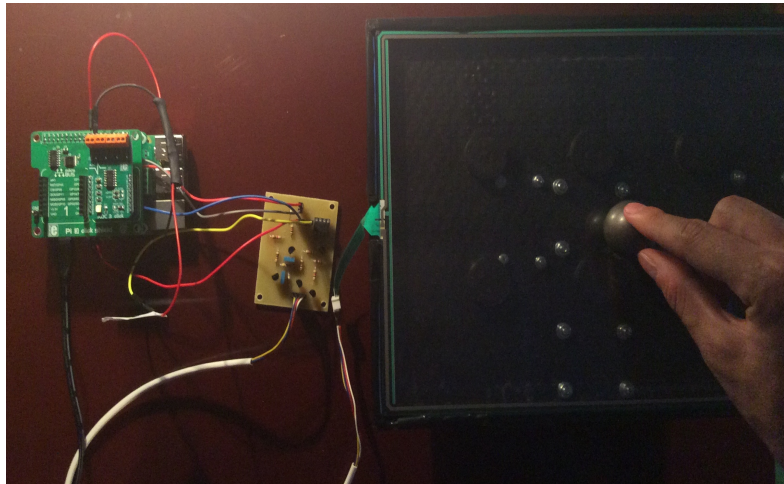


Figura 50: posición de la bola con el conversor ADC

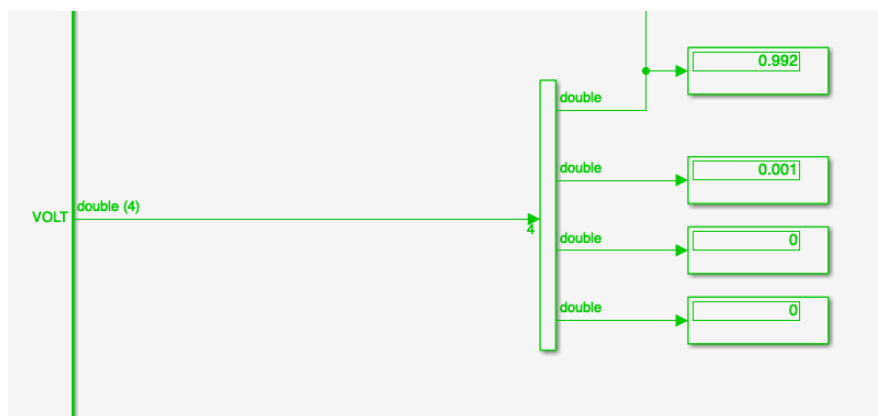


Figura 51: scopes de los canales mostrando los resultados 1

Como podemos observar en [51], el primer scope es el que se corresponde con la lectura del eje X, mientras que el segundo scope es el del eje Y. Los otros dos canales no los tengo configurados porque no nos son útiles. Por tanto, obtenemos una medida de 0,992 V, prácticamente igual que en el caso de [48].

Ahora hacemos lo mismo, pero con el eje Y. Para ello, alternamos las tensiones del DRIVEA y del DRIVEB y conectamos el divisor de tensión entre la lectura del eje Y y el canal 2 del conversor ADC. Ponemos la esfera en el mismo lugar que en [50] y obtenemos el resultado de [52].

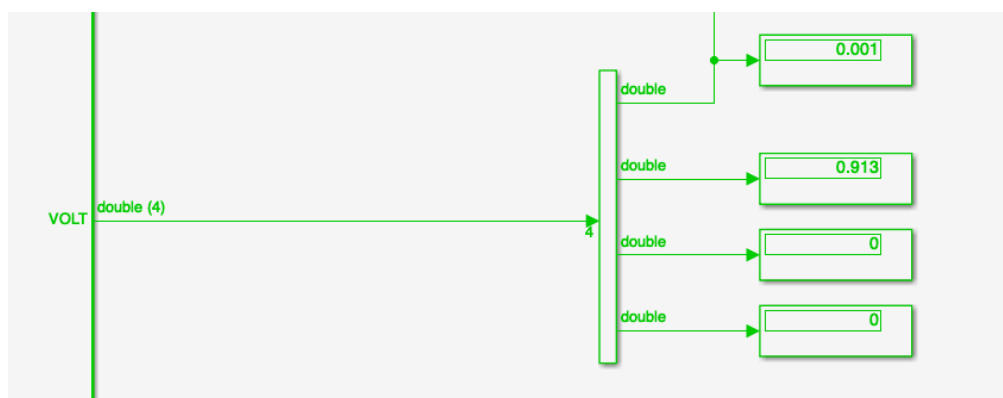


Figura 52: scopes de los canales mostrando los resultados 2

Como podemos observar, esta vez el que marca es el canal 2, el cual da una medida de 0,913V, que cuadra con la posición en la que he puesto la esfera.

Ahora, el último paso es conseguir que nos midan los dos canales (correspondientes a las lecturas de los ejes X e Y) a la vez. Estas lecturas deben ser continuas y periódicas, por lo que debemos estar constantemente alternando de tensión los DRIVEA y DRIVEB para ir leyendo las coordenadas en

ambos ejes X e Y. Por ello, debemos utilizar 2 GPIO de la Raspberry e ir alternando tensiones en ellos cada 10 ms, ya que queremos la medida de cada eje cada ese período de tiempo. Además, como la Raspberry solo da 3,3V en la parte de los GPIO, debemos utilizar un Level Converter, el cual pasa de 3,3 V a 5V en dichos GPIO. Por tanto, el Level Converter irá conectado por una parte a los DRIVEA y DRIVEB del circuito de adaptación y por otra a 2 GPIO de la Raspberry, que en este caso serán el GPIO 21 y el GPIO 20. La conexión del Level Converter podemos verla en [53].



Figura 53: conexión del Level Converter

En [53], por el lado de alto voltaje (HV) hay 4 conexiones: 5V, tierra y luego HV4 va conectado al DRIVEA y HV1 al DRIVEB. De la misma forma, en el lado de baja tensión (LV) hay otras 4 conexiones: 3,3V, tierra y luego LV4 va conectado al GPIO21 y el LV1 al GPIO20.

A la hora de probar el Level Converter para asegurarnos que transformaba los 3,3V procedentes de la Raspberry a los 5V que recibía el circuito de adaptación, nos dimos cuenta que el DRIVEB estaba mal diseñado internamente. Faltaba un amplificador operacional “buffer” con impedancia infinita que lo aislase de la resistencia de salida de la pantalla, la cual producía un divisor de tensión y, en lugar de pasar 5V al lado de alta tensión, apenas pasaban 3,5V. Como podemos ver en [39], el DRIVEA sí lo tenía, por lo que funcionaba correctamente. Por ello, diseñamos el nuevo circuito de adaptación incluyendo el operacional del DRIVEB, obteniendo [54].

Además, la forma de alternar las tensiones en los DRIVES se realiza mediante el Simulink asociado al conversor ADC, el cual está entero en el anexo. En él, cada 10 ms, se alternan las tensiones en los DRIVES al mismo tiempo que se va cambiando el canal activo del conversor ADC de 1 a 2 y de 2 a 1, sucesivamente.

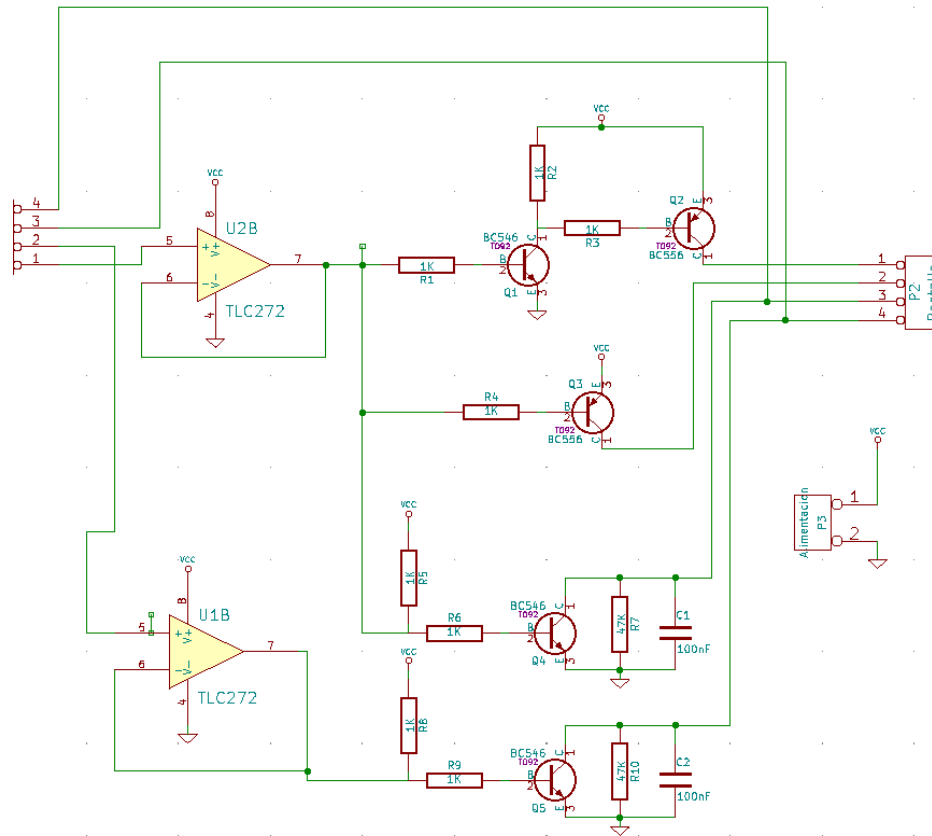


Figura 54: nuevo diseño del circuito de adaptación

Una vez tenemos todo conectado, y con el nuevo circuito de adaptación de la pantalla, realizamos la prueba para la esfera en la misma posición que [50], con el objetivo de obtener las tensiones de ambos ejes. Una vez lo probamos, obtenemos los resultados de [55].

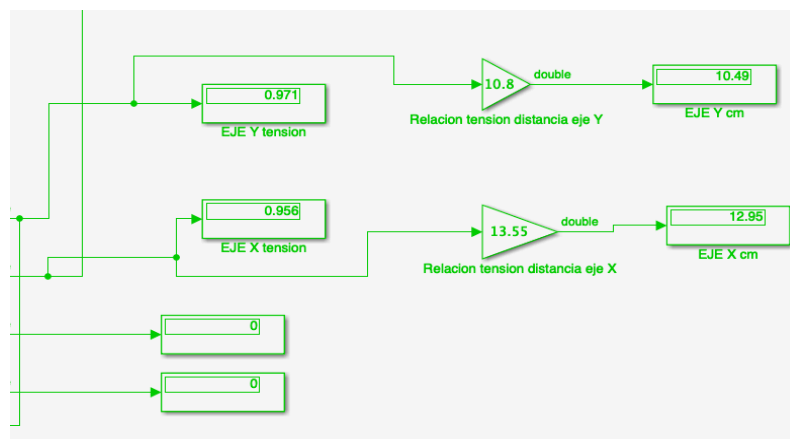


Figura 55: scopes de los canales mostrando los resultados 3

1.6.3. IMU (Inertial measurement unit)

Este último sensor empleado se trata de una unidad electrónica que mide la velocidad angular y aceleración de un objeto mediante acelerómetros y giroscopios. Con el acelerómetro medimos la aceleración en los 3 ejes mientras que con el giroscopio medimos la velocidad angular de esos ejes. Tiene múltiples usos, desde robots hasta satélites pasando por drones, cuya función en estos últimos es clave. Un dron no podría volar sin una serie de IMU que lo vayan “guiando” en todo momento. En nuestro caso, es muy útil para saber la inclinación de la pantalla en cada uno de los ejes. Son aparatos muy sensibles, por lo que hay que tener mucho cuidado, puesto que podría suponer un problema para el futuro control.

Por tanto, la IMU es capaz de captar cambios de posición angular en cada uno de los ejes (nos interesan los ejes X e Y) y de pasar la información a la Raspberry. Para ello utiliza el método de comunicación SPI (Serial Peripheral Interphase), comunicación en serie.

Como comentamos anteriormente, la IMU estará colocada en la parte inferior de la bandeja, con el fin de captar sus variaciones angulares en ambos ejes. Su conexión física con la Raspberry se realiza mediante un cable cinta como podemos ver en [56]. Como la Raspberry está colocada en la placa de metacrilato, esta estará bastante alejada de la bandeja, por lo que necesitaremos un cable cinta relativamente amplio.

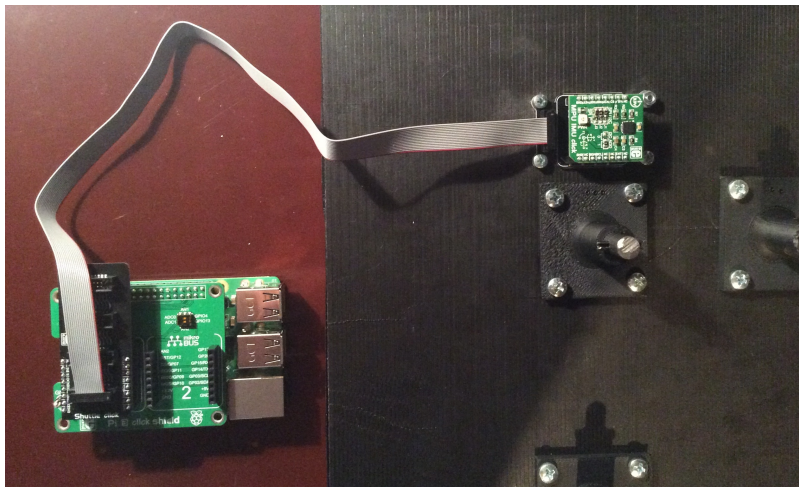


Figura 56: conexión entre IMU y Raspberry

Utilizaremos Simulink para inicializar la IMU y mediremos las variaciones de los ángulos mediante los scopes de los diagramas de bloques correspondientes, que podremos encontrar en los anexos correspondientes. Esto podemos verlo en [57], donde partiendo de la posición inicial (con la bandeja

totalmente plana), giramos la bandeja repetidamente unos 20° hacia la derecha respecto del eje X y luego la volvemos a poner en posición plana.

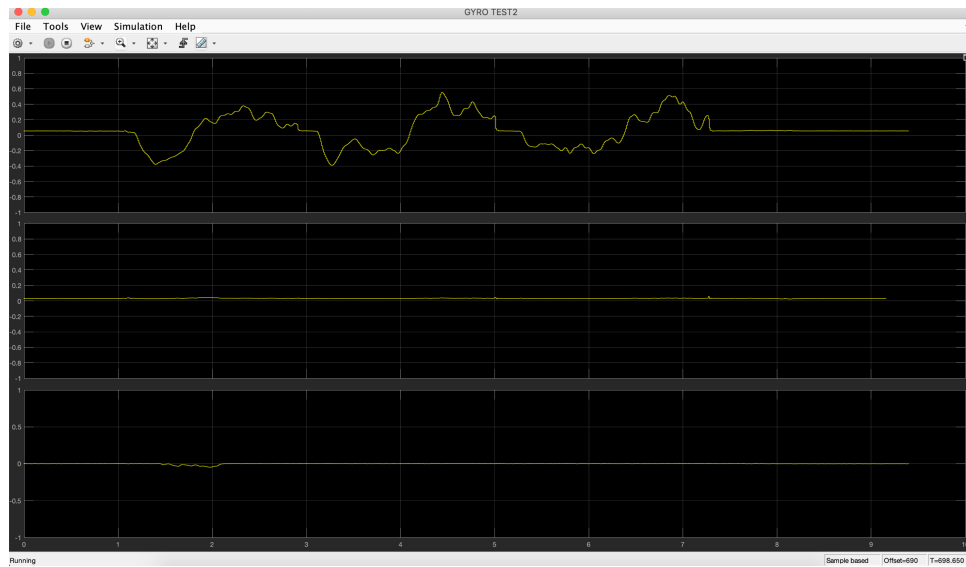


Figura 57: scope de la IMU al girar la bandeja en el eje X repetidamente

Como podemos observar, la gráfica de arriba es para el eje X, mientras que la del medio es para el eje Y. Por último, la de abajo sería para un hipotético eje Z, el cual no nos interesa en nuestro proyecto, por lo que lo hemos desactivado. Además, vemos que el eje X no está inicialmente ubicado en 0, sino unas centésimas por debajo. Esto es debido a que la superficie sobre la que lo estoy apoyando no es del todo plana. Como he dicho antes, la IMU es un aparato extremadamente sensible.

Ahora hacemos lo mismo que en [57], pero desplazando la bandeja hacia la derecha respecto del eje Y, cuyo resultado se plasma en [58].

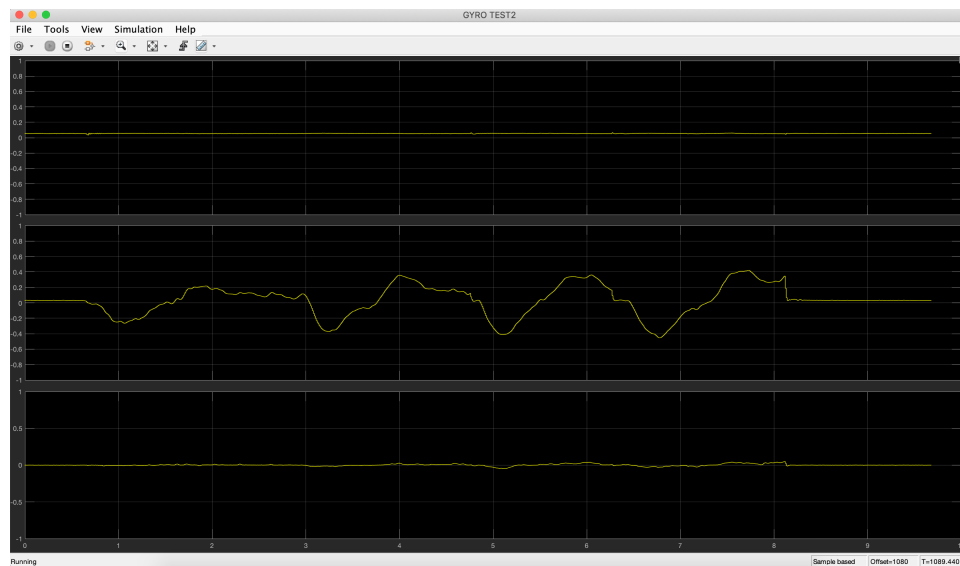


Figura 58: scope de la IMU al girar la bandeja en el eje Y repetidamente

Por último, giramos repetidamente la bandeja 20° en dirección eje X y lo mismo en dirección eje Y, por lo que, como podemos ver en [59], ambos ejes se verán afectados.

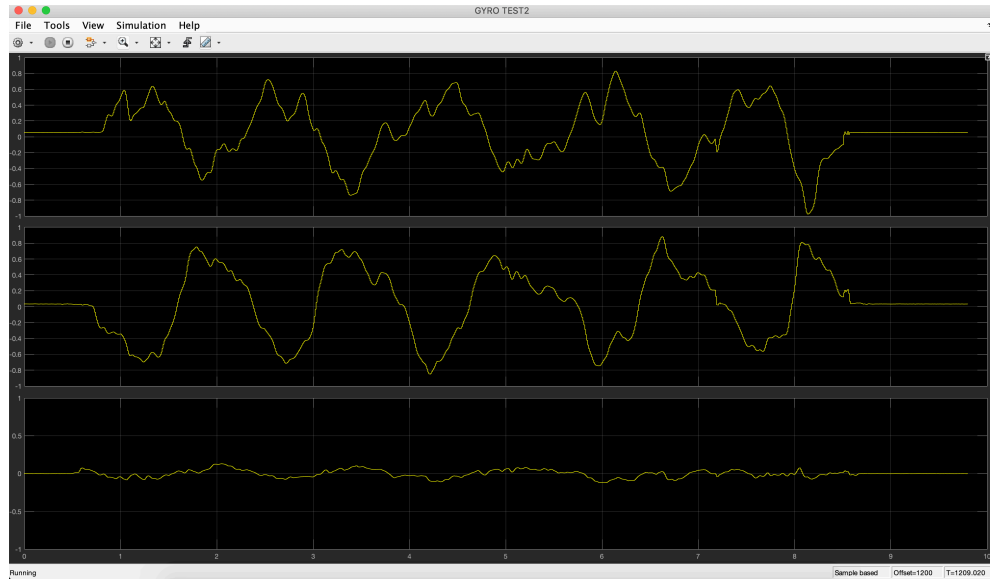


Figura 59: scope de la IMU al girar la bandeja en ambos ejes X e Y repetidamente

En nuestro diagrama de bloques de Simulink, el cual estará en el anexo correspondiente, tenemos varios scopes que nos dan diferentes medidas, pero a nosotros solo nos interesa el que aparece en [60], que es el acelerómetro. Este scope se llama GIRO TEST3, aunque también nos podría llegar a interesar el giroscopio, que es el GIRO TEST2.

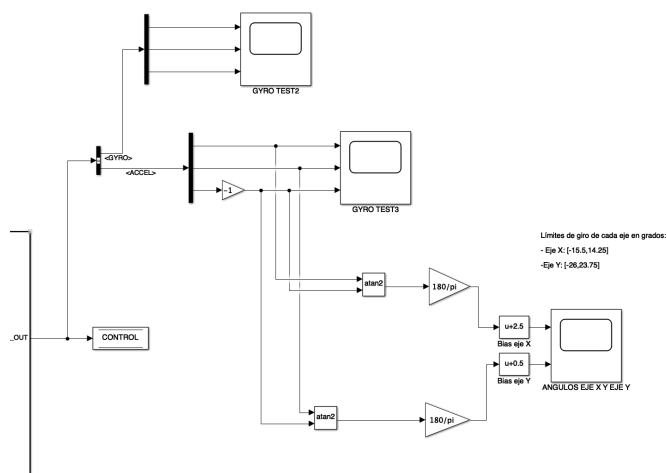


Figura 60: scopes útiles del diagrama de bloques de la IMU

Por tanto, estos son los sensores que vamos a utilizar en nuestro sistema y con los que vamos a estar constantemente tomando datos para su posterior uso en el control de la estructura.

1.7. MODELADO DEL SISTEMA

En este apartado vamos a explicar el modelado de nuestro sistema “ball on plate”, teniendo en cuenta las simplificaciones oportunas y distinguiendo entre los distintos elementos que afectan a dicho modelado.

Como sabemos, el modelo “ball on plate” se trata de una mejora del antiguo modelo “ball and beam”, en el cual se empleaba una viga sobre la que sostener una esfera, como se puede ver en [61]. Matemáticamente hablando, este modelo era más sencillo, puesto que solo se controlaba un eje mediante un solo servo. No obstante, en el modelo “ball on plate” se controlan los dos ejes X e Y, lo que hace que sea un sistema más complejo de modelar.

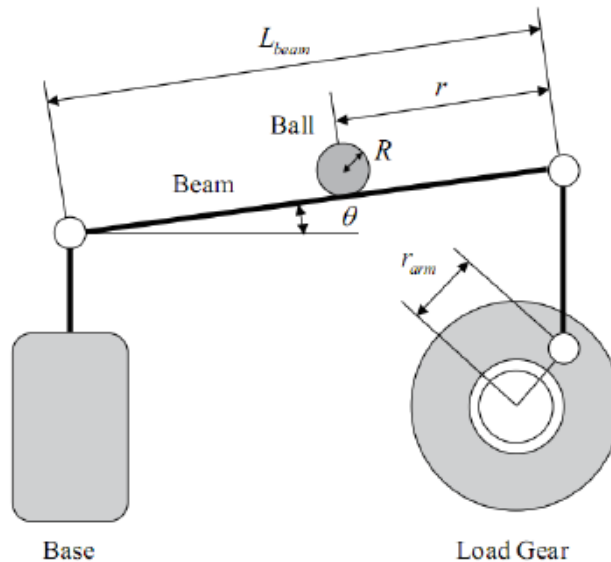


Figura 61: sistema “ball and beam” [JOSE14]

Antes de empezar, debemos hacer una serie de simplificaciones, las cuales son: la esfera no desliza sobre la pantalla resistiva, la esfera es completamente homogénea y simétrica (esto es importante para las ecuaciones de la esfera, como veremos más adelante) y la esfera y la pantalla están en contacto en todo momento [DEEJ17].

Lo primero que necesitamos para arrancar nuestro sistema es darle un ángulo de referencia a cada uno de los servos empleados, que serán θ_{xref} y θ_{yref} . Estos ángulos de referencia son los ángulos a los que queremos que se muevan los servos, uno en el eje X y otro en el eje Y. Sin embargo, el servo no es un actuador perfecto, por lo que habrá una pequeña variación entre los ángulos de referencia y los verdaderos ángulos a los que se mueven los servos, que serán θ_x y θ_y . Esto vendrá

dado por un sistema de segundo orden que podemos expresar, para cada uno de los ejes, de la siguiente forma:

$$\theta_{xref} = \frac{1}{wn^2} * \frac{d^2\theta_x}{dt^2} + \frac{2\zeta}{wn} * \frac{d\theta_x}{dt} + \theta_x$$

De esta ecuación, despejando, podemos sacar que (hay que tener en cuenta que $\frac{d^2\theta_x}{dt^2} = \ddot{\theta}_x$ y que $\frac{d\theta_x}{dt} = \dot{\theta}_x$):

$$\ddot{\theta}_x = wn^2 * (\theta_{xref} - \theta_x) - wn * 2\zeta * \dot{\theta}_x \quad (1)$$

Esta sería para el eje X, mientras que para el eje Y sería la siguiente:

$$\theta_{yref} = \frac{1}{wn^2} * \frac{d^2\theta_y}{dt^2} + \frac{2\zeta}{wn} * \frac{d\theta_y}{dt} + \theta_y$$

Despejando, obtenemos que (teniendo en cuenta que $\frac{d^2\theta_y}{dt^2} = \ddot{\theta}_y$ y que $\frac{d\theta_y}{dt} = \dot{\theta}_y$):

$$\ddot{\theta}_y = wn^2 * (\theta_{yref} - \theta_y) - wn * 2\zeta * \dot{\theta}_y \quad (2)$$

Por tanto, con respecto a los servos ya tenemos dos ecuaciones (1) y (2). Ahora tenemos que relacionar el ángulo de los servos con los ángulos de la bandeja, los cuales llamaré q_x y q_y . Una vez damos un ángulo de referencia al servo, el servo se mueve para alcanzar dicho ángulo. El movimiento del servo hace que se mueva también la bandeja, habiendo una relación mecánica entre el ángulo del servo y el de la bandeja. Esta relación viene dada por las siguientes dos ecuaciones:

$$(r_1 * \cos q_x - r_1 * \cos \theta_x)^2 + (r_1 * \sin q_x - r_1 * \sin \theta_x + l_1)^2 = l_1^2 \quad (3)$$

$$(r_2 * \sin \theta_y - r_2 * \sin q_y * \cos q_x + l_2)^2 + (r_2 * \cos q_y - r_2 * \cos \theta_y)^2 + (r_2 * \sin q_y * \sin q_x) = l_2^2 \quad (4)$$

Los parámetros r_1 , r_2 , l_1 y l_2 vienen dados de la geometría de nuestro sistema, como podemos observar en [62] y [63]. En [62], O1 y O2 sería el lugar donde colocamos los servos, mientras que O es donde

está colocada la junta universal del brazo central. Si todo estuviese bien, tendría que coincidir que $r_1 = r_2$ y que $l_1 = l_2$.

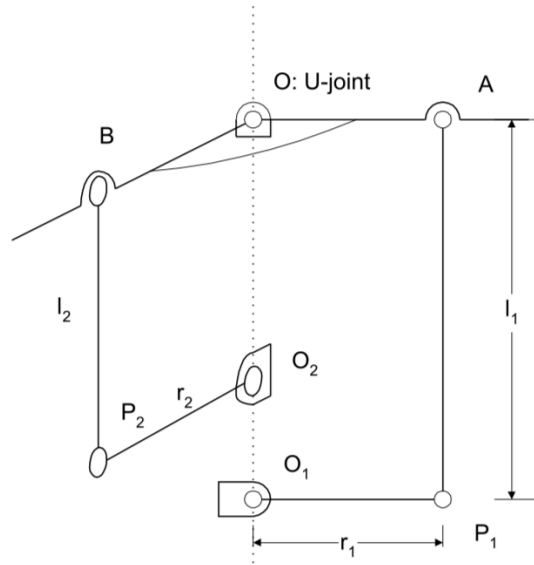


Figura 62: estructura mecánica del sistema 1 [AWTA02]

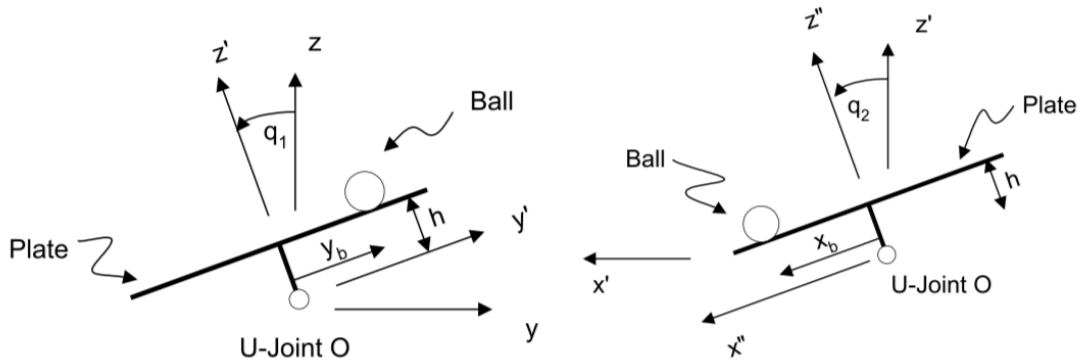


Figura 63: estructura mecánica del sistema 2 [AWTA02]

Por tanto, la relación entre los ángulos de los servos y los de la bandeja viene dada por las ecuaciones no lineales que he mostrado arriba. Sin embargo, para movimientos pequeños con respecto a la posición de equilibrio, se cumplen una serie de simplificaciones tal que llegamos a que:

$$q_x = \theta_x \quad y \quad q_y = \theta_y$$

No obstante, esto es solo una pequeña observación, ya que, a la hora del modelado del sistema, voy a tener solo en cuenta las ecuaciones no lineales (3) y (4).

Ahora, con las ecuaciones (1), (2), (3) y (4), hacemos cálculos y despejamos $\ddot{q}_x, \dot{q}_x, \ddot{q}_y$ y $\dot{q}_y$, que serán las derivadas de 4 de las 8 variables de estado que nos interesan. Las otras 4 derivadas las sacaremos con las ecuaciones de la esfera.

Con respecto a la esfera, sus ecuaciones se pueden sacar o por las leyes de Newton o por las ecuaciones de Lagrange. Para asegurarnos, lo hacemos de las dos maneras, donde obtenemos las siguientes ecuaciones no lineales:

$$m_b * g * r_b * \sin q_y * \cos q_x - m_b * r_b * [(h + r_b) * \ddot{q}_y - y_b * \ddot{q}_x * \sin q_y - x_b * \dot{q}_y^2 - x_b * \dot{q}_x^2 * \sin^2 q_y + (h + r_b) * \dot{q}_x^2 * \sin q_y * \cos q_y - 2 * \dot{y}_b * \dot{q}_x * \sin q_y + \ddot{x}_b] - I_b * (\ddot{x}_b / r_b) + \dot{q}_y = 0 \quad (5)$$

$$(m_b * g * r_b * \sin q_x) + m_b * r_b * [x_b * (\ddot{q}_x * \sin q_y + \dot{q}_y * \dot{q}_x * \cos q_y) - (h + r_b) * (\ddot{q}_x * \cos q_y - \dot{q}_y * \dot{q}_x * \sin q_y) + \dot{q}_y * \dot{q}_x * (h + r_b) * \sin q_y - y_b * \dot{q}_x^2 + x_b * \dot{q}_y * \dot{q}_x * \cos q_y + 2 * \dot{x}_b * \dot{q}_x * \sin q_y + \ddot{y}_b] + I_b * (\ddot{y}_b / r_b) - \ddot{q}_x * \cos q_y + \dot{q}_y * \dot{q}_x * \sin q_y = 0 \quad (6)$$

De estas dos ecuaciones, despejamos las derivadas de nuestras dos siguientes variables estado, es decir, la aceleración de la esfera en el eje X, $\ddot{x}_b$, y la aceleración de la esfera en eje Y, $\ddot{y}_b$. Además, debemos tener en cuenta que:

$$\dot{x}_b = \frac{dx_b}{dt} \quad y \quad \dot{y}_b = \frac{dy_b}{dt}$$

Por tanto, ya tendríamos todas las derivadas de nuestras 8 variables de estados. Estas variables de estado son las siguientes: la posición de la esfera en el eje X, x_b , la posición de la esfera en el eje Y, y_b , la velocidad de la esfera en el eje X, $\dot{x}_b$, la velocidad de la esfera en el eje Y, $\dot{y}_b$, la posición angular de la bandeja en el eje X, q_x , la posición angular de la bandeja en el eje Y, q_y , la velocidad angular de la bandeja en el eje X, $\dot{q}_x$ y la velocidad angular de la bandeja en el eje Y, $\dot{q}_y$.

1.8. CONCLUSIONES Y TRABAJOS FUTUROS

En primer lugar, podemos decir que se han cumplido los dos bloques principales de objetivos, tanto la parte de la estructura como la de los sensores. Sin embargo, el bloque de objetivos extra no ha sido capaz de cumplirse debido a la falta de tiempo motivada por los problemas que fueron surgiendo durante la realización del proyecto, por lo que no se pudo implementar el control.

Por una parte, hubo muchos problemas a la hora de diseñar las piezas de la estructura. Esto es debido a que la impresora 3D del laboratorio de fabricación utilizaba como material plástico, lo que hacía que las medidas de las piezas no fueran tal cual las diseñadas mediante SolidEdge. Por ello, cada pieza tuvo que diseñarse varias veces. Además, hubo especial conflicto con la bandeja ya que, al ser de gran tamaño, su impresión se demoró mucho en el tiempo. Pensé en volver a imprimirla, pero no había tiempo, por lo que tuve que hacer una serie de retoques al haber salido defectuosa de su impresión. Primero, tuve que lijar su interior para ampliar su superficie de modo que cupiese la pantalla resistiva. Después, tuve que dar calor a las esquinas para volver a darles forma, ya que salieron de la impresión totalmente deformes.

Una vez solucionados estos problemas, se llevó a cabo el montaje de la estructura, que transcurrió sin ningún tipo de problema. Si eso comentar que al principio se pensó en unir los 3 brazos a la bandeja mediante ventosas, pero viendo la poca capacidad de adhesión de las mismas, se decidió utilizar piezas 3D atornilladas. Además, no me decidía entre utilizar juntas universales o rótulas esféricas, pero al final me decanté por las primeras.

Con respecto a los sensores, hubo solo un problema con la IMU. Esta estaba programada para estar conectada al puerto 1 del *click shield*. Al principio no nos dimos cuenta y la conectamos al puerto 2, por lo que tardamos mucho tiempo en poder testearla. Hablando del conversor ADC, como por defecto estaba limitado a 2V, no nos llegaban de la pantalla resistiva las tensiones superiores a 2V, por lo que tuvimos que diseñar unos divisores de tensión, teniendo cuidado con las resistencias que cogíamos, de forma que la resistencia de salida de la pantalla no afectase al divisor. Por último, la pantalla resistiva nos dio varios problemas, ya que esta no era del todo plana, lo que afectaba a la hora de la lectura de las coordenadas de la esfera. Además, al principio, como he comentado antes, la pantalla no cabía en la bandeja. Una vez lijé la superficie interior para aumentarla, no hubo problema. En cuanto a los servos inteligentes, tuve problemas con ellos, por lo que no pude ponerlos en marcha mediante Simulink. Simplemente pude comprobar que funcionaban mediante un programa llamado LewanSoul, que me permitía saber la posición de cada servo y moverlo según quisiese.

En vista de los problemas surgidos, recomendaría, en primer lugar, volver a imprimir la bandeja sobre la que se apoya la pantalla o incluso utilizar una bandeja real con las dimensiones adecuadas. Además,

recomendaría cambiar la pantalla resistiva por una capacitiva o utilizar una cámara para el procesamiento de imágenes. Por último, cambiaría de servos, ya que los LX-16A son bastante complejos en cuanto a su funcionamiento. Una vez hecho estas cosas, procedería a continuar con el proyecto, que básicamente sería implementar un control simple al sistema, como por ejemplo un control P. Después, probaría otros tipos de controles distintos, entre los que destacan: el PD, PI, PID, control por realimentación de estado y control por lógica difusa.

El objetivo final del proyecto sería no solo que la bandeja pueda mantener la bola en una posición fija, sino que esta sea capaz de hacer que la esfera siga una trayectoria definida, como por ejemplo un círculo o un rectángulo.

1.9. BIBLIOGRAFÍA

1. [QUAN13] Dr. Dennis Bernstein, Dr. Stephen Williams, Dr. Kelly Cohen, Dr. Shirley Dyke, Dr. Keyvan Hashtrudi – Zaad, Dr. Tong Guofeng, Dr. Victor Becerra, Dr. Roxana Saint – Nom, Dr. Yongpeng Zhang, Dr. YangQuan Chen, Dr. Ben Cazzolato. “The Rotary Control Lab: A Modular Single Source Solution You Can Control”. Datasheet *QUANSER*, pág. 7. Toronto, Canada. 2015. [Última visita: agosto 2019]
<https://www.quanser.com/wp-content/uploads/2017/04/Rotary-Brochure-online.pdf>
2. [TYLE12] Tyler Kroymann. “Ball and Plate PID control with 6 DOF Stewart platform”. *SIA Magazin. Full Motion Dynamics*. March 28, 2018. [Última visita: agosto 2019]
<https://siamagazin.com/ball-and-plate-pid-control-with-6-dof-stewart-platform/#>
3. [SOLU99] Solutions4U. “Ball & Plate System”. Datasheet *GOOGOL Technology*, pág 1. Puchong, Selangor, Malasya. [Última visita: agosto 2019]
http://www.solutions4u-asia.com/PDT/GT/LAB/PDF/S4U_BallnPlate.pdf
4. [DAVI16] David Hamor. “Ball and Plate GPL3, LQ control of Ball and Plate system controlled by Arduino MEGA 2560”. *Project Hub*. [Última visita: agosto 2019].
<https://create.arduino.cc/projecthub/davidhamor/ball-and-plate-c48027>
5. [MUKT13] Mukta Nigam, Priyank Paras Jain. “Real time control of ball and beam system with model reference adaptive control strategy using MIT rule”. International Conference on Computational Intelligence and Computing Research. United States, 2013, organized by the Institute of Electrical and Electronics Engineers. [Última visita: agosto 2019].
<https://ieeexplore.ieee.org/document/6724180>
6. [MIAD08] Miad Moarref, Mahboobeh Saadat, G.R. Vossoughi. “Mechatronic design and position control of a novel ball and plate system”. 16th Mediterranean Conference on Control and Automation (MED). Congress Centre, Ajaccio, France. June 25-27, 2008. Organized by the Institute of Electrical and Electronics Engineers. [Última visita: junio 2019].
https://www.researchgate.net/publication/224326638_Mechatronic_design_and_position_control_of_a_novel_ball_and_plate_system

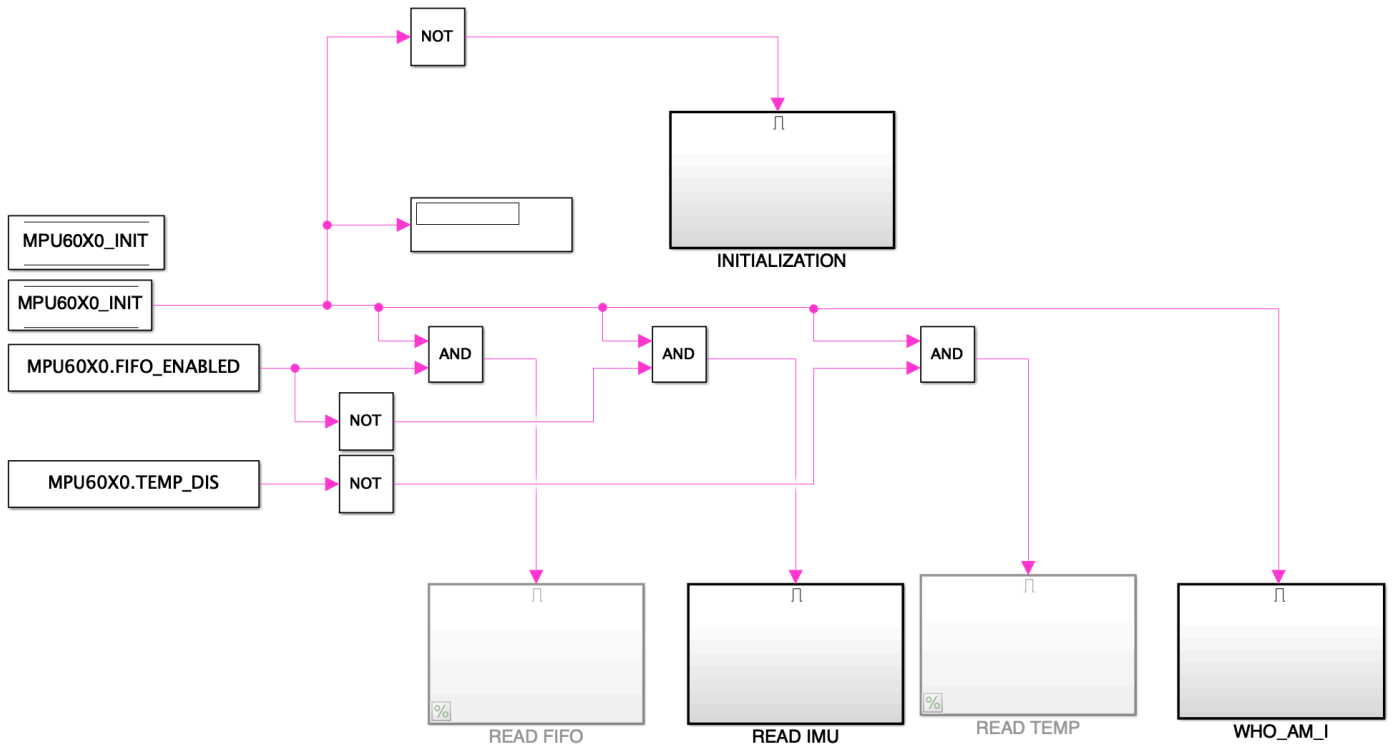
7. [TKUH18] T – Kuhn. “Keeping a Ball Bouncing With 4 Mics”. Disponible en blog *Electronic Dust*. July 22, 2018. [Última visita: junio 2019].
<https://electrondust.com/2018/07/22/stepper-juggler/>
8. [FILE16] López Ortega Filemón, Martha Cárdenas Maciel, Rogelio Serrano Zepeda y Cristina Danahe Ortega Rodríguez. “Plataforma experimental de control ball on plate utilizando Labview y Arduino”. *Revista Aristas Investigación Básica y Aplicada*, vol. 5, núm. 9, año 2016. Congreso Internacional de Investigación de Tijuana. [Última visita: agosto 2019].
<http://201.151.251.123/ftp/oficios17/academica/a/41.PLATAFORMA%20EXPERIMENTAL%20DE%20CONTROL%20BALL%20ON%20PLATE%20UTILIZANDO%20LABVIEW%20Y%20ARDUINO.pdf>
9. [JAVI16] Javier Elío. “¿Os acordáis de las pantallas resistivas?”. Artículo del periódico *El Español*. Mayo de 2016. [Última visita: agosto 2019].
<https://elandroidelibre.espanol.com/2016/05/seccion-retro-pantallas-resistivas.html>
10. [OSCA09] Óscar Beamonte. “Pantallas táctiles: capacitivas vs resistivas”. *Xataka Móvil*, febrero de 2009. [Última visita: agosto 2019].
<https://www.xatakamovil.com/desarrollo/pantallas-tactiles-capacitivas-vs-resistivas>
11. [EDNA16] Edna Carolina Moriones Polanía. “Diseño, implementación y control de sistema de balance ball on plate”. *Revista SENNOVA*, vol. 2, núm. 2, año 2016. Bogotá, Colombia. [Última visita: junio 2019].
<http://revistas.sena.edu.co/index.php/sennova/article/view/553>
12. [PEDR15] Pedro de Otaola Arca. “Construcción y diseño del sistema de control de posición de una esfera sobre una plataforma móvil”. Trabajo Fin de Grado, año 2015. Universidad Pontificia de Comillas ICAI, Madrid. [Última visita: agosto 2019].
https://www.iit.comillas.edu/pfc/PFC_Navigate.php?frmrequestedpage=pasado&requestedtitulacion=IEM_N&requestedano=2015
13. [DAVI17] David Heritage. “Ball and Plate Arduino PID Control”. *Instructables circuits*. Junio de 2015. [Última visita: agosto 2019].

<https://www.instructables.com/id/Ball-and-Plate-Arduino-PID-Control/>

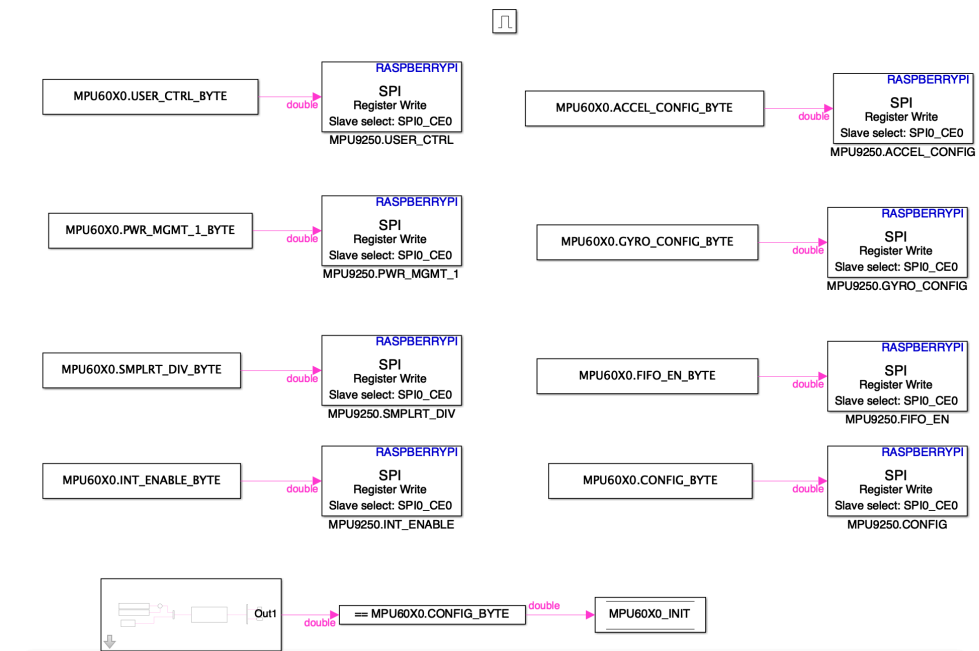
14. [JOSE14] José Oniman de Arquino Limaverde Filho, Eugenio Liborio Feitosa Fortaleza. “Control of the ball and beam using Kalman Filter. A Flatness Based Approach”. Conference anais do XX Congresso Brasileiro de Automática. Belo Horizonte, Minas Gerais. 20 a 24 de Setembro de 2014. [Última visita: agosto 2019].
https://www.researchgate.net/publication/272237344_Control_of_the_Ball_and_Beam_using_Kalman_Filter_-_A_Flatness_Based_Approach
15. [DEEJ17] DeeJay Stander, Santiago Jiménez – Leudo, Nicanor Quijano. “Low – cost Ball and Plate Design and Implementation for Learning Control Systems”. 3rd Colombian Conference on Automatic Control. Cartagena, Colombia. 18-20 Oct. 2017. Organized by the Institute of Electrical and Electronics Engineers. [Última visita: agosto 2019].
<https://ieeexplore.ieee.org/abstract/document/8276472>
16. [AWTA02] Shorya Awtar, C. Bernard, N. Boklund, A. Master, D. Ueda and Kevin Craig. “Mechatronic design of a ball-on-plate balancing system”. *Revista Elsevier*, volume 12, issue 2, march 2002, pages 217-228. [Última visita: agosto 2019].
<https://www.sciencedirect.com/science/article/abs/pii/S0957415801000629>

II. Anexo A: Diagramas Simulink

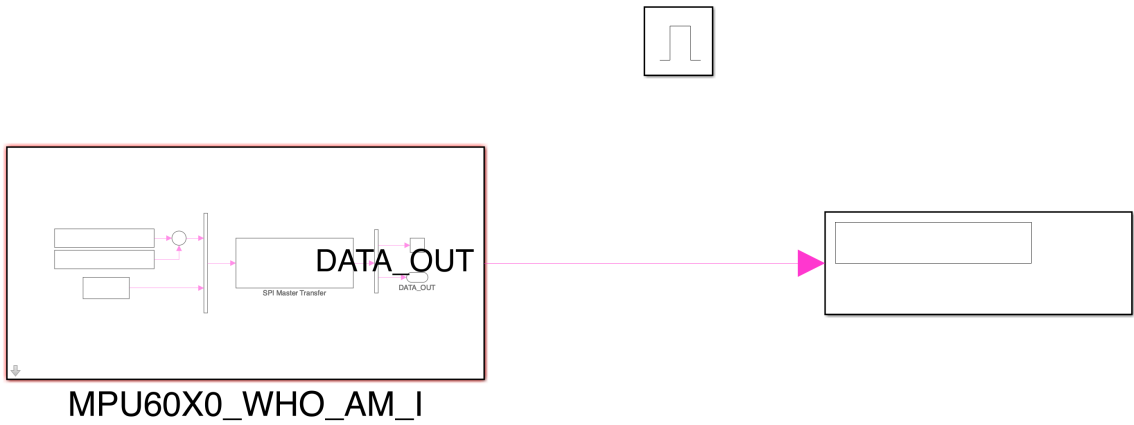
IMU GENERAL



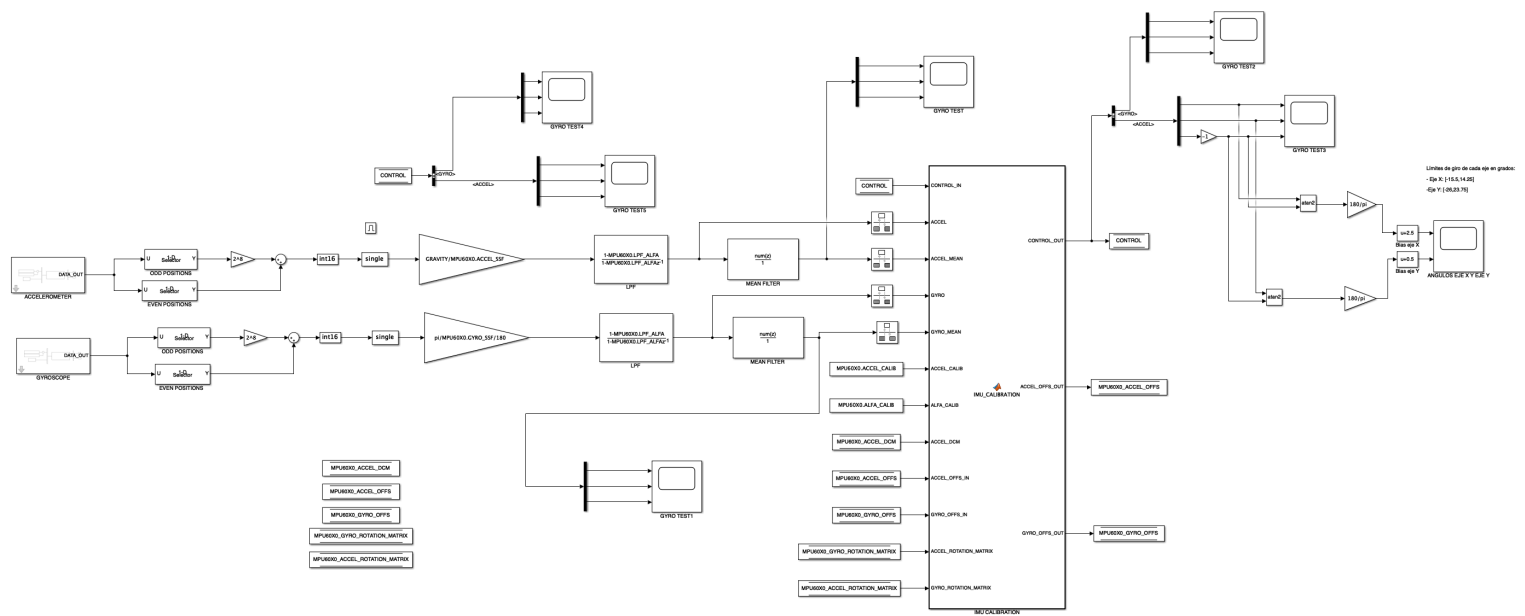
Inicialización IMU



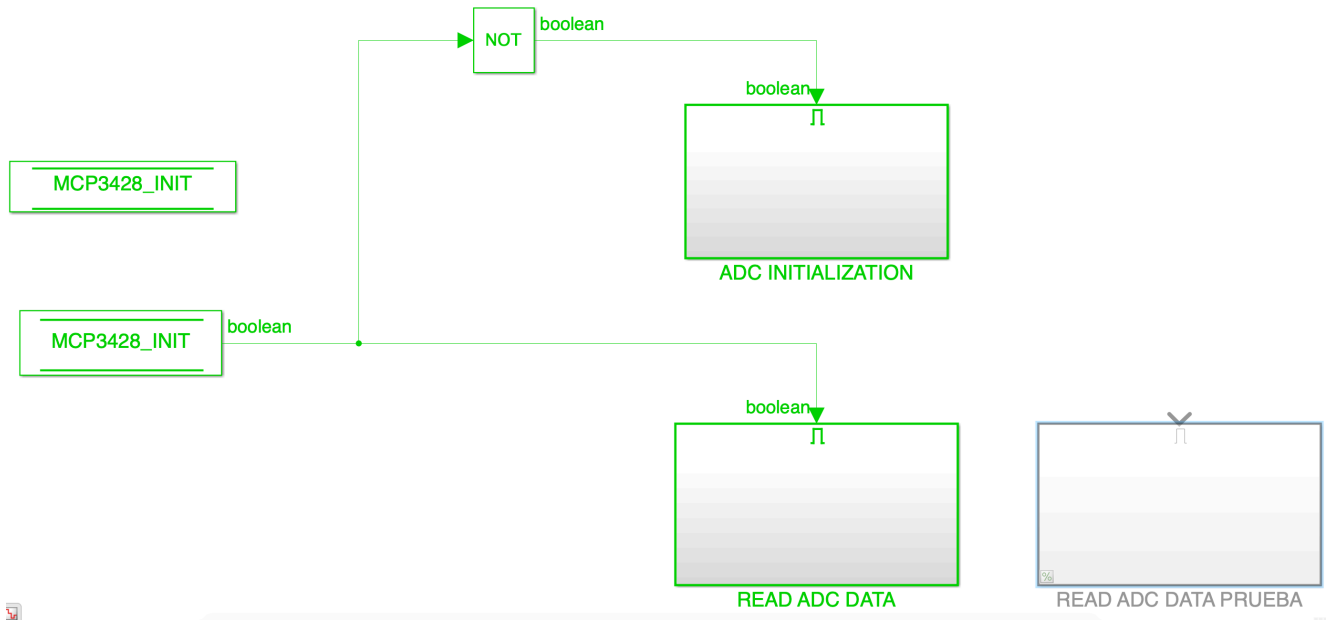
Who am I IMU



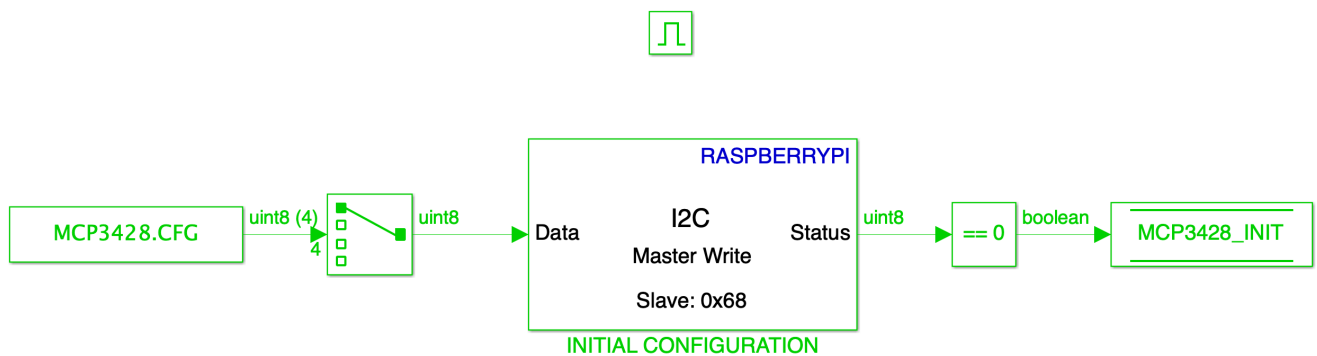
Lectura IMU



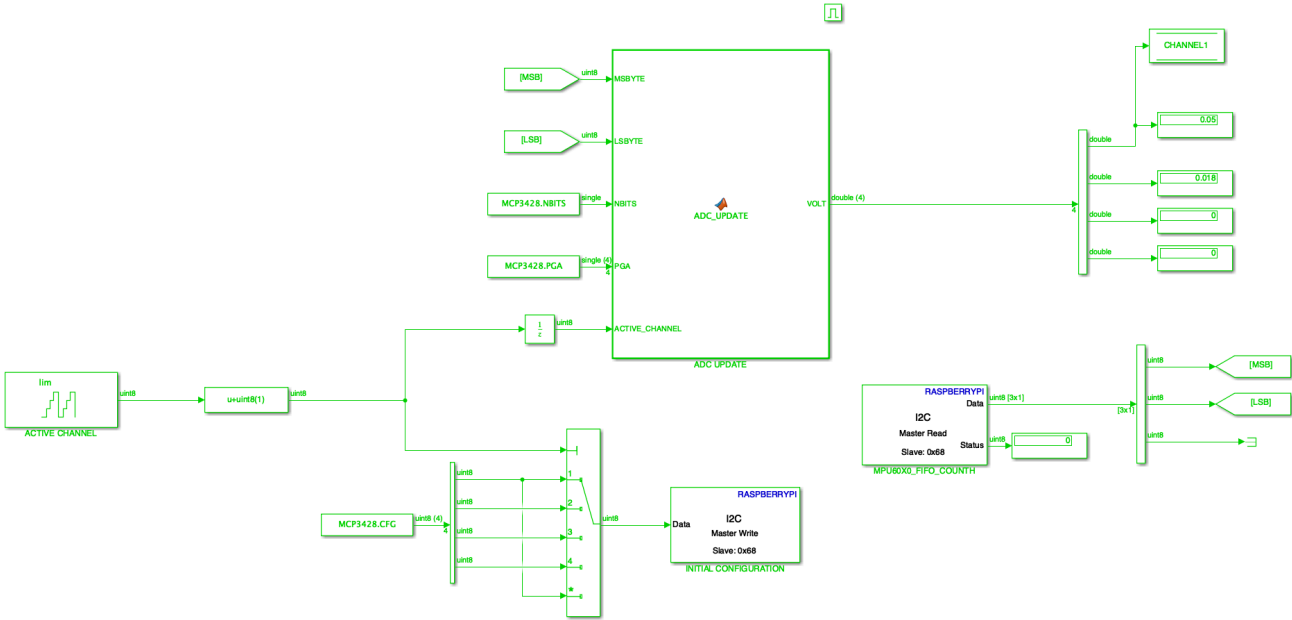
CONVERSOR ADC GENERAL



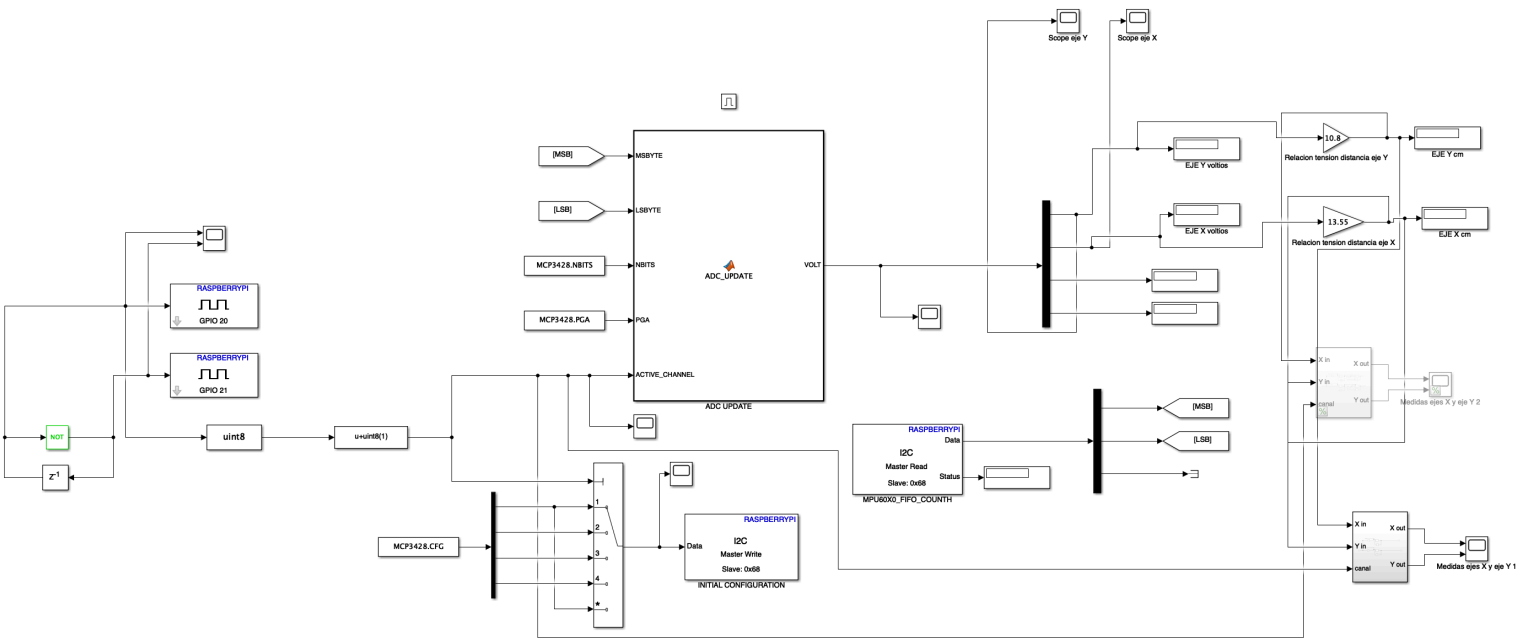
Inicialización ADC



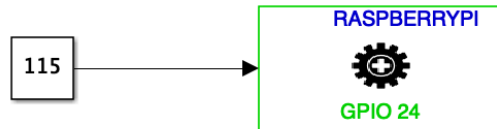
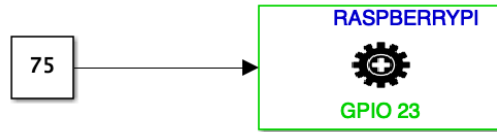
Lectura ADC (solo un canal a la vez)



Lectura ADC (ambos canales a la vez)



Servos



III. Anexo B: Código Matlab

Para poner en marcha la IMU, empleamos el CONFIG_MPU60X0.m:

```
%% MPU-60X0
%-----
MPU60X0.SAMPLING_TIME = 1e-3;
% I2C ADDRESS
MPU60X0.ADDRESS = bin2dec('1101000');
MPU60X0.ADDRESS_WRITE = bin2dec('11010001');
MPU60X0.ADDRESS_READ = bin2dec('11010000');
%-----
% Mask for reading (SPI)
MPU60X0.READ_FLAG = bin2dec('10000000');
%-----
% FIFO
MPU60X0.FIFO_ENABLED = boolean(0);
MPU60X0.FIFO_LENGTH = single(10);
%-----
% SPI or I2C
% / 0. I2C / 1. SPI
MPU60X0.I2C_IF_DIS = boolean(1);
%-----
% I2C MASTER ENABLE FOR AUXILIARY I2C (MAGNETOMETER)
MPU60X0.I2C_MST_EN = boolean(0);
%-----
% TEMPERATURE SENSOR ENABLE
MPU60X0.TEMP_DIS = boolean(1);

%-----
% IMU_ORIENTATION
MPU60X0.ORIENTATION = uint8(14); % Roll180Yaw270

%-----
% ACCELEROMETER SENSITIVITY SCALE FACTOR
MPU60X0.AFS_SEL='10';
switch(bin2dec(MPU60X0.AFS_SEL))
    case 0
        MPU60X0.ACCEL_SSF=16384;
    case 1
        MPU60X0.ACCEL_SSF=8192;
    case 2
        MPU60X0.ACCEL_SSF=4096;
    case 3
        MPU60X0.ACCEL_SSF=2048;
end
%-----
% GYROSCOPE SENSITIVITY SCALE FACTOR
MPU60X0.FS_SEL='10';
switch(bin2dec(MPU60X0.FS_SEL))
    case 0
        MPU60X0.GYRO_SSF=131;
    case 1
        MPU60X0.GYRO_SSF=65.5;
    case 2
        MPU60X0.GYRO_SSF=32.8;
    case 3
        MPU60X0.GYRO_SSF=16.4;
end
%-----
% DIGITAL LOW PASS FILTER (DLPF)
% Bits [2:0] in CONFIG
MPU60X0.DLPF_CFG='000';
%-----
% TEMPERATURE SENSITIVITY SCALE FACTOR
MPU60X0.TEMP_SSF = 340;
% TEMPERATURE OFFSET (DEGREES)
MPU60X0.TEMP_OFFS = 35;
%-----
% CLOCK SOURCE
%MPU60X0.CLKSEL: Bits [2:0] in MPU60X0.PWR_MGMT_1
MPU60X0.CLKSEL='001';
% '000': Internal 8MHz oscillator
```



```

% '001': PLL with X axis gyroscope reference
% '010': PLL with Y axis gyroscope reference
% '011': PLL with Z axis gyroscope reference
% '100': PLL with external 32.768kHz
% '101': PLL with external 19.2MHz
% '110': Reserved
% '111': Stops the clock and keeps the timing generator in reset
%-----
% IMU ORIENTATION
MPU60X0.ORIENTATION = uint8(14); % Roll180Yaw270

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% MPU-60X0 Register Map
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-----
% SELF-TEST REGISTERS
MPU60X0.SELF_TEST_X = hex2dec('0D');
MPU60X0.SELF_TEST_X_BYTE = bin2dec('00000000');
MPU60X0.SELF_TEST_Y = hex2dec('0E');
MPU60X0.SELF_TEST_Y_BYTE = bin2dec('00000000');
MPU60X0.SELF_TEST_Z = hex2dec('0F');
MPU60X0.SELF_TEST_Z_BYTE = bin2dec('00000000');
MPU60X0.SELF_TEST_A = hex2dec('10');
MPU60X0.SELF_TEST_A_BYTE = bin2dec('00000000');
%-----
% SAMPLE RATE DIVIDER
MPU60X0.SMPLRT_DIV = hex2dec('19');
if MPU60X0.FIFO_ENABLED == 0
    MPU60X0.SMPLRT_DIV_BYTE = bin2dec('00000111');
else
    SMPLRT_DIV = 1000*SAMPLING_TIME/MPU60X0.FIFO_LENGTH;
    if SMPLRT_DIV == 1
        MPU60X0.SMPLRT_DIV_BYTE = bin2dec('00000000');
    elseif SMPLRT_DIV == 2
        MPU60X0.SMPLRT_DIV_BYTE = bin2dec('00000001');
    elseif SMPLRT_DIV == 4
        MPU60X0.SMPLRT_DIV_BYTE = bin2dec('00000011');
    elseif SMPLRT_DIV == 5
        MPU60X0.SMPLRT_DIV_BYTE = bin2dec('00000100');
    else
        disp('ERROR IN THE FIFO LENGTH DEFINITION')
    end
end
clear SMPLRT_DIV
%-----
% CONFIGURATION
MPU60X0.CONFIG = hex2dec('1A');
MPU60X0.CONFIG_BYTE = bin2dec(['00000' MPU60X0.DLPF_CFG]);
MPU60X0.GYRO_CONFIG = hex2dec('1B');
MPU60X0.GYRO_CONFIG_BYTE = bin2dec(['000' MPU60X0.FS_SEL '000']);
MPU60X0.ACCEL_CONFIG = hex2dec('1C');
MPU60X0.ACCEL_CONFIG_BYTE = bin2dec(['000' MPU60X0.AFS_SEL '000']);
%-----
% % MOTION DETECTION THRESHOLD %%%%%%%%% NI IDEA DE PARA QUE ES %%%%%%%%%
% MPU60X0.MOT_THR = hex2dec('1F');
% MPU60X0.MOT_THR_BYTE = bin2dec('00000000');
%-----
% FIFO ENABLE
MPU60X0.FIFO_EN = hex2dec('23');
% Enable FIFO gyro and accelerometer register
if MPU60X0.FIFO_ENABLED
    % [TEMP XG YG ZG ACCEL SLV2 SLV1 SLV0]
    MPU60X0.FIFO_EN_BYTE = bin2dec('01111000');
else
    MPU60X0.FIFO_EN_BYTE = bin2dec('00000000');
end
%-----
% I2C CONFIGURATION
MPU60X0.I2C_MST_CTRL = hex2dec('24');
% I2C master clock frequency = 400 kHz
MPU60X0.I2C_MST_CLK = '1101';
MPU60X0.I2C_MST_CTRL_BYTE = bin2dec(['0000' MPU60X0.I2C_MST_CLK]);
MPU60X0.I2C_SLV0_ADDR = hex2dec('25');

```

```

MPU60X0.I2C_SLV0_ADDR_BYTE = bin2dec('00000000');
MPU60X0.I2C_SLV0_REG = hex2dec('26');
MPU60X0.I2C_SLV0_REG_BYTE = bin2dec('00000000');
MPU60X0.I2C_SLV0_CTRL = hex2dec('27');
MPU60X0.I2C_SLV0_CTRL_BYTE = bin2dec('00000000');
MPU60X0.I2C_SLV1_ADDR = hex2dec('28');
MPU60X0.I2C_SLV1_ADDR_BYTE = bin2dec('00000000');
MPU60X0.I2C_SLV1_REG = hex2dec('29');
MPU60X0.I2C_SLV1_REG_BYTE = bin2dec('00000000');
MPU60X0.I2C_SLV1_CTRL = hex2dec('2A');
MPU60X0.I2C_SLV1_CTRL_BYTE = bin2dec('00000000');
MPU60X0.I2C_SLV2_ADDR = hex2dec('2B');
MPU60X0.I2C_SLV2_ADDR_BYTE = bin2dec('00000000');
MPU60X0.I2C_SLV2_REG = hex2dec('2C');
MPU60X0.I2C_SLV2_REG_BYTE = bin2dec('00000000');
MPU60X0.I2C_SLV2_CTRL = hex2dec('2D');
MPU60X0.I2C_SLV2_CTRL_BYTE = bin2dec('00000000');
MPU60X0.I2C_SLV3_ADDR = hex2dec('2E');
MPU60X0.I2C_SLV3_ADDR_BYTE = bin2dec('00000000');
MPU60X0.I2C_SLV3_REG = hex2dec('2F');
MPU60X0.I2C_SLV3_REG_BYTE = bin2dec('00000000');
MPU60X0.I2C_SLV3_CTRL = hex2dec('30');
MPU60X0.I2C_SLV3_CTRL_BYTE = bin2dec('00000000');
MPU60X0.I2C_SLV4_ADDR = hex2dec('31');
MPU60X0.I2C_SLV4_ADDR_BYTE = bin2dec('00000000');
MPU60X0.I2C_SLV4_REG = hex2dec('32');
MPU60X0.I2C_SLV4_REG_BYTE = bin2dec('00000000');
MPU60X0.I2C_SLV4_DO = hex2dec('33');
MPU60X0.I2C_SLV4_DO_BYTE = bin2dec('00000000');
MPU60X0.I2C_SLV4_CTRL = hex2dec('34');
MPU60X0.I2C_SLV4_CTRL_BYTE = bin2dec('00000000');
MPU60X0.I2C_SLV4_DI = hex2dec('35');
MPU60X0.I2C_MST_STATUS = hex2dec('36');
%-----
% INTERRUPTION REGISTERS
MPU60X0.INT_PIN_CFG = hex2dec('37');
MPU60X0.INT_PIN_CFG_BYTE = bin2dec('00000000');
MPU60X0.INT_ENABLE = hex2dec('38');
MPU60X0.INT_ENABLE_BYTE = bin2dec('00000000');
MPU60X0.INT_STATUS = hex2dec('3A');
%-----
% SENSOR READOUT REGISTERS
MPU60X0.ACCEL_XOUT_H = hex2dec('3B');
MPU60X0.ACCEL_XOUT_L = hex2dec('3C');
MPU60X0.ACCEL_YOUT_H = hex2dec('3D');
MPU60X0.ACCEL_YOUT_L = hex2dec('3E');
MPU60X0.ACCEL_ZOUT_H = hex2dec('3F');
MPU60X0.ACCEL_ZOUT_L = hex2dec('40');
MPU60X0.TEMP_OUT_H = hex2dec('41');
MPU60X0.TEMP_OUT_L = hex2dec('42');
MPU60X0.GYRO_XOUT_H = hex2dec('43');
MPU60X0.GYRO_XOUT_L = hex2dec('44');
MPU60X0.GYRO_YOUT_H = hex2dec('45');
MPU60X0.GYRO_YOUT_L = hex2dec('46');
MPU60X0.GYRO_ZOUT_H = hex2dec('47');
MPU60X0.GYRO_ZOUT_L = hex2dec('48');
MPU60X0.EXT_SENS_DATA_00 = hex2dec('49');
MPU60X0.EXT_SENS_DATA_01 = hex2dec('4A');
MPU60X0.EXT_SENS_DATA_02 = hex2dec('4B');
MPU60X0.EXT_SENS_DATA_03 = hex2dec('4C');
MPU60X0.EXT_SENS_DATA_04 = hex2dec('4D');
MPU60X0.EXT_SENS_DATA_05 = hex2dec('4E');
MPU60X0.EXT_SENS_DATA_06 = hex2dec('4F');
MPU60X0.EXT_SENS_DATA_07 = hex2dec('50');
MPU60X0.EXT_SENS_DATA_08 = hex2dec('51');
MPU60X0.EXT_SENS_DATA_09 = hex2dec('52');
MPU60X0.EXT_SENS_DATA_10 = hex2dec('53');
MPU60X0.EXT_SENS_DATA_11 = hex2dec('54');
MPU60X0.EXT_SENS_DATA_12 = hex2dec('55');
MPU60X0.EXT_SENS_DATA_13 = hex2dec('56');
MPU60X0.EXT_SENS_DATA_14 = hex2dec('57');
MPU60X0.EXT_SENS_DATA_15 = hex2dec('58');
MPU60X0.EXT_SENS_DATA_16 = hex2dec('59');

```

```

MPU60X0.EXT_SENS_DATA_17 = hex2dec('5A');
MPU60X0.EXT_SENS_DATA_18 = hex2dec('5B');
MPU60X0.EXT_SENS_DATA_19 = hex2dec('5C');
MPU60X0.EXT_SENS_DATA_20 = hex2dec('5D');
MPU60X0.EXT_SENS_DATA_21 = hex2dec('5E');
MPU60X0.EXT_SENS_DATA_22 = hex2dec('5F');
MPU60X0.EXT_SENS_DATA_23 = hex2dec('60');
%-----
% I2C SLAVE DATA
MPU60X0.I2C_SLV0_DO = hex2dec('63');
MPU60X0.I2C_SLV0_DO_BYTE = bin2dec('00000000');
MPU60X0.I2C_SLV1_DO = hex2dec('64');
MPU60X0.I2C_SLV1_DO_BYTE = bin2dec('00000000');
MPU60X0.I2C_SLV2_DO = hex2dec('65');
MPU60X0.I2C_SLV2_DO_BYTE = bin2dec('00000000');
MPU60X0.I2C_SLV3_DO = hex2dec('66');
MPU60X0.I2C_SLV3_DO_BYTE = bin2dec('00000000');
%-----
% I2C MASTER DELAY CONTROL
MPU60X0.I2C_MST_DELAY_CTRL = hex2dec('67');
MPU60X0.I2C_MST_DELAY_CTRL_BYTE = bin2dec('00000000');
%-----
% SIGNAL PATH RESET
MPU60X0.SIGNAL_PATH_RESET = hex2dec('68');
MPU60X0.SIGNAL_PATH_RESET_BYTE = bin2dec('00000000');
%-----
% % MOTION DETECTION CONTROL %%%%%%%%% NI IDEA DE PARA QUE ES %%%%%%%%%
% MPU60X0.MOT_DETECT_CTRL = hex2dec('69');
% MPU60X0.MOT_DETECT_CTRL_BYTE = bin2dec('00000000');
%-----
% USER CONTROL: FIFO and SPI enable
MPU60X0.USER_CTRL = hex2dec('6A');
MPU60X0.USER_CTRL_BYTE = bin2dec(['0' num2str([MPU60X0.FIFO_ENABLED MPU60X0.I2C_MST_EN
MPU60X0.I2C_IF_DIS]) '0000']);
%-----
% POWER MANAGEMENT
MPU60X0.PWR_MGMT_1 = hex2dec('6B');
MPU60X0.PWR_MGMT_1_BYTE = bin2dec(['0000' num2str(MPU60X0.TEMP_DIS) MPU60X0.CLKSEL]);
MPU60X0.PWR_MGMT_2 = hex2dec('6C');
MPU60X0.PWR_MGMT_2_BYTE = bin2dec('00000000');
%-----
% FIFO CONTROL AND INFORMATION
MPU60X0.FIFO_COUNTH = hex2dec('72');
MPU60X0.FIFO_COUNTH_BYTE = bin2dec('00000000');
MPU60X0.FIFO_COUNTL = hex2dec('73');
MPU60X0.FIFO_COUNTL_BYTE = bin2dec('00000000');
MPU60X0.FIFO_R_W = hex2dec('74');
MPU60X0.FIFO_R_W_BYTE = bin2dec('00000000');
%-----
% WHO AM I
MPU60X0.WHO_AM_I = hex2dec('75');
MPU60X0.WHO_AM_I_BYTE=hex2dec('68');
%-----

%% IMU BUS DEFINITION
IMU_INI.GYRO = zeros(3,1,'single');
IMU_INI.ACCEL = zeros(3,1,'single');
IMU_INI.GYRO_MEAN = zeros(3,1,'single');
IMU_INI.ACCEL_MEAN = zeros(3,1,'single');
% IMU PROCESSING - ADDITIONAL FILTERING
IMU_INI.ACCEL_FILT_FREQ = 20; % Frequency in Hz
IMU_INI.GYRO_FILT_FREQ = 20; % Frequency in Hz
% IMU NOISE (standard deviation)
IMU_INI.GYRO_NOISE = single(0.02); % rad/s
IMU_INI.ACCEL_NOISE = single(0.25); % m/s^2
IMU_INI.SAMPLING_TIME = MPU60X0.SAMPLING_TIME;

```

Para poner en marcha el conversor ADC, usamos el CONFIG_MCP3428.m:

```
% Slave address for ADC converter MCP3428
MCP3428.ADDRESS = bin2dec('01101000');
MCP3428.ADDRESS_WRITE = hex2dec('D0');
MCP3428.ADDRESS_READ = hex2dec('D1');

% Configuration register for CH1, OSM, 240 SPS, PGA=1 (12 bits)
MCP3428.CONFIG_REG_100 = bin2dec('10000000');
% Configuration register for CH2, OSM, 240 SPS, PGA=1 (12 bits)
MCP3428.CONFIG_REG_200 = bin2dec('10100000');
% Configuration register for CH3, OSM, 240 SPS, PGA=1 (12 bits)
MCP3428.CONFIG_REG_300 = bin2dec('11000000');
% Configuration register for CH4, OSM, 240 SPS, PGA=1 (12 bits)
MCP3428.CONFIG_REG_400 = bin2dec('11100000');
% Configuration register for CH1, OSM, 60 SPS, PGA=1 (14 bits)
MCP3428.CONFIG_REG_101 = bin2dec('10000100');
% Configuration register for CH2, OSM, 60 SPS, PGA=1 (14 bits)
MCP3428.CONFIG_REG_201 = bin2dec('10100100');
% Configuration register for CH3, OSM, 60 SPS, PGA=1 (14 bits)
MCP3428.CONFIG_REG_301 = bin2dec('11000100');
% Configuration register for CH4, OSM, 60 SPS, PGA=1 (14 bits)
MCP3428.CONFIG_REG_401 = bin2dec('11100100');
% Configuration register for CH1, CCM, 240 SPS, PGA=1 (12 bits)
MCP3428.CONFIG_REG_110 = bin2dec('10010000');
% Configuration register for CH2, CCM, 240 SPS, PGA=1 (12 bits)
MCP3428.CONFIG_REG_210 = bin2dec('10110000');
% Configuration register for CH3, CCM, 240 SPS, PGA=1 (12 bits)
MCP3428.CONFIG_REG_310 = bin2dec('11010000');
% Configuration register for CH4, CCM, 240 SPS, PGA=1 (12 bits)
MCP3428.CONFIG_REG_410 = bin2dec('11110000');
% Configuration register for CH1, CCM, 60 SPS, PGA=1 (14 bits)
MCP3428.CONFIG_REG_111 = bin2dec('10010100');
% Configuration register for CH2, CCM, 60 SPS, PGA=1 (14 bits)
MCP3428.CONFIG_REG_211 = bin2dec('10110100');
% Configuration register for CH3, CCM, 60 SPS, PGA=1 (14 bits)
MCP3428.CONFIG_REG_311 = bin2dec('11010100');
% Configuration register for CH4, CCM, 60 SPS, PGA=1 (14 bits)
MCP3428.CONFIG_REG_411 = bin2dec('11110100');

%% General parameters
% Reference voltage
MCP3428.VREF = single(2.048); % V
% Samples per second
MCP3428.SPS = uint8(240); % Hz
% Number of bits
MCP3428.NBITS = uint8(12);
% Conversion mode
% / 0: Continuous conversion / 1: One-shot conversion
MCP3428.MODE = boolean(1);
% Configuration bytes for each channel
MCP3428.CFG = [MCP3428.CONFIG_REG_100 MCP3428.CONFIG_REG_200 ...
    MCP3428.CONFIG_REG_300 MCP3428.CONFIG_REG_400]';
% Programmable gain amplifier for each channel
MCP3428.PGA = uint8([1 1 1 1]);
% Total number of channels converted
MCP3428.NUM_CH = 2;

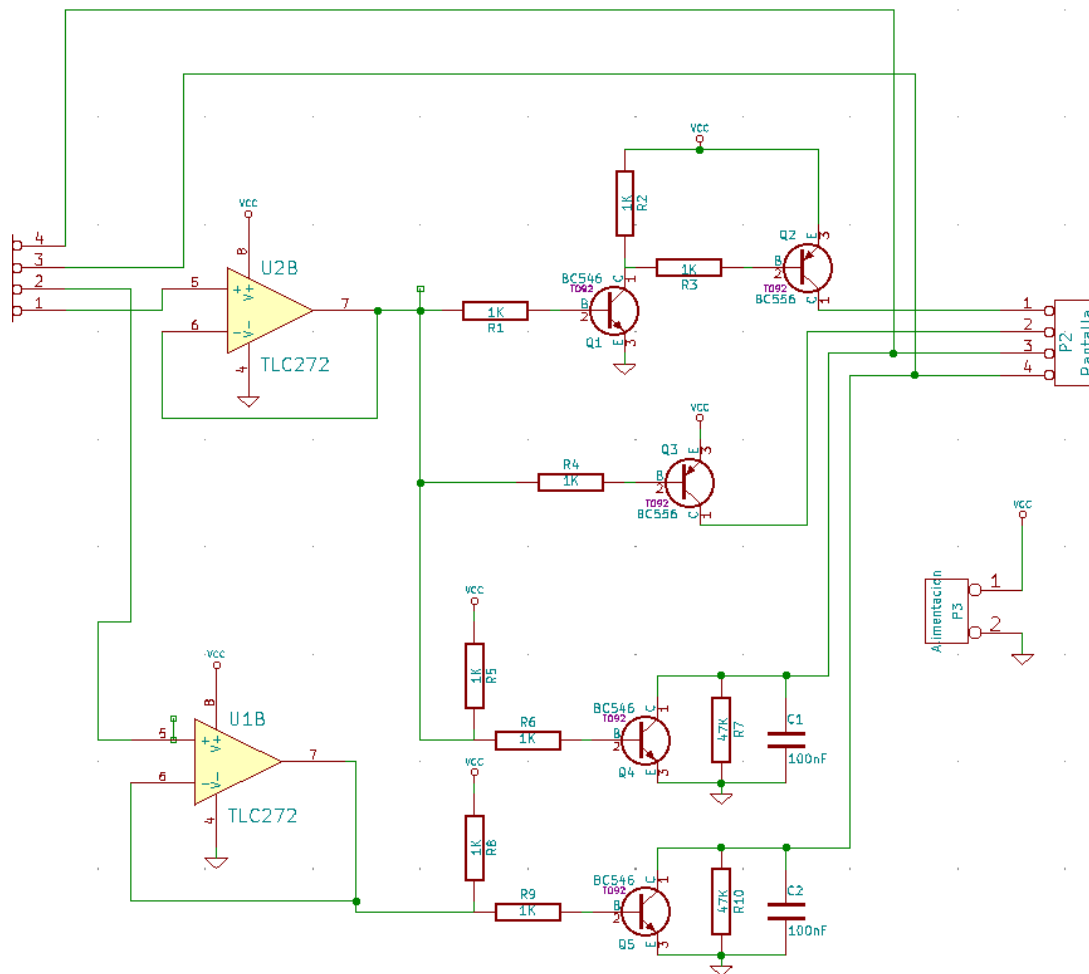
%% Ini Setup
ADC_INI.CHANNEL1 = 0;
ADC_INI.CHANNEL2 = 0;
ADC_INI.CHANNEL3 = 0;
ADC_INI.CHANNEL4 = 0;
```

DOCUMENTO 2:

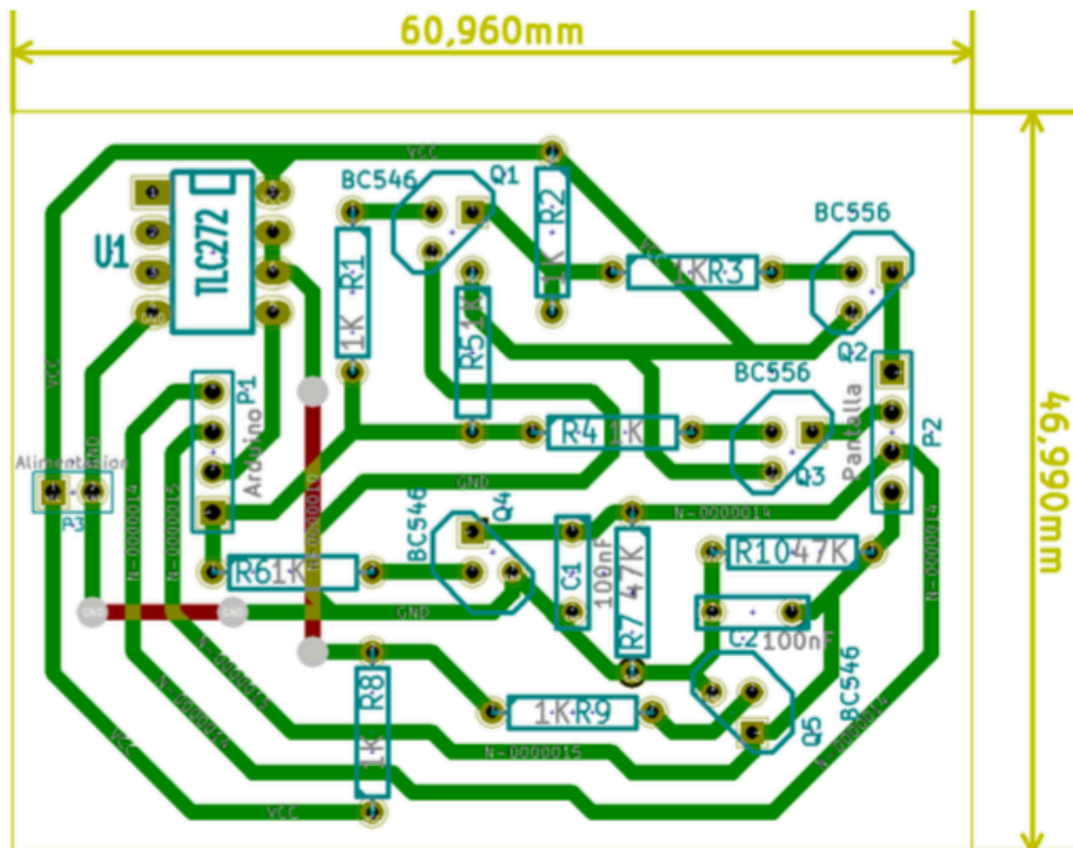
PLANOS

1. Circuito de adaptación de la pantalla resistiva

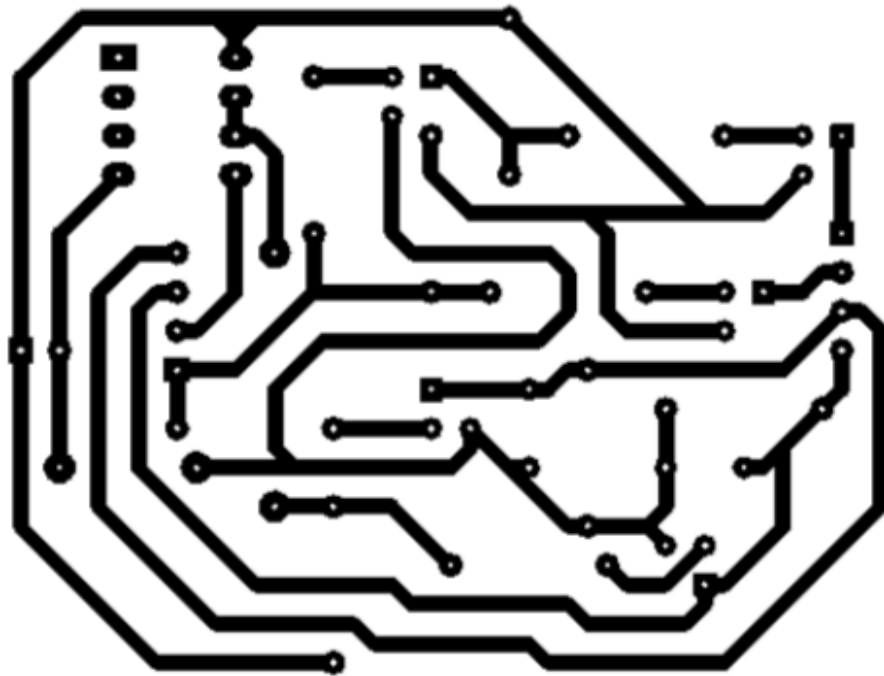
1.1. Esquema final de los componentes:



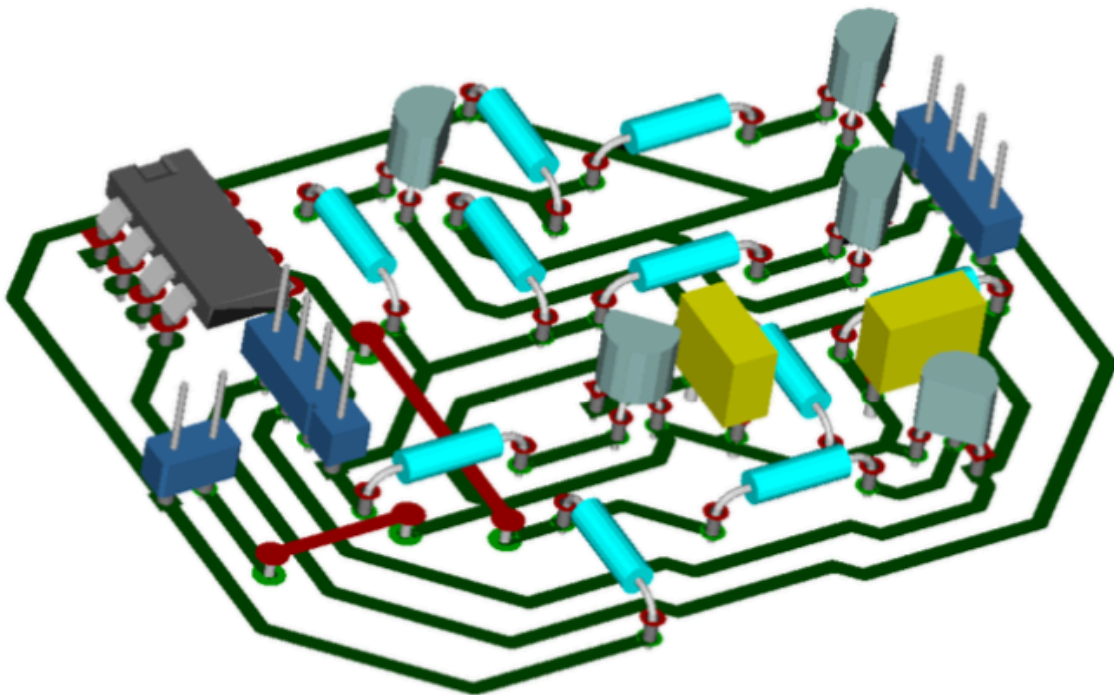
1.2. Diseño del circuito sobre placa: (*)



1.3. Plano de pistas: (*)



1.4. Circuito de adaptación completo en perspectiva: (*)



(*) Nota: en los tres últimos planos, falta el amplificador operacional “buffer” del DRIVEB, el cual está situado fuera de la placa del circuito de adaptación en una placa protoboard.

DOCUMENTO 3:

PRESUPUESTOS

1. Mediciones

En este apartado, se indicarán las diferentes partes que integran el proyecto, agrupadas de forma homogénea en distintas partidas, indicando las cantidades de cada parte.

1.1. Material

1.1.1. Hardware

Material	Cantidad
Piezas impresas 3D	11
Impresora 3D <i>Stratasys</i>	1
Pantalla resistiva	1
Juntas universales	3
Varilla de acero (1 m)	1
Servos LX-16A	4
Placa de metacrilato (85 x 40 cm)	1
Raspberry Pi	2
IMU	1
Conversor ADC	2

1.1.2. Software

Material	Cantidad
Licencia de uso académico Matlab	1
Licencia de uso académico Simulink	1

1.2. Proyecto de ingeniería

1.2.1. Investigación

Tarea	Horas
Búsqueda de información	10

1.2.2. Diseño y construcción

Tarea	Horas
Diseño de todas las piezas con SolidEdge	20

Impresión 3D de todas las piezas	120
Diseño y montaje del circuito de adaptación de la pantalla	5
Construcción de la estructura sin sensores ni actuadores	150
Construcción de la estructura con sensores y actuadores	20

1.2.3. Prueba de la estructura

Tarea	Horas
Prueba de sensores y obtención de medidas	80
Prueba de los servos (actuadores)	20

1.2.4. Documentación del proyecto

Tipo de documento	Horas
Anexo A	1
Anexo B	10
Resumen	2
Memoria	75
Presentación	5

2. Precios unitarios

En este apartado, se indicará el costo unitario de cada parte del apartado anterior. Incluye el precio unitario del material y de las horas empleadas.

2.1. Material

2.1.1. Hardware

Material	Precio unitario €
Piezas impresas 3D	35
Impresora 3D <i>Stratasys</i>	2700
Pantalla resistiva	86,95
Juntas universales	45
Varilla de acero (1 m)	45
Servos LX-16A	13,13

Placa de metacrilato (85 x 40 cm)	10
Raspberry Pi	35,75
IMU	30
Conversor ADC	7,50

2.1.2. Software

Material	Precio unitario €
Licencia de uso académico Matlab	250
Licencia de uso académico Simulink	250

2.2. Proyecto de ingeniería

2.2.1. Investigación

Tarea	Precio unitario €
Búsqueda de información	30

2.2.2. Diseño y construcción

Tarea	Precio unitario €
Diseño de todas las piezas con SolidEdge	30
Impresión 3D de todas las piezas	30
Diseño y montaje del circuito de adaptación de la pantalla	35
Construcción de la estructura sin sensores ni actuadores	35
Construcción de la estructura con sensores y actuadores	35

2.2.3. Prueba de la estructura

Tarea	Precio unitario €
Prueba de sensores y obtención de medidas	35
Prueba de los servos (actuadores)	35

2.2.4. Documentación del proyecto

Tipo de documento	Precio unitario €
Anexo A	35
Anexo B	35

Resumen	35
Memoria	35
Presentación	35

3. Sumas parciales

Este apartado se configura en base a los dos apartados anteriores, indicando las cantidades de cada una de las partes, su precio unitario y el importe parcial de cada una de ellas.

3.1. Material

3.1.1. Hardware

Material	Cantidad	Precio unitario €	Total €
Piezas impresas 3D	11	35	3853
Impresora 3D <i>Stratasys</i>	1	2700	2700
Pantalla resistiva	1	86,95	86,95
Juntas universales	3	45	135
Varilla de acero (1 m)	1	45	45
Servos LX-16A	4	13,13	52,52
Placa de metacrilato (85 x 40 cm)	1	10	10
Raspberry Pi	2	35,75	71,50
IMU	1	30	30
Conversor ADC	2	7,50	15

TOTAL: 6998,97€

3.1.2. Software

Material	Cantidad	Precio unitario €	Total €
Licencia de uso académico Matlab	1	250	250
Licencia de uso académico Simulink	1	250	250

TOTAL: 500€

3.2. Proyecto de ingeniería

3.2.1. Investigación

Tarea	Horas	Precio unitario €	Total €
Búsqueda de información	10	30	300

TOTAL: 300€

3.2.2. Diseño y construcción

Tarea	Horas	Precio unitario €	Total €
Diseño de todas las piezas con SolidEdge	20	30	600
Impresión 3D de todas las piezas	120	30	3600
Diseño y montaje del circuito de adaptación de la pantalla	5	35	175
Construcción de la estructura sin sensores ni actuadores	150	35	5250
Construcción de la estructura con sensores y actuadores	20	35	700

TOTAL: 10325€

3.2.3. Prueba de la estructura

Tarea	Horas	Precio unitario €	Total €
Prueba de sensores y obtención de medidas	80	35	2800
Prueba de los servos (actuadores)	20	35	700

TOTAL: 3500€

3.2.4. Documentación del proyecto

Tarea	Horas	Precio unitario €	Total €
Anexo A	1	35	35
Anexo B	10	35	350

Resumen	2	35	70
Memoria	75	35	2625
Presentación	5	35	175

TOTAL: 3255€

4. Presupuesto general

En este apartado se indicarán cada una de las partidas parciales con sus correspondientes costos, y finalmente, la suma de todas ellas, que constituyen el costo total del proyecto.

Material	Coste €
Hardware	6998,97
Software	500

Proyecto de ingeniería	Coste €
Investigación	300
Diseño y construcción	10325
Prueba de la estructura	3500
Documentación del proyecto	3255

TOTAL: 24878,97€