Universidad de Valladolid

ESCUELA DE INGENIERÍAS INDUSTRIALES

UNIVERSIDAD DE VALLADOLID

ESCUELA DE INGENIERIAS INDUSTRIALES

Grado en Ingeniería mecánica

Diseño, prototipado y puesta a punto de un rascacielos a escala provisto de excitador y disipador pendulares

Autor:

Prieto García, Marcos

Tutores:

Lorenzana Ibán, Antolín

Magdaleno González, Álvaro

C.A., I.T., M.M.C. y Teoría de Estructuras

Valladolid, junio 2020

Resumen

En este Trabajo de Fin de Grado se realiza el diseño y construcción de un prototipo de un rascacielos a escala, sobre el que se instala un sistema motorizado capaz de inducir fuerzas armónicas similares a las que induciría el desprendimiento de torbellinos debidos a la acción del viento. También se instala un disipador inercial de vibraciones (TMD). En este experimento se pretende mostrar la influencia de un TMD sobre las estructuras de manera sencilla y visual.

Con el fin de realizar este ensayo se utilizan una serie de elementos como el excitador, el TMD, el amortiguador o el bloqueador, que ha sido necesario fabricar. Estos componentes han de ir colocados sobre la estructura, para ello se han diseñado unos carenados que los soportan y revisten la estructura. Para controlar la maqueta se ha desarrollado un programa de Arduino. Por último, se han cableado aquellos elementos que necesitan ser alimentados.

Palabras cables: Prototipo, Diseño, Arduino, Carenados, TMD.

Abstract

Present end-of-degree project explains the design and construction of a skycraper prototype on scale. A motorized system is able to introduce armonic forces similar to the influence that the vortex shedding produced by an air stream is installed above the structure. The experiement is intented to show Tuned Mass Damper influence on the structure in an easy and visual way.

In order to make this test, a group of elements are necesary to be produced, such as a shaker, the TMD, a shock absorber or an element that blocks the TMD. These components have to be placed on the prototype. For that reason three diferent fairings have been designed, aimed at holding them and covering the structure. The test is controlled by a program developed in Arduino. Finally, wiring some of these elements to a power source is necessary in order to achieve their purpose.

Keywords: Prototype, Design, Arduino, Fairings, TMD.

ÍNDICE

1. Introducción

1.1. Motivación del trabajo

A lo largo de la historia, el comportamiento de las construcciones ante las posibles cargas, tanto estáticas como dinámicas, a las que se pueden ver sometidas, ha sido objeto de estudio para conseguir hacer unos edificios cada vez más resistentes frente a ellas.

Con este fin se han ido desarrollando cambios en los materiales y las técnicas de construcción. Muchos de estos cambios se han descubierto en el pasado, pero no se les ha encontrado aplicaciones hasta tiempos más recientes [1].

Es conocido que los griegos adoptaron ciertas medidas en sus construcciones, como hacer zapatas individuales para hacer sus edificios más dúctiles y que fuesen más duraderos y resistentes a los terremotos. Y el resto de las civilizaciones, que les han seguido, han ido implementando sus propias mejoras, desde los romanos hasta nuestros tiempos [1].

Este hecho ha tomado más relevancia desde finales del siglo XIX con la construcción de los primeros rascacielos, ya que este tipo de estructuras son más sensibles ante estos fenómenos por su gran altura. El diseño de los rascacielos de primera generación estaba basado en la fuerza, su construcción se basaba en muros cortina con un pesado revestimiento de mampostería, lo que les dotaba de una gran rigidez, y, por tanto, los hacía más resistentes ante este tipo de fenómenos. [2]

Figura 1.1. Home Insurance Building (Chicago), considerado el primer rascacielos de la historia [3].

El avance en la tecnología de la ciencia de materiales ha permitido el uso de secciones de acero y hormigón, lo que conlleva que las nuevas generaciones de rascacielos hayan podido ser más económicos y eficientes estructuralmente [2]. Este hecho es aún más visible en nuestros días, dada la tendencia a hacer rascacielos cada vez más esbeltos. Entendiendo como esbeltez de un edificio la relación entre la sección de la estructura y su altura [4].

Esta característica los hace todavía más sensibles ante posibles cargas dinámicas provocadas por fenómenos naturales, tales como terremotos o fuertes vientos, porque son más propensas a desarrollar fenómenos de vibraciones de gran amplitud.

Esto muestra la importancia de saber cuáles son los modos de vibración de estas estructuras, para evitar que estén dentro del rango de frecuencias al que se pueden ver sometidas por estas cargas dinámicas. Para que no entren en resonancia con ellas, porque si esto ocurriese puede condicionar su uso e incluso su integridad. [4]

En el caso concreto de este trabajo la excitación que se simula en la maqueta es la generada por el desprendimiento de los torbellinos de viento al chocar este con el rascacielos. Esta excitación se simula con el movimiento de un servomotor que tiene una masa excéntrica.

Se va a explicar más profundamente el fenómeno del desprendimiento de los torbellinos que hace oscilar a los rascacielos.

La denominada calle de torbellinos de Kármán consiste en un desprendimiento alternado y periódico de torbellinos, que se produce en un flujo de corriente, aguas abajo de una estructura no fuselada situada perpendicularmente a dicha corriente uniforme que incide sobre la estructura. [5]

Figura 1.2. Calle de vórtices de von Kármán. [6]

La alternancia de estos torbellinos hace aparecer una serie de fuerzas transversales a la dirección del flujo y periódicas que pueden hacer oscilar la estructura. Para que se produzca un movimiento desmesurado de la estructura no es necesario que la velocidad de la corriente de aire sea muy elevada. Porque si la frecuencia del desprendimiento de estos torbellinos coincide con alguna frecuencia propia de la estructura hace entrar a esta en resonancia, pudiendo provocar un fallo catastrófico de la misma.

En general estas oscilaciones no son peligrosas porque se estudian previamente los efectos sobre las estructuras y se toman medidas para que las puedan resistir y no poner en riesgo la integridad del edificio. Las oscilaciones cuando son controladas no son peligrosas, de hecho, es así como deben responder las estructuras. Pero sí puede resultar muy incómodo para sus ocupantes, pudiendo incluso llegar a ser problemático para realizar una vida laboral o doméstica normal.

Por ejemplo, el Taipei 101, Figura 1.3, con 509 m de altura, ocasionalmente cuando se ve azotado por fuertes vientos puede llegar a oscilar una amplitud de 0,35 m para cada lado, lo que resulta en una aceleración de 0,2 ms^{-2}. Si la exposición es por un periodo corto de tiempo, la mayoría de las personas lo toleran sin problemas, pero si la exposición es larga, provoca mareos y náuseas. Dicho edificio está preparado, para llegar a tener amplitudes de hasta 0,15 m durante los tifones más fuertes [7]

Figura 1.3. Imagen del Taipei 101, en la ciudad de Taiwán. [8]

Esto es posible gracias al estudio y la investigación en métodos que hacen disminuir las oscilaciones de estos edificios. Algunas de estas innovaciones son la implantación de TMD (*Tuned Mass Damper*), Amortiguador de Masa Sintonizado, o la construcción de un núcleo de hormigón en vez de acero, haciendo los edificios mucho más robustos y resistentes.

Un TMD es un sistema que tiene como fin reducir la oscilación provocada sobre una estructura, como puede ser un rascacielos o un puente. El sistema suele estar formado por una gran masa y elementos de amortiguación. Su frecuencia de oscilación ha de sintonizarse para que coincida con alguna frecuencia de resonancia de la estructura principal. En el caso de los rascacielos, que es el más similar al que se ha realizado en este trabajo, suelen ir colocados en los pisos superiores. [9]

Antes de construir un edificio como estos, es necesario hacer una serie de simulaciones con maquetas a escala, para poder ver si el diseño realizado es adecuado, y medir las frecuencias propias de la estructura. Sobre estas maquetas se hacen ensayos en túneles de viento para ver la influencia de estos fenómenos sobre el futuro edificio y así también poder probar la labor que realizarían estas medidas de retención de la oscilación.

Es aquí donde entra el interés de este TFG, que permite simular de una manera sencilla y visual el efecto que produciría una corriente de viento que incidiese sobre un edificio. También se muestra la influencia que tiene el TMD sobre la estructura, para demostrar el funcionamiento y la eficiencia de este tipo de sistemas, que son utilizados a escala real en rascacielos existentes.

Quizá el TMD más conocido sea el del Taipei 101, Figura 1.4, en Taiwán, ya que puede visitarse por el público. Es una gran bola de 6 m de diámetro y de 660 toneladas de peso, lo que la convierte en el mayor del mundo. Esta bola se está instalada en el centro de la planta 87 y está colgada desde la planta 91 de este edificio, mediante 4 conjuntos de cables de acero, con 4 cables cada conjunto. Además, cuenta con 8 amortiguadores en su parte inferior para controlar su movimiento en caso de que el edificio se vea sometido a algún percance de los ya comentados. [9]

Figura 1.4. Imagen del TMD instalado en el rascacielos Taipei 101. [10]

Como dato curioso, se puede observar el vídeo del siguiente enlace, en el que se ve a este TMD en pleno funcionamiento, ya que el 20 de abril de 2015 hubo un terremoto en Taiwán, y este sistema de seguridad demostró su utilidad y eficacia. https://www.youtube.com/watch?v=Rrv8JM0LB_c

1.2. Antecedentes

Este proyecto es una evolución de un TFG previo.[11] En este TFG anterior se simula un edificio de dos plantas, mientras que en este la maqueta simula ser un rascacielos de 100 pisos de altura. Se busca ver cómo responde una estructura con una morfología muy distinta.

En cuanto a la diferencia entre las maquetas, es destacable el hecho de que en este trabajo se utiliza un prototipo de un edificio mucho más esbelto que nos va a permitir estudiar otro tipo de comportamiento de la estructura. Al ser la parte principal de la maqueta una barra de aluminio de pequeña sección, de $8x10^{-4}$ m^2 y una barra de gran longitud, 3 m, se podrá ver fácilmente el primer modo de vibración de un sistema barra empotrada-libre.

1.3. Objetivos y descripción del trabajo

Por lo descrito anteriormente, este Trabajo Final de Grado tiene como objetivos principales:

- Diseño y construcción del prototipo de un rascacielos a escala.
- Simulación del efecto que provoca el desprendimiento de los torbellinos.
- Atenuación de la oscilación de la maqueta.

Poniendo estos tres objetivos en conjunto, podemos resumir que lo buscado en este TFG ha sido la construcción de un prototipo de un rascacielos a escala para la realización de ensayos dinámicos. En los que se genera una oscilación del prototipo mediante un excitador pendular y se pretende observar la eficacia que tienen los sistemas de control de la excitación, el disipador pendular o TMD.

Para poder llevarlo a cabo, se han de realizar una serie de objetivos secundarios que implican el diseño, la construcción y el desarrollo de una serie de elementos.

- Montaje de la estructura base, que simula ser el núcleo del rascacielos y sus cimientos.
- Diseño en CATIA V5 y fabricación de los 3 tipos de carenados que revisten la estructura, carenado genérico, porta servo y porta TMD.
- Diseño y construcción del TMD (*Tuned Mass Damper*).
- Diseño y construcción del bloqueador del TMD.
- Construcción del excitador.
- Aprendizaje de lenguaje de programación de Arduino para el desarrollo de los programas necesarios para la realización del ensayo. [12]
- Alimentación del Arduino y los servomotores.
- Sintonización del TMD y el excitador con la estructura.
- Construcción y posicionamiento del amortiguador.

Para generar la excitación que simula el efecto del desprendimiento de torbellinos de viento sobre la estructura se ha utilizado un servomotor MG995 con una masa colocada al final de la barra unida a su eje, es este movimiento pendular el que genera la inercia suficiente para provocar la oscilación de la maqueta. El control de este servomotor se realiza con una placa Arduino, mediante un programa que genera el movimiento pendular del servo. [13]

Para controlar la oscilación de la estructura, se dispone de un TMD (*Tuned Mass Damper*), cuyo funcionamiento se detalla más profundamente a lo largo de este documento. Puede definirse de manera sencilla como un péndulo con una masa, que oscila de manera sintonizada con la estructura para amortiguar su movimiento y disminuir la amplitud de oscilación.

Figura 1.5. Estructura desnuda de la maqueta.

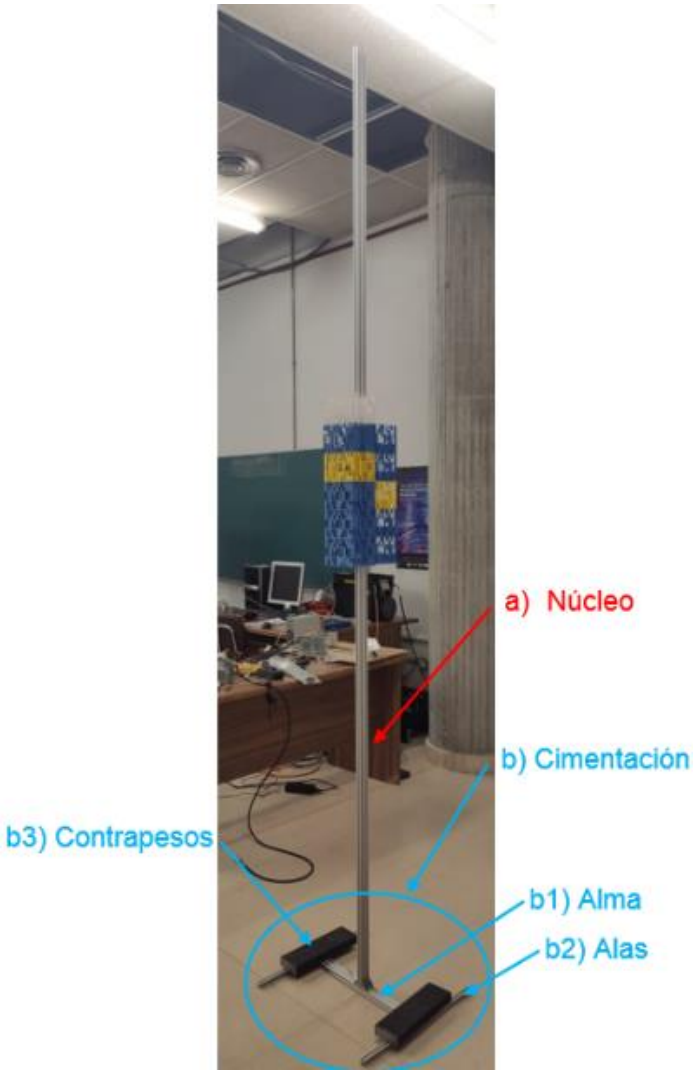


Figura 1.6. Estructura de la maqueta, revestida y lastrada. Se indican algunas de las partes del prototipo.

Se tiene la posibilidad de bloquear y desbloquear el TMD para poder ver la influencia que tiene en la oscilación de la maqueta. Este bloqueo se realiza mediante el bloqueador, formado por el servomotor SG90, controlado por la placa de Arduino, y el brazo bloqueador que se trata de una barra que en su extremo cuenta con un trozo de goma que es lo que contacta con el TMD, evitando su movimiento. [14]

Con el fin de dotar a la estructura de un aspecto exterior similar al de un rascacielos, se ha realizado el diseño de los carenados que revisten el núcleo central. Algunos de ellos albergan los componentes necesarios para realizar la simulación, por lo que han de soportar su peso y las cargas generadas por el movimiento de estos.

Por último, otra de las partes más importantes del trabajo es el programa realizado en Arduino, que es el que controla los elementos que generan y controlan la oscilación de la estructura, cuyas características y funciones se comentarán en el apartado destinado al software.

Este programa ha sido desarrollado en conjunto con un grupo de programadores de la Universidad de Oviedo. Ellos, al ser expertos en programación, han realizado el programa encargado de la visualización de las aceleraciones, y una aplicación desde la que poder controlar el ensayo y observar las gráficas de las aceleraciones desde un ordenador o un móvil.

Nosotros nos hemos encargado del desarrollo de la programación que controla los servomotores y de la realización de la parte mecánica, que implica el diseño, construcción y sintonización de aquellos elementos necesarios para llevar a cabo la experimentación.

1.4. Organización del trabajo

En cuanto a la organización de este documento, comienza con una introducción, en la que se habla de la motivación del trabajo, los antecedentes en los que se apoya este TFG y una descripción general del mismo.

A continuación, se enumeran las distintas partes necesarias para la realización de este trabajo. En primer lugar, se detalla la parte estructural y el proceso de diseño seguido para desarrollar el carenado genérico.

El tercer capítulo se centra en la descripción de los componentes mecánicos que forman parte de la maqueta que son responsables de la realización de la experimentación, así como del proceso seguido hasta la obtención de las versiones finales de cada uno de ellos.

Después se explican los circuitos necesarios para alimentar todos los elementos que requieren corriente eléctrica para su funcionamiento. En el quinto capítulo se definen los componentes electrónicos utilizados en el ensayo, su alimentación y cómo se comunican entre ellos.

Posteriormente, se describen los programas utilizados para controlar los elementos usados en el ensayo. Especialmente los desarrollados en Arduino, tanto los de desarrollo propio, como los desarrollados por los programadores con los que se ha colaborado.

En el séptimo y octavo apartado se explica todo lo relacionado con el montaje de la maqueta realizado en los ensayos tomados como referencia y cómo se ha de actuar para la realización del ensayo de manera correcta.

En el siguiente capítulo se exponen las conclusiones a las que se llega y también algunas posibles líneas futuras sobre las que se podría continuar desarrollando este trabajo. Después se expone la bibliografía consultada para poder desarrollar este trabajo.

Por último, se encuentran los anexos en los que se incluyen los planos de los elementos utilizados en la maqueta, su montaje en la estructura y los códigos de los programas explicados en el sexto capítulo.

2. Diseño parte estructural

En este primer apartado se va a describir la estructura base de este TFG, y los componentes utilizados para construirla. Esta maqueta está formada por 4 barras de aluminio, de dos tipos de perfiles diferentes, unidos mediante escuadras, que se fijan usando tornillos y tuercas que se introducen en los carriles de los perfiles.

A continuación, vamos a comentar las características más relevantes de estos componentes.

2.1. Perfiles de aluminio

Entre las diversas opciones que podrían considerarse para la construcción de esta maqueta, se ha optado por elegir perfiles de aluminio, como los mostrados en la Figura 1.6. Se trata de un material que reúne las características mecánicas suficientes para la función que desempeña, no es necesario un material muy resistente ya que no va a ser una barra que tenga que soportar una gran carga, ni va a tener una función de responsabilidad.

Además, se busca una configuración de barra, en cuanto a material y forma, a la que se pueda hacer balancear con la oscilación producida por el excitador pendular. Un material más resistente condicionaría la utilización de un péndulo más largo o con una mayor masa, que haría necesario un servomotor más potente y caro.

Otro de los factores que hacen de las barras de aluminio la mejor opción, es que al ser barras de gran longitud si fuesen pesadas su manipulación podría ser complicada. Sin embargo, entre los perfiles comprados el más pesado tiene una densidad lineal de 0.8 Kg/m, lo que facilita su transporte y montaje. Además de ser un material bastante económico.

Por último, era preferible escoger unas barras que tuviesen canales para poder ubicar los elementos necesarios para la realización de la experimentación sin tener que perforar la barra para poder engancharlos.

Estos perfiles los hemos obtenido del catálogo de la empresa Fasten Sistema SI, están hechos de una aleación de aluminio Al Mg Si 0,5 F25 (6060/6063), tiene una densidad de $2.7x10^6$ gr/m^3. Además, cuenta con una capa de anodizado de 15 µm, que le proporciona una gran resistencia a la corrosión y una dureza superficial de entre 250-300 HV, hecho que es realmente beneficioso ya que la dureza del material sin tratar es de 75 HB. [10]

A continuación, se adjunta la tabla de características del material, donde se especifican algunas características a mayores de las ya mencionadas:

Datos técnicos Aluminio anodizado color natural
Technical data Aluminium natural anodized

Longitud estándar **Standard length**	6 m.	**Límite elástico** **Tensile strength**	200 N/mm²
Aleación **Alloy**	Al Mg Si 0,5 F25 (6060/6063)	**Módulo de elasticidad transversal** **Elasticity module E**	aprox. 70000 N/mm²
Densidad **Density**	2,7 gr/cm³	**Dureza Brinell** **Hardness**	75 HB
Espesor anodizado **Anod. layer thickness**	15 µm	**Coeficiente de dilatación** **Coefficient of dilation**	$23,8 \cdot 10^{-6} \cdot K^{-1}$
Dureza anodizado **Anod. layer hardness**	250-350 HV	**Punto de tensión** **Ductile yield**	A5 > 10% - A10 > 8%

Figura 2.1. Características de la aleación de aluminio con el que están hechas las barras del TFG. [15]

El primer tipo de barra utilizado es un perfil básico de 20x20 mm, contamos con dos barras de 0.5 m de este tipo. Estas son las alas (b2, Figura 1.6.) de la base en forma de 'H', sobre la que se encuentra montada la barra principal de la maqueta. Su finalidad es la de dotar de estabilidad a la estructura.

A continuación, se adjunta una tabla donde se especifican las características más relevantes de este perfil:

Perfil básico 20x20 **Basic profile 20x20**

	Kg	Ix cm4	Iy cm4	Wx cm3	Wy cm3
5049	0,44	0,66	0,66	0,66	0,66

Figura 2.2. Características del perfil básico 20x20mm. [15]

Este perfil cuenta con 4 ranuras de 6 mm de abertura, una en cada una de sus caras. El centro de la barra también está vacío, que lo hace más ligero.

En cuanto a la elección del perfil del que será la barra principal de la maqueta, se planteaban dos posibilidades, la de un perfil de 20x40 mm o de 30x60 mm. Finalmente se decidió escoger la opción de 20x40 mm porque al tener menor inercia, resultaría más sencillo hacer oscilar esta barra.

De este tipo de perfil se han utilizado en este trabajo dos barras, una de 3m que es la barra principal de la maqueta, se encuentra en posición vertical y ejerce como núcleo del rascacielos, sobre ella se anclan los carenados del revestimiento. Y otra, de 0.5 m que es el alma (b1, Figura 1.6.) de la base de la estructura, está apoyada sobre uno de sus laterales de mayor superficie. Es sobre ella sobre la que está anclada el núcleo de la estructura.

Estas barras son de un perfil compuesto, cuya forma es la resultante de juntar dos perfiles de 20x20 mm como los explicados en el subapartado anterior.

A continuación, se adjunta una tabla proporcionada por el fabricante donde se detallan más características de este perfil que pueden resultar útiles para este trabajo:

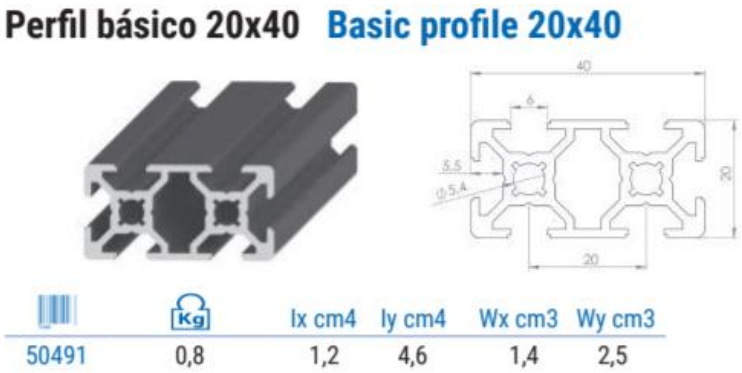


Figura 2.3. Características del perfil básico 20x40mm. [15]

Entre estas características cabe destacar la diferencia que existe entre las inercias de sus ejes principales X e Y, ya que la inercia del eje Y es aproximadamente 3.80 veces la del eje X. Esto es buscado ya que se desea que la barra principal de la maqueta sólo oscile según su eje de menor inercia, y lo haga en la menor medida posible sobre su eje de mayor inercia. De este modo se simplificarán tanto la ejecución del excitador como la del absorbedor, al quedar confinado el movimiento a un plano.

Este perfil cuenta con 6 ranuras de 6 mm de apertura, es por estas ranuras por las que se introducen unas tuercas de la misma forma para el montaje de la estructura y para el anclaje de los carenados que soportan a los servos, placas de Arduino y TMD.

2.2. Predimensionado del núcleo

Este apartado justifica la elección de la longitud del núcleo de la maqueta. En base a esta medida, y en función a ella, se van a detallar los cálculos de la primera frecuencia de resonancia teórica de la estructura y de la longitud del TMD.

En un primer momento se barajaban dos posibles longitudes, 3 m o 4 m. Como se ha comentado en el apartado anterior se eligió la opción de 3 m, por resultar más manejable a la hora de transportarla y manipularla.

Antes de realizar los cálculos se ha de definir lo que es la frecuencia natural de un sistema y el fenómeno de resonancia. La frecuencia natural o de resonancia de un sistema, en este caso la estructura base de la maqueta, es una característica propia de dicho sistema, es decir, sólo depende de sus propiedades mecánicas, masa, rigidez y de las condiciones de contorno. Y es independiente de factores externos a él, como pueden ser fuerzas externas que actúen sobre él. [16]

El fenómeno de resonancia se produce cuando la frecuencia de excitación coincide con alguna de las frecuencias propias del sistema. Este fenómeno produce valores máximos de amplitud de oscilación.

En la experimentación de este trabajo se busca generar una oscilación de la estructura mediante un péndulo excitador accionado mediante un servomotor. Para después, súbitamente dejar de excitar la estructura y hacerla parar mediante un TMD, que para resultar efectivo ha de estar sintonizado con la estructura, es decir, su frecuencia de oscilación ha de ser la misma que la frecuencia natural de la estructura.

Por tanto, teniendo fijada ya la longitud de la barra y el tipo de perfil elegido, conocemos las características de esta barra. Se puede calcular ya la frecuencia de resonancia teórica de la barra desnuda, considerando el sistema como una barra empotrada-libre, que puede tener una masa concentrada en su extremo libre, mediante la siguiente expresión:

$$f_n = \frac{1}{2\pi}\left[\frac{3EI}{L^3(M_i + 0.2357m_b)}\right]^{1/2} \qquad \text{Ec 2.1.}$$

Donde:

- f_n: frecuencia natural del sistema.
- E: módulo de Young o de elasticidad del material.
- I: momento de inercia de la barra.
- M_i: masa concentrada en el extremo libre de la barra.
- m_b: masa total de la barra.
- L: longitud de la barra.

Para el caso concreto de este sistema, considerándolo sólo como una barra desnuda y sin ninguna masa colocada en su extremo:

- $E = 70000\,\frac{N}{mm^2} = 7x10^{10}\,\frac{N}{m^2}$
- $I = 1.2\ cm^4 = 1.2x10^{-8}m^4$
- $M_i = 0\ kg$
- $L_b = 3m$
- $m_b = \rho_b \cdot L_b = 0.8\,\frac{kg}{m} \cdot 3m = 2.4kg$

Por tanto:

$$f_n = \frac{1}{2\pi}\left[\frac{3 \cdot 7x10^{10}\,\frac{N}{m^2} \cdot 1.2x10^{-8}m^4}{3^3(0 + 0.2357 \cdot 2.4kg)}\right]^{1/2} = 2.044Hz$$

Una vez calculada la frecuencia de resonancia del primer modo de vibración de la estructura, podemos determinar la longitud que deberá tener el TMD.

Como en este caso el disipador de energía de la estructura es un péndulo, la frecuencia de oscilación de este elemento está determinada por su longitud. Despejando de la expresión del cálculo de esta frecuencia de oscilación, podemos obtener la longitud que ha de tener el péndulo, Ec 2.2.

$$T = 2\pi\sqrt{\frac{l_p}{9.81}} \rightarrow \frac{1}{f_n} = 2\pi\sqrt{\frac{l_p}{9.81}} \rightarrow f_n = \frac{1}{2\pi}\sqrt{\frac{9.81}{l_p}} \rightarrow l_p = \frac{9.81}{(2\pi f_n)^2}$$

$$l_p = \frac{9.81}{(2\pi f_n)^2} \qquad \text{Ec 2.2.}$$

Sustituyendo, por el valor de esta frecuencia de oscilación:

$$l_p = \frac{9.81}{(2\pi f_n)^2} = \frac{9.81}{(2\pi \cdot 2.044)^2} = 0.05947m$$

Estos cálculos son válidos para el caso concreto simplificado que se ha explicado, por eso sólo ha de tomarse como una aproximación. Ya que en caso real que nos encontramos en la maqueta, habría que añadir el peso de los carenados y los elementos que estos soportan repartidos por toda la longitud de la barra. No valdría con la expresión mostrada anteriormente, ya sólo es válida para el caso concreto de una masa puntual colocada en su extremo.

Pero si podemos estimar que será necesario un péndulo de mayor longitud ya que, todos los elementos que se añadan al núcleo harán aumentar la masa del sistema y, por tanto, a la vista de la expresión de la frecuencia natural del sistema, Ec 2.1., esta disminuirá. Y según la relación mostrada en la Ec 2.2. se deduce que habrá de tener mayor longitud.

La longitud final de este péndulo se calculará experimentalmente, una vez este montada la estructura con la configuración de los componentes deseada. Y se explicará en el apartado en que se habla de este elemento.

2.3. Montaje

En este apartado se va a explicar cómo se ha realizado el montaje de la maqueta.

La manera en la que se unen estas barras es mediante escuadras de 20x20x17 mm, que se fijan utilizando tornillos y unas tuercas especiales con la forma de la ranura de los perfiles, que se introducen por ellos. Como los tornillos utilizados son un poco más largos que la profundidad que tienen los canales de los perfiles, para evitar dañarlos se añaden 3 pequeñas arandelas.

Figura 2.4. Base de la estructura sin lastrar y unión entre las barras de la base y entre la base y la barra principal de la estructura.

La unión entre las barras de la base se hace con 2 de estas escuadras en el plano que forma esta base. Se ancla el lado de menor anchura de la barra central de la base con uno de los laterales de las barras de perfil 20x20 mm. La unión entre la base y la barra principal del trabajo se realiza mediante 4 de estas escuadras.

Se pretende que la unión entre las barras que forman la base, y la que es objeto de estudio, que representa el núcleo central del rascacielos, sea lo más similar posible a un empotramiento ideal. Para dotar a la estructura de estabilidad se lastra este conjunto de barras con dos masas de 12.5 kg.

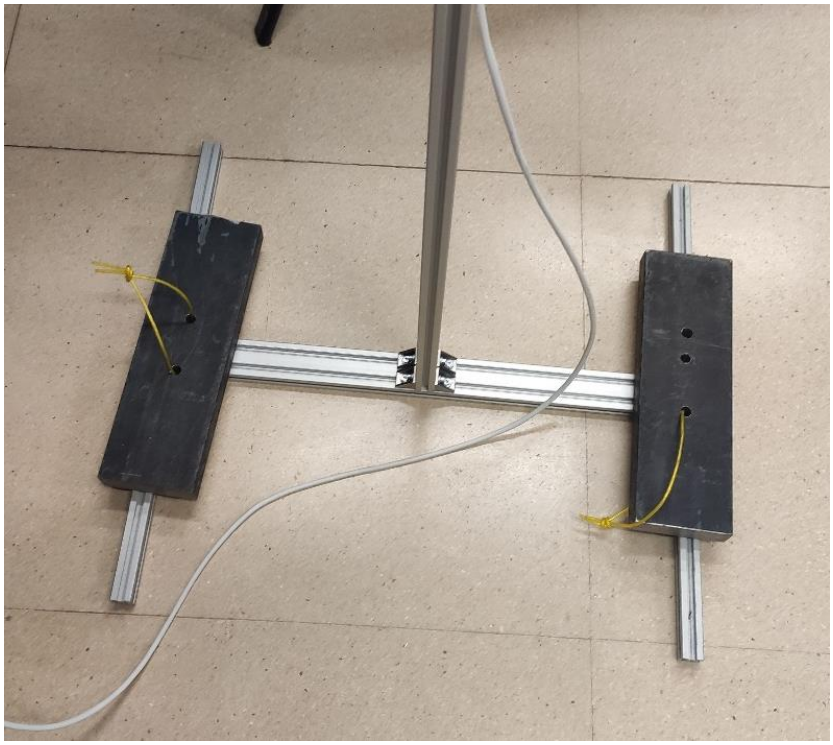


Figura 2.5. Base de la estructura lastrada y unión de la base con la barra principal del trabajo.

Una vez construida la estructura completa colocaremos sobre la barra central los carenados que revestirán la estructura para darle un aspecto más similar al de un rascacielos real.

2.4. Carenado genérico

Este es el carenado base, sobre él se ha ido desarrollando el diseño de los carenados especiales. Es el más simple de los 3 tipos de carenados.

Todos los carenados, tanto las pruebas que finalmente no han formado parte de la estructura final como los que sí que lo han hecho, están diseñados en CATIA V5 y fabricados mediante impresión 3D con una impresora Creality Ender 3 Pro, que es el modelo del que dispone el Departamento de Estructuras de la Escuela de Ingenieros Industriales.

Todos han sido fabricados con el mismo material, PLA (Ácido poliláctico), que es un material de impresión barato y fácil de utilizar.

Figura 2.6. Caja de una de las bobinas con las que se han realizado los carenados.

Se compra en bobinas de 1 kg, de un diámetro de hilo de 1.75 mm. Su temperatura de fusión está entre 145 ºC-160 ºC, el fabricante recomienda imprimir en un rango de temperaturas de 200 ºC-220 ºC. Siendo la temperatura con la que se obtiene la mejor relación velocidad/calidad 210 ºC. [17]

Las principales características de este material son su rápida solidificación, esto es importante sobre todo en piezas de pequeño tamaño, ya que en el proceso de impresión es necesaria que cada capa esté suficientemente fría cuando se va a depositar la siguiente. El PLA tiene una muy pequeña deformación por tensión térmica y así se evita que se produzca un abombamiento de las primeras capas por la diferencia térmica entre la cama, a 80 ºC y el filamento que está entre 200 ºC-220 ºC. [17]

El objetivo buscado en el diseño del carenado genérico es ser un revestimiento para el núcleo central de la estructura, dándole una mejor estética no dejando la barra de aluminio desnuda.

Estos módulos solo van a tener que soportar su peso propio, no se montará sobre ellos ningún elemento externo, y sólo se verán influidos por el movimiento de la estructura completa, que no supone ninguna carga relevante.

La explicación del proceso de diseño seguido en el desarrollo de este carenado se va a realizar explicando los cambios y las características de las versiones más importantes que se han ido realizando, que en general coinciden con aquellas versiones que se han impreso.

- 1ª versión

En un primer momento se consideró hacer unos módulos de unas dimensiones de 60x40 mm cada uno, lo que daría una planta por piso de 60x80mm. Con una altura de 90 mm, que estaría dividida en 3 pisos de 30 mm cada uno.

Este primer prototipo cuenta con unos rigidizadores que ocupaban toda la altura del módulo y con todas las ventanas completamente tapadas para darle mayor robustez a la estructura.

El método de enganche de este primer carenado son dos apéndices en forma de 'T' que se introducen por la ranura de la barra desde su extremo libre. Estos enganches se extienden en toda la altura del carenado, es decir, 90 mm.

Las paredes, rigidizadores y enganches tienen un grosor de 0.6 mm, menos las ventanas, que sólo tienen un grosor de 0.1 mm.

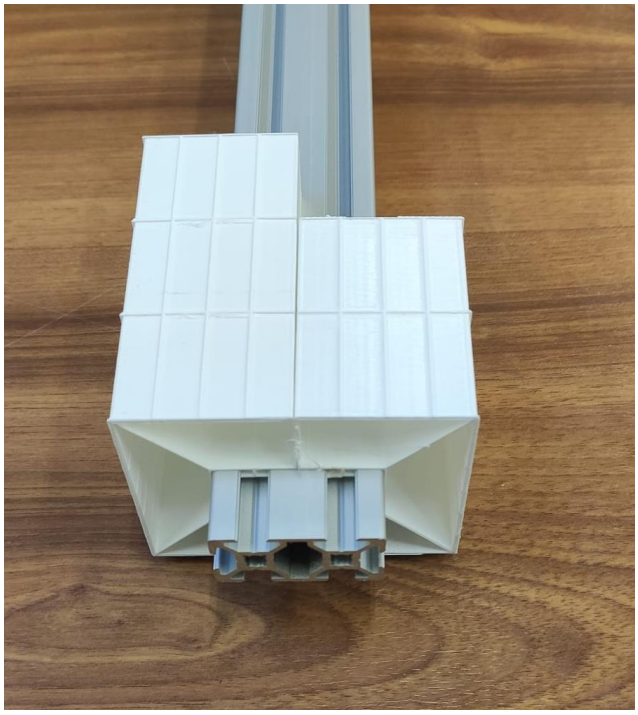


Figura 2.7. 1ª versión impresa del carenado genérico, en la que pueden verse los enganches en forma de 'T'.

En cuanto al diseño de las ventanas exteriores, estas tienen unas dimensiones de 29.4x9.4 mm, con una separación entre ellas de 0.6 mm, tanto en horizontal como en vertical. Mientras que en los extremos de cada uno de los tres laterales que tienen ventanas, el tamaño de los tabiques es de 0.3 mm.

De este modelo se imprimieron dos carenados, para poder ver cómo quedaría el montaje de dos carenados puestos juntos, que forman 3 pisos completos. En la impresión de uno de ellos, falló la impresora y sólo se pudieron imprimir dos de los pisos, como puede verse en las Figuras 2.8. y 2.7.

Esta primera prueba tiene un aspecto importante a mejorar, que es el enganche. Al tener forma de 'T' no permite encajar o desencajar el módulo, sino que hay que deslizarlo hasta el extremo de la barra para poder sacarlo. Esto resulta problemático en la experimentación, puesto que, si se quisieran cambiar los módulos de lugar, habría que sacarlos todos.

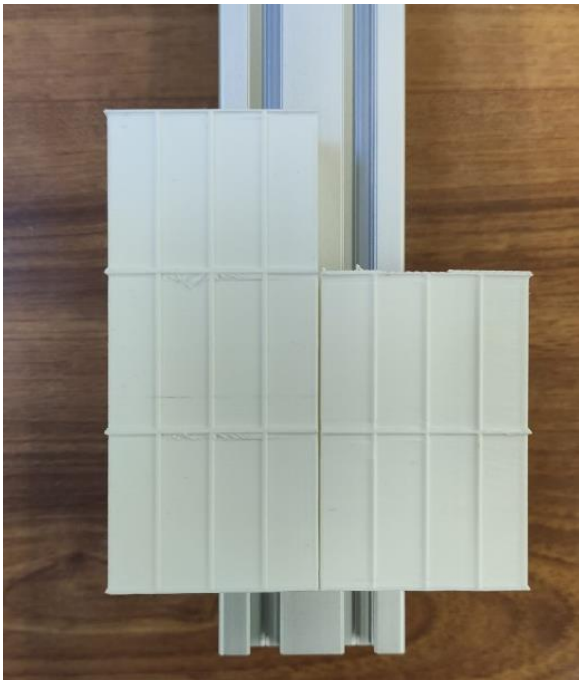


Figura 2.8. Primeras impresiones 3D del carenado genérico insertas en uno de los perfiles de aluminio.

Otro problema que encontramos en los enganches es que no eran capaces de soportar los carenados en una posición determinada por sí mismos, y se deslizan hacia abajo.

- 2ª versión

En esta segunda versión la principal variación respecto a la anterior es en cuanto al enganche, ya que se cambia de forma y longitud. Pero también se realizan algunos cambios en cuanto al diseño exterior y la base del carenado.

Este nuevo enganche es triangular, es sólido, tiene de 6 mm de base, 1.9 mm de altura, y una longitud de 30 mm, coincidiendo con el piso central del carenado. La punta de este triángulo se encuentra en el punto medio de la pared en la que se encuentra, a 10 mm de la pared posterior del carenado. Se trata de un saliente que nos permite clipar y desclipar el módulo.

Otro de los cambios realizados, fue añadir una base 0.6 mm de grosor en la parte inferior del carenado, su finalidad es la de facilitar el asentamiento del carenado a la cama de impresión de la impresora 3D.

Y, por último, se han vaciado algunas de las ventanas, para mejorar el aspecto del carenado. Pero se han mantenido las dimensiones de las ventanas y tabiques.

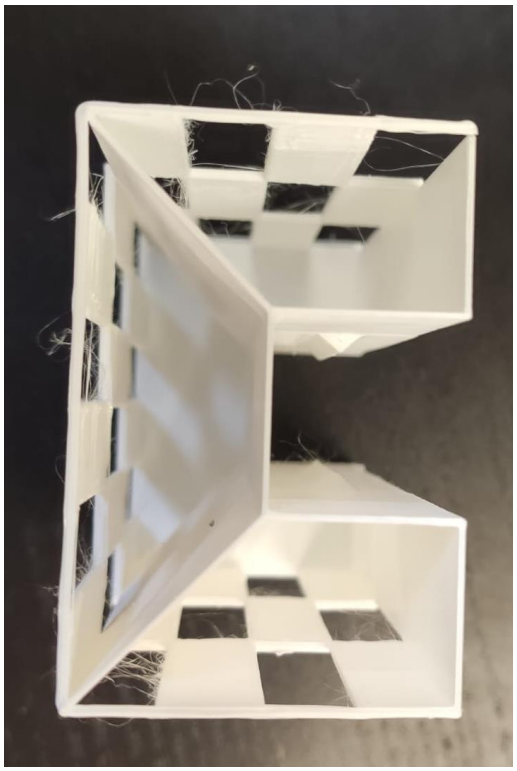


Figura 2.9. Vista superior de la impresión de la 2ª versión del carenado genérico donde se puede apreciar la forma de los enganches.

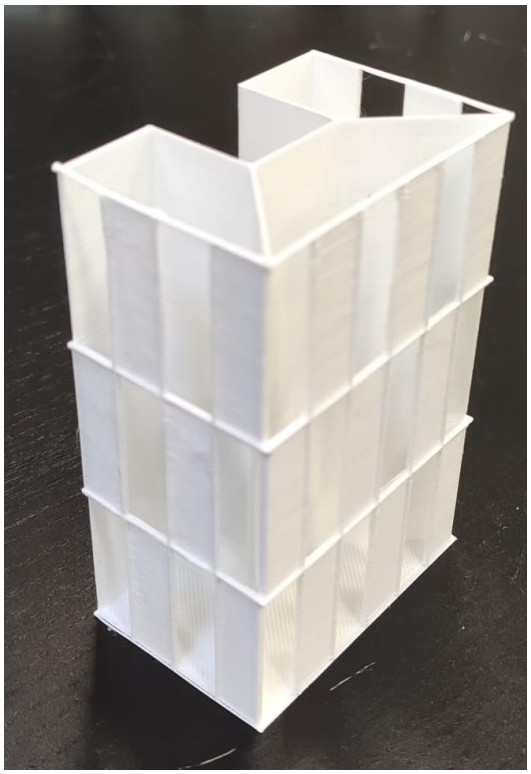


Figura 2.10. 2ª versión del carenado genérico donde se muestra el diseño exterior y la base.

Pese a que este módulo se podía clipar y desclipar, daba problemas, porque para realizar esta acción el carenado ha de deformarse elásticamente para que se pueda encajar la barra en el hueco del carenado.

Y este módulo es muy rígido, ya que cuenta con unos rigidizadores que ocupan toda la altura del carenado, la adicción de la base lo hace aún más rígido y los enganches sólidos no se deforman para entrar en la guía. Todo esto hace necesaria más fuerza para poder hacer el clipaje. Añadido al hecho de tener algunas ventanas vacías y unos tabiques entre ventanas de un grosor de sólo 0.6mm al realizar el clipaje algunos de los tabiques se rompen.

- 3ª versión

Una vez hechas estas dos primeras impresiones se vio que con el tamaño de estos carenados se obtendría un rascacielos demasiado esbelto, por lo que se optó por hacer unos carenados más grandes.

Se pretendió hacer la maqueta proporcional a un rascacielos real, por lo que se toman como ejemplo las torres gemelas, que tenían una altura de 415 m de altura y una planta cuadrada de 64 m de lado. Escalando esto a la maqueta de este trabajo, que tiene 3 m de altura, nos saldrían unos carenados de 462 mm de lado.

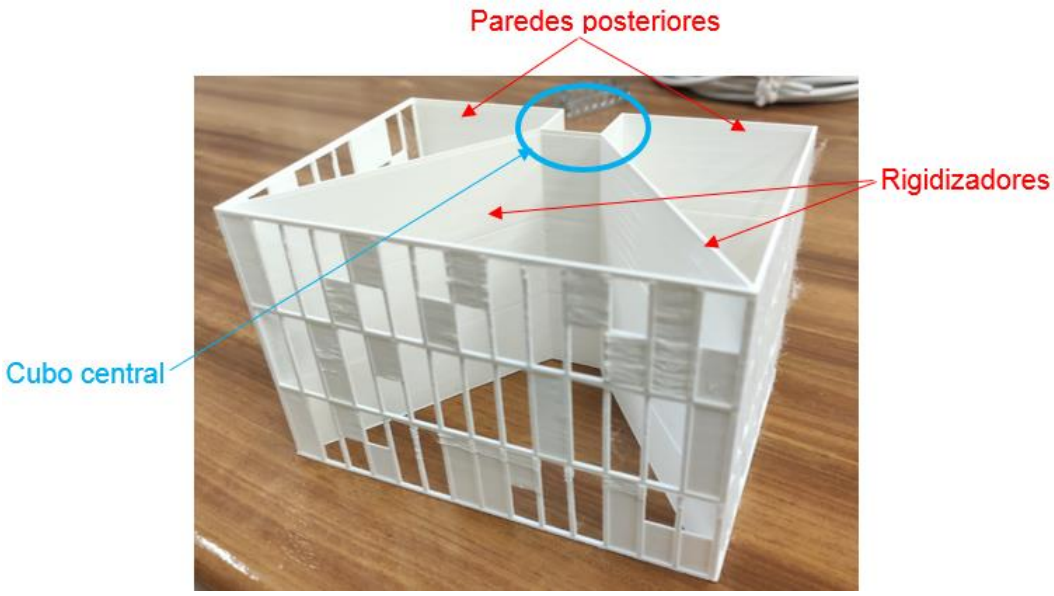


Figura 2.11. Impresión de la tercera versión del carenado genérico. Se indican algunas partes del carenado.

Con la impresora 3D de la que se dispone, no podríamos realizar estos carenados. Ya que, las piezas de mayor tamaño que se podrían realizar son de 220x220 mm de planta. Por lo que hubo que descartar la opción de hacerlo proporcional a las torres gemelas y se optó por una solución intermedia.

Se diseñaron unos módulos de 140x100 mm cada uno que da como resultado una planta rectangular de 140x200 mm. De esta manera, la maqueta final es más proporcionada en relación con su altura, no siendo tan esbelta como lo era antes.

En esta nueva versión, la principal modificación es la del cambio de tamaño del carenado, como ya se ha comentado. Pero también se ha pretendido resolver la problemática encontrada en la 2ª versión, que es la rigidez del módulo, para ello, lo primero que se hizo fue retirar la base, ya que, para mejorar el pegado de las primeras capas de impresión, que era la finalidad de esta base, puede aumentarse la temperatura de la cama de impresión.

Figura 2.12. Vista de detalle de los clipajes del carenado genérico.

En segundo lugar, se han modificado los clipajes. Sigue manteniéndose su tamaño, son dos triángulos de 6 mm de base y 1.9 mm de altura, que tienen una longitud de 30 mm a lo largo del piso central del carenado. Pero se modifica su posición, ya que se acerca 0.4 mm hacia el fondo del hueco en el que se encaja el perfil para que coincida con el centro del carril de la barra.

En esta modificación se ha realizado un vaciado del triángulo, de manera que el grosor de clipaje sea de 0.6 mm en la punta del triángulo. También se ha hecho un hueco en la pared posterior a ellos, es un hueco rectangular de 6 mm de anchura y que se extiende 5 mm más por arriba y por abajo del clipaje. La función de este hueco es dotar de mayor flexibilidad a los clipajes para que puedan deformarse a la hora de introducirse en la guía del perfil.

La otra gran variación que se puede ver en esta versión es en las ventanas. En primer lugar, se ha cambiado el diseño exterior de ventanas tapadas y descubiertas, dejando algunas de ellas vacías a la mitad de su tamaño. Este diseño está basado en la configuración de ventanas del edificio del aulario IndUVA. Cabe destacar que este diseño exterior es el definitivo para todos los carenados que se han impreso, también para los carenados especiales.

Otra modificación llevada a cabo en cuanto a las ventanas es que se ha reducido su tamaño, pasan a tener unas dimensiones de 28x8.35 mm. De esta manera se puede aumentar la anchura de los tabiques entre ventanas a 1.5 mm para que sean más robustos.

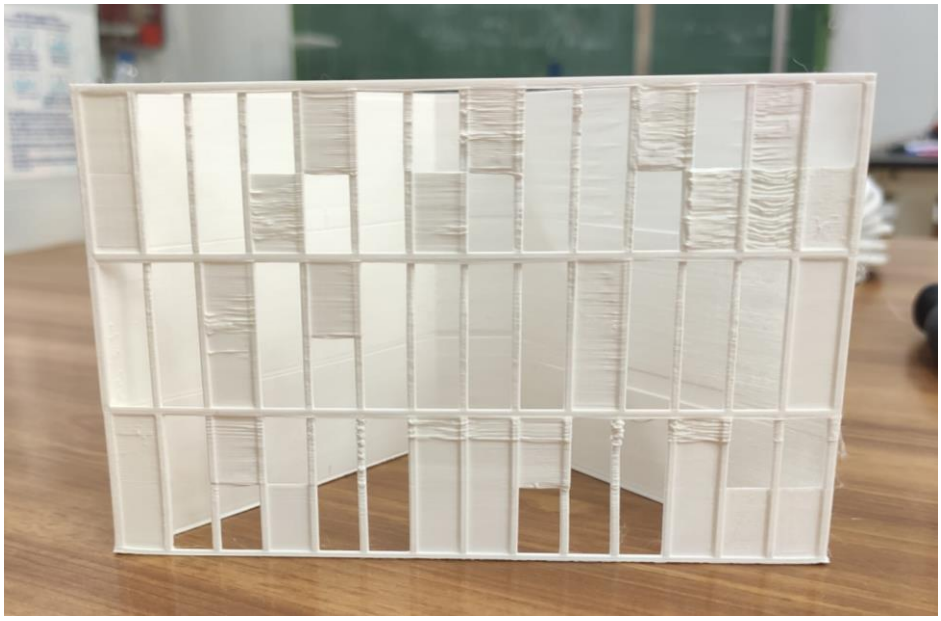


Figura 2.13. Vista frontal de la 3ª versión del carenado genérico, donde se puede observar el nuevo diseño exterior de las ventas.

La última modificación realizada en esta versión es en cuanto al grosor de las ventanas y tabiques. Los tabiques que hasta este momento tenían un grosor de pared de 0.6 mm, pasan a ser de 1.2 mm. Y las ventanas cubiertas, que hasta ahora tenían un espesor de 0.1 mm tendrán serán ahora de 0.6 mm de grosor.

Con todos estos cambios obtenemos un carenado mucho menos rígido que en la versión anterior, y unas ventanas y tabiques más resistentes para que no se rompan al introducirlos en la barra.

Esta tercera versión del carenado genérico no tiene ningún error como los existentes en versiones anteriores. Cumple perfectamente con los requisitos que se le piden, puede anclarse y desanclarse directamente y con facilidad de la posición que se quiera. Es capaz de mantenerse en la posición requerida sin necesidad de ningún elemento de sujeción. Además, el diseño exterior es más agradable que en las primeras versiones. Los tabiques al ser más gruesos no se rompen al manipular el módulo.

El principal aspecto para mejorar que tiene esta versión son sus rigidizadores. Cumplen con su función, que es que el carenado mantenga su forma rectangular, y que las paredes laterales y la frontal no vibren en exceso una vez que se monte en la maqueta y se realice la experimentación. Sin embargo, se utiliza mucho material para este fin, y podría optimizarse este diseño.

- **4ª versión**

En esta versión se intenta optimizar este carenado. Para ello se prueba un carenado genérico exactamente igual que el de la tercera versión, pero en el que se eliminan los rigidizadores por completo, para comprobar si el módulo mantiene su forma y las paredes se mantienen suficientemente rígidas por sí mismas.

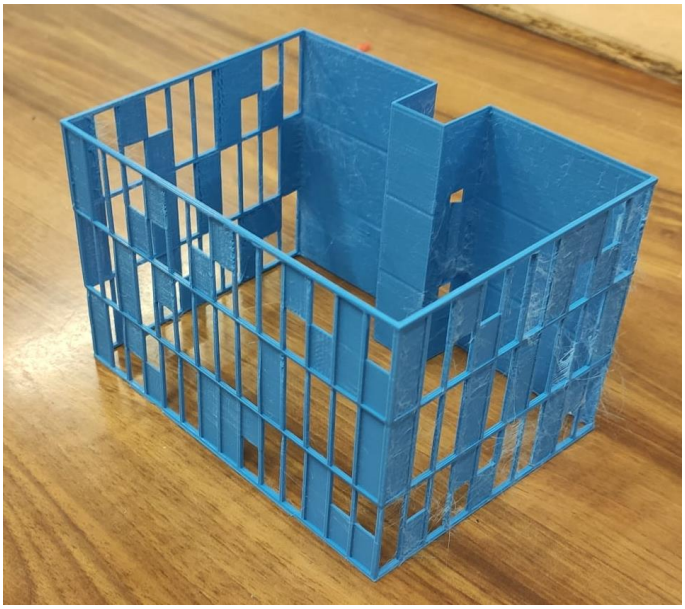


Figura 2.14. Versión sin rigidizadores del carenado genérico.

Esta opción se desestimó rápidamente, porque la ausencia de rigidizadores provocaba que las paredes posteriores de los carenados (las que no tienen ventanas) se combasen, como puede verse en la Figura 2.15.

Además, las paredes laterales y la frontal, al estar sujetas sólo con las paredes posteriores, vibran mucho con el movimiento generado en la experimentación, llegando incluso a producir ruido.

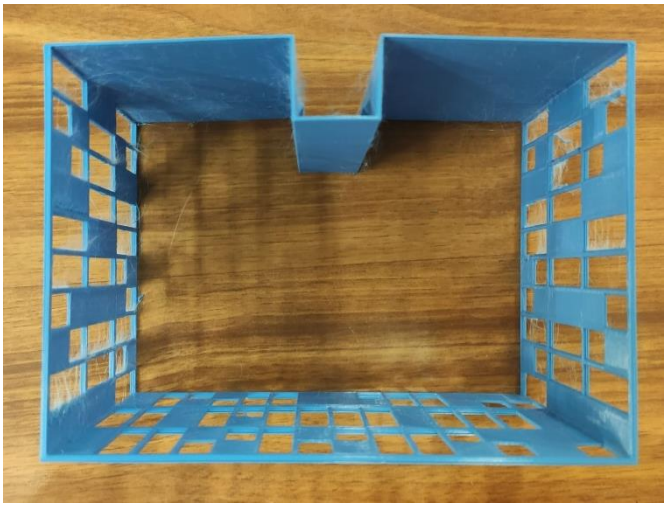


Figura 2.15. Vista superior de la 4ª versión del carenado genérico en la que puede verse como las paredes posteriores están combadas por la ausencia de rigidizadores.

- 5ª versión

Por último, se adoptó una medida intermedia, que los rigidizadores ocupen solamente el piso inferior de los carenados. Para ver si de esta manera obtenemos unos módulos que guardan su forma, y sujeten las paredes laterales y la frontal, a la vez que se ahorra material.

Con esta modificación llegamos al punto final del desarrollo de este carenado, ya que estos rigidizadores nos permiten ahorrar algo de material en cada una de las impresiones y cumple con las condiciones que se buscan para estos elementos.

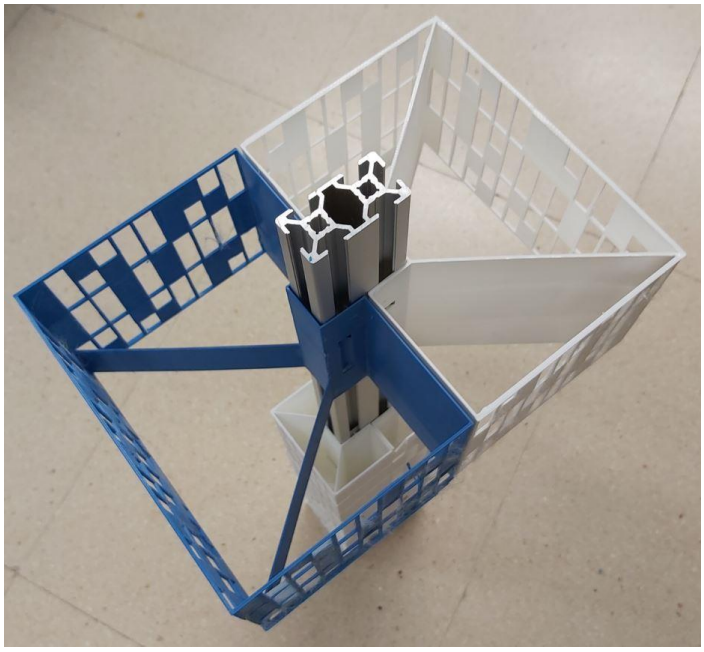


Figura 2.16. El carenado de la izquierda es la 5ª y definitiva versión del carenado genérico. A la derecha 4ª versión del carenado genérico.

De esta versión definitiva se han impreso 4 carenados, aunque en la experimentación final sólo hay dos de ellos montados porque los otros dos se han cedido al equipo de trabajo de la Universidad de Oviedo, con la que se ha colaborado en este TFG, para la réplica de la maqueta que han montado ellos.

3. Diseño parte mecánica

En este apartado se va a comentar el diseño y construcción de los diferentes componentes que forman la parte mecánica del trabajo. En primer lugar, se va a comentar el proceso de diseño de los dos carenados especiales, que son los encargados de albergar a los elementos que generan o reducen la oscilación de la estructura. Después, se explican estos componentes y sus funciones.

3.1. Carenado porta servo

La idea del diseño de este carenado es la de disimular en él, los elementos que ha de soportar. Para ello, se parte del carenado genérico, y se modifica hasta obtenerlo. Se pretende que, al observar la estructura completa, los elementos que han de ir montados sobre ella y que son los encargados de controlar su movimiento, no estén al descubierto, si no que estén ocultos dentro de estos carenados especiales para disimular su presencia.

Este carenado es el encargado de albergar y soportar el *shaker*, que es el servomotor MG995, encargado de generar la excitación.

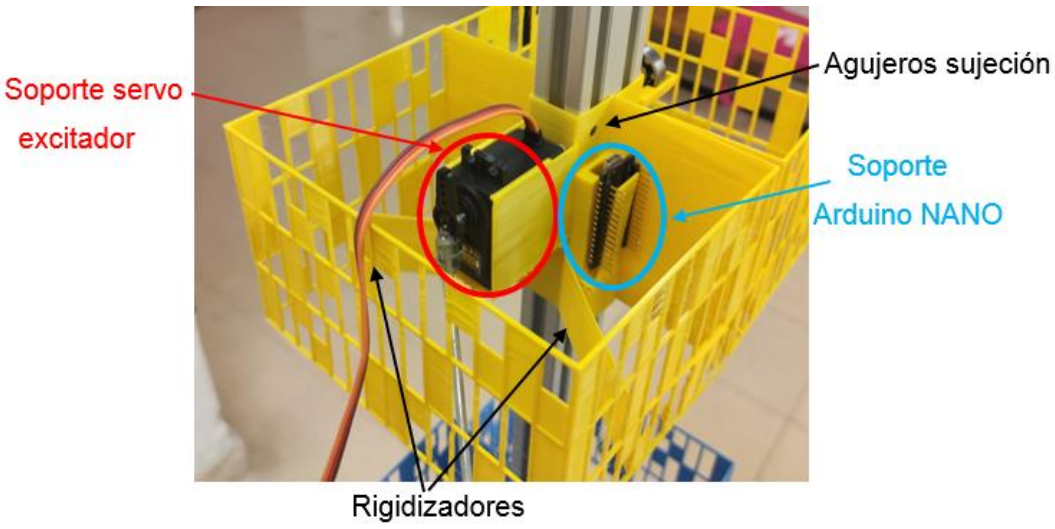


Figura 3.1. Impresión de la 3ª versión del careando porta servo con sus elementos insertos. Se indican las partes más importantes del módulo.

Dado que en las versiones que se han impreso de este carenado no ha sufrido grandes variaciones en cuanto a los elementos que lo forman y el lugar que estos ocupan dentro del propio carenado, se van a explicar aquellos aspectos generales que lo caracterizan. Las modificaciones que se han realizado en estas diferentes versiones han sido principalmente cambios en las dimensiones de los diferentes elementos.

Al igual que los carenados genéricos, dispone de rigidizadores, pero tienen que ser diferentes, ya que es necesario un espacio libre en la parte delantera del carenado para que pueda oscilar el péndulo, por lo que estos rigidizadores no llegan a las esquinas delanteras del módulo. Su extremo, la parte más adelantada está a 38.98 mm de la pared delantera. Por esto, la parte frontal del módulo es menos rígida que las de los carenados genéricos, sufriendo una mayor vibración, aunque no es excesiva. Cabe destacar que estos rigidizadores no se han modificado en todo el proceso y son iguales desde el primer diseño.

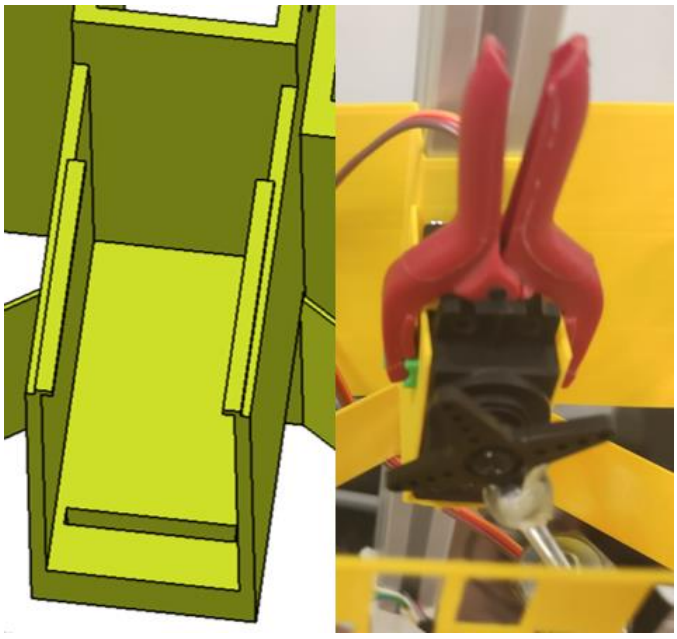


Figura 3.2. Ejemplos de algunas de las posibilidades que se han probado para el soporte del servomotor excitador.

Otro elemento común en todas las versiones es el soporte del servomotor, Figura 3.2. Este elemento va colocado en la parte central del carenado, su pared posterior es coincidente con la pared del carenado que está en contacto con el perfil de aluminio. Se trata de un cubo rectangular con las tapas superior y frontal vacías para poder insertar el servomotor. El soporte cuenta en la parte inferior con un rigidizador con un grosor 0.6 mm ayudarle a soportar los esfuerzos producidos por el excitador.

Además, cuenta con un agujero en la tapa inferior en el que se encaja una de las alas que posee el servomotor para ayudar a sujetarlo. En las últimas versiones, se le han añadido unas pletinas en la parte superior con este mismo fin, y el de disminuir las vibraciones cuando el servo está en funcionamiento.

Por último, el soporte para placa de Arduino se trata de un cubo rectangular con unas dimensiones finales de 13x19x46 mm, al que se le ha realizado un vaciado para poder introducir en él la placa de Arduino NANO. Una vez realizado el vaciado, queda una pequeña pletina de 1x12.3x26 mm, para sujetar la plaquita. Está situado en la parte posterior de la pared en la que se encuentra uno de los clipajes.

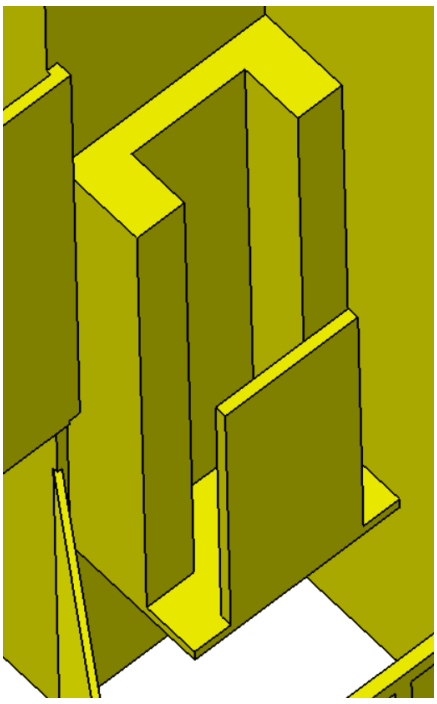


Figura 3.3. Soporte para Arduino NANO de la versión final del carenado pota servo.

Se han impreso 4 versiones, las diferencias entre ellas se detallarán exhaustivamente más adelante. Cabe destacar que se descartaron otras opciones anteriores antes de realizar la primera impresión, en las que las ubicaciones de los elementos eran distintas a las definitivas. Se van a mostrar brevemente algunas de estas versiones para dar una idea del proceso seguido hasta conseguir la versión definitiva.

- Versiones no impresas

En primer lugar, se va a mostrar la primera idea que se desarrolló en CATIA V5, Figura 3.4. En un principio la placa de Arduino NANO no iba a ir inserta por sí misma en el carenado, sino que se engancharía en una protoplaca de dimensiones 82.5x53.5 mm, y sería esta la que se insertaría en el carenado.

Es por este motivo por el que en las primeras pruebas tienen un soporte para el Arduino tan grande. Este soporte estaba ubicado en la pared posterior del carenado, en vez de la ubicación final.

En cuanto al soporte para el servo, tenía unas paredes muy gruesas de 2.75 mm de grosor y un nervio en la parte inferior de 10 mm de anchura.

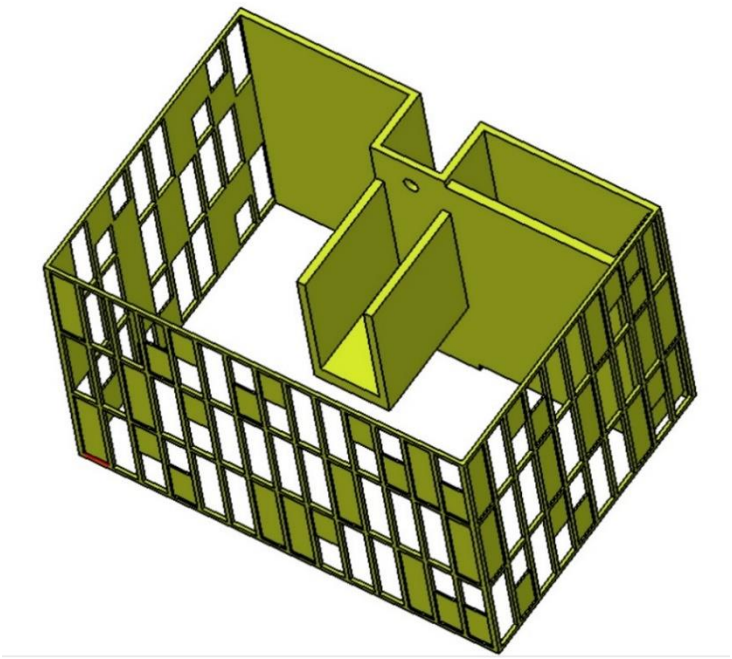


Figura 3.4. Primera prueba del carenado porta servo.

Otro hecho particular de este carenado son los dos agujeros que tiene, uno por encima y otro por debajo del soporte del servomotor. Su finalidad es la de ayudar a la sujeción del carenado mediante tornillos.

En la siguiente prueba, Figura 3.5. se varío la posición del soporte de Arduino NANO enganchado en una protoplaca, y se colocó en la que ha sido la posición definitiva. También se retiró una de las paredes del soporte para poder manipular la placa sin tenerla que sacar del soporte. Esta fue la primera vez que se realizó el agujero en el soporte del servo para introducir una de las alas del servomotor.

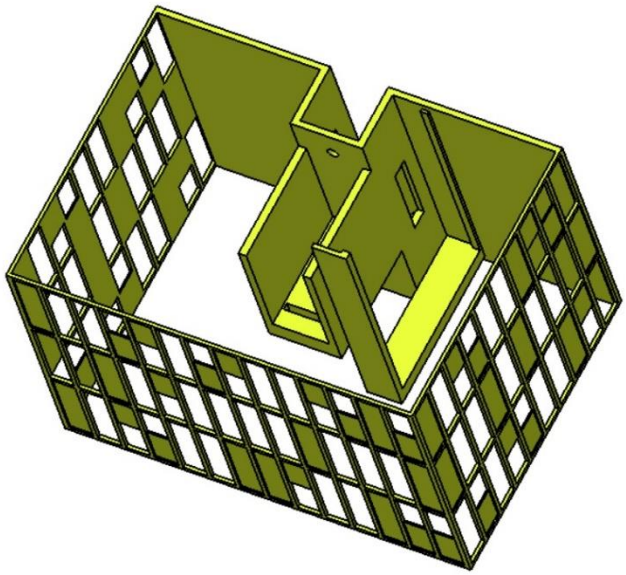


Figura 3.5. Segunda prueba del carenado porta servo.

La última de estas pruebas no impresas, Figura 3.6. cabe destacar que es un carenado hermano de la primera versión impresa. Son exactamente iguales, con la salvedad de que esta prueba no impresa cuenta con un soporte para albergar un acelerómetro.

Este soporte se ideó porque al comienzo de la realización del proyecto, existía la posibilidad de realizar la medición de las aceleraciones de la estructura mediante un acelerómetro independiente, en vez de utilizando la IMU del Arduino NANO 33 IoT. La duda existía por la posibilidad de que con este

acelerómetro el programa de Arduino a realizar fuese más sencillo. Esta posibilidad se descartó, por al final resultar más compleja.

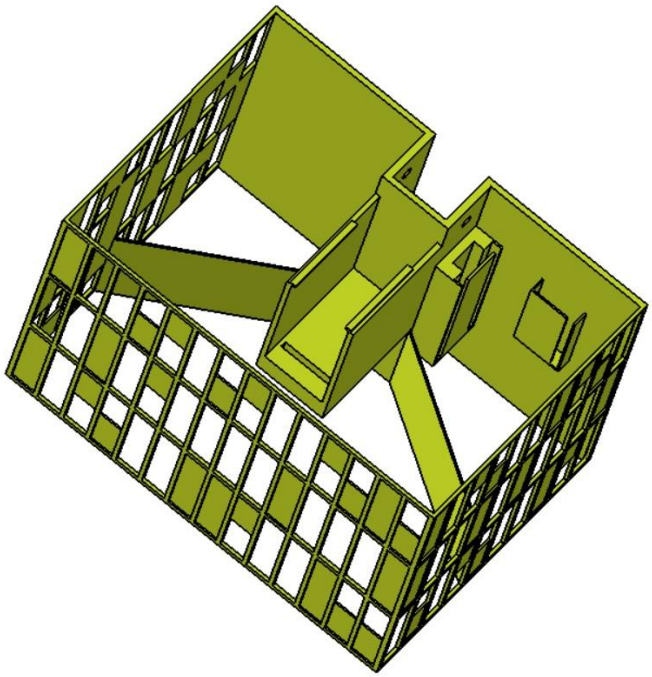


Figura 3.6. Prueba del carenado porta servo con soporte para un acelerómetro.

- <u>1ª versión</u>

En esta primera versión hubo un problema durante la impresión y sólo pudo imprimirse algo más de un piso, no obstante, fue suficiente para percibir varios problemas.

Antes de enumerar esta serie de problemas, se van a resaltar algunas de las principales características de este carenado. En esta versión, aunque no llegaron a imprimirse, se han añadido dos pletinas en la parte superior del soporte del servomotor para ayudar a la sujeción de este. Estas pletinas están situadas en la parte frontal del soporte y se extienden 29 mm hacia la parte posterior de este. Tienen una anchura de 2 mm, a 1 mm de la cara lateral exterior del soporte, y tienen un grosor de 1 mm.

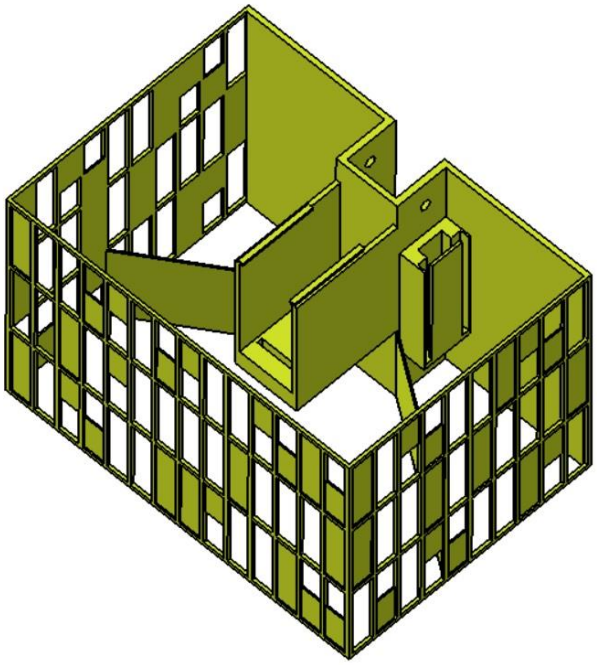


Figura 3.7. Imagen de la primera versión completa del carenado porta servo.

Otro elemento que ha cambiado respecto a las versiones no impresas han sido los agujeros destinados a la sujeción del carenado mediante tornillos. Antes eran dos, y se encontraban en la cara en la que está sujeta el soporte, por encima y por debajo de él. Y en esta prueba se han movido a las paredes laterales del hueco trasero, donde se encuentran los enganches. Tienen un diámetro de 4 mm. Se han realizado 4, dos en cada lateral, a 10mm del extremo superior e inferior, respectivamente.

Los principales problemas se encontraban en las dimensiones del soporte del servo y del de Arduino. En cuanto al soporte del servo resultaba muy estrecho, tenía unas dimensiones de 41.5x21.5x41.5 mm. Podía encajarse el servo porque sólo se había impreso la parte inferior, pero si se hubiese acabado la impresión hubiese sido muy complicado conseguir introducirlo. Además, la ranura en la que se debe introducir una de las alas del servo, de 19x2 mm, era demasiado pequeña, en ambas dimensiones del agujero.

Para el soporte del Arduino se parte de un cubo de 19x10x46 mm al que se le realiza un vaciado para para ubicar la placa de Arduino. El problema era similar al del soporte del servo, se había hecho un hueco demasiado pequeño para la plaquita. Podía introducirse el Arduino, pero porque sólo se imprimió la parte inferior del soporte. Las medidas que más complicaban la inserción de la plaquita son las que están destacadas en la imagen siguiente:

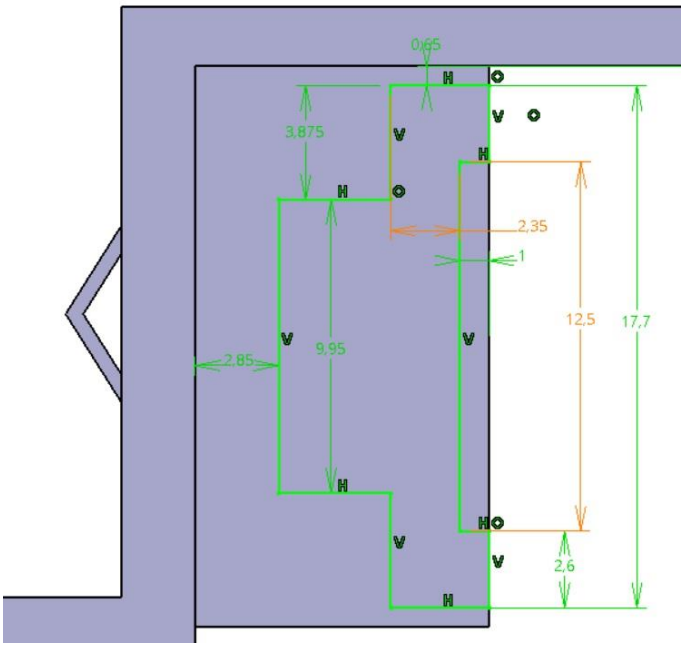


Figura 3.8. Medidas que dificultan la inserción de la plaquita de Arduino en su soporte.

La longitud de la pletina de 12.5 mm es demasiado grande, por lo que choca con los pies de los pines de la placa de Arduino, se reducirá en las siguientes versiones. Y el espacio entre la parte posterior de la pletina y la pared siguiente, de 2.35 mm, también hace que choquen la cabeza de los pines con la pared del soporte.

Otra de las posibles mejoras es la de disminuir el grosor de las paredes posteriores del carenado, que son de 2 mm en vez de 0.6 mm, como en el carenado genérico. El motivo de haber aumentado el grosor de estas paredes respecto a las de los carenados genéricos es para hacer más resistente al módulo, por tener que soportar el peso y fuerzas generadas por los elementos que soporta. No obstante, al realizar esta impresión, parecía un gasto de material inútil, por lo que se optaría por reducirlo en versiones posteriores.

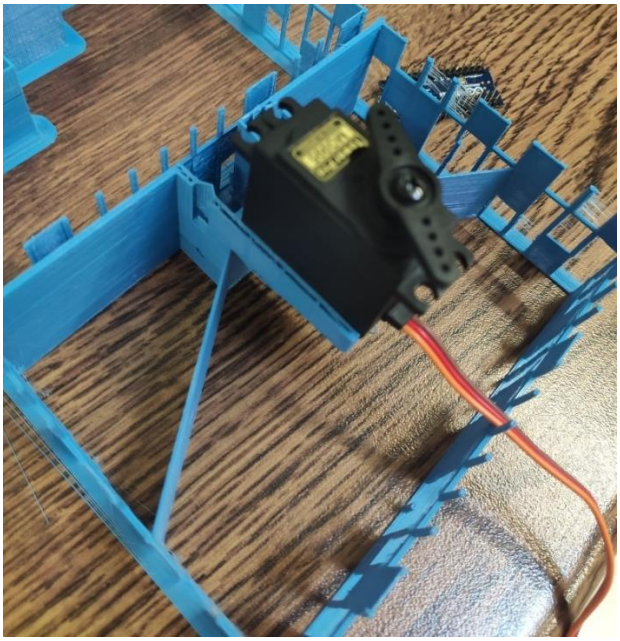


Figura 3.9. Primera versión impresa del carenado porta servo, con el servomotor inserto.

También las paredes del soporte del servo estaban sobredimensionadas para las cargas que tendrían que resistir. Las paredes laterales tendrían un grosor de 1.75 mm, y la base del soporte de 7 mm en esta versión.

Y, por último, cambiar el nervio del soporte del servo, que en esta prueba llevaba hasta la parte frontal de dicho soporte, para que no llegue a sobrepasar la ranura para el ala del servomotor.

- **2ª versión**

De esta versión se han impreso 2 unidades. En ella se pretendían solucionar los problemas encontrados en la versión anterior.

Se reduce el espesor de las paredes posteriores, y se dejan del mismo grosor que las del carenado genérico, 0.6 mm. También se han adelgazado las paredes del soporte del servo y se han dejado con un espesor de 1 mm.

En cuanto a las posibles mejoras comentadas anteriormente, también se ha variado el nervio del soporte del servo, en vez de llegar hasta el extremo del soporte, ahora sólo lo hace hasta el comienzo de la ranura para las alas del servomotor, como puede verse en la Figura 3.10.

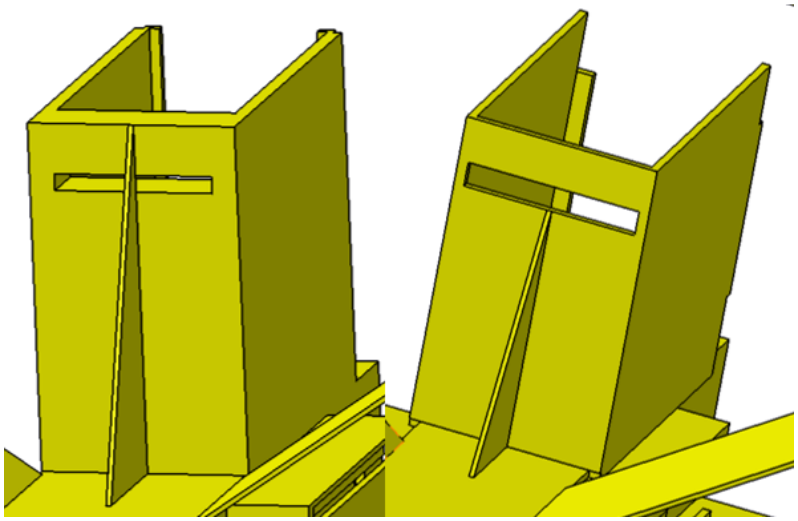


Figura 3.10. Comparación del nervio del soporte del servomotor entre la 1ª versión, foto izquierda y la 2ª versión, foto derecha.

En cuanto al soporte del servo se han ampliado su anchura y su longitud, ahora el hueco para introducir el servomotor es de 43x22x41.5 mm, y también se ha aumentado 1 mm la longitud de la ranura para introducir el ala del servo, que pasa a ser de 20x2 mm.

Este aumento de la anchura del soporte del servo, junto con la variación de los grosores de las paredes del soporte y de las paredes posteriores, hace necesaria la extensión de la pared posterior 0.4 mm a cada lado.

En cuanto al soporte del Arduino, se han variado las medidas comentadas en el apartado anterior que dificultaban la inserción de la placa. Se ha reducido la longitud de la pletina a 12.4 mm, y la distancia entre la pletina y la pared posterior más cercana se ha aumentado de 2.35 mm a 3 mm. Al hacer mayor de esta distancia, y tener que mantener un cierto espacio con la pared de los enganches, es necesario aumento del tamaño del soporte, se amplía 3 mm el ancho del cubo, y pasa a tener unas dimensiones de 19x13x46 mm.

Además de resolver los problemas detectados en la versión anterior, también se ha realizado un rediseño del soporte de la placa de Arduino, eliminando los rebordes laterales con los que contaba la primera prueba. Estos tenían un grosor de 0.65 mm, y no tenían sentido, ya que no iban a ayudar a la sujeción de la placa y sólo dificultaban su inserción.

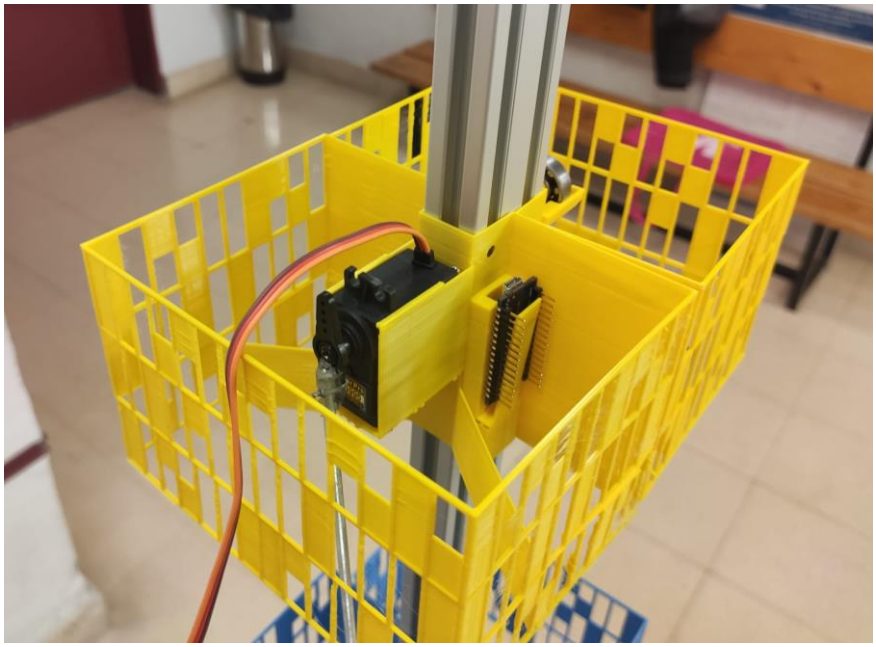


Figura 3.11. segunda versión impresa del carenado porta servo colocado sobre la estructura, con el servomotor y el Arduino montados.

Por último, vamos a comentar los fallos encontrados en este modelo. En cuanto al soporte del servo, es demasiado largo y hay demasiado espacio en la parte posterior, esto supone que al accionar el servomotor se produzca una mayor vibración. La ranura de este soporte seguía siendo demasiado pequeña, porque para poder colocar el servomotor hubo que aumentar su tamaño mediante acciones mecánicas. Y las pletinas superiores chocaban con el ala superior del servomotor, por lo que no sujetaban al servo convenientemente.

En cuanto al soporte del Arduino, la inserción seguía siendo complicada porque el hueco sigue siendo demasiado pequeño, esto puede verse en la Figura 3.11. donde se muestra cómo ha de combarse la pletina para poder introducir la placa de Arduino. Por este motivo se puede deducir que la parte superior de esta pletina no va a realizar una labor demasiado relevante en la sujeción del Arduino y sólo dificulta la inserción, por lo que se reducirá su altura en la siguiente prueba. Las otras dimensiones que dificultan este proceso son las mismas que en la versión anterior.

- 3ª versión

En primer lugar, se van a comentar los cambios realizados en el soporte del servomotor. Resolviendo la problemática de la versión anterior se ha reducido la longitud del soporte a 38.5 mm. Las pletinas se han movido a la parte central del soporte para evitar que contacten con el ala superior del servomotor, se encuentran a 7.5 mm de la parte frontal del soporte. También se ha reducido su tamaño de 29 mm a 22 mm, y se ha aumentado su anchura de 2 mm a 3 mm y su grosor de 1 mm a 1.2 mm.

El último cambio realizado en este soporte ha sido aumentar las dimensiones de la ranura de la tapa inferior, que ahora será de 21.5x3 mm, para que el ala del servo quepa por ella sin problemas.

En el soporte para el Arduino, se realizan tres modificaciones. Se reduce el ancho de la pletina, hasta dejarlo en 12.3 mm, para poder introducir el Arduino NANO con mayor facilidad. También, como se comentó en la versión anterior, se ha reducido la longitud de esta pletina en 20 mm, hasta dejarla en una medida de 26 mm. Con el mismo fin se aumenta la distancia entre la parte interior de la pletina y la pared posterior hasta dejarlo en 4 mm. Estas medidas son las que están remarcadas en naranja en la Figura 3.12.

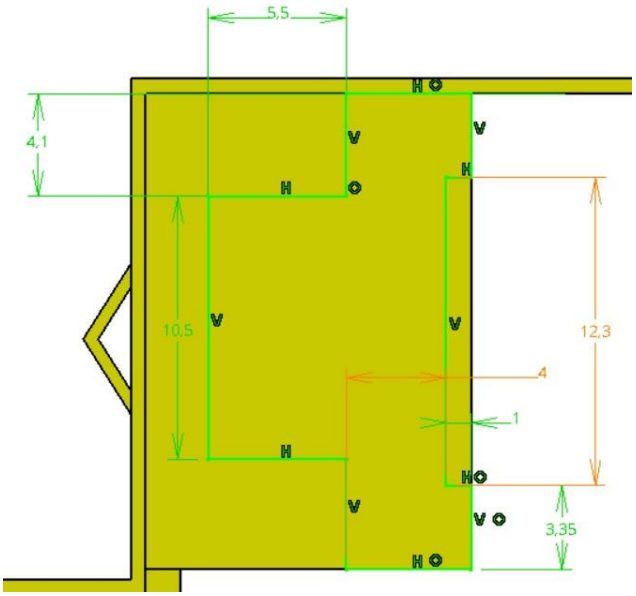


Figura 3.12. Medidas cambiadas en el soporte de la placa de Arduino.

Tras haber realizado distintos ensayos, experimentado con versiones anteriores de este carenado cargadas con todos los elementos que alberga y sin haberlas sujetado mediante tornillos, se observó que el carenado no se deslizaba por la barra, por lo que los agujeros para introducir los tornillos son prescindibles.

Este hecho muestra el buen trabajo realizado en el diseño de los enganches de los carenados, ya que tienen la flexibilidad suficiente para permitir encajar y desencajar los módulos de la barra. Así como la rigidez necesaria para mantener los carenados en la posición deseada aun estando cargados con elementos externos, y ser sometidos a cargas dinámicas.

Con estas variaciones se obtiene un carenado que cumple con los requerimientos que se le exigen. En cuanto al soporte del Arduino, cumple con sus funciones debidamente, ya que se puede introducir en él la placa de Arduino y es capaz de sujetarla, por lo que no se plantean posibles mejoras de este elemento.

El soporte del servomotor cumple con su principal función de manera suficiente. Sin embargo, al accionar el servo para excitar la estructura, el propio servomotor tiene un cierto movimiento lateral dentro del soporte. Esto es debido a que, para poder insertar fácilmente el servo, el soporte es más ancho que el propio servo. Esto es un hecho que se pretendía mejorar en versiones posteriores, pero no se pudo volver a imprimir por la alerta sanitaria COVID-19.

Es un pequeño detalle que mejorar, pero esta versión del carenado es suficiente para cumplir con las funciones que tiene encomendadas. Para sujetar al servo motor y así evitar esta vibración durante la experimentación, se colocó una pinza de manera temporal, y sólo hasta que se pudiese imprimir la siguiente versión.

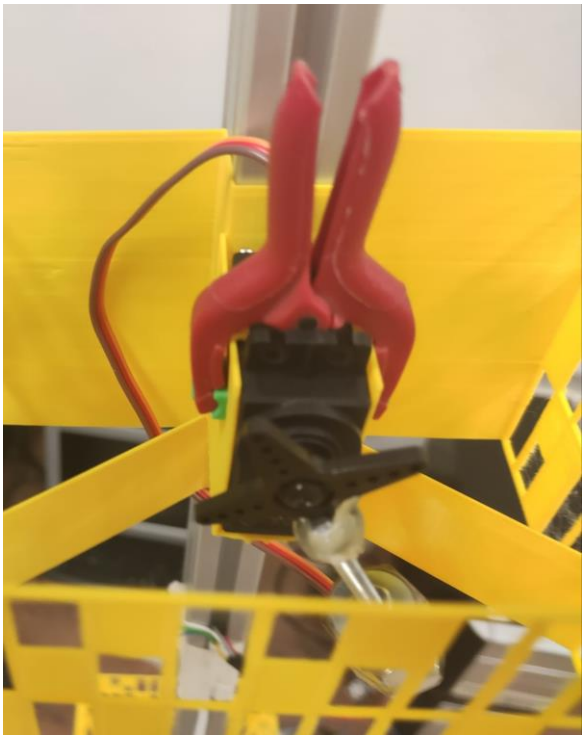


Figura 3.13. Servomotor sujeto con una pinza para evitar su oscilación lateral. Inserto en la 3ª versión del carenado porta servo.

3.2. Carenado porta TMD

Este carenado es el más complejo al albergar un gran número de elementos. Tiene el fin de albergar al TMD, al servo bloqueador y la placa de aluminio. Dispone de un soporte para el Arduino NANO pero no va a tenerlo que sujetar, porque irá inserto en el módulo porta servo. Si cuenta con este elemento es porque en las primeras ideas, la parte electrónica del trabajo se iba a realizar mediante dos placas de Arduino, por lo que se introdujeron soportes para Arduino en los dos carenados especiales. Finalmente, sólo se utilizará un Arduino NANO que irá en el carenado porta servo.

Por el hecho de que el soporte para el Arduino es igual en los dos carenados, y que el proceso seguido en su diseño ha sido igual en los dos módulos, se han ido desarrollando al mismo tiempo. No se va a volver a explicar en este apartado.

Al igual que en el proceso seguido en el carenado porta servo, se ha partido del genérico y se ha ido desarrollando, introduciendo los diferentes componentes que lo forman.

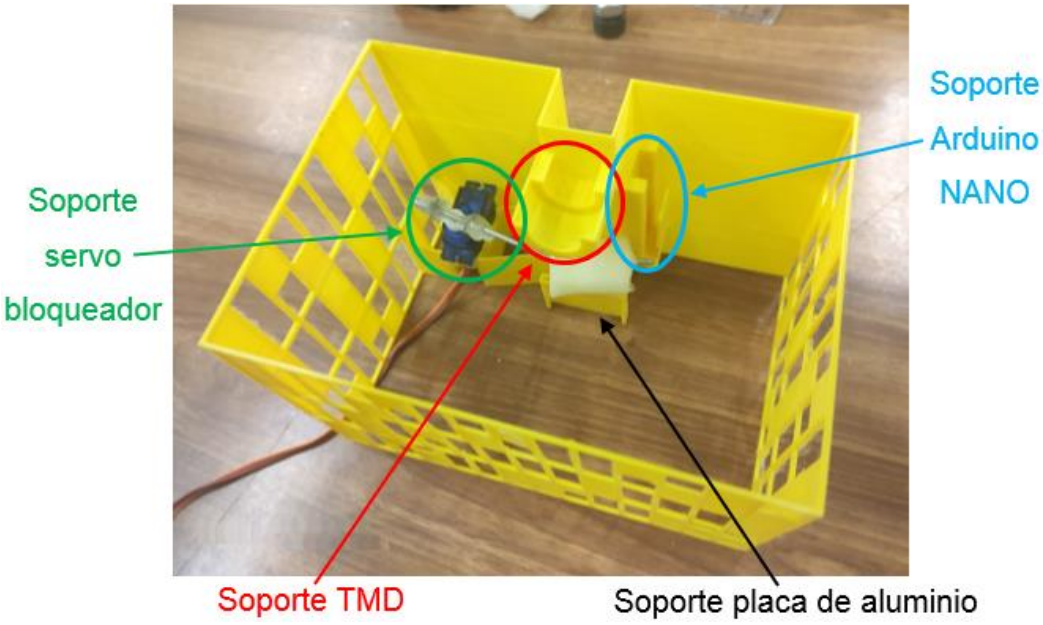


Figura 3.14. Última versión impresa del carenado porta TMD. Se indican algunas de las partes más relevantes.

Este carenado oculta y sujeta los elementos comentados anteriormente. Y tiene además un diseño funcional que permite a cada uno de ellos tener la movilidad requerida para cumplir con su cometido. Este diseño ha requerido un largo proceso en el que se ha ido mejorando el módulo en base a los problemas que se han ido encontrando en las sucesivas impresiones.

En primer lugar, se van a explicar las características principales de este carenado, y la finalidad que tienen los elementos que lo componen. Estos son independientes de las dimensiones y no van a variar en las diferentes versiones.

Después se comentarán algunos de los aspectos más relevantes de aquellas pruebas de este carenado que no han llegado a imprimirse. Y posteriormente se explicarán los cambios realizados entre las sucesivas versiones y sus características más importantes, hasta llegar a la definitiva.

El elemento principal de este carenado es el soporte para el TMD, que está situado en la parte central, sujeto a la pared. Se trata de un semicilindro al que se realizan dos vaciados, uno más profundo para insertar los cojinetes del TMD y otro para que entren la varilla roscada y las tuercas que forman este elemento.

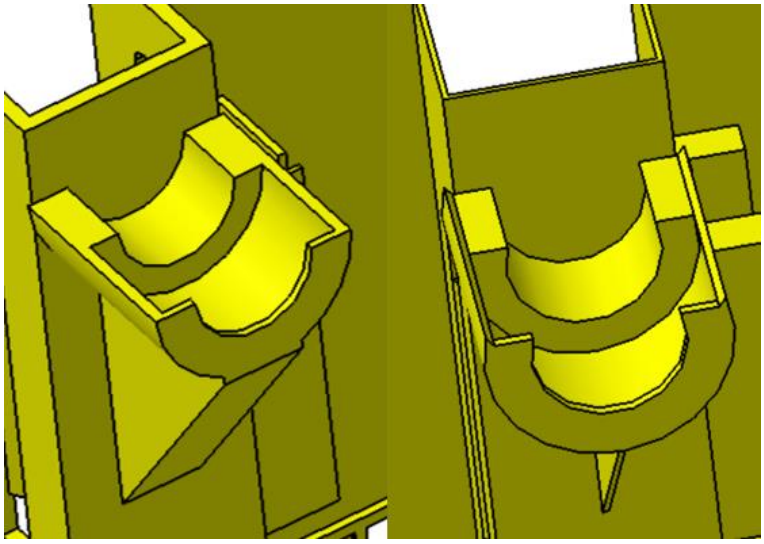


Figura 3.15. Comparación entre los soportes del TMD de la primera versión impresa, foto izquierda, y la versión final, foto derecha.

Al lado izquierdo del soporte del TMD se encuentra el soporte del servo bloqueador. Este es un elemento que ha variado mucho en forma, tamaño y posición, ya que en principio se iba a utilizar un servomotor del mismo modelo que el excitador, MG995. Finalmente se ha utilizado un servo SG90 porque reúne unas mejores condiciones para lo que se requiere en este carenado, como se explicará más adelante.

El soporte del servomotor ha de estar ubicado en una de las paredes a los lados del cubo central, porque las paredes con ventanas no son suficientemente resistentes para aguantarlo. Dado que a la derecha del soporte del TMD se encuentra el del Arduino, el del servomotor deberá ir al lado izquierdo del TMD.

Dicho elemento está sujeto a la estructura del carenado mediante 4 rigidizadores, dos de ellos lo unen con la pared posterior, otro con el cubo central del carenado y el último con la pared exterior. Este es coincidente con una de las ventanas tapadas para disimular su presencia desde el exterior.

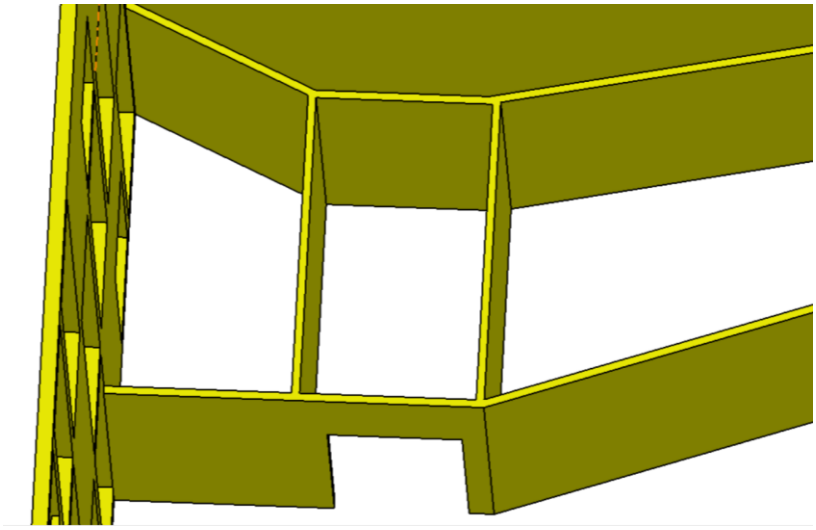


Figura 3.16. Soporte del servomotor SG90, donde pueden verse sus nervios de anclaje.

El soporte para albergar este elemento es un cubo de 23.3x13.1x22.5 mm, con unas paredes de 0.6 mm, y cuya parte frontal está situada a 30.9 mm de la pared posterior del carenado.

El último elemento singular de este carenado es el que sostiene a la placa de aluminio que forma parte del amortiguador de la estructura. Este soporte se encuentra en la parte central del carenado, debajo del el que sostiene al TMD y está sujeto a la misma pared que el anterior mediante 3 nervios.

Se trata de un cubo rectangular de 31.2x3.2 mm, al que se le realiza un vaciado central para crear un hueco de 30x2 mm y se retira parte de la pared frontal, dejando sólo en la parte superior una porción de 2.5 mm de achura.

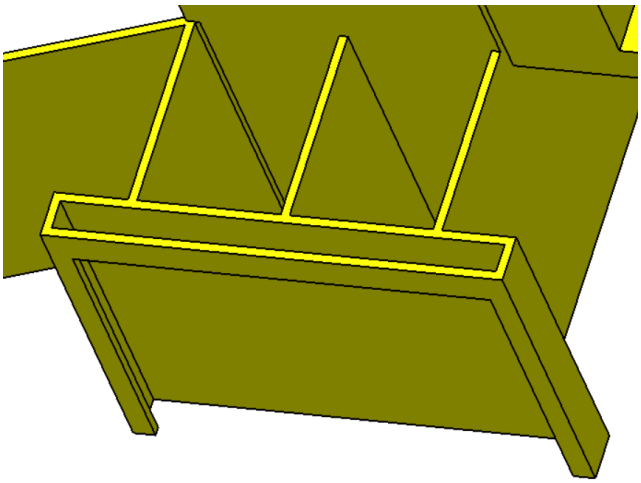


Figura 3.17. Soporte para la placa de aluminio con 3 anclajes al soporte.

Una característica muy importante es la distancia entre este soporte y la pared posterior, ya que va a condicionar la separación entre la placa de aluminio y el imán. Esta distancia marca la eficiencia del amortiguador, que está formado por estos dos componentes.

Un hecho característico de este carenado es que no dispone de rigidizadores como los otros. Esto es por la necesidad de introducir tantos elementos en él. Ya que para que el TMD pueda oscilar ha de tener un hueco en la parte delanter, lo que impide hacer unos rigidizadores como los del carenado genérico. Además, la situación de la placa de aluminio impide tener unos rigidizadores como los del porta servo.

Por lo que finalmente se ha elegido la opción de no incluir rigidizadores, lo que implica que la pared frontal del carenado sea menos estable y tenga mayores vibraciones cuando se hace oscilar la maqueta. Es de reseñar que las vibraciones producidas en este módulo no son demasiado grandes por lo que tampoco suponen un gran inconveniente.

- Versiones no impresas

Se van a exponer aquellas pruebas que se realizaron al comienzo del diseño de este carenado y que no llegaron a imprimirse.

Como puede verse en la siguiente foto, Figura 3.18. la primera prueba que se realizó tan siquiera disponía de un soporte para albergar el servomotor, que es la mayor diferencia que puede apreciarse a primera vista. El diseño del soporte del Arduino es muy similar al de la primera versión no impresa del carenado porta servo.

Por último, en cuanto al soporte del TMD, se trata de un semicilindro de radio 12.5 mm, y 16 mm de longitud que tiene un espesor de pared de 1.6 mm, que está sujeto por un nervio de 10 mm de grosor.

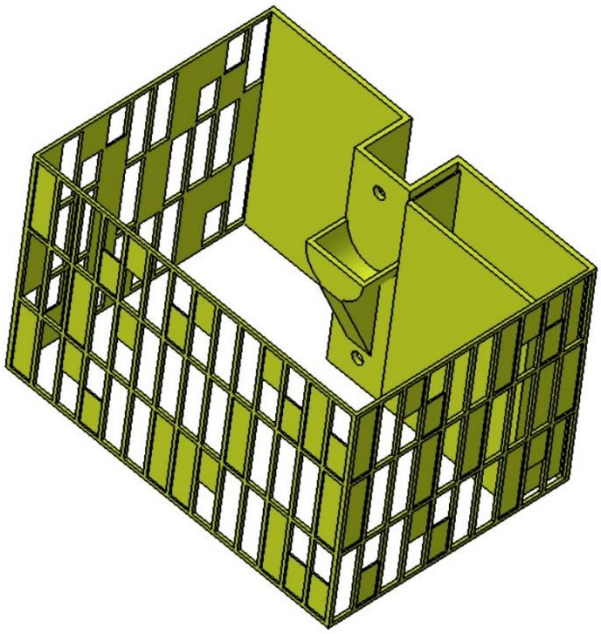


Figura 3.18. Primera prueba realizada en Catia del carenado porta TMD.

Este carenado dispone de dos agujeros de radio 2.5 mm, uno por encima y otro por debajo del soporte del TMD, para ayudar a sujetarle mediante tornillos. Los agujeros están ubicados a 10 mm del extremo del carenado, superior e inferior respectivamente.

En la segunda prueba que se realizó, Figura 3.19. ya se incluía un soporte para el servomotor MG995. Este elemento está ubicado a la izquierda del soporte del TMD y aprovecha la pared posterior. Tiene un grosor de pared de 2.75 mm, y cuenta con una ranura para insertar una de las alas del servo.

Un problema importante de esta ubicación y disposición del servo es que no permitía bloquear el TMD en cualquier punto de su recorrido de oscilación, por lo que se variará en posteriores versiones.

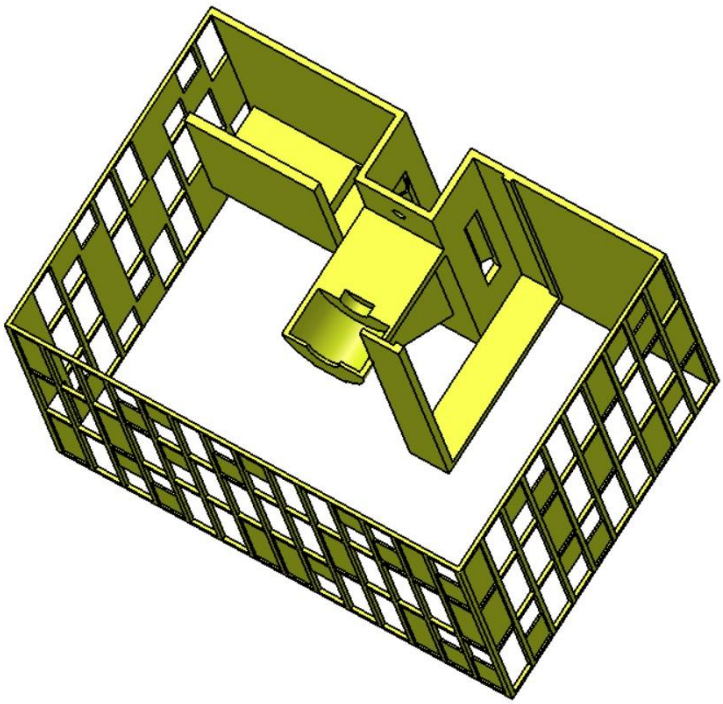


Figura 3.19. Segunda prueba del carenado porta TMD.

El otro elemento destacable de este módulo es el soporte del TMD, que ya dispone de hueco para las tuercas con las que cuenta el TMD. Aunque en su parte trasera sigue siendo pequeño para que entre la barra roscada de este elemento. El hecho más característico de este soporte es su longitud, de 33.5 mm, necesaria para evitar el choque con el soporte para la protoplaca en la que iría enganchado el Arduino.

De haber impreso la 2ª prueba, la longitud de este elemento podría haber resultado problemática porque al estar el TMD tan separado de la barra central de la estructura, generaría mayores fuerzas de torsión en el carenado. Está sustentado por un nervio de 10 mm de grosor.

Al igual que en el caso del porta servo, la tercera prueba no impresa, Figura 3.20. es "hermana" de la primera impresión, con la única diferencia de que la versión no impresa cuenta con un soporte para albergar un posible acelerómetro. Que, como ya se ha comentado anteriormente, era una posibilidad que se barajaba para medir la aceleración a la que se somete a la estructura.

Las características de esta versión se comentarán en la descripción de la primera versión impresa.

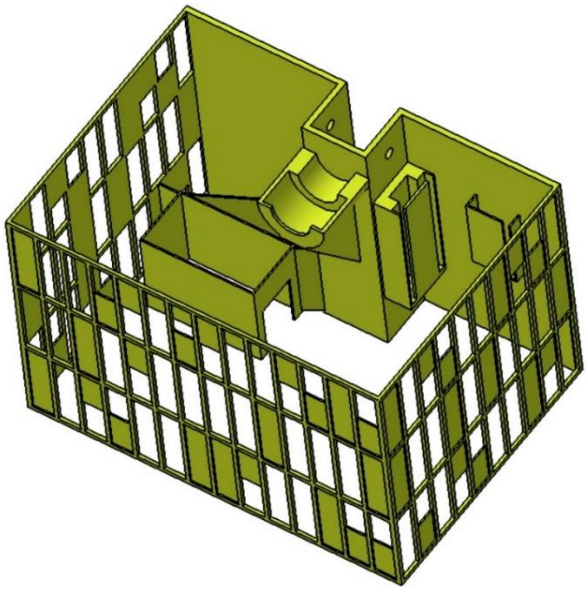


Figura 3.20. Tercera prueba no impresa del carenado porta TMD.

- **1ª versión**

En la primera versión que se imprimió de este careando hubo un problema durante la impresión, se realizó casi la mitad de la altura total, aproximadamente 40 mm. Como llegaron a imprimirse algunos de los componentes, aunque sea parcialmente, se pudieron valorar algunos de los elementos de este módulo.

En primer lugar, se van a comentar algunas de las características del soporte del servomotor. Este sostén se ancla a las paredes del carenado mediante 3 nervios, dos de ellos nacen de las esquinas de la parte central. Y el tercero lo hace desde la pared con ventanales más cercana, de hecho, se aprovecha una de las ventanas opacas para disimularlo. Estos tres anclajes se unen a las esquinas traseras del soporte.

Una variación importante que se realiza en esta primera impresión es la de los agujeros con los que se ancla el carenado a la estructura. En vez de tener dos en la pared posterior, aquella a la que está sujeta el soporte del TMD, dispone de 4, que se ubican en las paredes laterales. En donde se encuentran los enganches, dos agujeros en cada lado. Uno por encima y otro por debajo del saliente, a 10 mm del extremo correspondiente del carenado.

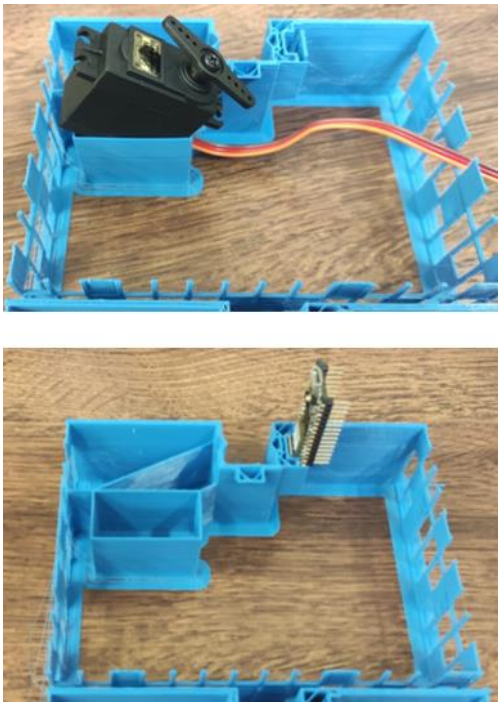


Figura 3.21. Primera impresión del carenado porta TMD. Prueba de la inserción de los elementos que ha de albergar.

El principal problema que presenta, y que puede observarse en la Figura 3.21. es que el soporte para el servomotor bloqueador es demasiado corto. En cuanto a la anchura, las dimensiones son adecuadas porque la inserción se puede realizar sin problemas.

El propósito de este soporte es introducir el servomotor por la parte superior y apoyarlo en las paredes del carenado con las alas del servo. Por ello el soporte de 32 mm tiene menos altura que el servo, 42.9 mm.

El hueco en que debe insertarse el servomotor tiene unas dimensiones de 41.5x21.5 mm que son mayores que el servo, 40.7x19.7 mm pero no son suficientes para poder introducir el servo en él. El principal inconveniente es que no se había tenido en cuenta la conexión de los cables que se encuentra en uno de los laterales del servo, y que tiene un pequeño reborde de plástico, Figura 3.22. En versiones posteriores se irá modificando este soporte.

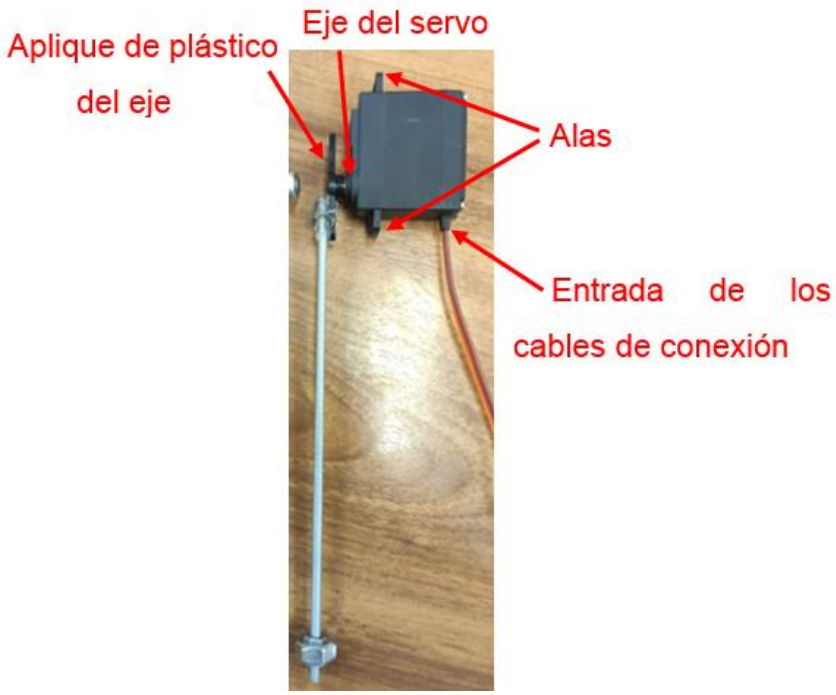


Figura 3.22. Servomotor MG995, en que pueden verse las alas y la conexión de los cables.

De esta impresión también puede deducirse que el nervio del soporte del TMD es demasiado grueso, y ha de reducirse en pruebas posteriores.

- **2ª versión**

En esta versión se pretende resolver los problemas encontrados en la anterior. De esta versión se han impreso 3 unidades del carenado, un primer intento en el que hubo un fallo en la impresión y sólo se imprimieron unos 20 mm, pero que fue suficiente para comprobar que algunos de los cambios realizados funcionaban. Otra para la maqueta objeto de este TFG y otra para el grupo con el que se ha colaborado en la realización del trabajo.

El mayor de los errores de la 1ª versión se encuentra en el soporte del servomotor, por lo que en esta prueba se realizan una serie de modificaciones sobre él, tanto en relación con las dimensiones como con su diseño.

En primer lugar, se ha de aumentar la longitud del soporte, que es la dimensión más problemática porque se ha de tener en cuenta la entrada de los cables al servomotor. Pero el aumento desmesurado de esta dimensión puede ser inconveniente, al dificultar la labor del servomotor. Ya que cuando se requiera que el TMD esté bloqueado es el servomotor el que tiene que pararlo, y si este se mueve dentro de su habitáculo, el bloqueo no será óptimo.

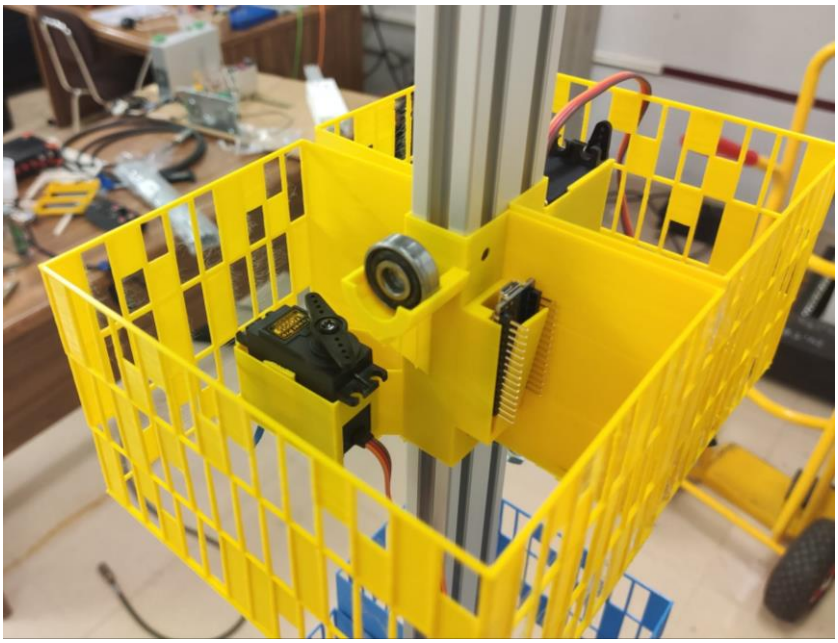


Figura 3.23. 2ª versión del carenado porta TMD en que se puede apreciar el agujero realizado para el cableado del servomotor.

Por todo ello, se ha buscado una solución que cumpla con ambas necesidades, que el servomotor quepa en el soporte, pero sin tener demasiada holgura. Se ha aumentado la longitud del soporte 3 mm hasta tener 44.5 mm, y se ha retirado parte de una de las paredes del soporte, aquella que coincide con el lateral del servo donde se encuentran los cables, como puede verse en la Figura 3.23.

De esta manera, introduciendo primero la esquina del servomotor más cercana a los cables de conexión y colocando este reborde en el agujero del que dispone el soporte, se consigue una inserción del servomotor adecuada a las necesidades requeridas.

Como puede verse en la Figura 3.23. se ha calzado el servomotor para evitar un pequeño movimiento que se generaba cuando la estructura oscilaba. No se va a variar el ancho del soporte en versiones posteriores, ya que es preferible tener que adoptar está solución a reducir la anchura y que la inserción sea más complicada.

El otro cambio que se ha realizado en esta versión es la reducción del grosor del nervio del soporte del TMD, que era de 10 mm y ha pasado a ser de 0.6 mm.

Esta es la primera ocasión en la que se ha impreso por completo este carenado y que por tanto se ha podido probar el soporte del TMD. La intención es la de enganchar el TMD al soporte mediante sus dos cojinetes, introduciéndolos en la parte delantera del soporte, a la que se ha hecho un vaciado de 11.05 mm de diámetro y 14.25 mm de longitud.

También dispone de un vaciado menor, con un diámetro de 7.5 mm y que se extiende 10 mm, desde el final del primer vaciado hasta la pared posterior, donde termina el soporte. Este segundo vaciado es necesario ya que ha de tener hueco para poder introducir la barra roscada sobre la que están los cojinetes, y la tuerca necesaria para la fijación de los primeros. La tapa frontal de este soporte también tiene hecho un agujero del mismo tamaño, que tiene la misma finalidad que el comentado en el párrafo anterior.

Tras la prueba de la inserción del TMD y el servomotor bloqueador podemos concluir que se ha llegado una solución que cumple con los requisitos de manera adecuada, al poder introducir todos los elementos en él.

Es destacable el diseño del soporte del TMD, ya que siendo la primera vez que llega a imprimirse, la inserción de los cojinetes en él es prácticamente perfecta. Tiene las dimensiones suficientes para que se puedan introducir estos cojinetes sin demasiado esfuerzo, realizando una pequeña fuerza hasta que encaja. Y una vez insertos el soporte realiza presión sobre ellos para que no se salgan durante la experimentación.

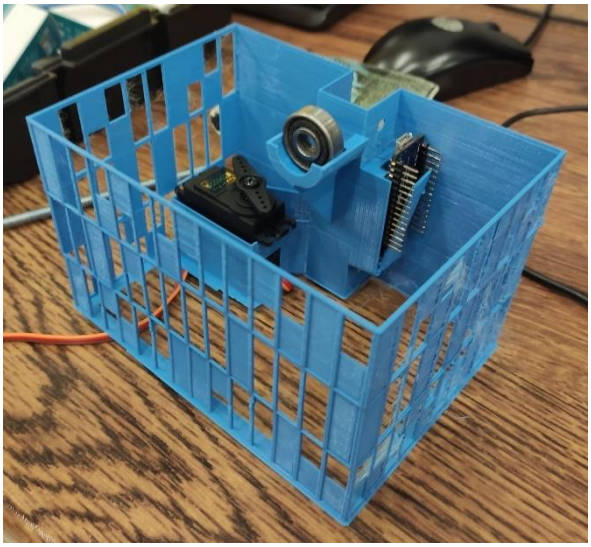


Figura 3.24. 2ª versión impresa del carenado porta TMD.

Si bien es cierto que hay algún aspecto de este elemento a mejorar, ya que con el paso del tiempo y tras la realización de numerosas inserciones del TMD, las paredes del soporte pierden rigidez y los cojinetes tienen cierta holgura dentro del hueco.

Por último, esta versión no cuenta con un soporte para la placa de aluminio que forma parte del sistema de amortiguación de la estructura. Esto es debido a que en el momento en que se realizó este módulo, se planteaban otras opciones para la sujeción de la placa en las que este carenado no intervenía.

La opción más relevante de las que se barajaron era la de hacer otro carenado especial que debería ir necesariamente debajo del porta TMD, y cuyo fin sería el albergar esta placa. Pero con la intención de evitar tener que hacer más carenados especiales que condicionen la posición de los distintos módulos en la estructura, se prefirió optar por intentar introducir la placa dentro del porta TMD, como podrá verse en la siguiente versión.

- **3ª versión**

En esta versión se han realizado dos modificaciones principales, el cambio de posición del soporte del servomotor, y la introducción del soporte de la placa de aluminio. En cuanto a la variación hecha en el soporte del servomotor, sólo se ha modificado su posición.

El motivo del giro de 90º del soporte del servomotor es para evitar el choque del bloqueador, elemento que se le coloca en el servo para bloquear el TMD, con el nervio del soporte de este elemento. Al girar el servo se gana la distancia suficiente para evitar este choque.

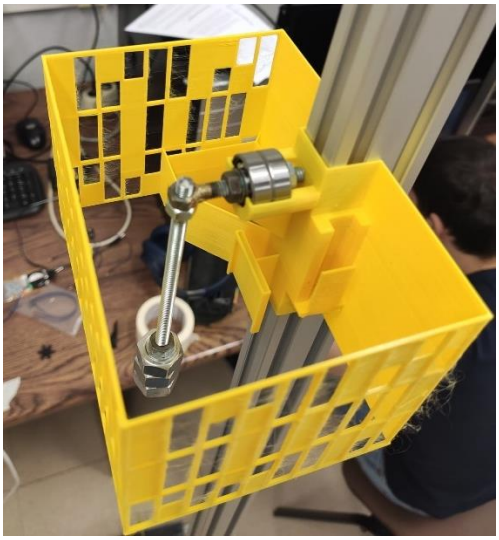


Figura 3.25. Impresión de la tercera versión del carenado porta TMD.

El otro gran cambio es el soporte de la placa de aluminio, que se añade debajo del soporte del servomotor. Se trata de una caja a la que se realiza un vaciado para poder introducir por ella una placa de aluminio de 2 mm de espesor, a la que le han realizado dos ranuras de 0.6 mm.

Este elemento está anclado a la pared posterior del carenado mediante un anclaje de 13.4 mm de longitud, 0.6 mm de grosor y 22.5 mm de altura, la misma que el propio soporte, comenzando en la parte inferior.

El principal problema encontrado en esta versión es que el cambio de posición del soporte del servo impide colocar una placa de aluminio muy ancha, porque chocaría con él. Y se busca tener una placa lo más ancha posible, para que durante la oscilación del TMD, el imán no tenga mayor amplitud que el límite de la placa, para que la amortiguación sea la máxima posible.

Este elemento cuenta con algún aspecto más a mejorar, ya que el soporte de esta placa sólo está enganchado por un nervio, por lo que no es muy rígido, y podría llegar a romperse si se manipulase demasiado.

- 4ª versión

Esta es la última versión que se ha impreso. En primer lugar, vamos a enumerar las variaciones que se han hecho en ella respecto a versiones anteriores. Para después hablar de aquellos aspectos que se puedan mejorar.

Se ha modificado el diseño del soporte del TMD para hacer que, tras la realización de muchas inserciones de los cojinetes, el soporte no pierda fuerza y siga sujetándolos debidamente. Para ello se ha aumentado el ángulo que abarca la semicircunferencia unos 20º de cada lado.

El mayor cambio realizado ha sido en cuanto al soporte del servomotor, ya que se ha optado por cambiar el modelo del servo bloqueador. Este hecho está motivado por los problemas generados por el tamaño del anterior servo. Dado que para la función que tiene el servo bloqueador no era necesario un modelo de servo tan potente como el MG995, y que en el laboratorio había disponibles varios servomotores SG90, se optó por utilizar uno de ellos.

Este nuevo modelo de servo es muy pequeño, sencillo y barato, no tiene un gran par, pero tampoco es necesario para bloquear el servomotor. Estas características lo hacen idóneo para la función que tiene que realizar. Por su reducido tamaño deja hueco suficiente para poder colocar una placa de aluminio lo suficientemente ancha para que el TMD no tenga mayor amplitud de oscilación que anchura la placa.

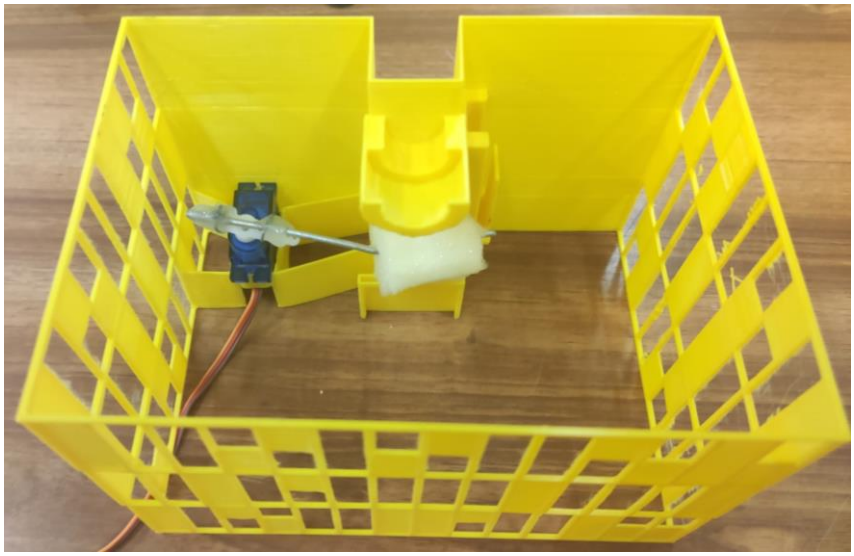


Figura 3.26. 4ª impresión del carenado porta TMD, con el servomotor SG90 inserto.

Este nuevo soporte está más atrás que el soporte de la placa de aluminio, que era el objetivo principal del cambio de modelo de servo. Al igual que en el soporte para el servo MG995 se retira parte de una de las tapas para el cableado del servo.

El último cambio realizado es el añadido de 2 anclajes más al soporte de la placa de aluminio para hacerlo más resistente. Estos dos nuevos rigidizadores coinciden con los extremos del cubo de la pared posterior y tienen las mismas dimensiones que el nervio existente.

Aunque este carenado ya es funcional, y se ha utilizado para realizar la experimentación, hay algunos aspectos que se pueden mejorar. El más importante es el aumento necesario de las dimensiones del soporte del servomotor, ya que en esta versión su inserción es muy costosa y sólo se consiguen introducir algunos milímetros, lo necesario para que se mantenga dentro.

De hecho, como puede verse en la Figura 3.27. fue necesario retirar la pared posterior del soporte para poder introducir el servo completamente para realizar la simulación.

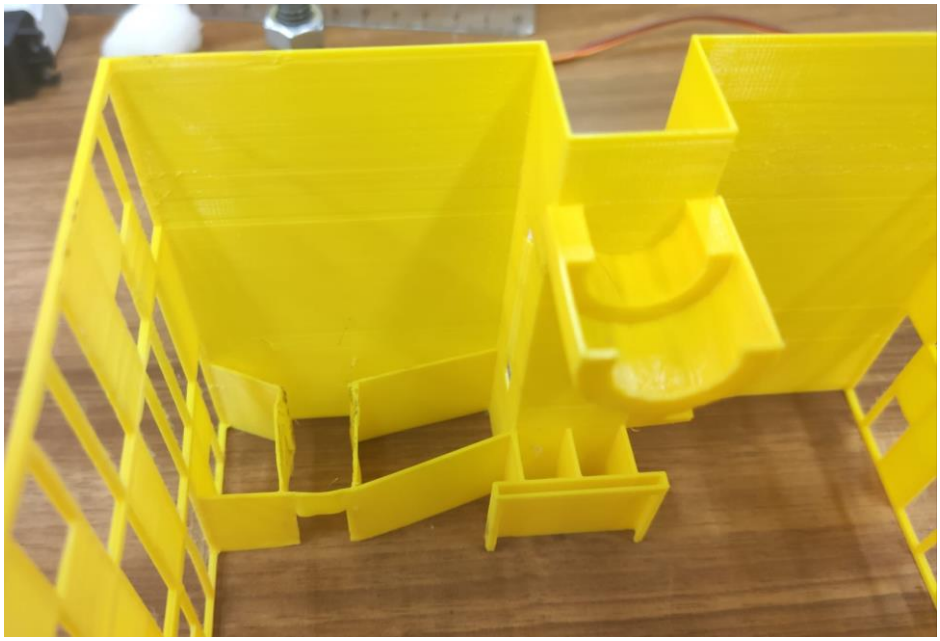


Figura 3.27. 4ª versión del carenado porta TMD, con el soporte para el servo SG90 al que se la ha retirado la pared posterior.

Al haber aumentado las dimensiones del soporte del servo, se ha de evitar que choque con la placa de aluminio, para ello habría que desplazarlo hacia la pared posterior. Este desplazamiento está limitado por el choque del ala del servo con la pared trasera.

Así que es necesario aumentar la separación del soporte de la placa con la pared a la que está sujeta. Aunque también existe una limitación en cuanto esta distancia, porque no debe contactar con el TMD, ni con el imán, ni con las masas.

Esta modificación estaba preparada para realizarla en posteriores versiones, pero no se ha podido volver a imprimir.

3.3. Excitador o shaker

El péndulo excitador es uno de los elementos más importantes de este proyecto, dado que es el responsable de generar con su movimiento, la inercia necesaria para conseguir la oscilación de la maqueta. Se encuentra alojado en uno de los carenados especiales, en concreto en el porta servo.

Figura 3.28. Primer montaje del péndulo excitador de 0,15 m de longitud. Se indican las partes más importantes del servo.

Está formado por una serie de componentes, de los cuales, el más importante es el servomotor MG995, que es el que genera el movimiento de este elemento. Sujeto a él, se encuentra el brazo sobre el que se colocan las masas necesarias en este elemento.

A continuación, se va a explicar el funcionamiento de los servomotores y las características de este modelo en concreto. Para posteriormente hablar del brazo y las masas que forman este componente.

- Servomotor MG995

Un servomotor es un tipo de motor eléctrico que permite controlar en todo momento la posición de su eje de salida. El control se realiza mediante una señal que hace girar al servo un cierto ángulo y luego se mantiene fijo en esa posición. [18]

La señal de control se transfiere desde los pines digitales PWM de la placa de Arduino al pin de control del servo. Las siglas PWM (Pulse-width modulation), significan modulación por ancho de pulsos, es decir, el ancho del pulso es lo que determina el ángulo de giro del servo. Esta relación entre la anchura del pulso y el giro del servomotor varía dependiendo del modelo de servomotor. [19]

En el caso concreto de los servos utilizados en este trabajo se detalla en las hojas de especificaciones que la señal se envía a una frecuencia de 50 Hz, es decir, en pulsos de 20 ms. Por lo tanto, un pulso de 0.5 ms en un ciclo de 20 ms (50 Hz), haría ir al servo a la posición 0º, mientras que uno de 1.5 ms, lo haría ir a la posición de 90º, y uno de 2.5 ms a 180º, como se puede ver en la figura 3.29:

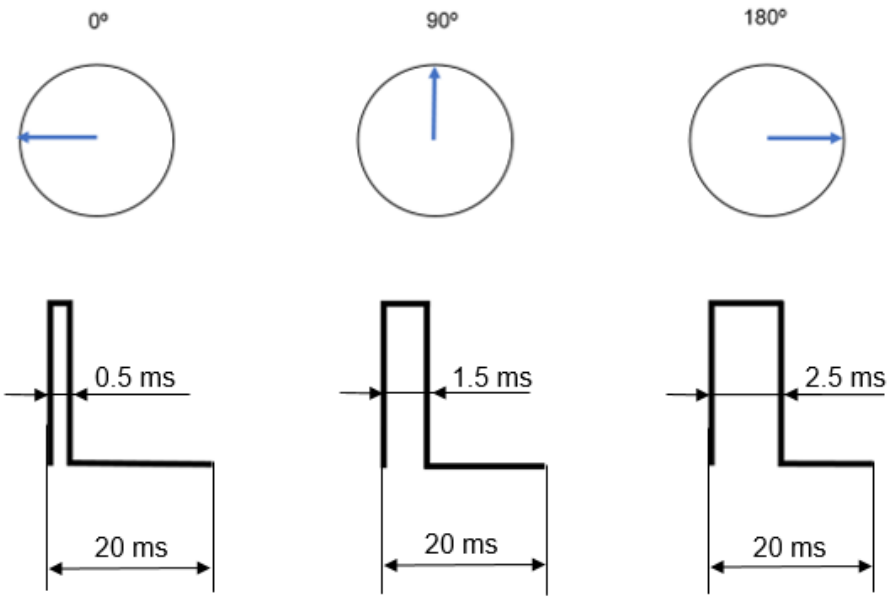


Figura 3.29. Relación entre el ancho de pulso y la posición del servomotor.

La capacidad de control de este tipo de motores los hace muy útiles para aplicaciones de robótica en las que es necesaria gran precisión de movimientos. Además, son capaces de mantenerse en una posición fija, cosa imposible para los motores DC tradicionales.

Los servomotores están formados por un motor de corriente continua (DC), un tren de engranajes, un sensor de desplazamiento y una placa de control. [18]

El motor DC es el encargado de generar el movimiento. El tren de engranajes reductores se encarga de reducir la velocidad de giro del motor DC para aumentar el par en el eje de salida del servo. [18]

El sensor de desplazamiento es generalmente un potenciómetro que permite conocer en todo momento la posición del eje del motor. La placa de control se encarga de comparar el voltaje de referencia, que es la posición deseada en este momento, la que le pide el usuario, con la lectura del potenciómetro, que es el que marca la posición del servo en ese instante. Una vez obtenida esta diferencia se mueve la posición del eje hasta la ubicación deseada. [18]

El servo encargado de generar la excitación es un servo MG995. Se ha escogido este modelo en concreto, porque resulta ser una solución idónea para su aplicación y con un precio adecuado.

Se trata de un servomotor de 40.7x19.7x42.9 mm, con un gran par para su tamaño, de 0.8336 N·m si se le alimenta a 4.8 V y 0.9807 N·m a 6 V, por lo que es capaz de mover una gran masa. Esta es la característica principal que se busca en el servo excitador porque es necesaria una inercia suficiente para hacer balancearse toda la estructura.

Por el contrario, no es un servomotor demasiado rápido, pero no es una característica que resulte demasiado determinante para el uso que tiene en el experimento. A continuación, se exponen en una tabla algunas de las especificaciones más importantes de este servomotor:

Tabla 3.1. Características servo MG995. [13]

Peso	55g
Dimensiones	40.7x19.7x42.9mm
Par	0.8336N·m a 4.8V
	0.9807N·m a 6V
Velocidad de giro	0.2s/60° a 4.8V
	0.16s/60° a 6V
Voltaje de operación	4.8V – 7.2V
Rango de temperaturas	0°C – 55°C

Un factor determinante que se ha de tener en cuenta es el periodo de oscilación del servo, que va a determinar la frecuencia con la que oscila el péndulo. Y para conseguir el fenómeno de resonancia y que la amplitud de oscilación de la maqueta sea la máxima posible, la frecuencia de excitación ha de ser la misma que la frecuencia natural de la maqueta.

Dado que el periodo es el inverso de la frecuencia, y que mediante el programa de Arduino se controla el periodo de oscilación del servomotor. Conociendo la frecuencia de resonancia de la maqueta, se puede adecuar los parámetros del servomotor para conseguir que la maqueta entre en resonancia.

Pero la frecuencia natural de la maqueta no es siempre la misma, varía según los elementos que tenga insertos, ya que está relacionada con el peso, entre otras variables. Por eso, para conseguir los mejores resultados posibles, se ha conocer la frecuencia natural de la estructura en cada una de las condiciones posibles, y adaptar las variables funcionamiento del servo a ella.

- Brazo y masas

Estos dos componentes van montados uno sobre otro. El brazo es una varilla roscada de diámetro 3 mm, sobre la que se colocan 3 tuercas de 6 mm de diámetro interior, cuyo cometido es ser la masa compacta oscilante que hace moverse a la maqueta.

La sujeción de la barra del péndulo y el servo se hace utilizando un aplique de plástico, que se coloca en el eje del servo para facilitar la transmisión del movimiento. Esta unión se realiza utilizando un alambre enrollado alrededor de ambos elementos, y silicona que se aplica con una pistola termofusible, con el fin de hacer la unión más fuerte y segura. Evitando que la unión tenga un movimiento indeseado.

Estas 3 tuercas se encuentran en el extremo final de la varilla, lo más cercanas posible al final de esta, ya que cuanto mayor sea la longitud del péndulo, distancia entre el centro de giro, que en este caso es el punto de sujeción con el servo y la posición de la masa, mayor será la inercia que generará este péndulo. Este mismo efecto se podría conseguir aumentando la masa del péndulo.

Para explicar este concepto de la inercia del péndulo, tomaremos el péndulo como un sistema de una barra que tiene en su extremo una masa puntual. Se puede expresar la inercia de este conjunto como la inercia de la barra, I_B, más la inercia de la masa puntual, I_M. Los valores de las inercias de algunos elementos son conocidas, como es el caso de los elementos que nos incumben:

$$I_{Brazo} = \frac{m_B \cdot L_B^2}{3} \qquad \text{(Ec 3.1.) [20]}$$

$$I_{Masa} = m_M \cdot r^2 \qquad \text{(Ec 3.2.) [20]}$$

$$I_{Total} = I_{Brazo} + I_{Masa} \qquad \text{(Ec 3.3.)}$$

Donde:

- m_B: masa del brazo.
- L_B: longitud del brazo.
- m_M: masa de la masa puntual.
- $r = L_B$: distancia de la masa puntual al centro de giro, en este caso, es igual a la longitud del péndulo.

Por tanto, para una varilla dada, la forma de aumentar la inercia es incrementar el peso de la masa puntual, o colocarla más lejos del centro de giro del péndulo, es decir, aumentando el parámetro *r*.

Volviendo a la explicación de los componentes del péndulo. Como las tuercas que hacen de masa son de un diámetro nominal mucho mayor que el de la varilla roscada, no pueden ir roscadas en ella. Para mantenerlas fijas y centradas, evitando un movimiento indeseado entre ellas, se ha insertado entre las tuercas y la varilla, un tubo plástico, cuya función habitual es la de revestir cables.

Además, con el mismo objetivo que este tubo de plástico, las tuercas están sujetas por dos conjuntos tuerca-arandela de 3 mm de diámetro, roscados en la varilla, estando uno de cada lado del conjunto de las tuercas.

Se han construido dos péndulos, que están formados por los elementos ya comentados, y sólo se diferencian en su longitud. El péndulo principal, con el que se ha realizado la experimentación, está formado por una varilla roscada de 0.15 m de longitud, en cuyo extremo se encuentra la masa oscilante.

Con este péndulo de 0,15 m, se consigue la inercia necesaria para hacer oscilar la maqueta fácilmente. Su principal inconveniente es precisamente su longitud, ya que, si el servo se mueve más allá de los límites angulares determinados por el programa de Arduino, el péndulo choca con el carenado. Pudiendo llegar a producir la rotura de una de las paredes del carenado, o de la unión del péndulo.

En teoría este problema no debería producirse dado que en el programa que controla el servomotor se ha limitado a que la amplitud máxima sea de 20º hacia cada lado de la vertical del servomotor. De esta manera, habría todavía un límite de unos 10º hacia cada uno de los lados, hasta que se produjese el choque.

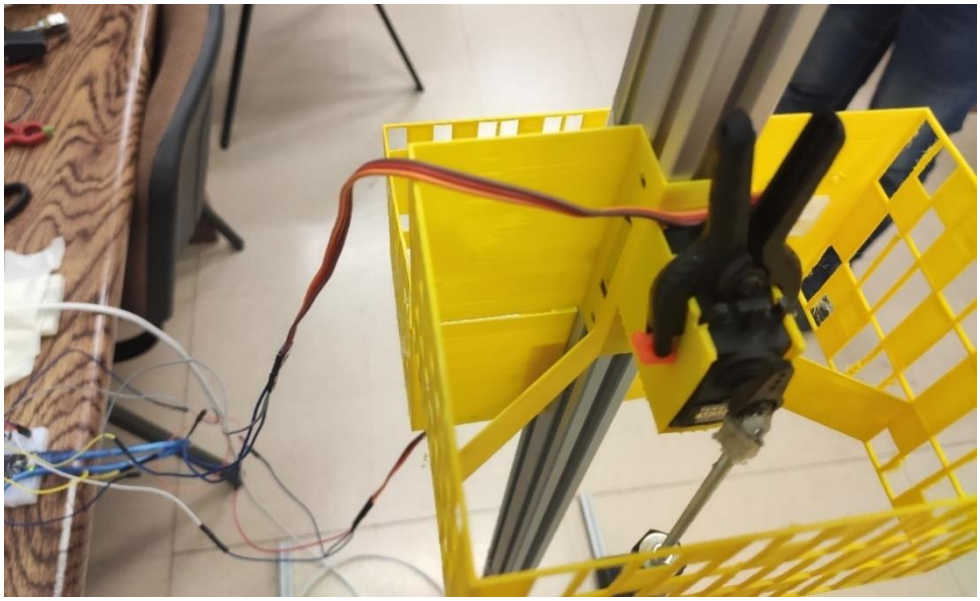


Figura 3.30. Servo excitador y péndulo largo insertos en el carenado porta servo, pueden verse las conexiones entre el servo y la placa de Arduino durante una prueba.

Se puede asegurar que una vez el servo entra en funcionamiento, el choque no se va a producir. Cuando sí se ha producido este choque ha sido durante el proceso de conexión o desconexión de los servos, que por algún motivo que no hemos llegado a comprender, en ciertas ocasiones al llegar corriente o al quitársela, los servos hacen un movimiento repentino y aleatorio, que los hace girar más allá de los límites establecidos. Se ha investigado sobre este tema en diversas fuentes, sin llegar a obtener un motivo, ni una solución.

Como medio para intentar solventar este problema del choque del péndulo con el propio carenado, se ha construido otro péndulo de menor longitud, cuya barra es de 0.06 m de longitud. Siendo esta la única diferencia entre ambos péndulos.

Si bien, con este péndulo resolvemos el problema comentado, al ser la barra de menor longitud, tiene menor inercia, lo que dificulta la oscilación de la maqueta. Si que se consigue una cierta oscilación, pero de menor amplitud de la que se consigue con el péndulo más largo, por lo que ha sido este último el que se ha utilizado para los ensayos realizados.

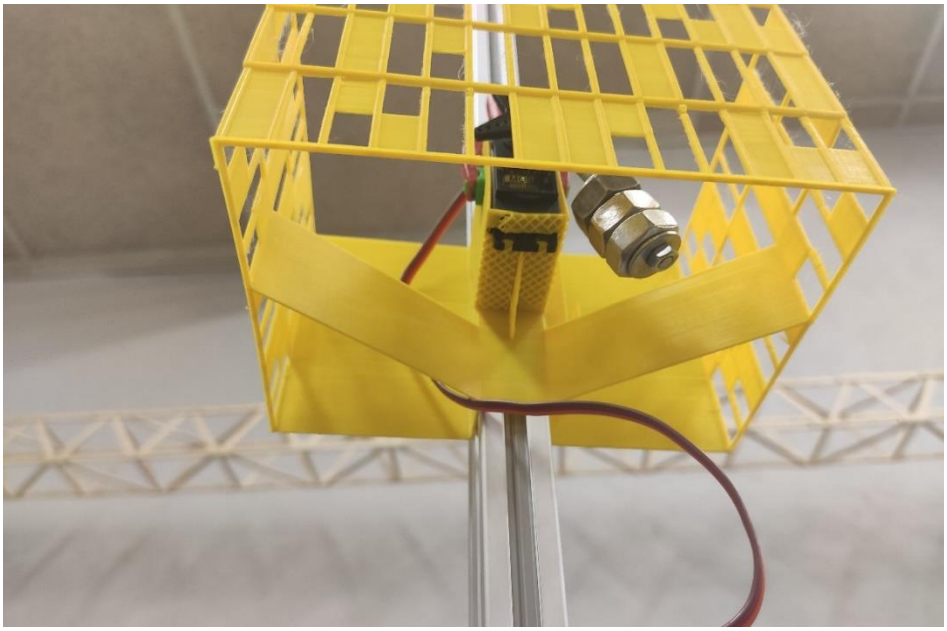


Figura 3.31. Servo excitador y péndulo corto insertos en el carenado porta servo.

Por último, como ya se ha comentado, se pretende que el péndulo oscile respecto a la posición vertical que tiene el propio péndulo cuando está colocado en el carenado. El servo es capaz de controlar y conocer su posición en todo momento, pero en función de en qué posición se inserte el enganche al que está unida la barra, puede oscilar en una posición no deseada y producir los problemas de choque con el carenado enunciados anteriormente.

Para evitarlo, es importante que si se ha desenganchado el péndulo del servo excitador por el motivo que sea, y antes de volver a ponerlo en marcha. Se ha de sacar el servo del carenado, colocar el péndulo y hacerlo oscilar para saber si el recorrido que realiza es el adecuado o no. En caso de no serlo, se varía la posición del péndulo el ángulo necesario y se vuelve a realizar la prueba antes de volver a insertar el servo en el carenado hasta tener el péndulo en la posición deseada.

3.4. TMD (Tuned Mass Damper)

El TMD o disipador pendular, es otro de los principales elementos de la maqueta, es el encargado de absorber las vibraciones mediante su balanceo. Si este componente se encuentra bloqueado ha de mitigar el efecto del excitador. Este elemento va alojado en el carenado porta TMD.

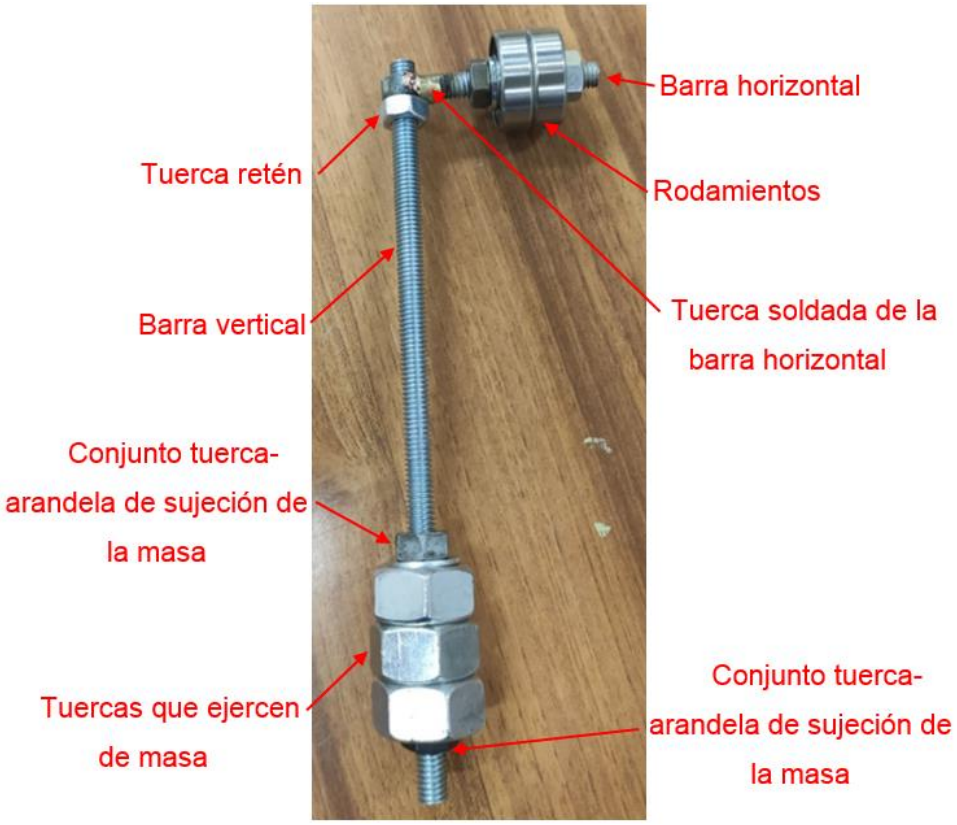


Figura 3.32. TMD pendular. En modificaciones posteriores se colocó un imán en la barra vertical, y se disminuyó la longitud de la barra soldada. Con indicaciones de las partes del TMD.

El TMD es el elemento más complejo de los que se han construido. Está formado por dos barras roscadas de 3 mm de diámetro, la de menor longitud, de 0.025 m, cuando está colocada en el carenado está en posición horizontal, paralela al suelo. Esta tiene una tuerca soldada en uno de sus extremos, mediante la cual se unen las dos barras de este componente. Esta barra más larga, de 0.105 m, es perpendicular al suelo y por tanto a la barra horizontal.

En la barra de menor longitud se encuentran alojados dos rodamientos, que son los dos rodamientos centrales de dos *fidget spinner*, como el de la Figura 3.33. Se han utilizado solo los centrales porque son de mejor calidad que los rodamientos de los lóbulos de este juguete.

Figura 3.33. Fidget spinner.[21]

Como no están roscados en su circunferencia interior, para fijarlos se ha aprovechado la tuerca soldada en la barra más corta y una tuerca de 3 mm para impedir el movimiento en el otro lado.

Estos rodamientos tienen además la función de ser los elementos mediante los que se encaja el TMD en el carenado. Lo hacen mediante clipaje en el soporte para el TMD, en el que se ha realizado un vaciado de 11.05 mm para albergar estos rodamientos.

Para fijar la posición de la barra horizontal en la barra larga, y que no se vaya soltando la rosca con la oscilación del propio TMD, se coloca una tuerca justo debajo de la unión entre ellas, que actúa como retén.

Figura 3.34. Detalle de los rodamientos y la unión entre las dos barras.

Siguiendo hacia debajo de la barra más larga, nos encontramos con dos tuercas roscadas juntas, en las que además deben coincidir sus 6 lados de forma paralela. Por el hecho de que sobre ellas irá colocado el imán del amortiguador y este ha de ser perfectamente paralelo a la placa de aluminio que también va alojada en este carenado.

Por último, cerca del final de la barra hay 3 tuercas de 6 mm de diámetro, iguales que las del péndulo excitador, sujetas de la misma manera que ellas. Y que también tienen la misma función, hacer de masa oscilante.

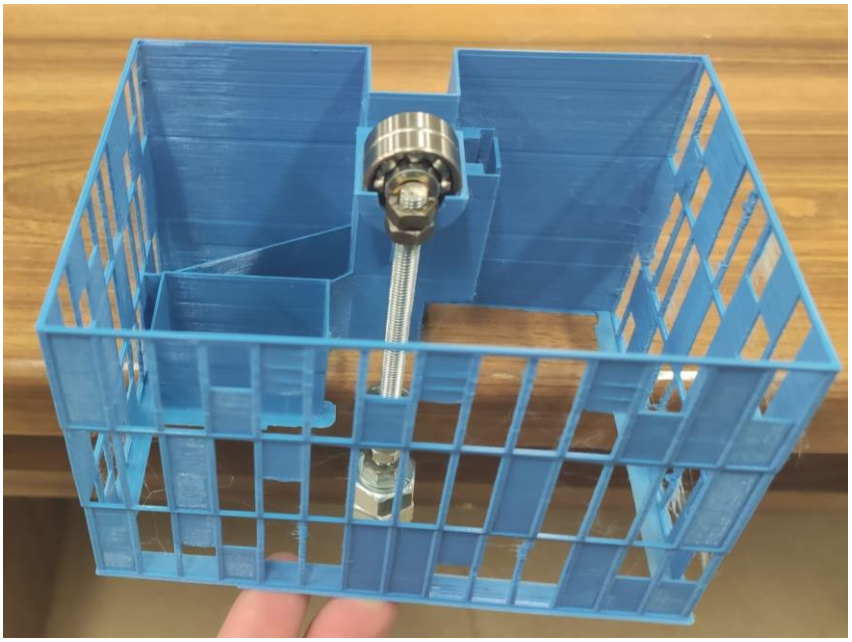


Figura 3.35. TMD definitivo inserto dentro de la 2ª versión impresa del carenado porta TMD.

Es muy importante la longitud que tenga este péndulo, es decir, la distancia entre el punto de unión de las dos barras y las tuercas que ejercen como masa. Pues es esta longitud la que va a determinar el periodo de oscilación del TMD.

Es por eso, que en este elemento no se encuentra la masa justo al extremo de la barra, porque se busca algo distinto que en el caso del excitador. En este caso es necesario, para que el TMD sea realmente eficaz, que su frecuencia de oscilación coincida con la frecuencia de resonancia de la maqueta, que también ha de ser la misma que la frecuencia de excitación del *shaker*, como ya se ha comentado en el apartado anterior.

El control de este periodo de oscilación se hace mediante el ajuste de la longitud del péndulo. Y al igual que en el caso del excitador, lo idóneo sería ajustar el periodo de oscilación del TMD según la configuración de elementos que haya montados, que van a determinar la frecuencia de resonancia de la maqueta.

Se trata de un elemento en el que no se ha tenido que realizar casi ninguna modificación, más allá de la longitud de la barra horizontal, que en las primeras construcciones realizadas era más larga.

Este cambio ha sido motivado porque es conveniente que el TMD, que tiene una masa relevante en el conjunto de la estructura, no se encuentre muy alejado del núcleo de la barra central de la estructura. Puesto que a mayor distancia se generan mayores fuerzas de torsión indeseables, que pueden provocar oscilaciones de la estructura en planos diferentes al deseado.

3.5. Amortiguador

Este es un elemento formado por dos componentes, una placa de aluminio, de dimensiones 100x50x2 mm y un imán de neodimio. Su finalidad es la disipar la mayor cantidad de energía posible para disminuir la oscilación de la estructura. Esto es posible gracias a las corrientes de Foucault que crean pérdidas de energía a través del efecto Joule, que convierte esta energía en calor. [22]

Las corrientes de Foucault son inducidas cuando un conductor, en este caso la placa de aluminio se mueve dentro de un campo magnético, el creado por el imán, o viceversa. También se crean cuando el conductor está situado en un campo magnético variable, pero no es el caso en este trabajo. Las corrientes son producidas porque se genera un movimiento de los electrones en el conductor. [22]

En el caso del amortiguador utilizado en la maqueta, es el imán el que se mueve oscilando cerca de la placa.

Estas corrientes no siempre son adecuadas, ya que también se generan en los núcleos de los transformadores, produciendo pérdidas. Pero como se muestra en este trabajo pueden ser aprovechadas como freno o amortiguador, o para producir calor, como en los hornos de inducción. [22]

El amortiguador se complementa con el TMD para ayuda a que la estructura deje de oscilar en un menor tiempo. De hecho, el imán está montado sobre el propio péndulo del TMD y la placa de aluminio está inserta en el mismo carenado que este, el carenado porta TMD.

El principal factor a tener en cuenta en este sistema es el control de la distancia que hay entre el imán y la placa, porque cuanto menor sea esta distancia más energía se logrará disipar, sin llegar a tocarse.

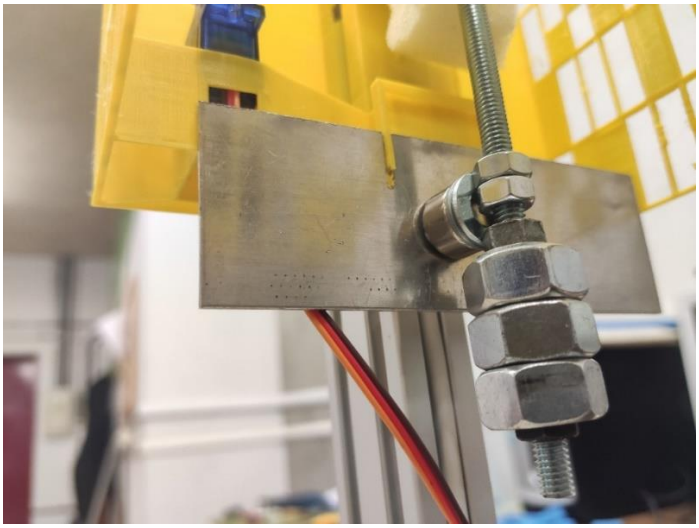


Figura 3.36. Sistema de amortiguación de la maqueta montado sobre el carenado porta TMD.

Con el fin de poder enganchar la placa de aluminio al carenado, se realizan en ella dos ranuras de 0.6 mm de anchura y 15 mm de longitud. Las ranuras se han realizado con una sierra para metal, y se introducen en las paredes laterales del soporte asegurando la sujeción de la placa.

Para situar al imán en la posición deseada, se cuenta con dos tuercas, sobre las que se pega el imán. Como es necesario que la distancia entre el imán y la placa de aluminio sea la mínima posible, para que la disipación de energía sea mayor, se añaden 3 arandelas entre el imán y estos tornillos. En los TMD reales de los edificios, este sistema de disipación de energía suele estar formado por amortiguadores.

3.6. Servo bloqueador

Este elemento es el encargado de bloquear y desbloquear el TMD, se trata de una varilla de acero de 1.5 mm de diámetro y 78 mm de longitud, que se denomina brazo bloqueador, y que está unida a un servomotor SG90, que es el que realiza el movimiento.

En primer lugar, se va a comentar las características del servo utilizado en el bloqueador, y posteriormente el brazo.

- Servomotor SG90

Se trata de un servo de pequeñas dimensiones, 32x23x12 mm y muy poco peso, 14,7 g, siendo estos los motivos principales por los que se le ha escogido. No era la primera opción, para esta aplicación se iba a utilizar un servo como el del *shaker*, pero se desestimó, porque al ser de mayor tamaño generaba problemas a la hora de posicionarlo en el carenado, como se explica más detalladamente en el apartado 3.2.

Este es un servomotor menos potente que el utilizado para generar la excitación de la maqueta, tiene un par de 0.2452 N·m. Pero es un servo más rápido, tiene una velocidad de giro de 0,1 s/60º, el doble de la que tiene el servo excitador cuando se alimenta a 4,8 V. Además de ser muy barato, ya que una unidad se puede comprar por 2€.

Sin embargo, en esta aplicación es más relevante la velocidad de giro del servo que el par, y estas son precisamente las características de este modelo, por lo que es el servomotor ideal para la función que debe cumplir en el experimento.

A continuación, se resumen en una tabla las principales características de este modelo:

Tabla 3.2. Características servo SG90. [14]

Peso	14.7g
Dimensiones	32x23x12mm
Par	0.2452N·m
Velocidad de giro	0.1s/60º
Voltaje de operación	4.8V – 6V

- Brazo bloqueador

La varilla está doblada en uno de sus extremos formando un ángulo de unos 150º, Figura 3.37, de manera que la parte final de ella es paralela a la pared posterior del carenado cuando se encuentra en la posición de bloqueo.

En este extremo libre tiene pegado con pegamento termofusible un trozo de Goma-Eva, que es el que contacta con el TMD. A esta goma se le han realizado una serie de incisiones para evitar que el TMD resbale sobre ella, intentando que el bloqueo sea lo más perfecto posible.

En el proceso de diseño de este brazo se han barajado otras opciones, debido a que no estaba definida aún la ubicación definitiva del servo, por lo que tampoco podía determinar aún la forma que debía tener.

Se llegó a soluciones en las que, una vez activado el bloqueo, sólo se conseguía parar el movimiento del TMD en una dirección dejando libre el movimiento en la otra.

Figura 3.37. Varilla del servo bloqueador del TMD, hecho con Goma-EVA.

Buscando una opción en la que se bloquease el TMD en las dos direcciones en las que tiene permitido el movimiento, se llegó a esta solución en cuanto a posición del servo y forma del bloqueador.

Una vez determinado esto, se han probado diversas opciones en cuanto al material que entra en contacto con el TMD. La primera opción fue la de dejar la varilla desnuda, pero el contacto metal con metal hacía que resbalase, generando una oscilación indeseada.

La siguiente opción que se barajó, fue la de utilizar un pequeño trozo de espuma de cojín cortada de forma prismática. Con este material se conseguía un mejor bloqueo, pero al ser blando y esponjoso permitía un pequeño movimiento. Esta opción es la que puede verse montada en la Figura 3.38.

Por último, se probó con la opción que se ha utilizado finalmente, la Goma-EVA, que es con la que mejores resultados se han obtenido. Produciendo un bloqueo prácticamente perfecto, aunque en ocasiones, si el contacto no es adecuado, si se puede producir una pequeña vibración.

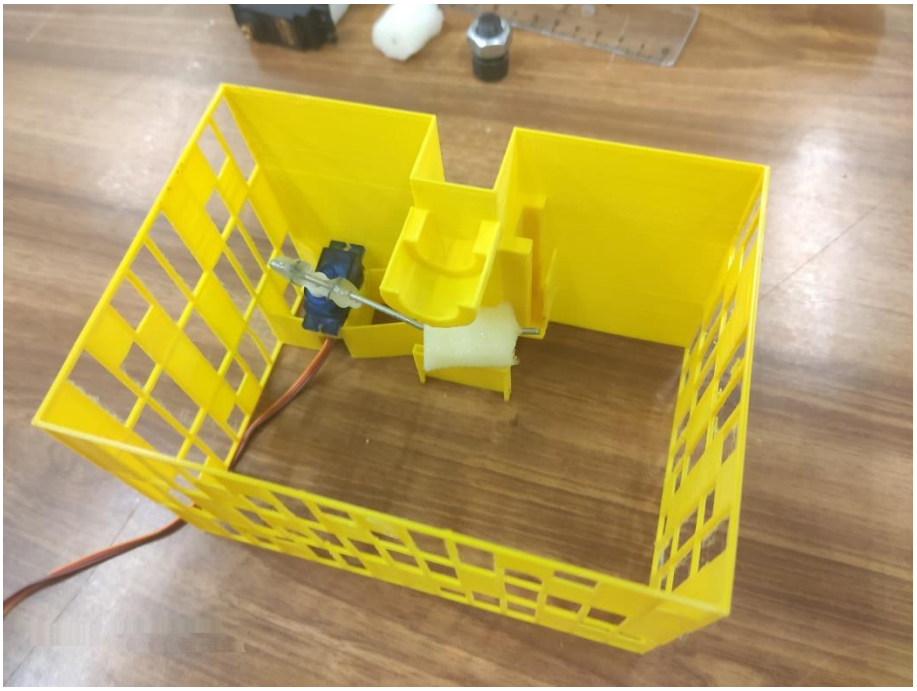


Figura 3.38. Servo bloqueador, con la varilla hecha con espuma, inserto en la versión final del carenado porta TMD.

La forma en que este bloqueador está sujeto al servo, es igual a como lo hace el péndulo excitador con el servo MG995. La varilla se une al adaptador de plástico que está unido a la salida del eje del servo, esta unión se hace en primer lugar mediante dos pequeños trozos de alambre, uno a cada lado de dicha pieza. Y después, para asegurar la unión, se utiliza pegamento termofusible cubriendo la varilla, la pieza plástica y los trozos de alambre.

4. Parte eléctrica

En este apartado se va a especificar como es el montaje del circuito necesario para poder alimentar y controlar los servomotores que se montan sobre la estructura en el ensayo.

No obstante, con la imposibilidad de ir al laboratorio debido a la alerta sanitaria COVID-19 no se ha podido llevar a cabo el montaje del circuito que se pretendía hacer. Así que se van a detallar las dos posibilidades, la que se pretendía hacer y la última prueba que se ha llegado a poner en marcha.

En cuanto al circuito para alimentar los servomotores, se dispone de una fuente de alimentación de voltaje regulable que va conectada a la corriente. En primer lugar, y esto es común para las dos opciones que se explican, hay que determinar el voltaje de este elemento.

Según las hojas de especificaciones de los servos, el excitador, modelo MG995 puede ser alimentado en un rango de voltaje entre 4.8 V a 7.2 V, y el servo bloqueador, SG90, entre 4.8 V a 6 V. Entre las opciones que nos permite esta fuente de alimentación, elegimos un voltaje de 5.6 V que está dentro de los valores óptimos de los dos servos.

Esta fuente de alimentación tiene diversas salidas, en este caso utilizaremos el de 7.40x0.90 mm porque el adaptador del que se dispone es de esta medida. Este adaptador tiene dos polos, positivo y negativo, a los que se conectan dos cables, el positivo lleva la energía a los servomotores y el negativo es la tierra.

Para poder alimentar los dos servos con la misma fuente de alimentación habría que realizar una división de cada uno de estos cables en dos, esto se puede hacer de forma relativamente fácil, pelando los cables insertos en el adaptador, que han de contar con al menos dos hilos cada uno y empalmándolos a dos de los hilos de un cable de 6 hilos y 3 m de longitud.

De esta manera, ya tendríamos resuelta la alimentación de todos los elementos, y sólo faltaría explicar la conexión entre estos servos y la placa de Arduino, que es la que controla su movimiento.

Cada uno de los servomotores tiene 3 cables, el de conexión, naranja, el de alimentación, rojo, y el de tierra, marrón. El control de los servos se realiza mediante la conexión con los pines digitales del Arduino, por lo que son necesarios dos cables, uno que una el pin D5 de la placa con el cable naranja del servo excitador y otro entre el pin D6 de la placa con mismo cable del servo bloqueador.

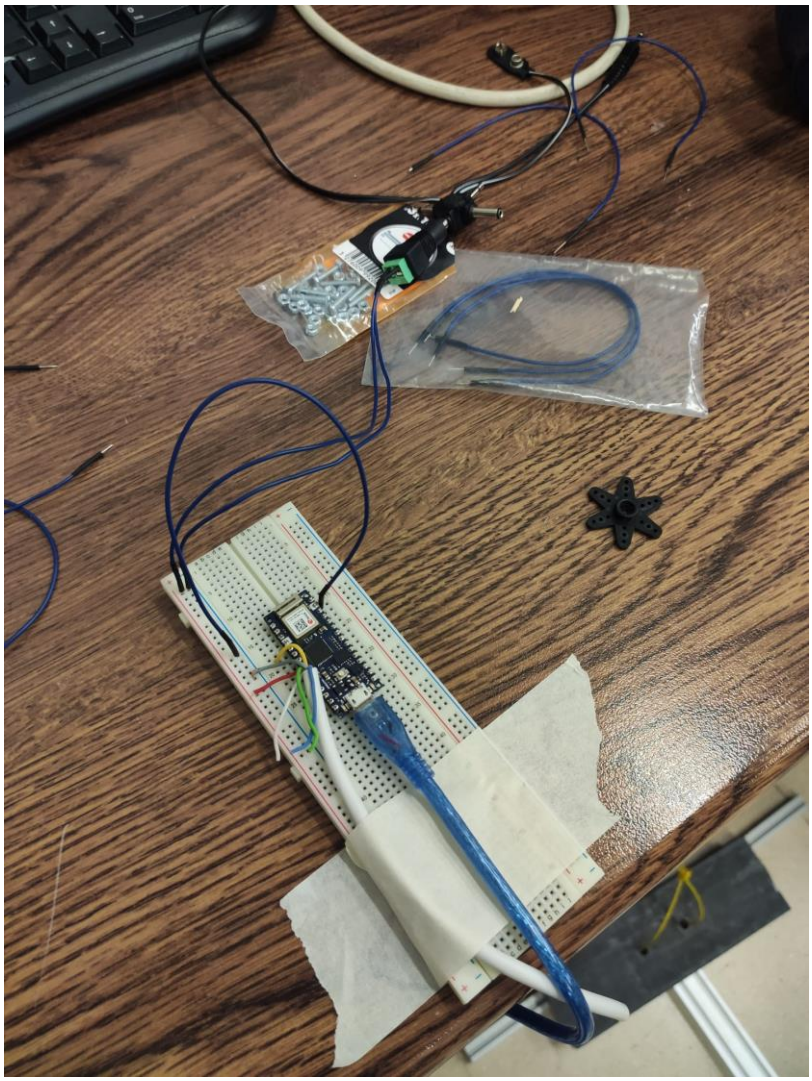


Figura 4.1. Montaje de los circuitos realizados en el proyecto. Pueden verse la salida de la fuente de alimentación, el adaptador, los cables de alimentación de la protoplaca, el Arduino y el cable de 6 hilos utilizado.

El cable que une el Arduino con el servo MG995 va a ser pequeño porque van insertos en el mismo carenado, y los propios cables del servo, según las especificaciones del fabricante son de 0.3 m, incluso el cable podría ir inserto directamente en la placa.

En cuanto al cable de conexión entre el Arduino y el otro servomotor va a ser de una longitud indefinida, ya que va a depender de la distancia a la que se coloquen el carenado porta servo y el porta TMD. Esto es fácil de resolver pues se dispone de una serie de cables de unos 0.1 m de longitud, que pueden irse conectando entre sí, dado que algunos de ellos disponen de conector "hembra".

Ahora, se va a explicar cómo se realizó la última prueba en el laboratorio, que conlleva una serie de cambios respecto a lo que se ha explicado anteriormente.

En esta situación la alimentación de los servos se realiza mediante las columnas '+' y '-' de la protoboard. Insertando los polos correspondientes del adaptador, de esta manera ya se dispone de más de 50 pines desde los que poder llevar la alimentación a los servos mediante cables.

Se cuenta con un cable de 6 hilos de unos 3 m de longitud, para conectar esta protoplaca con los servos. Se utilizan 4 de estos hilos para alimentar los servomotores.

El procedimiento que se siguió es el siguiente, se peló el revestimiento exterior del cable para dejar los 6 hilos libres. Luego se pelaron estos 4 hilos que se van a utilizar, y como en principio iba a ser una solución temporal para realizar una prueba, se recubrieron con estaño las puntas de estos cables, usando un soldado, para facilitar su inserción en los pines de la protoboard y de los servos.

La forma correcta de operación, y que en condiciones normales se hubiera llevado a cabo, es cortar un cable como los que pueden verse en la Figura 4.1. Para aprovechar sus extremos, que proporcionan unas conexiones entre los elementos más seguras, con menor riesgo de que se suelten y realizando una inserción más sencilla.

Los servomotores se pueden alimentar llevando uno de los cables que salen de la columna '+' y otro de la '-' de la protoboard a cada uno de ellos.

Sólo falta por aclarar la forma en que se conecta el control de los servos con el Arduino. Este se encuentra inserto en la protoplaca, colocado encima de una mesa, como puede verse en la Figura 4.1. La conexión se hace utilizando los dos hilos que quedan libres del cable de 6 hilos, y el procedimiento utilizado es el mismo, se recubren las puntas peladas de estos cables con estaño.

De esta manera se consigue realizar un procedimiento que sería suficiente para hacer oscilar y parar la maqueta según se requiera, controlando el sistema desde el ordenador. Aunque no es la solución idónea, ni la pretendida, supone una opción perfectamente funcional con aspectos a mejorar. Lo único que esta prueba no nos permite realizar, es la medición de las aceleraciones de la estructura, no por no poderlas medir con el Arduino, sino por no disponer de medios suficientes para poderlo colocar sobre ella.

5. Parte electrónica

5.1. Arduino

Existe gran variedad de placas de Arduino, que dan una gran oferta entre la que elegir en función de las necesidades de cada situación. En este TFG se comenzó utilizando un Arduino UNO, que se trata de la placa más básica, robusta y utilizada de las que fabrica Arduino.

Dispone de 6 entradas analógicas, 14 pines de entrada y salida digital y desde ella se puede alimentar otros elementos a un voltaje de 3.3 V o de 5 V. Cuenta además con conexión USB, un conector de alimentación para ser alimentada desde una fuente de corriente alterna, de corriente continua o una batería. [23]

Figura 5.1. Placa Arduino UNO REV 3. [23]

Pese a que en un primer momento se trabajó con una placa Arduino UNO, al disponer en el Departamento de 3 Arduino NANO se ha optado por usar una de ellas, por ser más pequeña, más fácil de ubicar en los carenados y tiene menos peso. Y para las funciones que ha de realizar en la maqueta, es mejor que el UNO ya que este no dispone de IMU para medir las aceleraciones a las que está sometida la estructura y el NANO sí.

De 3 modelos de Arduino NANO disponibles en el departamento se eligió el modelo Arduino NANO 33 IoT por ser el ideal para las funciones que ha de realizar, ya que dispone de una IMU de 6 ejes y ser la mejor opción en cuanto a comunicaciones con otros elementos.

Las siglas IoT (*Internet of Things*), Internet de las Cosas, que aparecen en el nombre de este Arduino muestran que es una muy buena opción para la comunicación con sensores u otras placas. Está especialmente diseñado para realizar conexiones Wi-Fi y Bluethooth, con una comunicación segura ya que cuenta con el chip criptográfico $Microchip^{®}$ ECC608. Es capaz de realizar conexiones con dispositivos BLE (*Bluethooth Low Energy*), para enviar datos a un teléfono móvil. [24]

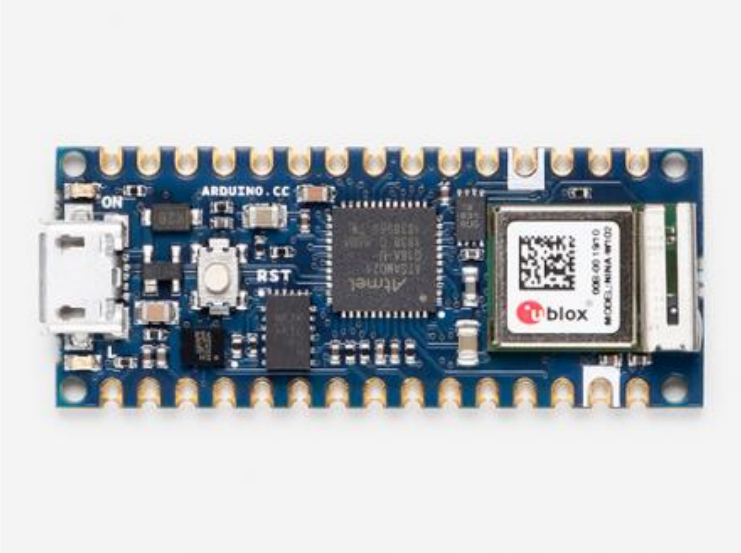


Figura 5.2. Placa Arduino NANO 33 IoT. [24]

Además, existen proyectos de la comunidad de Arduino como puede ser Blynk, que permite operar una placa desde un teléfono, o IFTTT, para controlar enchufes inteligentes, entre otras. [24]

Una de las características más interesantes de esta placa es que cuenta con una IMU de 6 ejes (Inertial Measurement Unit). Es una unidad electrónica que obtiene información del movimiento y cinemática del elemento en que se encuentre instalada, en este caso la placa. Esta IMU es la LSM6DS3, está formada por un acelerómetro 3D, y un giroscopio 3D. [24]

Esta placa tiene tres cometidos principales, generar y controlar la oscilación del servomotor excitador, controlar el movimiento del servomotor bloqueador y medir y registrar los datos de la aceleración de la estructura.

En los ensayos realizados, las placas de Arduino utilizadas, han estado insertas en protoplacas o protoboard, que son unas placas en las que se pueden insertar elementos electrónicos y cables para construir circuitos sin necesidad de soldar los componentes. Estas placas pueden verse en las Figuras 4.1. y 5.3.

Tiene orificios que están conectados entre sí mediante pequeñas láminas metálicas. Cuenta con dos tipos distintos de orificios, en los laterales, los buses, que son dos columnas en cada lado, una '+' en rojo, para la alimentación y otra '-' en azul, para la tierra del circuito. Estas columnas cuentan con 50 pines repartidos en grupos de 5. Los pines de cada una de estas columnas están conectados entre sí, pero los de columnas distintas no lo están.

En la parte central de la placa se encuentra el otro tipo de orificios, las pistas, que son aquellos en los que se insertan los diferentes elementos que componen el circuito. Cuenta con 61 filas de 10 pines cada una, las filas están numeradas y las columnas están indicadas mediante letras.

Cada una de las filas está separada en dos grupos de 5, con una separación entre estos dos grupos de pines, el canal central, que sirve para mantener aislados los pines de cada uno de los lados en caso de que fuera necesario. Los pines de cada una de estas filas están conectados entre sí, pero los pines de diferentes filas no lo están.

Estas protoboard suponen una gran opción a la hora de realizar pruebas de circuitos, ya que pueden utilizarse tantas veces como se quiera y son fáciles de utilizar. Sin embargo, no están preparadas para trabajar con componentes de gran potencia, y con una corriente de entre 3A y 5A. [25]

5.2. Conexión y alimentación

En este apartado se va a explicar cómo se realiza la alimentación del Arduino y la conexión de este con el ordenador. La placa utilizada en este trabajo solo puede alimentarse desde el propio ordenador, así que la alimentación y la transferencia de datos se realiza por el mismo canal, mediante un cable tipo USB 2.0.-micro USB tipo b.

Esta transferencia de datos es en dos sentidos, en primer lugar, para subir el programa desde el ordenador a la placa, y después para realizar una comunicación constante para activar y desactivar los servos o cambiar las variables de oscilación del servo excitador.

En primer lugar, se va a explicar el montaje completo que se pretendía realizar. Dado que el Arduino ha de ir colocado sobre el carenado porta servo, y para poder alimentar y transferir datos desde el ordenador a la placa de Arduino, estos elementos tienen que ir necesariamente conectados entre sí. Hace necesario un cable de conexión con unos conectores como los ya comentados, y que sea de gran longitud para tener la libertad de poder colocar este carenado a la altura que se desee dentro de la maqueta.

Este cable como ya se ha comentado, ha de ser del tipo USB 2.0.-micro USB tipo b, siendo la conexión usb 2.0. para el ordenador y el otro tipo para el Arduino. Buscando entre las posibilidades que reúnan estas condiciones, se encontró una solución idónea en un cable de 6 metros a un precio adecuado. Con este cable se resuelve la conexión entre estos dos elementos.

Al igual que en el apartado 4, se va a explicar ahora como se ha realizado el montaje en la última prueba de la experimentación.

El principal cambio se produce en la posición de la placa de Arduino, ya que al no disponer del cable de conexión entre ordenador y Arduino de una longitud suficiente es imposible realizar el ensayo con la placa montada sobre el carenado. Esta estaba enganchada en una protoplaca situada en una mesa junto a la maqueta.

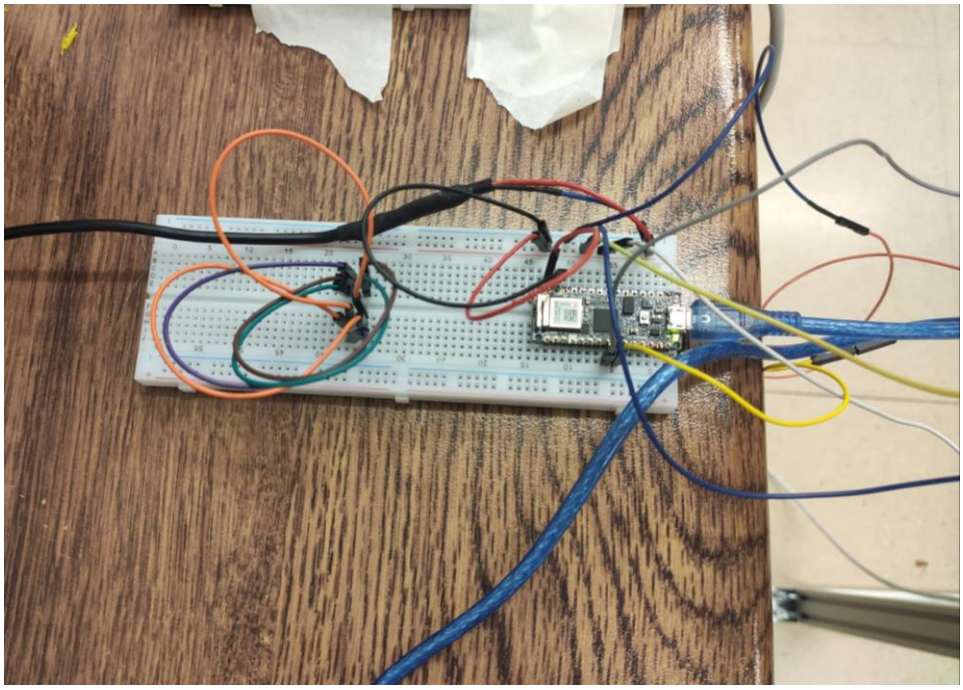


Figura 5.3. Conexiones de la placa de Arduino realizadas durante una prueba junto a los colaboradores de la universidad de Oviedo,

El problema que produce esta posición del Arduino es que no se han podido medir las aceleraciones a las que se encuentra sometida la estructura. Aunque es de destacar que si se puede medir y observar la gráfica de las aceleraciones que sufre la propia placa de Arduino, y si se hubiera dispuesto de este cable se podría haber realizado esta medición.

5.3. Comunicación entre elementos

Este apartado se centra en explicar brevemente el programa de Arduino con el que se ha realizado la programación, algunas de las maneras en las que se pueden comunicar la placa y el ordenador, y cómo puede compartirse el código creado.

El software utilizado para realizar la programación necesaria en este trabajo es el programa que proporciona de manera gratuita la propia empresa Arduino. Este programa puede utilizarse de dos maneras, en su plataforma online, o descargar el programa y trabajar *offline*.

El entorno propio de desarrollo del programa está escrito en Java, pero el lenguaje que han de utilizar los usuarios es C++ que es sencillo de aprender y utilizar.

Este es un programa de código abierto, en que los usuarios pueden crear sus propias de librerías y compartirlas con el resto de la comunidad. Se tiene libertad a la hora de crear estas librerías siempre que se cumplan las especificaciones marcadas por los desarrolladores.

Algunas de estas librerías vienen preinstaladas en el propio programa y definen ciertos comandos que permiten controlar diferentes elementos que suelen utilizarse en este tipo de programas, como pueden ser sensores de temperatura, servomotores o acelerómetros.

Figura 5.4. Gestor de librerías del software Arduino.

Pero en la mayoría de los casos hay que instalarlas por tratarse de elementos muy concretos. Aunque el proceso de descarga de estas es muy sencillo, dentro de la pestaña *Herramientas → Administrar Bibliotecas...* se accede al gestor de librerías, que dispone de un buscador en el que, introduciendo, por ejemplo, el modelo de sensor del que se dispone aparece un listado de librerías compatibles.

Sobre ellas aparece el botón *Instalar*, pinchando sobre él automáticamente se instala la librería y ya puede utilizarse. Junto con la librería se descargan una serie de ejemplos con los que testar los elementos y su programación. Se puede acceder a ellos desde la pestaña *Archivo* → *Ejemplos*, y buscando la librería descargada.

Cabe comentar la facilidad con la que es posible compartir el código generado en este software, ya que dispone de dos comandos dentro de la ventana *Editar*, *Copiar al foro* y *Copiar como HTML.* El primer comando copia todo el código de Arduino para poderlo pegar en un editor de texto o en el foro de la comunidad de Arduino. El segundo, copia el código y lo transforma a lenguaje *HTML*, que es un código que suele utilizarse en el desarrollo de páginas web.

A continuación, se va a explicar cómo se realiza la comunicación entre el ordenador y la placa de Arduino, en los diferentes programas utilizados en el desarrollo del TFG. En primer lugar, se va a comentar como es este proceso en el programa desarrollado por nosotros, y después, se explica el método utilizado en el programa creado por los colaboradores de la Universidad de Oviedo.

Los puertos serie son la forma más habitual de comunicar una placa de Arduino con un ordenador, pero también se usan para la comunicación con otros sistemas como los módulos bluetooth. Esta forma de comunicación es la que se ha utilizado en las distintas pruebas que hemos ido desarrollando para el control de los servos.

En todos los modelos de placas de Arduino, hay al menos un puerto serie, también llamados UART o USART, aunque hay algunas que disponen de hasta 3 puertos serie adicionales, como es el caso del Arduino mega.

El Arduino NANO, que es el que se utiliza en este proyecto, cuenta con un puerto serie, formado por los pines 0 y 1, compartido con el puerto USB de la placa. Es conveniente sólo utilizar estos pines para comunicarse con el ordenador, ya que si se conectase otro elemento podría causar problemas al cargar el programa a la placa. [26]

El puerto 0 también denominado RX es el encargado de la recepción de datos, y el 1 o TX se encarga de la transmisión. Esta transferencia de datos en uno y otro sentido se realiza según una secuencia de bits. [27]

La manera más sencilla para la transmisión de datos mediante el puerto serie es la misma que para subir el programa a la placa, es decir, utilizando un cable que conecte los puertos USB de los dos elementos. En el caso de la placa Arduino NANO, el tipo de conexión USB de la que dispone es micro USB tipo b, por tanto, se ha de disponer de un cable que tenga esos dos tipos de puertos.

Una vez realizada la conexión física, se ha de indicar en el código la velocidad a la que se quiere realizar la transmisión, en baudios, esto suele realizarse en el bucle setup() del programa mediante el comando **Serial.begin()**. Ahora clicando en el **Monitor Serie**, situado en la parte superior derecha de la pantalla, ya se puede enviar y recibir información.

Figura 5.5. Muestra de algunos de los comandos más utilizados para la comunicación mediante el puerto serie.

Un hecho para tener en cuenta es que el mismo número de baudios que se definen dentro del comando Serial.begin(), se han de establecer también dentro del Monitor Serial. Ya que si no coinciden no se podrá realizar la transferencia de datos entre estos elementos. Es un detalle simple, pero que suele dar lugar a problemas.

El monitor serie dispone de dos zonas, una para enviar los datos requeridos, y otra en la que se muestran los datos recibidos.

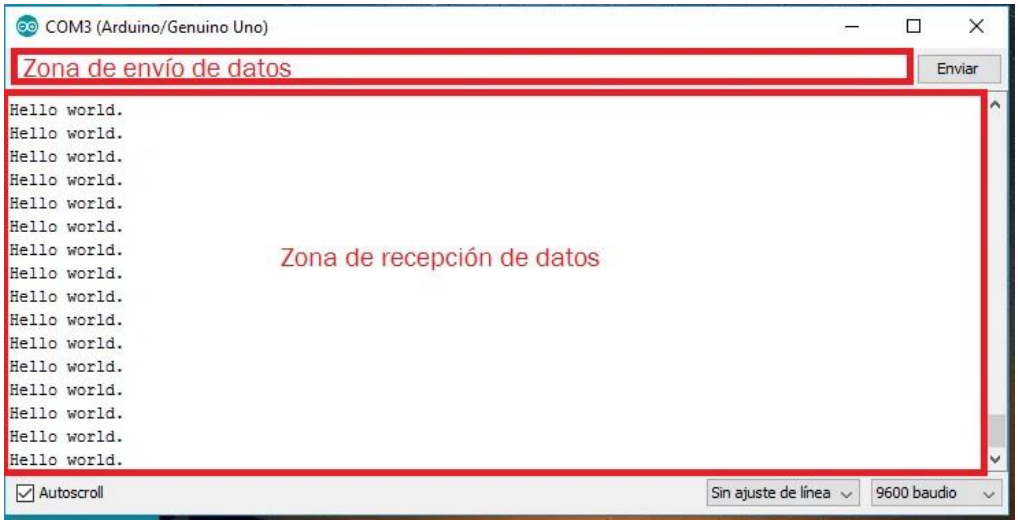


Figura 5.6. Monitor serial del software de Arduino, con sus dos zonas principales marcadas. [26]

Algunos de los comandos más útiles y utilizados referidos al puerto serie son **Serial.print(),** que imprime datos al puerto serie como código ASCII, **Serial.read()** que lee el primer byte entrante desde el buffer serie o **Serial.available()** que da el número de bytes disponibles para leer en el puerto serie, que están almacenados en el buffer. [28]

En el programa creado por nuestros colaboradores la comunicación entre los diferentes elementos se realiza mediante el protocolo de comunicación MQTT, que permite la comunicación M2M (*Machine to Machine*), que comunica entre sí diferentes dispositivos IoT. Estos dispositivos son básicamente cualquiera que tenga la posibilidad de conectarse a internet. [29]

El MQTT es un servicio de mensajería *push* con patrón publicador/suscriptor, es decir, los clientes se conectan a un servidor central llamado *broker*, que en este caso es el programa *Mosquitto*. Este servidor se organiza mediante *topics* que se organizan jerárquicamente, y que sirven para filtrar la información. Los clientes se suscriben a los *topics* que necesiten y pueden publicar un mensaje en él o recibir aquellos mensajes que lleguen a ese *topic*. En todo momento, los clientes están conectados al *broker* a la espera de que llegue información, o que quieran enviar un mensaje. [30]

En este caso se han utilizado 5 topics, uno para cada una de las posibles necesidades que se requieren en este trabajo. Uno para los datos recogidos por el acelerómetro, otros dos para activar o desactivar cada uno de los servos, y otros dos para las dos variables de operación del servo excitador.

Los *topics* para controlar los servos, sólo se utilizan cuando se quiere activar o desactivar uno de los servos, cada uno de los dos tiene su propio *topic*. Tanto la amplitud de oscilación del *shaker*, como su periodo pueden variarse, por lo que cada uno de ellos tiene su propio *topic*, que sólo se usa cuando se varía uno de estos parámetros. Y, por último, el *topic* de los datos del acelerómetro está utilizándose prácticamente siempre ya que la gráfica de la aceleración se muestra en tiempo real.

Han creado un entorno para controlar la maqueta. Este entorno está formado por la placa de Arduino, en la que se carga el programa desarrollado en el software de Arduino, y la interfaz o página web que han creado, desarrollada en *HTML* JavaScript. La comunicación entre ambos se realiza mediante el protocolo MQTT.

En este caso concreto, el funcionamiento sería así:

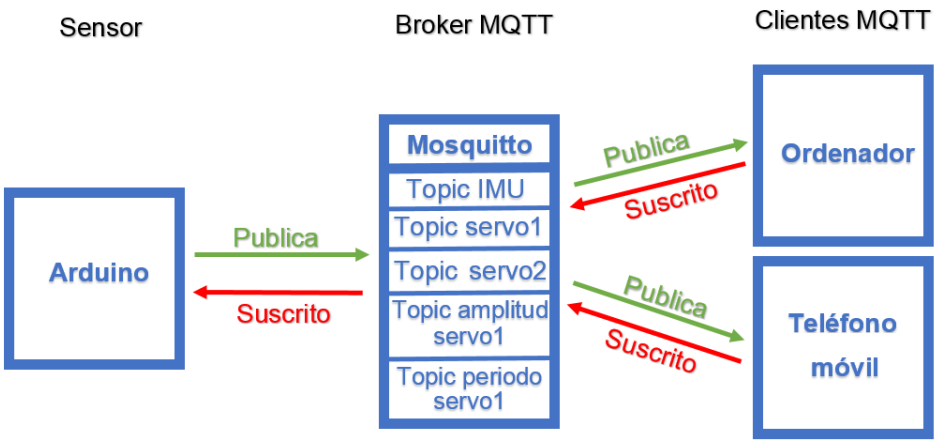


Figura 5.7. Esquema del funcionamiento del protocolo MQTT de este trabajo.

Este entorno no sólo nos permite controlar la maqueta desde el ordenador mediante la página web, sino que además podríamos controlarla desde otros dispositivos, como puede ser un teléfono móvil. Para ello sólo sería necesario descargar una aplicación para conectar el dispositivo móvil a MQTT, estas aplicaciones pueden encontrarse en cualquier plataforma de distribución de aplicaciones móviles.

Una condición indispensable para poder realizar este tipo de conexiones entre diferentes dispositivos es que todos ellos estén conectados a la misma red Wi-Fi.

El proceso de comunicación que sigue este entorno es el siguiente. El usuario opera mediante la página web, Fig 6.1. desde la que pueden controlar los dos servos, activarlos o desactivarlos, o modificar las variables de operación del *shaker.* En el momento en que se modifica alguno de estos comandos, el programa publica la información en el *broker*, *Mosquitto*, dentro del *topic* que corresponda. Y la información que se haya publicado llega al Arduino, que la ejecuta, debido a que es el encargado del control de los servomotores.

Pero esta comunicación no es sólo en este sentido, pues el Arduino, que actúa como sensor, está constantemente publicando en el *broker,* en otro *topic*, el que corresponda a los datos del acelerómetro la información de la aceleración a la que se encuentra sometida. De manera inversa es ahora la interfaz que recoge esta información y la publica en la pantalla en forma de grafica.

6. Parte software o control

En este apartado se explican los objetivos que deben cumplir los programas desarrollados para realizar el ensayo sobre la maqueta. En primer lugar, se comentarán las características del programa desarrollado con nosotros, y después, las del entorno creado por los colaboradores de la Universidad de Oviedo.

Estos programas han de realizar 3 tareas fundamentales, que son:

- Generar y controlar la oscilación del servomotor excitador, que es el que produce el movimiento que hace oscilar toda la estructura.
- Controlar el movimiento del servomotor encargado del bloqueo y desbloqueo del TMD.
- Medir y registrar los datos de la aceleración a la que se encuentra sometida la propia placa de Arduino y, por tanto, la estructura.

El código de estos programas se muestra en los Anexos I y II.

6.1. Programas de desarrollo propio

Como parte fundamental y como aprendizaje necesario en la realización de este TFG, se han desarrollado tres programas que son capaces de realizar las funciones que les son requeridas, pero de manera independiente, por lo que para poder realizar un ensayo que esté controlado por estos programas, serían necesarias tres placas de Arduino diferentes.

Podría haberse continuado con la evolución de estos programas para acabar trabajando con los tres en uno solo, y solo necesitar un Arduino. Pero fruto de la colaboración ya comentada, se ha optado por utilizar el programa que ellos han desarrollado, porque con su conocimiento y experiencia, el programa que ellos han hecho es seguramente mejor que el que nosotros hubiéramos conseguido. A continuación, se van a explicar cada uno de los tres.

- Programa del excitador.

Es el programa principal, el más complejo y que más se ha tardado en desarrollar, es el responsable de hacer oscilar el servomotor excitador, MG995. Se trata de un programa en que sólo se utiliza una librería, **Servo.h**, que tiene como fin definir los comandos necesarios para controlar el servo.

Cuenta con seis variables, tres relacionadas con el servo, tipo **const int**, que van a ser constantes durante todo el tiempo de ejecución del programa. Una para definir el pin desde el que se controla, y otras dos para definir su pulso máximo, 2500 y mínimo, 500. Otra de las variables es tipo **char**, que es un carácter para definir que opciones de las que dispone el programa queremos elegir. Las otras dos son tipo **float**, y que son las que están relacionadas con el tiempo de ciclo y la amplitud del servo.

Cuenta con tres bucles o funciones, el primero es **void setup()**, es uno de los bucles básicos de un programa de Arduino, sólo se ejecuta una vez, y en este programa sólo se utiliza para definir los parámetros de funcionamiento del servo e inicializar el puerto serie a 9600 baudios. El siguiente es el bucle **void loop()**, es el otro bucle básico, y este se ejecuta de manera continua mientras se esté ejecutando el programa.

Las primeras líneas de código de este bucle son las responsables de hacer oscilar el servo. Se manda ir al servo a una posición determinada mediante el comando **servo.write()**, esta posición está definida mediante las operaciones (90-amplitud) y (90+amplitud), esto es debido a que se pretende hacer oscilar a que el servo tiene un rango de actuación de 0º a 180º, y no podrían utilizarse las posiciones +amplitud y -amplitud, porque el servo no es capaz de ir a esta última.

Justo después de cada una de estas líneas, mediante el comando **delay()** se hace “esperar” al servo en dicha posición un tiempo igual a la mitad del tiempo de ciclo que se haya definido. Estas cuatro líneas de código se repiten constantemente mientras se esté ejecutando el programa, lo que hace que el servo oscile entre estas dos posiciones en un tiempo total igual al tiempo de ciclo.

La parte final de esta segunda función, son tres bucles tipo **if**, su función es la de mostrar por pantalla que opción se ha elegido. Se cuenta con tres opciones o modos, el modo 'a' es el que permite variar la amplitud de oscilación, el modo 't' permite modificar el periodo y el modo 'c' centra al servo, es decir, le manda al ángulo 90º.

Es importante que al final de este segundo bucle se inicialice la variable 'Modo' que es la que permite elegir la opción que se desea. Para ello simplemente se ha de igualar a un valor distinto a las opciones con las que cuenta el programa, para que al repetirse el bucle no entre en los bucles **if.**

Por último, está la tercera función, **void serialEvent()**, esta, si hay datos en el puerto serie los agrega en una cadena hasta que se encuentra un salto de línea, y entonces reestablece el valor nulo. Este hecho es relevante a la hora de operar con el programa, ya que para elegir cualquiera de los modos, se ha de comenzar escribiendo la letra correspondiente al modo que se quiera elegir.

Si lo que se pretende es variar las condiciones de operación del servo, se eligen los modos 'a' o 't', se ha de escribir a continuación de la letra el nuevo valor que se desee, y después enviar la información, que genera un salto de línea. De esta manera se consigue que el programa haya leído toda la cadena que le hemos mandado.

Por ejemplo, si se quiere variar la amplitud a 20º, habría que introducir la cadena 'a20', o si se quiere variar el tiempo de ciclo a 800 ms, se debe introducir 't800'.

Este proceso se produce porque una vez que se le envía esta cadena el programa lee el primer byte de la cadena, que debe ser una de las letras, y la iguala a la variable 'Modo' utilizando el comando **Serial.read()**, que sólo lee el primer byte de los que llegan por el puerto serie.

Entonces mediante el comando **switch(Modo)** se crea un menú que tiene las tres posibilidades ya comentadas, y en función al valor que tenga esta variable entra en uno u otro de los posibles casos. Este proceso tiene una metodología determinada, se ha de escribir **case 'a', case 'c'** o **case 't'**, y al final de cada uno de los casos se tiene que escribir **break**, para que salga del caso.

En el supuesto de que se elija Modo=a, se iguala la variable amplitud al comando **Serial.parseInt()**, que devuelve el primer entero válido en el puerto serie, en este caso el nuevo valor de la amplitud e ignora aquellos caracteres que no sean números, en esta ocasión la 'a'. Si el modo elegido es el 't', el programa hace lo mismo, pero igualándolo a la variable tiempo de ciclo. Por último, si se elige el modo 'c', el servo se va a su posición 90º y espera ahí durante 5000 ms.

- Programa bloqueador

Este programa tiene la misma estructura que el anterior, debido a que al igual que en el otro caso, se busca controlar el movimiento de un servomotor. En este caso se trata del servo bloqueador, que es del modelo SG90.

Se utiliza la misma biblioteca, la necesaria para controlar un servomotor, y se dispone de 4 variables, las tres del servo, que definen su pulso máximo, 2500 y mínimo, 500 y el pin desde el que se controla. La última variable es llamada 'Modo' y tiene la misma función que la del otro programa.

La primera función **void setup()** sirve para inicializar el puerto serie a 9600 baudios, se definen los parámetros de funcionamiento del servo y se ordena al servo ir a una posición media entre el bloqueo y el desbloqueo, que son 20º, y mantenerse en ella durante 3 s. Esto se realiza con el fin de ubicar al servo, para asegurarnos de que su posición es la correcta.

La siguiente es la función **void loop()** que en esta ocasión cuenta con dos bucles **if()** dentro de él, que si coincide el modo elegido con la condición de entrada al bucle, muestran por pantalla la opción elegida. Al final del bucle es necesario inicializar la variable Modo a un valor diferente a los posibles modos, que en esta ocasión son dos, modo 'b' para bloquear el TMD y el modo 'd' para desbloquearlo.

Por último, está el bucle **void serialEvent()**, al que se entra cuando se recibe algo por el puerto serie. En este caso resulta más sencillo ya que sólo se puede decidir si bloquear o desbloquear el servo, y no se pueden modificar otras variables del servo. Por lo que en este programa sólo es necesario igualar la variable 'Modo' al comando **Serial.read()** que lee el primer byte.

Y al igual que lo explicado anteriormente se crea con el comando **switch(Modo)** un menú para elegir entre los posibles casos. Si la opción elegida es la 'b', se ordena al servo ir a la posición 10º, que bloquea el TMD, esto se realiza mediante el comando **servo.write()**. Este mismo comando es el que se utiliza en el caso 'd', pero en esta ocasión el servo gira hasta la posición 30º, que desbloquea el TMD.

- Programa acelerómetro

Este programa no se va a analizar en detalle dado que se trata de un ejemplo que ya viene creado dentro de la librería necesaria para poder utilizar la IMU de 6 ejes del Arduino NANO 33 IoT, El modelo es LSM6DS3, y dispone de un acelerómetro y un giroscopio de 3 ejes cada uno.

La manera de acceder a este ejemplo es en la ventana *Archivo → Ejemplos,* se busca la pestaña correspondiente a la IMU, que según la librería que hemos descargado se llama *Arduino_LSM6DS3*, y dentro de ella hay dos ejemplos, uno para el acelerómetro y otro para el giroscopio. En este caso el que nos interesa es el *SimpleAccelerometer*, que es el que corresponde al acelerómetro.

A pesar de tratarse de un ejemplo destinado a obtener los datos de aceleración, también está programado para mostrar por pantalla los datos del giroscopio y del magnetómetro, que sirve para medir la fuerza o dirección una señal magnética.

Este ejemplo permite obtener los datos de forma numérica por la pantalla de comunicación del puerto serie, y también muestra en forma de gráfica mediante el comando *Serial Plotter*, que se encuentra en la pestaña *Herramientas.*

6.2. Programas cedidos

En este punto se va a explicar el entorno que nos han cedido para la realización de este TFG, está formado por un programa creado en Arduino, que controla la propia placa y otro código creado en *HTML* para generar una aplicación desde la que controlar el ensayo.

La explicación se centra sobre todo en el programa de Arduino puesto que es el mismo tipo de programación que se ha utilizado en el desarrollo de nuestros programas. Mientras que apenas hemos adquirido conocimientos de programación en *HTML*, y la explicación se centrará sólo en explicar la interfaz, y las posibilidades que nos permite realizar.

Este entorno ha de cumplir con los mismos objetivos que los programas explicados en el apartado anterior.

6.2.1. Programa de Arduino

- Definición de librerías y variables

Este programa de Arduino es bastante complejo, está formado por siete funciones, pero antes de explicar que hace cada una de ellas, hay que aclarar las librerías utilizadas y las variables que se definen al comienzo.

Se utilizan 7 librerías, que son:

- SPI.h: permite comunicarse con uno o más periféricos.
- WiFiNINA.h: hace que el Arduino sea un servidor que acepta conexiones entrantes o que sea un cliente que realiza conexiones salientes.
- PubSubClient.h: hace que el Arduino funcione como un cliente MQTT.
- Servo.h: sirve para controlar los servomotores.
- Src/asyncServo/AsyncServo.h: permite mover un servo de manera no bloqueante.
- Arduino_LSM6DS3.h: biblioteca correspondiente al acelerómetro del Arduino NANO 33 IoT.
- ArduinoJson.h: está relacionada con la memoria de la placa.

A continuación, se definen los pines desde los que se controlan los servomotores, el servo excitador se controla desde el pin D5 y el servo bloqueador desde el pin D6.

Después es necesario configurar la red Wi-Fi a la que se conectarán los diferentes elementos, ha de ser una red privada, y la configuración se realiza mediante dos variables tipo **char** una para definir el nombre de la red **ssid** que en este caso ha de tener el nombre de la red Wi-Fi del Laboratorio de Estructuras que es donde se han realizado los ensayos. La otra es la variable **pass** que es la contraseña de la red. También se define la IP del *broker* de MQTT, que ha de ser la misma que la IP del dispositivo donde se instala el programa *Mosquitto*.

Las siguientes líneas de código están dedicadas a crear los cinco *topic*s utilizados en este entorno, esto se hace creando tantas variables tipo **char** como *topics* sean necesarios. Y justo después se asigna la asincronía a los dos servos.

También se crean dos variables tipo **bool** una para cada uno de los servos, estas variables pueden ser verdadero o falso, y su función es la de encender o apagar los servos. Se generan las variables de funcionamiento del servo excitador, una es la amplitud de oscilación, y la otra es una variable que impide que el servo no gire más de 1º cada 8 ms, porque si esto ocurre el servomotor MG995 patina y tiene un funcionamiento errático. Y, por último, se crean las instancias de los clientes Wi-Fi y MQTT, y las variables del tiempo para el envío de datos del acelerómetro.

- 1ª función

Es ahora cuando se inicia la primera función, **void servoSetup()**, que sirve para configurar el funcionamiento del servo excitador. En primer lugar, se definen las posiciones a las que tendrá que moverse el servo durante el proceso de excitación. También se definen los valores de la amplitud máximos y mínimos hasta los que puede moverse el servo sin llegar a chocar con el carenado porta servo.

Se crea un bucle para asegurar que el servo recorre como máximo 1º cada 8 ms, y que funciona de forma correcta. Por último, se muestra por pantalla un mensaje que alerta de que se ha cambiado la configuración de este servomotor, y los nuevos valores de operación.

- 2ª función

La segunda función es una función tipo **callback()**,se entra en ella cada vez que se publica un mensaje en uno de los *topic* a los que se está suscrito. Esta función recibe tres mensajes, uno correspondiente al *topic* en el que se ha publicado algo, otro que contiene aquello que se ha publicado, y otro que informa del tamaño que tiene el mensaje publicado.

Dentro de esta función lo que se hace es recoger estos mensajes en tres variables internas del bucle. Después cuenta con cuatro bucles **if**, uno por cada uno de los *topic* relacionados con los servomotores, y cuando el nombre del *topic* recibido coincide con uno de ellos, se entra en estos bucles.

Dos de ellos están relacionados con las variables de operación del servo excitador, en estos dos casos, iguala cada una de estas variables con el valor recibido. Y los otros dos sirven para activar o desactivar cada uno de los dos servomotores.

- 3ª función

Este es el bucle encargado de conectarse a la red Wi-Fi, esto se realiza mediante el comando **WiFi.begin(ssid,pass)**, que devuelve el mensaje 'WL_CONNECTED' cuando la conexión con la red Wi-Fi se ha realizado correctamente.

Cuando la conexión es correcta, el programa muestra el mensaje 'Conectado.', si no, muestra '.' y vuelve a intentar conectarse.

- 4ª función

Esta es la función que se encarga de la conexión con el *broker* MQTT, esto se realiza mediante un bucle interno. Para entrar en él se comprueba si el comando **client.connect()** devuelve el nombre del *topic* principal de aquellos a los que nos hemos suscrito. Los *topic*s están ordenados de manera jerárquica, y hemos de suscribirnos a uno principal que alberga otros más específicos.

Es por esto, que una vez dentro del bucle el programa devuelve que se ha conectado al *broker* de MQTT, y a continuación, se suscribe a los *topic* relacionados con el funcionamiento de los servomotores, lo muestra por pantalla y publica los valores iniciales de funcionamiento del servo excitador.

Si el programa no consigue conectarse al bucle que realiza todas estas funciones significa que la conexión con el *broker* no se ha realizado correctamente, mostrando un mensaje de error y reintentando la conexión después de un tiempo de espera de 5 s.

- 5ª función

Esta es una de las funciones básicas de un programa de Arduino, **void setup()**. En ella lo primero que se realiza es la inicialización del puerto serie a 115200 baudios, para posteriormente llamar dentro de ella a la función encargada de la configuración del Wi-Fi (3ª función).

Después se inicializa la IMU de la placa de Arduino mediante el comando **IMU.begin()**, y el servo bloqueador con el comando **servo.begin()** estableciendo desde qué pin se controla, y cuáles son sus pulsos máximo y mínimo. También se manda a este servo a la posición 45º.

A continuación, se establecen los detalles del servidor con **client.setServer(broker,1883)**, en la que se definen la IP del broker y el puerto al que hay que conectarse. Finalmente llama a la 2ª función mediante un puntero, con el comando **setCallback(callback)** que se activa cuando se publica algo nuevo en los *topics* a los que se ha suscrito.

- 6ª función

El cometido de esta función es el de obtener los datos de la IMU del Arduino, para ello se crean 3 variables internas para este bucle, tipo **float**, una para obtener el dato de la aceleración de cada eje.

A continuación, cuenta con un bucle **if** al que se entra cuando hay nuevos datos de la IMU disponibles, mediante el comando **IMU.accelerationAvailable()**, para después leer los valores de aceleración disponibles, y almacenarlos en las 3 variables anteriormente creadas, esto se realiza con **IMU.readAcceleration(x, y, z).**

Las siguientes 8 líneas de código de este bucle, están destinadas a crear una variable a la que le podemos definir su capacidad, de manera tal que haya sitio suficiente para albergar los datos de la aceleración recogidos para cada eje. Siempre se ha de dejar algo más de espacio que el estrictamente necesario, ya que el Arduino no trabaja bien con memoria variable, y si por algún motivo el tamaño de los datos fuese mayor del estimado, funcionaría peor.

Por último, se publica en el *broker* el mensaje que se ha guardado en esta variable, que es el correspondiente a los valores de la aceleración a los que se encuentra sometida la placa.

- **7ª función**

Esta es la última función del programa, se trata de la función **void loop()** que es otra de las funciones básicas, su característica principal es que se repite de manera continua mientras se está ejecutando el programa.

Lo primero que se realiza en ella es actualizar los datos de los servomotores, mediante **servo.update()** esto es necesario por haber hecho que su movimiento no sea autobloqueante.

Después hay dos bucles para activar o desactivar cada uno de los servomotores. Estos bucles funcionan usando las variables **bool** utilizadas en la 2ª función, si la variable de este tipo referente a uno de los servos es verdadera (*true)* se activa dicho servomotor, mientras que si es falsa (*false*) lo apaga.

El siguiente bucle tiene la finalidad de relanzar la 4ª función, llamada *reconnect,* sólo cuando no haya un usuario conectado al *broker*, para reconectarlo.

La siguiente línea de código es necesaria en los programas MQTT, se trata del comando **client.loop()**, su finalidad es la de permitir que el cliente MQTT procese los mensajes entrantes para enviar los datos de publicación y actualizar la conexión al servidor. Ha de ser llamada regularmente, por eso se encuentra alojada en esta función.

Por último, está el bucle responsable de la llamada a la función de la IMU, 6ª función, y, por tanto, de mostrar los datos y gráficas de la aceleración. Se comienza igualando una variable denominada *currentTIme* (tiempo actual) al comando **millis()**, este devuelve el número de milisegundos transcurridos desde que la placa comenzó a ejecutar el programa, es decir, cada vez que se recorre esta línea de código se actualiza el valor de la variable *currentTime,* teniendo cada vez un valor mayor.

Se cuenta con otra variable también relacionada con el tiempo, llamada *lastTime*, que inicialmente vale 0. Cuando la operación (*currentTime* – *lastTime*) sea mayor que 200 ms, es decir, cuando currentTime>200ms, se ejecuta el bucle mencionado anteriormente lanzando la función de la IMU, de manera que se actualizan los datos y gráficas de la aceleración.

Justo después, se iguala la variable *lastTime* a la función **millis()** que ahora tendrá un valor cercano a los 200 ms, y se sigue ejecutando el resto del código. Como ya se ha comentado anteriormente el valor de *currentTime* se va actualizando continuamente, hasta que se vuelva a cumplir la condición (*currentTime* – *lastTime*)>200, momento en que se ejecutará nuevamente el bucle de la IMU, actualizando el valor de *lastTime*.

Esto lo que genera es que cada 200 ms = 0.2 s se reenvíen los valores y las gráficas de la aceleración a la que se encuentra sometida la estructura en los 3 ejes.

6.2.2. Interfaz

La interfaz creada es una página web sencilla que puede ejecutarse con cualquier navegador y que permite controlar todas las funciones que se pueden realizar sobre la maqueta.

Figura 6.1. Interfaz para controlar la maqueta.

Como puede verse en la Figura 6.1. la interfaz cuenta en su parte superior con una gráfica en la que se muestra la aceleración a la que está sometida la estructura en cada uno de sus ejes. Esta gráfica permite escoger la aceleración de los ejes que se deseen, simplemente clicando en los 3 botones de su parte inferior.

Luego, están los dos botones para habilitar o deshabilitar los servomotores, según la configuración que se ve en la Figura 6.1. el servo excitador (Servo 1) estaría en funcionamiento y el servo deshabilitado (Servo 2) lo que en su caso supone estar bloqueando el TMD.

Por último, se encuentran dos barras en las que poder escoger los valores que se requieran para las variables de funcionamiento del servo excitador, es decir, la amplitud y el periodo de oscilación.

El propio programa está limitado para que siempre se cumpla la condición de que el servo excitador no tenga que recorrer una amplitud de 1º en un tiempo menor de 8 ms. Pueden darse situaciones en las que se pida una amplitud demasiado grande en un periodo muy pequeño, y automáticamente, el programa variará el periodo de oscilación hasta el tiempo mínimo necesario para que se cumpla la condición ya citada.

7. Prototipado

En este punto se detalla el montaje de la maqueta con el que se ha hecho el último ensayo realizado, con la placa de Arduino sobre la mesa, en vez de montada sobre el carenado porta servo.

Se clasifican los elementos utilizados entre aquellos que pueden ser comprados y aquellos que se han montado en el laboratorio a partir de estos elementos comprados.

Cabe destacar que la mayoría de los materiales utilizados para la realización de este proyecto han sido aprovechados a partir de material ya existente en el laboratorio.

Los únicos componentes comprados expresamente para este trabajo son los perfiles de aluminio y los elementos utilizados para unir los perfiles entre sí: tuercas, tornillos y arandelas. También es necesario el cable de conexión entre la placa de Arduino y el ordenador de 3 m de longitud, que no se ha podido llegar a comprar.

7.1. Compras

Los elementos que se han comprado y se utilizan en el TFG sin ser modificados son:

- Servo MG995.
- Servo SG90.
- Arduino NANO 33 IoT.
- Perfiles de aluminio, las cuatro barras que forman parte de la estructura, las tres de la base y la que es el núcleo de la maqueta.
- Los elementos necesarios para sujetar los perfiles de aluminio entre sí, como son las tuercas, tornillos y arandelas.
- Cables de conexión de un hilo.

- Las tuercas que hacen de masa en los péndulos del excitador y del TMD.
- El imán de neodimio del amortiguador.
- Todas las tuercas y arandelas necesarias para fijar las posiciones de diferentes componentes dentro de los elementos.
- La fuente de alimentación y el adaptador, para alimentar los servos.

Luego, los elementos que pueden comprarse y que se han modificado o que han formado parte de alguno de los componentes del trabajo son:

- El cable de seis hilos de 3 m de longitud que lleva la alimentación a los servos.
- Las barras roscadas que forman el péndulo del excitador y del TMD.
- La barra de acero y la goma Eva del bloqueador del TMD.
- Las bobinas de PLA que se utilizan para hacer los carenados.
- El cable de conexión entre el Arduino y el ordenador, de 0.3 m.
- Los dos rodamientos centrales de dos *fidget spinner.*
- La placa de aluminio del amortiguador.

7.2. Taller

Con algunos de estos elementos comerciales se ha realizado la construcción de una serie de componentes necesarios para poder construir la maqueta y que esta sea funcional. A continuación, se van a enumerar cuáles son estos elementos y cuáles sus componentes:

- TMD.
    - La barra horizontal del propio TMD, que está soldada a una tuerca en uno de sus extremos también es un elemento que ha de ser construido.
    - La tuerca que actúa como retén de esta barra.
    - Los dos rodamientos de los *fidget spinnet*, y las tuercas y arandelas que los fijan.
    - La barra roscada del péndulo, la vertical.

    - El imán de neodimio, y las tuercas y arandelas necesarias para su posicionamiento.
    - Las tuercas que hacen de masa, el trozo de revestimiento de cables que impide su movimiento, para que no se descentren y las tuercas y arandelas que ayudan a fijarlas.
- Péndulo del excitador.
    - La barra vertical roscada.
    - Las tuercas que hacen la función de masa, y las tuercas y arandelas que las fijan en el extremo de esta.
- Brazo bloqueador.
    - La barra de acero.
    - El trozo de Goma Eva colocado en el extremo del brazo.
- Carenados, el genérico y los dos especiales.
    - Hechos a partir de las bobinas de PLA.

7.3. Montaje maqueta experimentación

Este apartado se va a centrar en explicar dónde, cómo y cuáles han sido los elementos que forman parte en los ensayos de esta maqueta. Al igual que en apartados anteriores se van a explicar dos ensayos diferentes, uno el que se ha llegado a realizar, y otro, aquel que se hubiera realizado en condiciones normales.

En ambos casos, se parte de la estructura desnuda, formada por las tres barras de la base y el perfil principal. Esta estructura básica está lastrada mediante dos contrapesos para mantenerla lo más fija posible. Se irán colocando sobre núcleo los diferentes elementos necesarios en el ensayo.

Se pretendía realizar un montaje en el que al menos debería haber montados los dos carenados especiales, y a mayores de ellos se le podrían añadir tantos carenados genéricos como cupiesen en la estructura. Los dos carenados singulares deberían ir colocados a una altura que cuanto mayor sea, mejor se podrá apreciar el efecto de resonancia que se busca en esta experimentación.

En el carenado porta servo van inserto el servomotor encargado de la excitación, y la placa de Arduino que controla los servomotores y mide las aceleraciones de la estructura. En el carenado porta TMD se inserta el TMD, la placa de aluminio del amortiguador y el servo bloqueador.

Este ensayo estaría controlado mediante el entorno que nos han cedido. El Arduino deberá estar conectado al ordenador para que pueda estar alimentado mediante un cable de conexión USB 2.0.-micro USB de 3 m de longitud.

En cuanto a la alimentación de los servos se ha detallado en el apartado 4, pero para ella sería necesaria una fuente de alimentación, un adaptador y un cable de 3 m de longitud para hacer llegar la corriente a los servos. El control de estos servomotores se haría mediante cables de conexión entre ellos y el Arduino.

En la prueba que llegamos a realizar se colocaron un total de 10 carenados, de los cuales cuatro son carenados genéricos, dos de la versión definitiva, el otro de la 3ª versión impresa de este carenado, la que cuenta con unos rigidizadores que ocupan toda la altura del módulo, y el último de la 4ª, la que no dispone de rigidizadores.

También se cuenta con dos carenados porta servo, uno de ellos de la última versión que se ha llegado a imprimir, y el otro de la versión anterior, la 2ª que se imprimió de este tipo de módulo.

El resto de los carenados que se han montado en este ensayo son carenados porta TMD, cuatro en total. Al igual que en el caso anterior, uno de ellos es de la última versión impresa, otros dos son de la 2ª versión, y los otro de la 3ª versión.

Se han utilizado versiones anteriores de los distintos tipos de carenados para aprovechar que ya se habían impreso y no desecharlos, así hay mayor cantidad de la maqueta tapada y se les da un uso a estos carenados. Pero dentro de los carenados especiales, sólo realizan la función que tienen encomendada las últimas versiones de estos dos módulos.

En esta prueba, el carenado porta servo alberga el servo excitador que sujeta al péndulo, y el carenado porta TMD sostiene al TMD, al servo bloqueador y a la placa de aluminio que forma parte del amortiguador. La otra parte del amortiguador es el imán de neodimio que está ubicado en la barra vertical del propio TMD.

Se toma este ensayo como referencia de aquellos que se han llegado a realizar, puesto que es el más completo de los que se ha realizado, y en el que mejor se puede apreciar la influencia del TMD sobre la estructura.

Se tiene libertad a la hora de posicionar los carenados sobre la barra central de la estructura, y resulta interesante el hecho de modificar la altura de los carenados especiales y ver la influencia que tiene esta variación sobre la estructura. En este ensayo el carenado porta servo, se encuentra sobre los 2 m de altura, y el carenado porta TMD aproximadamente está 1 m por encima de la base de la estructura.

Aunque para que el efecto de resonancia que se busca sea más acusado, es adecuado colocar los carenados especiales lo más alto posible. Pero esto dificultaría la manipulación, la altura idónea sería alrededor de los 2 m.

La alimentación de los servos se hace mediante el uso de una protoplaca para tener más pines desde los que llevarles la energía. Para ello se usa un cable de seis hilos, que se disimula en la estructura aprovechando uno de los huecos del perfil principal de la maqueta.

Los aspectos que faltaban por mejorar en esta prueba fue que no se tenía en ese momento la capacidad de montar el Arduino sobre la estructura, por lo que no se han podido obtener medidas de las aceleraciones que sufre la estructura, que la parte del brazo del servo bloqueador que frena el TMD no está hecha de Goma Eva como en la versión definitiva, sino que es una versión anterior hecha de espuma.

Y que el control de los elementos se realizaba mediante dos placas de Arduino, una para cada servo, y se usaban los programas desarrollados por nosotros. Una de ellas es la placa Arduino NANO 33 IoT y otra una placa Arduino UNO. En este sentido es de destacar que se han realizado ensayos en los que se ha utilizado el entorno creado por los colaboradores de la Universidad de Oviedo, que puede controlar todos los elementos necesarios con un solo Arduino, pero no se dispone de documentos gráficos de ellas, por lo que se ha preferido tomar esta como referencia.

8. Manual de uso

En este punto se va a detallar como se realiza todo el proceso de experimentación en el prototipo. La intención que se busca en todo este procedimiento es la de mostrar de manera visual la función que realiza el TMD en la estructura.

Es un proceso que se divide en 3 fases, en cada una de ellas se excita la estructura, y podrá verse la incidencia real del TMD en función del estado en que se encuentre, bloqueado o desbloqueado.

Una vez se tiene la maqueta montada, con sus elementos colocados sobre ella, conectados de forma adecuada, y los programas ejecutados, se puede comenzar a realizar el ensayo.

- **1ª fase**

Se comienza con todos los elementos bloqueados. Pinchando en el comando de "Control del Servo 1" se activa el servo excitador y comienza a oscilar el prototipo. Si esta oscilación no es suficiente porque no esté sintonizada con la estructura, y no llega al estado de excitación deseado, se para la excitación y se calcula el periodo de oscilación de la maqueta.

Esto se realiza de forma experimental, haciendo oscilar la maqueta de manera manual, se le desplaza hacia un lado y se suelta, justo en ese momento se comienza a contar el tiempo que tarda la estructura en realizar un número de oscilaciones completas. Normalmente se ha hecho con diez.

Ahora, dividiendo el tiempo que tarda en realizar estos ciclos, entre el número de ciclos, se obtiene el periodo de oscilación natural de la estructura, que es el inverso de su frecuencia de resonancia. Este no es un método exacto, pero si que permite calcular un valor cercano, por lo que es una solución útil, y muy sencilla de ejecutar.

Una vez conocido este valor, se ajusta el periodo de oscilación tanto del *shaker* como del TMD para que estén sintonizadas con la estructura, y que el fenómeno que han de generar en el ensayo, respectivamente, sea efectivo.

Volviendo a activar el servo 1 o excitador, se debe llegar a un estado de oscilación suficiente porque el prototipo entra en resonancia con la fuerza de excitación. Este ha de llegar a un punto en el que su amplitud de oscilación ya no crezca más, es decir, que sea máximo para la excitación que se le está realizando.

Es ese el estado de oscilación que se estaba buscando, y es entonces cuando, se para el servomotor excitador para no seguir estimulando el movimiento de la estructura, y se le deja oscilar libremente.

La amplitud del movimiento oscilatorio de la maqueta va a ir decreciendo según vaya pasando el tiempo, pues el sistema va a ir perdiendo energía, hasta que transcurrido un tiempo se acabará parando. Y se podrá dar por concluida la primera fase.

- 2ª fase

Una vez se encuentra parada la estructura por completo, se puede comenzar la segunda fase del ensayo. Esta también ha de comenzarse, activando el servo excitador, clicando en el comando del programa "Control del Servo 1", para hacer oscilar la estructura de nuevo.

Una vez se llega al estado de excitación deseado, se repite la misma operación que en la fase anterior, se para el servo excitador. Pero en esta ocasión, en vez de dejar a la maqueta se pare por sí misma, justo después de parar el *shaker*, volviendo a clicar en el mismo comando que para activarlo. Se ha de mover el servo bloqueador para desbloquear el TMD, esto se realiza clicando en el comando "Control Servo 2" del programa.

De esta manera, el TMD ya tendrá el movimiento permitido, y realizará junto con el amortiguador, una atenuación de la oscilación de la maqueta, que la hará pararse por completo de manera mucho más rápida que en la primera fase.

- 3ª fase

En esta tercera y última fase, también se comienza accionando el servo excitador, pero en esta ocasión, a diferencia de las anteriores, el TMD estará desbloqueado.

Este elemento contrarresta el efecto del excitador, no permitiendo que la estructura llegue a un estado de excitación como los alcanzados en las fases anteriores. Y aunque si tiene un pequeño balanceo, la amplitud de este es mucho menor que la que se tenía al excitar la estructura con el TMD bloqueado.

Esto puede mostrarse también observando las gráficas de las aceleraciones a las que se encuentra sometida la estructura en cada una de las distintas fases. El diagrama de estas aceleraciones puede observarse en la interfaz del propio programa Mosquitto.

De manera que teniendo en cuenta todo el proceso:

- En la primera fase se observarían unas aceleraciones de gran magnitud, y que una vez se deja de excitar, el valor de la aceleración va reduciéndose lentamente.
- En la segunda fase se observaría una aceleración que llega a una magnitud igual que la de la primera fase, pero posteriormente, una vez se deja de excitar y se activa el TMD, ha de verse como el valor de la aceleración disminuye de manera muy rápida, gracias a este elemento.
- La última fase es el culmen de la experimentación, es la muestra perfecta del efecto que realiza el TMD sobre la estructura, no dejando que alcance unos valores de aceleración como los que se obtenían en las dos primeras fases.

Esto es extrapolable a lo que sucede en los rascacielos reales con los TMD de los que disponen, mostrando de una manera sencilla y visual, el impacto que estos elementos tienen sobre las estructuras sobre las que están montados. Y haciendo que sea viable construirlos en lugares que a priori no son adecuados para este tipo de edificios, como el caso ya comentado del Taipei 101.

9. Conclusiones y líneas futuras

9.1. Conclusiones

A la vista de lo expuesto en esta memoria, se puede concluir, que se cumple con todos los objetivos planteados al comienzo del trabajo.

Se ha logrado fabricar el prototipo de un rascacielos, sobre el que se ha llevado a cabo la experimentación deseada. Consiguiendo una excitación suficiente y demostrando la efectividad de los mecanismos de control de la oscilación. Siendo este proceso controlado desde una aplicación de manera telemática.

Para poder obtener este resultado ha sido necesaria la realización de una serie de trabajos:

- Se han diseñado y fabricado, tres tipos diferentes de carenados, obteniendo unos resultados que satisfacen los requisitos pedidos para cada uno de ellos.
- Se han desarrollado una serie de programas en Arduino con los que controlar la experimentación.
- Se han realizado las tareas necesarias para la sintonización y puesta a punto del excitador y del TMD con la estructura. Consiguiendo que tengan el efecto deseado en ambos casos.
- Se han fabricado todos los elementos que forman parte del ensayo, TMD pendular, péndulo excitador y bloqueador. Así como, se han realizado todas las conexiones tanto eléctricas como electrónicas necesarias para alimentar los servomotores y el Arduino, conectarlos entre sí, y la conexión del Arduino con el ordenador.

La realización de estas labores ha implicado la mejora o la adquisición de ciertas competencias.

- Utilización de CATIA V5 en el diseño de los carenados, el modulado de todos los elementos que forman parte del prototipo, y la realización de los planos de las piezas y de los conjuntos. Mejorando las capacidades en el uso de este programa.
- Adquisición de nuevos conocimientos de programación de placas de Arduino.
- Realización de trabajos de taller, tanto manuales en el montaje de la estructura y los diferentes elementos, como en el uso de herramientas tales como sierras manuales, sierras de disco o pistolas termofusibles.
- Realización trabajos de electricidad y electrónica, como son el montaje de los circuitos eléctricos o la soldadura de cables mediante soldadura de estaño.

9.2. Líneas futuras

En cuanto a posibles líneas futuras para poder continuar con este TFG, al tratarse de un trabajo en el que se trabajan varias disciplinas diferentes, hay varios posibles flancos de mejora.

- Automatizar el proceso de experimentación, es decir, este proceso se ha realizado de manera manual, y sería conveniente poder realizarlo de una forma más automática. Para ello habría que realizar una serie de cambios en los elementos que componen el prototipo.

En primer lugar, habría que variar el programa para que pueda calcular la frecuencia de resonancia de la estructura, esto puede hacerse accionando el *shaker* para excitar la maqueta, cuando la oscilación sea grande, se para el excitador. Sin accionar el TMD, ya que quiere medirse la frecuencia natural de la maqueta, la placa de Arduino debería comenzar a contar el tiempo que tarda el modelo en realizar un determinado núcleo de ciclos.

Dividiendo el tiempo entre el número de ciclos, se obtendría el periodo natural de la estructura. La frecuencia de resonancia será el inverso de este periodo. Es evidente que este no es un proceso preciso y que no tiene porqué obtenerse exactamente la frecuencia de resonancia, pero sí un valor bastante aproximado, y que se realiza de manera sencilla.

La siguiente modificación que debería realizarse sobre lo hecho en este trabajo, sería que una vez obtenida esta frecuencia de resonancia, el *shaker* y el TMD se sintonizarán automáticamente con ella. En el caso del excitador esto es bastante sencillo, ya que simplemente sería asignar el periodo de oscilación de la maqueta al periodo de oscilación del servomotor de este elemento.

En el caso del TMD sería más complejo pues la forma de modificar su periodo de oscilación es variando la longitud de su péndulo. Una opción para resolver esto podría ser contar con otro servomotor que controlase la distancia a la que se sitúe la masa de este elemento, de manera que su periodo de oscilación sea el mismo que el de la maqueta. Esto conllevaría también cambios en el programa de Arduino y en el carenado porta TMD.

- Resolver la problemática existente con los choques del *shaker* y el carenado. Para ello habría que hacer un péndulo excitador más corto como ya se ha probado en este trabajo, pero con él no se consigue inercia suficiente para hacer oscilar la maqueta, esto se solucionaría consiguiendo un servo que tenga mayor recorrido, pudiendo girar hasta 360º, para compensar la menor inercia del péndulo.
- Hacer un TMD multidireccional similar al del Taipei 101, en vez del TMD unidireccional utilizado en este trabajo.
- Introducir la posibilidad de activar o desactivar el amortiguamiento, esto se realizaría haciendo girar el imán del amortiguador 90º o 180º, para que no estuviera de frente a la placa de aluminio. Dicho giro debería estar controlado por otro servomotor para poderlo realizar automáticamente.

9.3. Consideraciones adicionales

Como colofón de este trabajo se va a hacer una serie de consideraciones sobre las repercusiones sociales, medioambientales, de seguridad y económicas asociadas a este TFG.

En el aspecto social, este TFG supone una manera sencilla y didáctica de demostrar la influencia que tienen estos sistemas de control de la oscilación en las estructuras, pudiendo extrapolarse a casos de edificios reales como lo comentado con el TMD del Taipei 101. Está experimentación por su sencillez permite ser expuesta y comprendida por personas no expertas en el ámbito ingenieril.

El impacto medioambiental de este trabajo es mínimo, ya que la mayor parte de los materiales utilizados en la construcción de este prototipo y todos sus componentes, han sido reutilizados de aquellos disponibles en el Laboratorio de Estructuras. El mayor impacto que puede haberse ocasionado es en cuanto a la impresión de múltiples versiones de los carenados, pero para no desaprovecharlas una vez impresas, se han montado sobre la estructura haciendo simplemente funciones de revestimiento. Incluso se han aprovechado materiales desechados de otros proyectos.

Algunos de estos elementos tales como los servomotores, la placa de Arduino o los propios perfiles de aluminio pueden volver a ser utilizados por otros estudiantes en el futuro para la realización de sus TFGs, puesto que tras la realización de este trabajo todos se encuentran en perfectas condiciones de funcionamiento.

En cuanto a la seguridad, en la realización de este TFG, se han utilizado algunas herramientas peligrosas, como distintos tipos de sierras, pero todos los trabajos llevados a cabo con ellas se han hecho cumpliendo con todas las medidas de seguridad pertinentes.

Por último, se hará una consideración del valor económico que ha tenido la consecución de este TFG, teniendo en cuenta el coste de los materiales utilizados, aunque no hayan sido comprados expresamente para este trabajo y el número de horas empleadas en él.

En cuanto a los materiales, cabe destacar que algunos de ellos como las barras roscadas del TMD o el excitador, no se contabilizan ya que suponen un gasto ínfimo, y de difícil cuantificación al ser partes de un todo mayor.

Tabla 9.1. Tabla de costes materiales.

Material/Componente	Coste (€)
Perfiles aluminio	68
Escuadras y tornillos	17
Arduino NANO 33 IoT	24
Servo MG995	15
Servo SG90	2
Total	**126€**

Para el coste de las horas empleadas en él, se considerará el coste de un ingeniero en formación, que está en torno a 20.000 €/año, por un tiempo de trabajo de aproximadamente 1.800 h/año, lo que resulta en 11.2 €/hora. Este cálculo estará desglosado en tres tablas, una de la fabricación del prototipo, otra del desarrollo de las distintas fases del trabajo y la redacción de los documentos:

Tabla 9.2. Tabla de costes de la fabricación del prototipo.

Fabricación prototipo	Horas	Coste hora (€/h)	Coste total (€)
Montaje perfiles	1	11,2	11,2
Impresión carenado genérico*	5	11,2	56
Impresión porta servo*	5	11,2	56
Impresión porta TMD*	5	11,2	56
Fabricación eje TMD	1	11,2	11,2
Montaje TMD	1	11,2	11,2
Montaje péndulo excitador	1	11,2	11,2
Montaje brazo bloqueador	1	11,2	11,2
Montaje carenados especiales	1	11,2	11,2
Total	**21h**	-	**235.2€**

* No se refiere al número de horas de trabajo de la máquina sino a la adecuación de los carenados para su uso en diferentes ensayos.

Tabla 9.3. Tabla de costes de las diferentes fases de desarrollo del proyecto.

Fases de desarrollo y programación (sin montaje componentes)	Horas	Coste hora (€/h)	Coste total (€)
Predimensionado maqueta	4	11,2	44,8
Modelado y diseño componentes	65	11,2	728
Desarrollo programación Arduino	65	11,2	728
Conexiones eléctricas	2	11,2	22,4
Puesta a punto prototipo	15	11,2	168
Elaboración planos	5	11,2	56
Total	**156h**	-	**1.747,2€**

Tabla 9.4. Tabla de costes de la fase documental del proyecto.

Fase documental	Horas	Coste hora (€/h)	Coste total (€)
Memoria	150	11,2	1680
Documentos gráficos (Elaboración planos, capturas)	10	11,2	112
Presentación	10	11,2	112
Total	**170h**	-	**1904€**

$$Total\ TFG = 126€ + 235{,}2€ + 1.747{,}2€ + 1904€ = 4012{,}4€$$

Bibliografía

[1] Kirikov, B.A. *History of earthquake resistant construction: from antiquity to our times.* 1ª Ed. (1992).

[2] DSpace@MIT. *Structural systems and tuned mass dampers of super-tall building: case study Taipei 101.* Recuperado desde: https://dspace.mit.edu/handle/1721.1/38947 [consulta: 5 abril 2020].

[3] Structuralia. (2018). *Home Insurance Building: la historia del primer rascacielos.* Recuperado desde: https://blog.structuralia.com/home-insurance-building-la-historia-del-primer-rascacielos [consulta: 6 abril 2020].

[4] Casado Sánchez, C.M. (2019). *Control de vibraciones en estructuras flexibles mediante amortiguadores de masa pasivos, adaptativos semiactivos y activos* (Tesis doctoral). Recuperado desde http://uvadoc.uva.es/handle/10324/40259.

[5] Barrero Gil, A; Alonso Rodrigo, G; Meseguer Ruiz, J; Astiz Suárez, M.A. (2007). *Ensayos en túnel de viento de un modelo aeroelástico del arco del puente sobre el río Tajo "Arcos de Alconétar".* Recuperado desde: http://www.hormigonyacero.com/index.php/ache/issue/view/285/241. [consulta: 8 mayo 2020].

[6] *Calle de vórtices de von Kármán.* Recuperado desde: https://es.wikipedia.org/wiki/Calle_de_v%C3%B3rtices_de_Von_K%C3%A1rm%C3%A1n. [consulta: 8 mayo 2020].

[7] Ucke, C. Schlichting, H-J. (2008). Oscillating Dolls and Skycrapers. *Physik in unserer Zeit. 39.* P. 139-141. Recuperado desde: https://pdfs.semanticscholar.org/ab30/3cc6ee3f5c34fbb8c52251cb20d314221cf9.pdf. [consulta: 11 abril 2020].

[8] Structuralia. (2016). *El impresionante rascacielos Taipei 101.* Recuperada desde: https://blog.structuralia.com/el-impresionante-rascacielos-taipei-101. [consulta: 10 abril 2020].

[9] Tuan, A.Y; Shang, G.Q. (2014). Vibration Control in a 101-Storey Building Using a Tuned Mass Damper. *Journal of Applied Science and Engineering, Vol 17*(No. 2). Recuperado desde: http://www2.tku.edu.tw/~tkjse/17-2/05-CE10302_1185.pdf.

[10] *Tuned mass damper of Taipei 101.* Recuperado desde: https://www.atlasobscura.com/places/tuned-mass-damper-of-taipei-101. [consulta: 10 abril 2020].

[11] Garzón Escudero, J. (2018). *Instrumentación de una maqueta de un edificio de dos plantas para su uso como demostrador de sistemas de control de vibraciones* (Trabajo de Fin de Grado). Recuperado desde: https://uvadoc.uva.es/handle/10324/31108.

[12] *Arduino – Home.* Recuperado desde: https://www.arduino.cc/. [consulta: 24 julio 2020]

[13] *MG995.* Recuperado desde: https://www.electronicoscaldas.com/datasheet/MG995_Tower-Pro.pdf. [consulta: 2 febrero 2020].

[14] *SERVO MOTOR SG90.* Recuperado desde: http://www.ee.ic.ac.uk/pcheung/teaching/DE1_EE/stores/sg90_datasheet.pdf. [consulta: 2 febrero 2020].

[15] *Componentes para mobiliario y equipamientos industriales.* V.17. Recuperado desde: https://www.antipoda.eu/pdf/fasten.pdf. [consulta: 30 enero 2020]

[16] Herráez. M. *Tema 9: Vibraciones de sistemas de 1 grado de libertad.* Máquinas y mecanismos.

[17] *PLA filamento 1.75mm.* Recuperado desde: https://www.bq.com/es/pla. [consulta: 28 mayo 2020].

[18] *¿Qué es y cómo funciona un servomotor?* Recuperado desde: http://panamahitek.com/que-es-y-como-funciona-un-servomotor/. [consulta: 2 febrero 2020]

[19] *Servomotor ¿Qué es y cómo funciona?* Recuperado desde: https://www.ingmecafenix.com/electricidad-industrial/servomotor/. [consulta: 2 febrero 2020]

[20] *Momento of Inertia.* Recuperado desde: http://hyperphysics.phy-astr.gsu.edu/hbasees/mi2.html. [consulta: 12 mayo 2020].

[21] *Los "Fidget Spinner" y el porqué de su éxito.* Recuperado desde: https://www.bestteacher-formacion.com/post/2017/05/18/los-fidget-spinner-y-el-porqu%C3%A9-de-su-%C3%A9xito. [consulta: 19 mayo 2020].

[22] *Corrientes de Foucault.* Recuperado desde: http://www.sc.ehu.es/sbweb/fisica/elecmagnet/induccion/foucault/foucault.htm. [consulta: 29 mayo 2020].

[23] *ARDUINO UNO REV 3.* Recuperado desde: https://store.arduino.cc/arduino-uno-rev3. [consulta: 18 marzo 2020]

[24] *ARDUINO NANO 33 IOT.* Recuperado desde: https://store.arduino.cc/arduino-nano-33-iot. [consulta: 20 mayo 2020].

[25] *¿QUÉ ES UNA PROTOBOARD?*. Recuperado desde: https://blog.330ohms.com/2016/03/02/protoboards/. [consulta: 21 mayo 2020].

[26] *Serial.* Recuperado desde: https://www.arduino.cc/reference/en/language/functions/communication/serial/. [consulta: 27 mayo 2020]

[27] *COMUNICACIÓN DE ARDUINO CON PUERTO SERIE.* Recuperado desde: https://www.luisllamas.es/arduino-puerto-serie/. [consulta: 27 mayo 2020].

[28] *Aprendiendo Arduino.* Recuperado desde: https://aprendiendoarduino.wordpress.com/2016/07/02/comunicacion-serie-arduino/. [consulta: 28 mayo 2020].

[29] *MQTT.* Recuperado desde: http://mqtt.org/. [consulta: 28 mayo 2020].

[30] *¿QUÉ ES MQTT? SU IMPORTANCIA COMO PROTOCOLO IOT.* Recuperado desde: https://www.luisllamas.es/que-es-mqtt-su-importancia-como-protocolo-iot/. [consulta: 28 mayo 2020]

Anexo I – Programa Arduino desarrollo propio

En este primer anexo, se van a mostrar 2 de los 3 programas desarrollados en Arduino para la realización del ensayo sobre el prototipo. El tercero no se incluye ya que se trata de un ejemplo que se incluye al descargarse la librería de control de la IMU del Arduino NANO 33 IoT, LSM6DS3.

- Programa control excitador

Este programa es el encargado de hacer oscilar al servomotor excitador, así como de controlar sus variables de operación.

```
#include <Servo.h>

Servo servo1;

const int pinservo=2;
const int pulsoMin=500;
const int pulsoMax=2500;
char Modo = '0';


float Tciclo=685;
float amplitud=20;

 void setup() {
  Serial.begin(9600);
   servo1.attach(pinservo,pulsoMin,pulsoMax);
   delay(3000);
  }
void loop() {
  servo1.write(90-amplitud);
  delay(Tciclo/2);
  servo1.write(90+amplitud);
  delay(Tciclo/2);


 //El siguiente bucle saca por pantalla el nuevo valor para la amplitud
  if (Modo == 'a'){
    Serial.print("La amplitud ha cambiado a: ");
    Serial.println(amplitud);
    }
 //El siguiente bucle saca por pantalla el nuevo valor del tiempo de ciclo
  else if (Modo == 't'){
      Serial.print("El tiempo de ciclo ha cambiado a: ");
      Serial.println(Tciclo);
    }
 //El siguiente bucle saca por pantalla que hemos elegido el modo de centrado del arduino
  else if (Modo == 'c'){
    Serial.println("Centrado");
    }
    Modo = 0;      //Inicializamos la variable del switch
}
```

```
void serialEvent(){
  Modo = Serial.read();
  Serial.print(Modo);
  switch(Modo){
    case 'a':
    amplitud = Serial.parseInt();
    Serial.println(Modo);
    break;

    case 't':
    Tciclo = Serial.parseInt();
    Serial.println(Modo);
    break;

    case 'c':
    servo1.write(90);
    Serial.println(Modo);
    delay(5000);
    break;
    }
  }
```

- **Programa control bloqueador**

Este programa se encarga de controlar el servo bloqueador, pudiendo bloquear o desbloquear el TMD.

```
#include <Servo.h>

Servo servo1;

const int pinservo=2;
const int pulsoMin=500;
const int pulsoMax=2500;
char Modo = '0';

 void setup() {
  Serial.begin(9600);
   servo1.attach(pinservo,pulsoMin,pulsoMax);
   servo1.write(20);
   delay(3000);
  }

 void loop() {
   if (Modo == 'b'){    //Bucle para sacar por pantalla el nuevo valor para la amplitud
     Serial.println("Bloqueo del TMD");
     }
   else if (Modo == 'd'){   //Bucle para sacar por pantalla el nuevo valor del tiempo de ciclo
       Serial.println("Desbloqueo del TMD");
     }
     Modo = 0;      //Inicializamos la variable del switch
 }

   void serialEvent(){
     Modo = Serial.read();
     switch(Modo){
       case 'b':
       servo1.write(10);
       break;

       case 'd':
      servo1.write(30);
       break;
       }
     }
```

Anexo II – Programa Arduino cedido

En este anexo se va a mostrar el código desarrollado en Arduino por los colaboradores de la Universidad de Oviedo, que realiza por las 3 funciones principales requeridas en este TFG.

```
//#include <SPI.h>
#include <WiFiNINA.h>
#include <PubSubClient.h>
#include <Servo.h>
#include "src/asyncServo/AsyncServo.h"
#include <Arduino_LSM6DS3.h>
#include <ArduinoJson.h>

#define PIN_SERVO_1 5
#define PIN_SERVO_2 6

//******** CONFIGURAR ESTO **************
//Config de red
char ssid[] = "StrucWiFi";
char pass[] = "8438389158";
//MQTT
char broker[] = "192.168.0.151";
//**************************************
char brokerUser[] = "";
char brokerPass[] = "";
//topic de salida de los datos del sensor
char ImuOutTopic[] = "/demostrador_up/IMU";
//topic de control y estado del servo1
char servoTopic[] = "/demostrador_up/servo";
//topic y estado del servo2
char servo2Topic[] = "/demostrador_up/servo2";
//topic  de la apertura del servo1
char servoDegreesTopic[] = "/demostrador_up/servoDegrees";
//topic del tiempo en ms de apertura de los grados del servo
char servoMsTopic[] = "/demostrador_up/servoMs";


AsyncServoClass servo1;
AsyncServoClass servo2;

bool enableServo = false;
bool enableServo2 = false;

int servoDegrees = 45;
int servoDelay = 8;

//Clientes wifi y Mqtt
WiFiClient espClient;
PubSubClient client(espClient);

//tiempos para envío de gráfico
long currentTime, lastTime;
```

```
    //configuración de servo1 al recibir mensajes de mqtt
    void servoSetup(int setDegrees, int setMS) {
      int minDegree = 90-setDegrees;
      int maxDegree = 90+setDegrees;

      minDegree = 93-(setDegrees/2);
      maxDegree = 93+(setDegrees/2);

      //por seguridad no permitimos que pase de estos valores (choca con la estructura si los pasa)
      if (minDegree < 78 || maxDegree > 120) {
        minDegree = 80;
        maxDegree = 120;
      }

      int ms = (setMS / setDegrees) / 2;
      //el servo necesita no menos de 8ms por cada ° de apertura, si no patinará y se terminará abriendo más o menos de lo debido
      if (ms < 8) {
        ms = 8;
      }

  //actualiza los parámetros del servo en el pin5
  servo1.begin(PIN_SERVO_1, 1, ms);
  servo1.add(minDegree);
  servo1.add(maxDegree);

  Serial.println("Config de servo1 modificada: ");
  Serial.println((String)"° Minimos: " + minDegree);
  Serial.println((String)"° MAximos: " + maxDegree);
  Serial.println((String)"setMS: " +setMS);
  Serial.println((String)"ms por cada °: " + ms);

}

//callback de mqtt, se entra en esta función cada vez que se publica un mensaje de un topico al que esta suscrito
void callback(char* topic, byte* payload, unsigned int length) {

  String receivedMessage = "";
  String topicName = String(topic);

  Serial.println((String)"Mensaje en el topic: " +topicName);

  //Convert the payload to String
  for (int i=0;i<length;i++) {
    receivedMessage += (char)payload[i];
  }
  receivedMessage = receivedMessage+"\0";

  if (topicName == servoDegreesTopic) {
    servoDegrees = receivedMessage.toInt();
    Serial.println((String)"Actualizados ° a: " + servoDegrees);
    servoSetup(servoDegrees, servoDelay);
  }

  if (topicName == servoMsTopic) {
    servoDelay = receivedMessage.toInt();
    Serial.println((String)"Actualizados ms a: " + servoDelay);
    servoSetup(servoDegrees, servoDelay);
  }
```

```
  if (topicName == servoTopic) {
    if (receivedMessage.compareTo("true") == 0) {
      enableServo = true;
    } else {
      enableServo = false;
    }
    Serial.println((String)"Actualizado estado del servo1: " + enableServo);
  }

    if (topicName == servo2Topic) {
    if (receivedMessage.compareTo("true") == 0) {
      enableServo2 = true;
    } else {
      enableServo2 = false;
    }
    Serial.println((String)"Actualizado estado del servo2: " + enableServo2);
  }
}

void setupWifi() {
  delay(100);
  Serial.print((String)"Conectando a red wifi: " + ssid);
  WiFi.begin(ssid, pass);
  delay(100);

  while(WiFi.status() != WL_CONNECTED){
    delay(500);
    Serial.print(".");
  }
  Serial.println("Connectado.");
}


    void reconnect() {
      while(!client.connected()){
        Serial.println((String)"Conectando a Broker MQTT: " + broker);
        //es importante que esta id no se reutilice en otro proyecto con mqtt, provoca conflictos
        if(client.connect("demoOsciladorUpint"), true){
          Serial.println("Conectado a MQTT.");
          client.subscribe(servoTopic);
          client.subscribe(servo2Topic);
          client.subscribe(servoDegreesTopic);
          client.subscribe(servoMsTopic);
          Serial.println("Suscrito a los siguientes topics: ");
          Serial.println(servoTopic);
          Serial.println(servo2Topic);
          Serial.println(servoDegreesTopic);
          Serial.println(servoMsTopic);
          //Al conectar al topic sobreescribo los datos
          //client.publish(servoTopic, "false", false);
          //client.publish(servo2Topic, "false", false);
          client.publish(servoDegreesTopic, "45", false);
          client.publish(servoMsTopic, "720", false);
        } else {
          Serial.println("Error al conectar al Broker MQTT, reconectando.");
          delay(5000);
        }
      }
    }
```

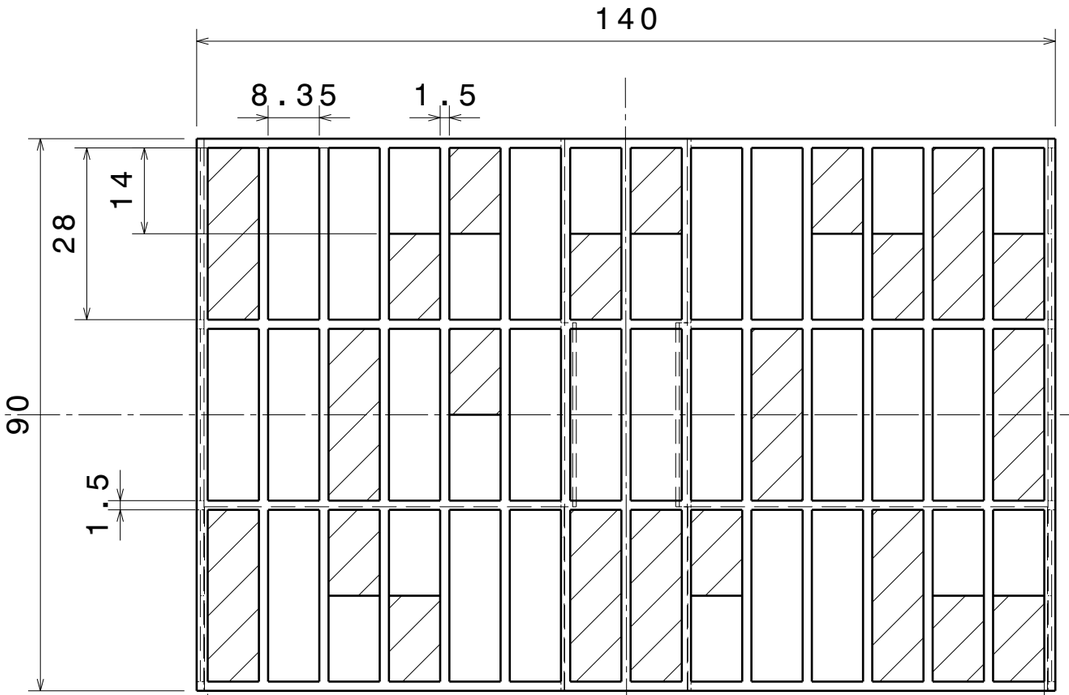
```
void setup() {
   /*while (!Serial) {
    ;
  }*/
  Serial.begin(115200);
  setupWifi();
  IMU.begin();
  servo2.begin(PIN_SERVO_2, 1, 16);
  servo2.add(45);
  client.setServer(broker, 1883);
  client.setCallback(callback);
}

/*
 * IMU Loop to get sensor data.
 */


void IMULoop() {
  float x, y, z;
  if (IMU.accelerationAvailable()) {
    IMU.readAcceleration(x, y, z);
   }

  const size_t capacity = JSON_OBJECT_SIZE(4);
  DynamicJsonDocument doc(capacity);
  doc["sensor"] = "IMU";
  doc["x"] = x;
  doc["y"] = y;
  doc["z"] = z;

  char buffer[512];
  size_t n = serializeJson(doc, buffer);
  client.publish(ImuOutTopic, buffer, n);

}


void loop() {
  servo1.update();
  servo2.update();

  if (enableServo) {
    servo1.loop();
  } else {
    servo1.stop();
  }

  if (enableServo2) {
    servo2.goTo(65);
  } else {
    servo2.goTo(105);
  }

  if (!client.connected()) {
    reconnect();
  }
  client.loop();

  currentTime = millis();
  if(currentTime - lastTime > 200) {
    IMULoop();
    lastTime = millis();
  }
}
```

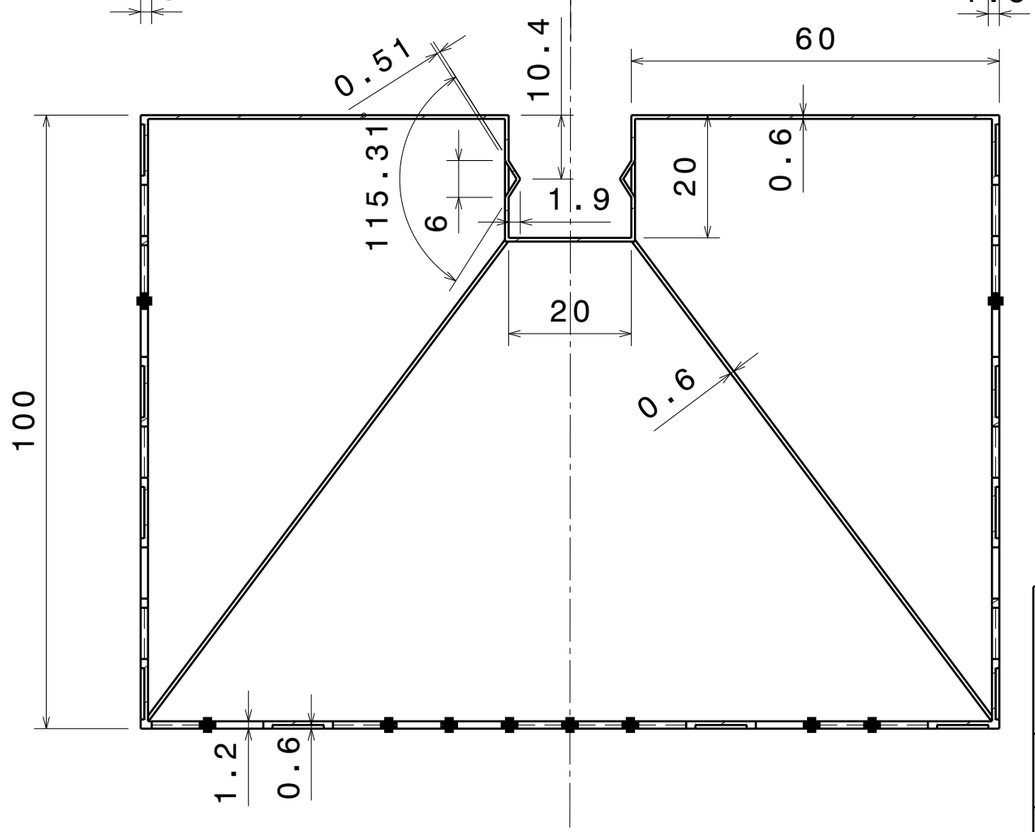
Anexo III – Planos

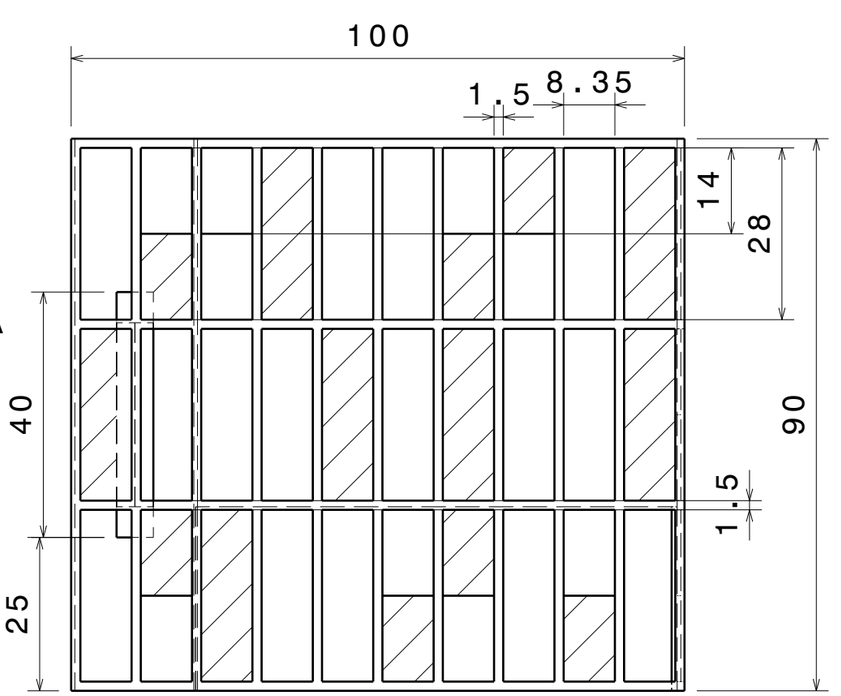
En este anexo se van a incluir los planos de algunos de los elementos utilizados en este TFG, y que se han explicado a lo largo de este TFG, más concretamente en los apartados *2. DISEÑO PARTE ESTRUCTURAL* y *3. DISEÑO PARTE MECÁNICA*.

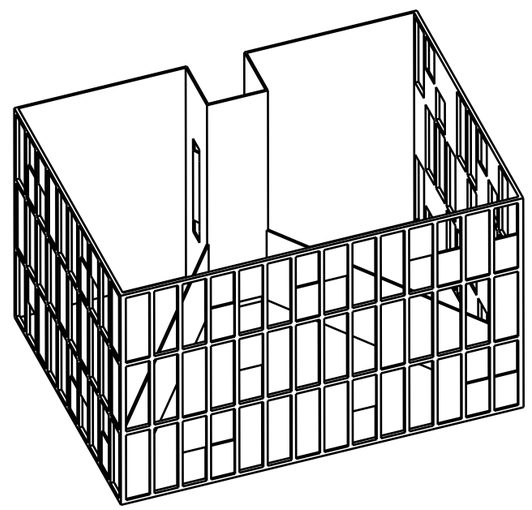
El orden en que se disponen estos planos es el siguiente:

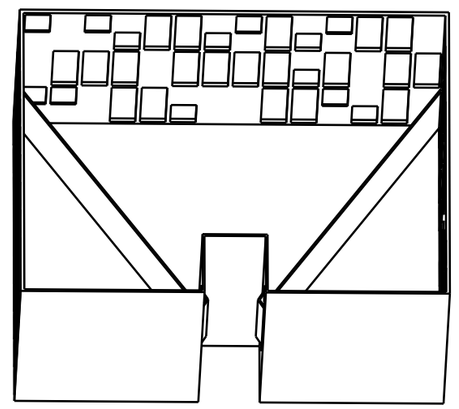
1- Carenado genérico.
2- Carenado porta servo.
3- Carenado porta TMD.
4- Estructura completa.
5- Explosionado estructura completa.
6- Explosionado carenado porta servo.
7- Explosionado carenado porta TMD.





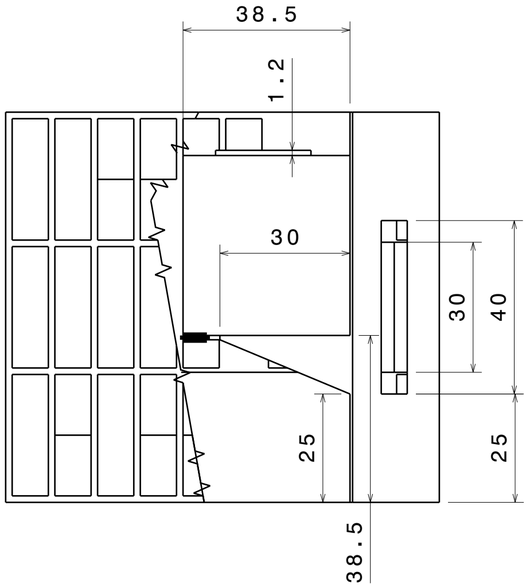


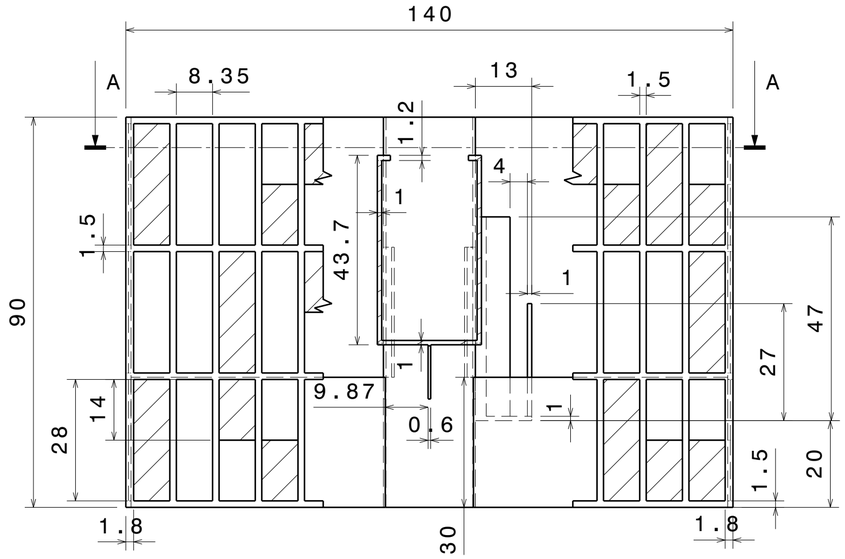


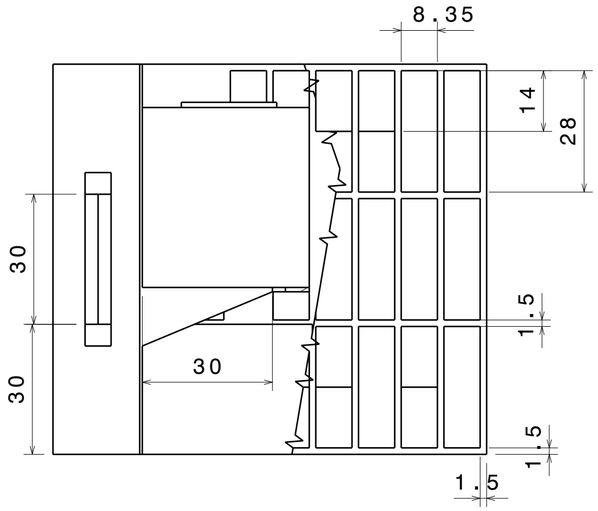


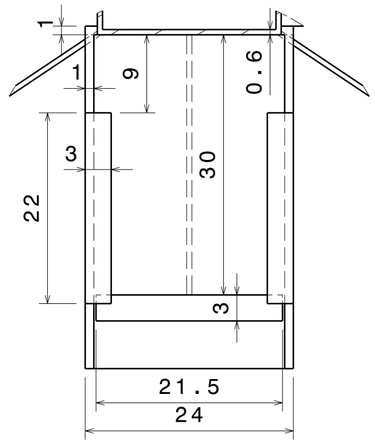
DASSAULT SYSTEMES

DRAWN BY Prieto, M.	DATE 28/03/2020	DRAWING TITLE Carenado genérico	
CHECKED BY Prieto, M.	DATE 28/03/2020	SIZE A3	DRAWING NUMBER 1
DESIGNED BY Prieto, M.	DATE 28/03/2020	SCALE 1:1	SHEET 1/7

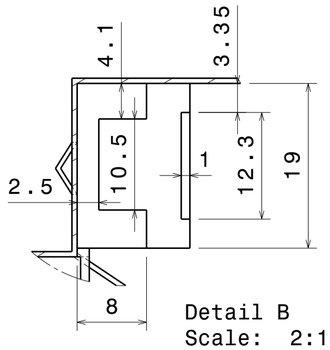




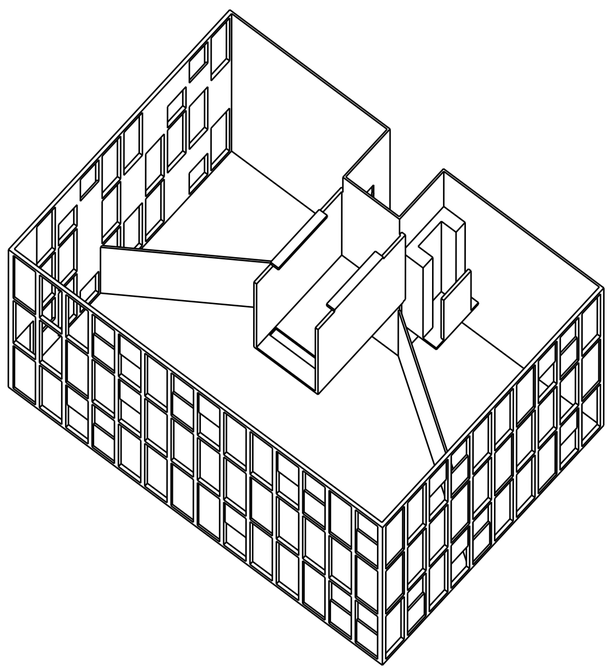


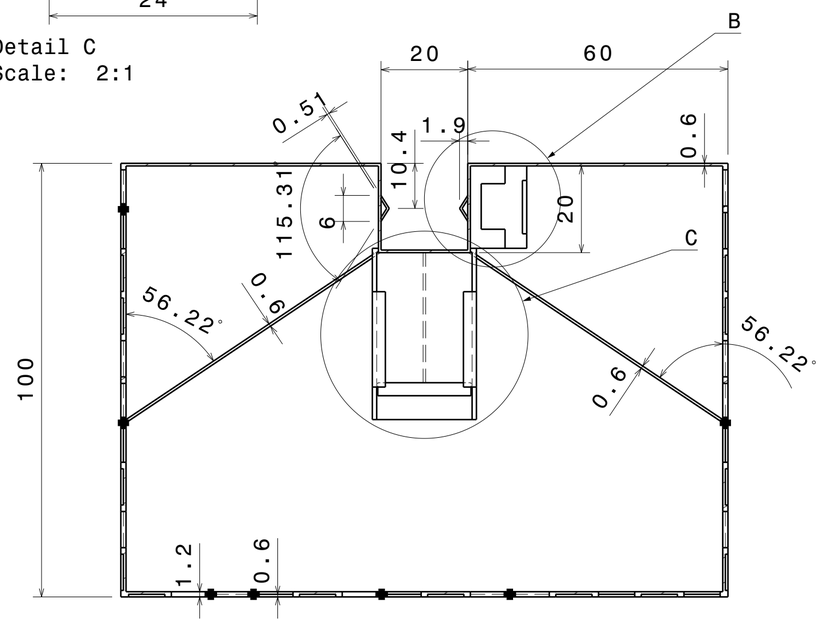


Detail C
Scale: 2:1



Detail B
Scale: 2:1

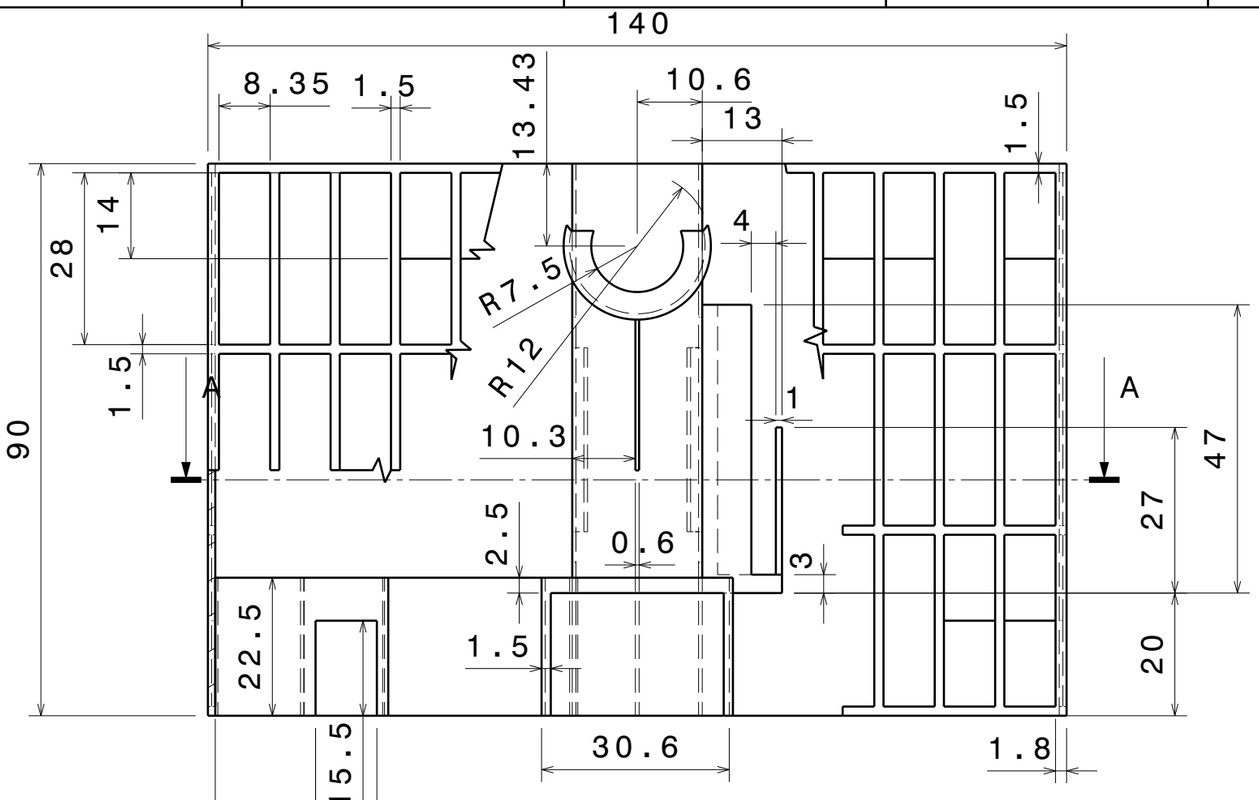


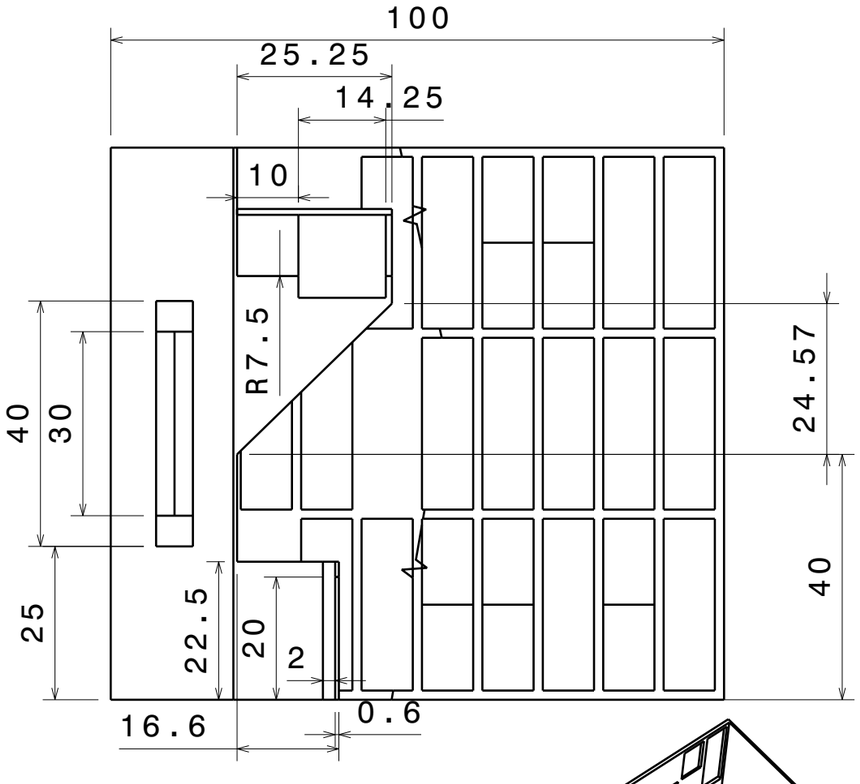


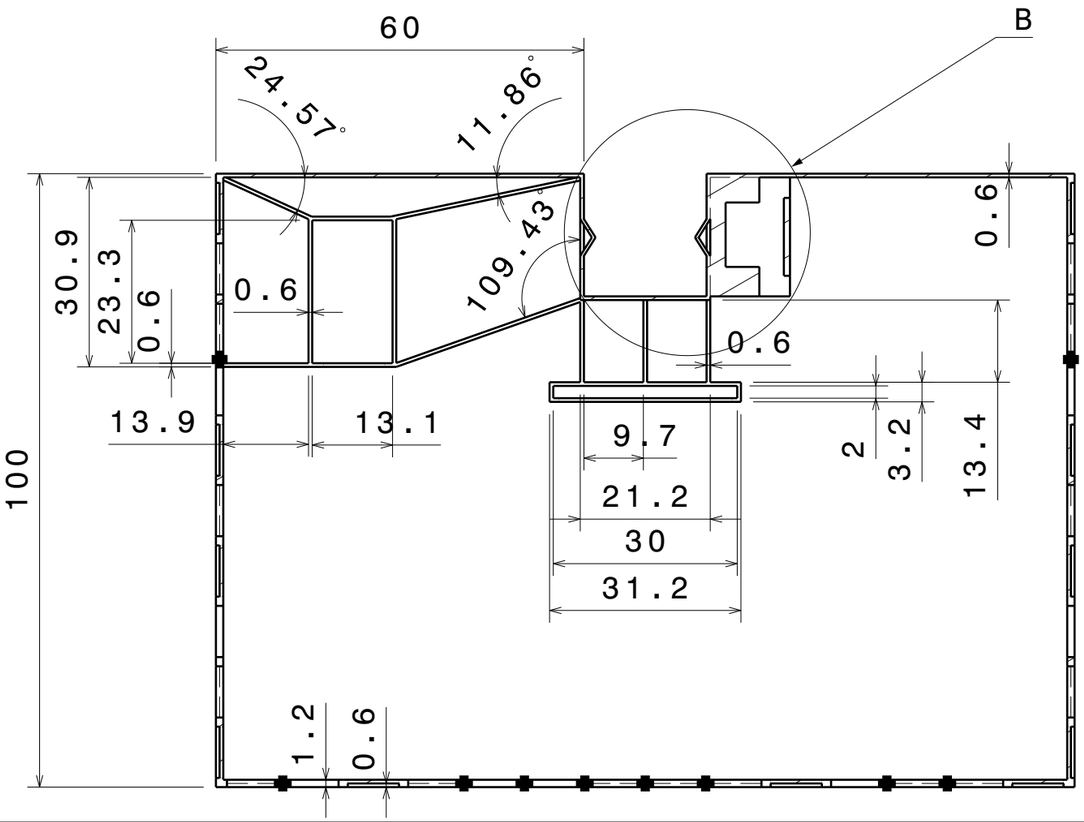
DASSAULT SYSTEMES

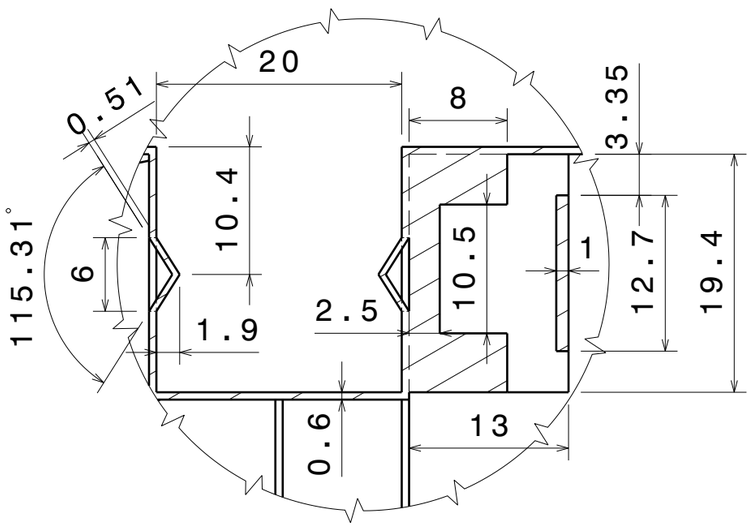
DRAWING TITLE
Carenado porta servo

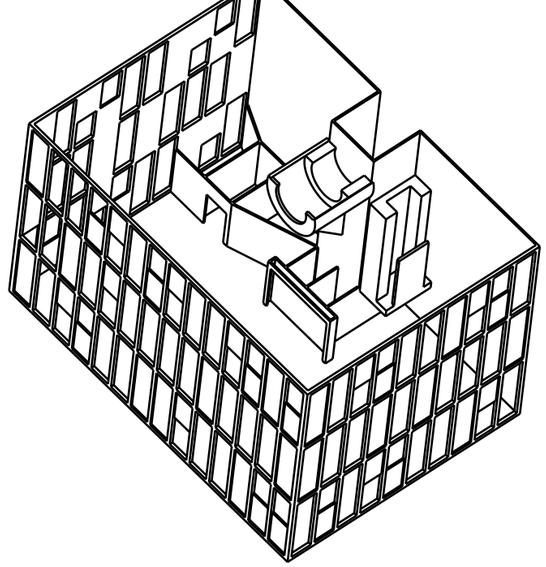
DRAWN BY	DATE	SIZE	DRAWING NUMBER
Prieto, M.	20/04/2020	A2	2
CHECKED BY	DATE		
Prieto, M.	20/04/2020		
DESIGNED BY	DATE	SCALE 1:1	SHEET 2/7
Prieto, M.	20/04/2020		











This drawing is our property.
It can't be reproduced
or communicated without
our written agreement.

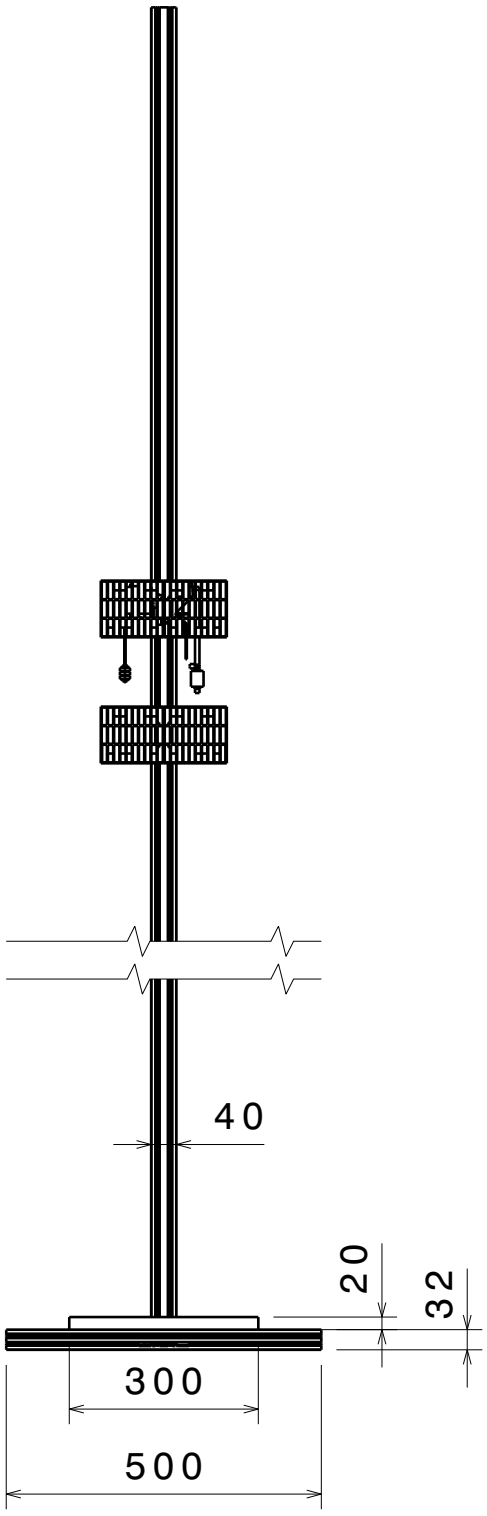
DASSAULT SYSTEMES

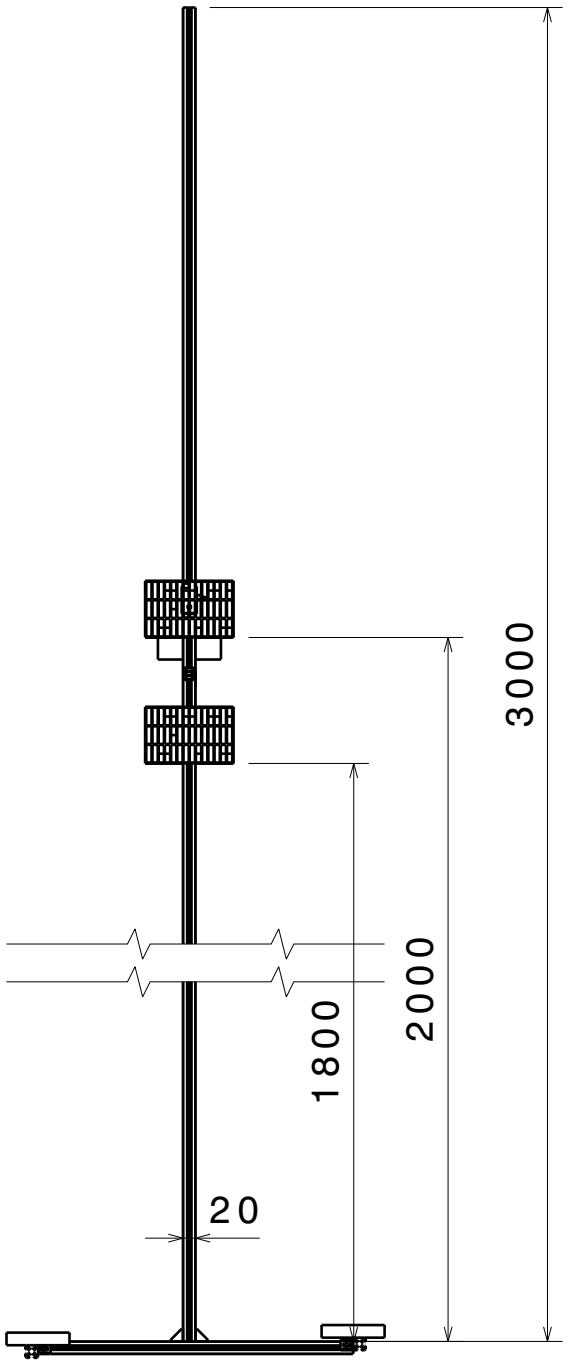
DRAWING TITLE

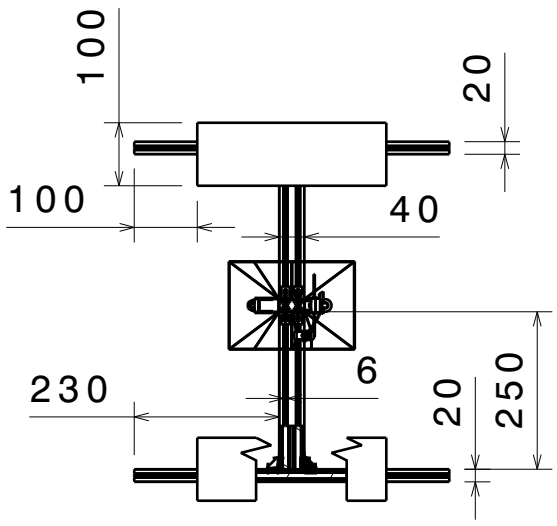
Carenado porta TMD

DRAWN BY	Prieto, M.	DATE	20/04/2020
CHECKED BY	Prieto, M.	DATE	20/04/2020
DESIGNED BY	Prieto, M.	DATE	20/04/2020

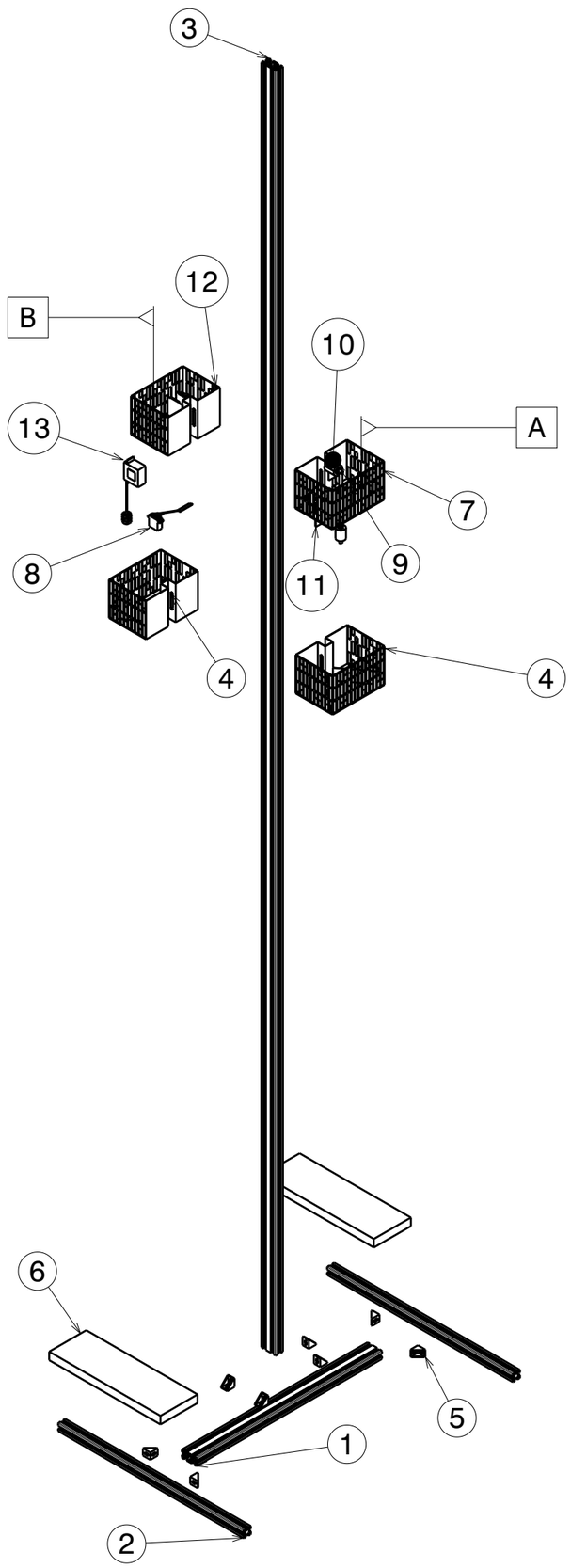
SIZE	DRAWING NUMBER	
A3	3	
SCALE 1:1		SHEET 3/7







This drawing is our property. It can't be reproduced or communicated without our written agreement.		DASSAULT SYSTEMES	
DRAWN BY Prieto, M.	DATE 29/04/2020	DRAWING TITLE Estructura completa	
CHECKED BY Prieto, M.	DATE 29/04/2020	SIZE A3	DRAWING NUMBER 4
DESIGNED BY Prieto, M.	DATE 29/04/2020	SCALE 1:12	SHEET 4/7



Bill of Material: Estructura aluminio

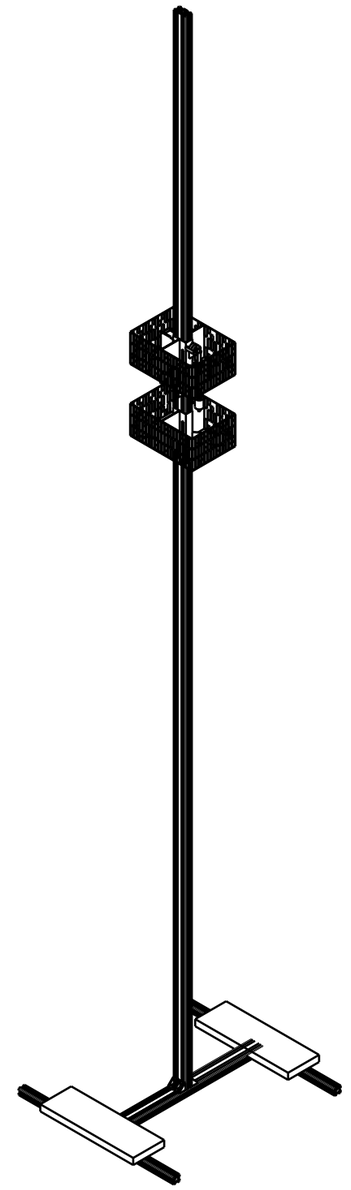
Number	Part Number	Quantity
1	Barra base	1
2	Barra 1 20x20	2
3	Barra central	1
A	Porta TMD	1
B	Porta servomotor	1
4	Carenado generico	2
5	Escuadra 1	8
6	Masas base estructura	2

Bill of Material: Porta TMD

Number	Part Number	Quantity
7	Porta TMD 110	1
9	TMD	1
11	Placa aluminio	1
8	Servo bloqueador	1

Bill of Material: Porta servomotor

Number	Part Number	Quantity
12	Part1	1
10	Arduino nano	1
13	Servo shaker	1



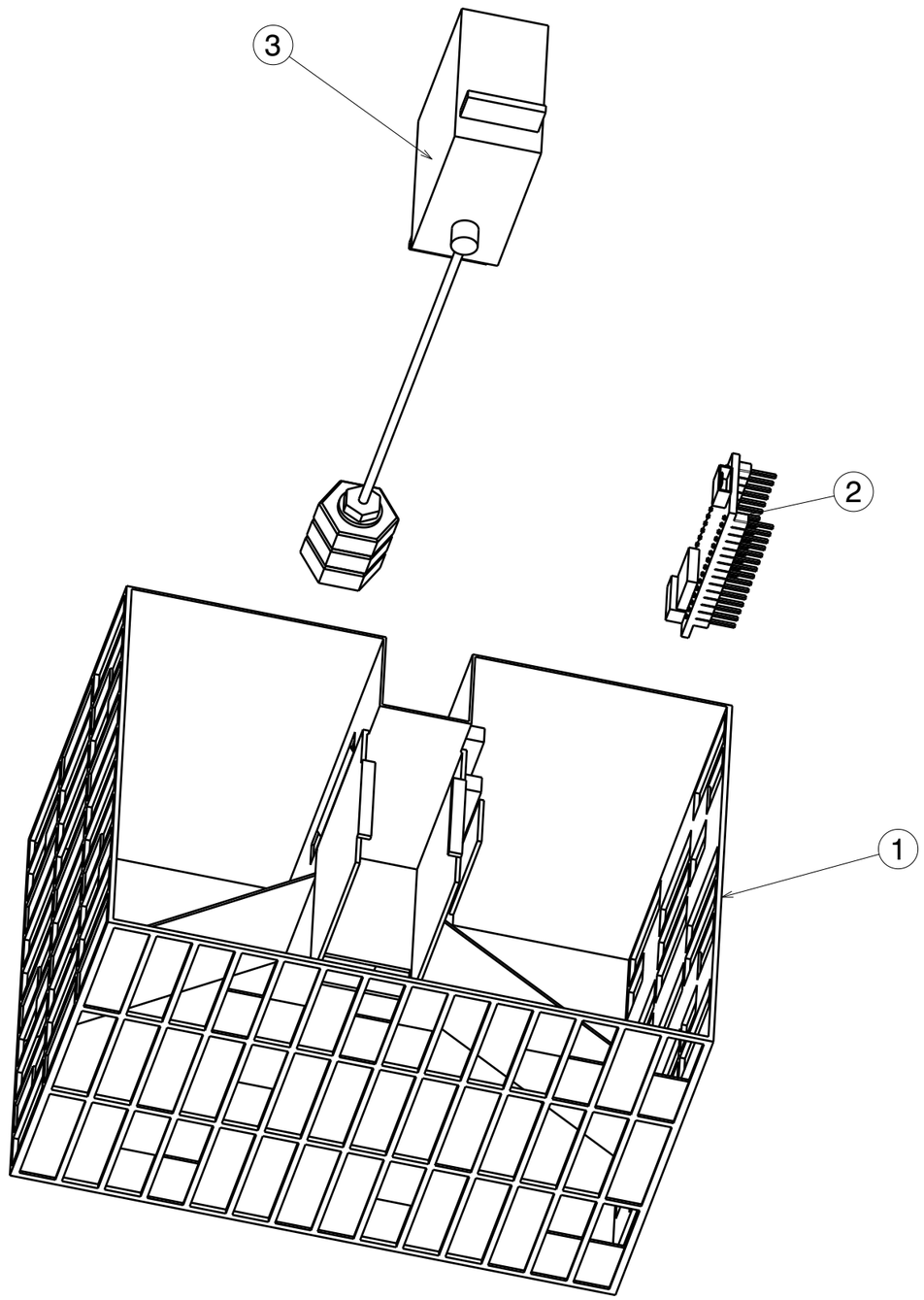
DASSAULT SYSTEMES

DRAWING TITLE: Explosionado estructura completa

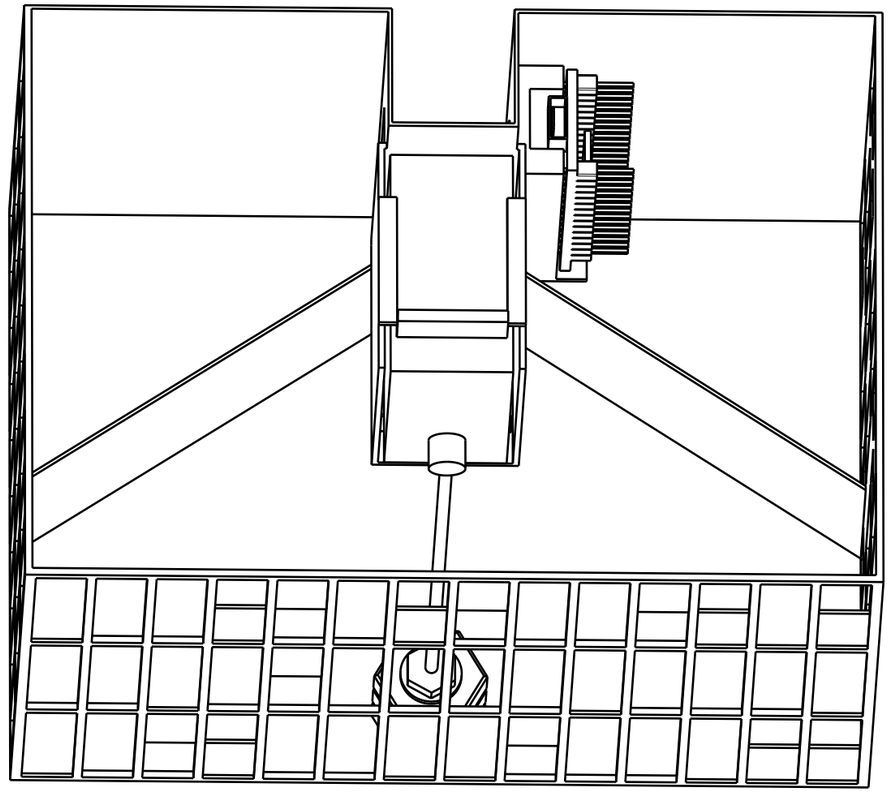
DRAWN BY: Prieto, M.	DATE: 29/04/2020		
CHECKED BY: Prieto, M.	DATE: 29/04/2020	SIZE: A3	DRAWING NUMBER: 5
DESIGNED BY: Prieto, M.	DATE: 29/04/2020	SCALE: 1:11	SHEET: 5/7

Bill of Material: Porta servomotor

Number	Part Number	Quantity
1	Porta servo	1
2	Arduino nano	1
	Servo shaker	1



Isometric view
Scale: 3:4

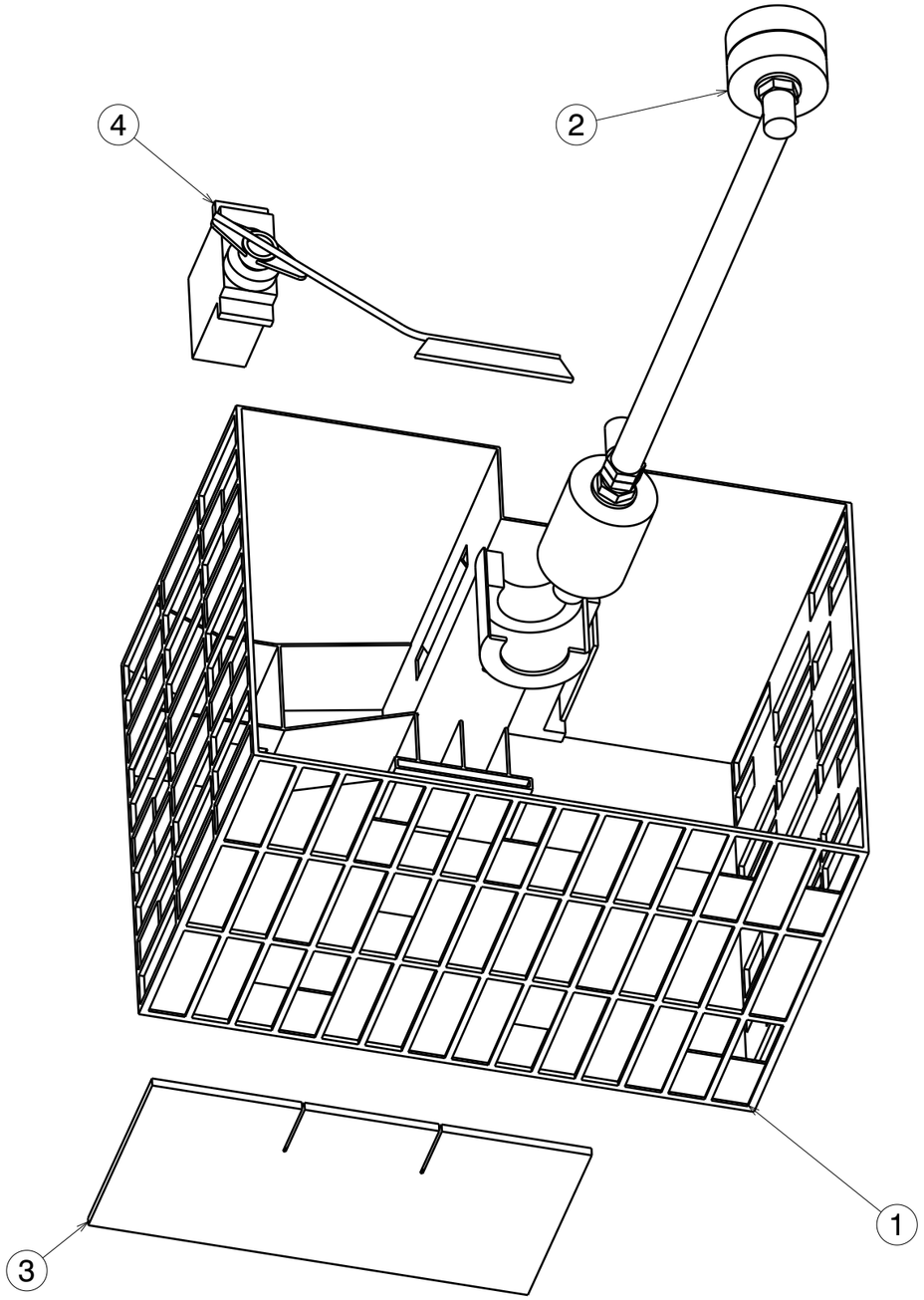


DRAWN BY	Prieto, M.	DATE 28/04/2020
CHECKED BY	Prieto, M.	DATE 28/04/2020
DESIGNED BY	Prieto, M.	DATE 28/04/2020

DASSAULT SYSTEMES

DRAWING TITLE: Explosionado carenado porta servo

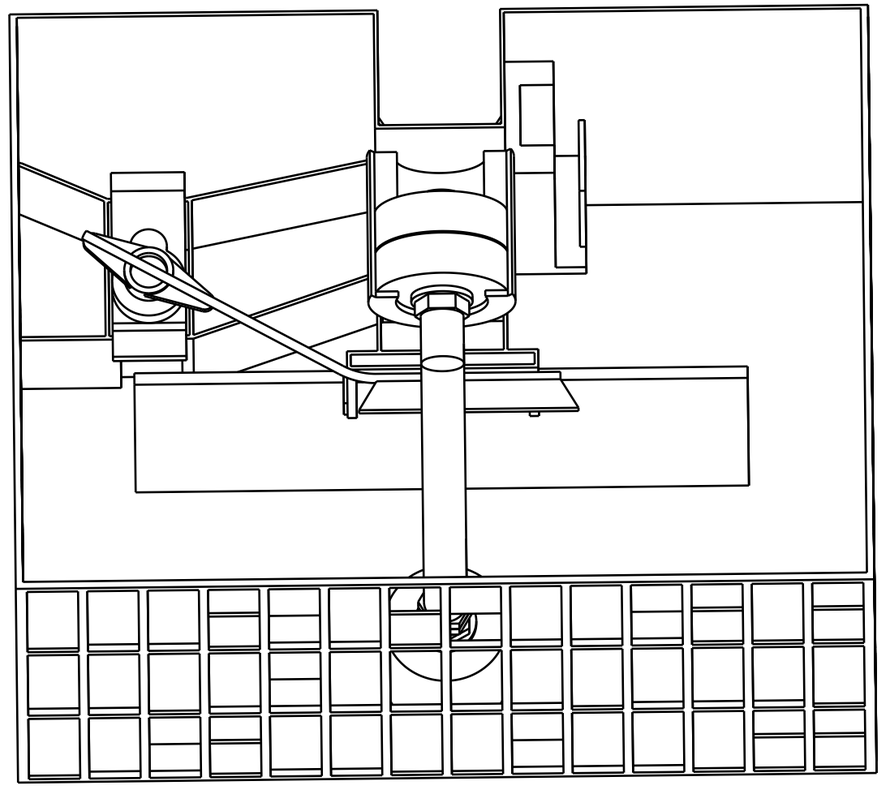
SIZE	DRAWING NUMBER	
A3	6	
SCALE 3:4		SHEET 6/7



Isometric view
Scale: 3:4

Bill of Material: Porta TMD

Number	Part Number	Quantity
1	Porta TMD 110	1
2	TMD	1
3	Placa aluminio	1
4	Servo bloqueador	1



DRAWN BY	Prieto, M.	DATE 27/04/2020
CHECKED BY	Prieto, M.	DATE 27/04/2020
DESIGNED BY	Prieto, M.	DATE 27/04/2020

DASSAULT SYSTEMES

DRAWING TITLE

Explosionado carenado porta TMD

SIZE	DRAWING NUMBER	
A3	7	
SCALE 3:4		SHEET 7/7