Universidad de Valladolid

UNIVERSIDAD DE VALLADOLID

ESCUELA DE INGENIERIAS INDUSTRIALES

Grado en Ingeniería Mecánica

Diseño, prototipaje y caracterización de una mesa sísmica de 1 grado de libertad a escala

Autor:

Morales Reyes, Judith

Tutor:

Lorenzana Iban, Antolín
Magdaleno González, Álvaro
C. A., I. T. y Mecánica de los Medios Continuos y Teoría de Estructuras

Valladolid, Abril 2017.

Agradecimientos

En primer lugar nombrar a mis padres y mi hermano, por enseñarme a luchar por lo que uno quiere, sin ellos nada hubiese sido posible.
Alex, gracias por tu paciencia, apoyo y comprensión. Dándome fuerzas cada día y recordándome que pase lo que pase, no cuenta donde estás, sino dónde vas a llegar.
Nunca me olvido de mis amigas, las que tengo cerca y las que no tanto, pues me entendéis en días buenos pero sobre todo en los malos, siempre estáis ahí y siempre os llevo conmigo.
Por último, al departamento de estructuras por esta oportunidad y todo lo que me habéis enseñado. Sobre todo a Álvaro por tu dedicación.

Resumen

El Trabajo de Final de Grado de Ingeniería Mecánica tiene como finalidad el diseño con herramienta CAD, mecanizado y montaje de los componentes de una mesa sísmica a escala.
Además se programa el motor, los sensores y controladores para el manejo del sistema, a través del software Arduino.
Posteriormente se caracteriza el sistema con el equipo del laboratorio, para conocer los límites de utilización y realizar las correcciones de los parámetros.
Todo ello con el objetivo final de obtener una mesa sísmica funcional y las especificaciones técnicas para su manejo.

Palabras clave

Mesa vibratoria
Excitación
Control
Límites de utilización
Caracterización

Abstract

The purpose of this Mechanic Engineering End of degree Project is achieving the design, by using CAD tool, machining and assembly of the componentes of scaled seismic table.
Furthermore the engine, sensors and controllers required for the use of this system are programmed by Arduino software.
Afterwards, the system is characterized by the laboratory equipment in order to find out about its utilization limits and conduct the suitable parameter corrections.
After all these approaches, the goal of obtaining a functional seismic table and the technical specifications for its use was finally achieved.

Keywords

Shaking table
Excitation
Control
Utilization limits
Characterization

Índice

1. Introducción

1.1. Justificación

Desde los inicios de la construcción de edificaciones ha preocupado su conservación a lo largo del tiempo, ya sea su resistencia estructural como por la acción de cargas estáticas o dinámicas, acciones climatológicas o acciones sísmicas del terreno.

Estas últimas son imprevisibles y su mayor componente es aleatorio, por lo que es interesante estudiar el comportamiento de las estructuras en este tipo de sucesos.

Todo ello ligado al uso de materiales cada vez más resistentes y la tendencia de diseño de edificaciones cada vez con más altura y la necesidad continua de abaratar costes, lleva a estructuras más vulnerables frente a la acción sísmica.

Por ello es importante dotarse de herramientas que puedan simular estas acciones y permitan realizar ensayos experimentales, para validar los cálculos numéricos de estas situaciones a pequeña escala, facilitando el desarrollo a gran escala o en fases más complejas.

Los avances tecnológicos y los softwares de adquisición libre permiten la accesibilidad a este campo en desarrollo continuo. Permitiendo el uso de controladores cada vez más precisos sin incurrir en grandes costes, para comparar la excitación empleada y la respuesta del sistema medida. Esto acerca los estudios a ámbitos de recursos más bajos, pero no con menos capacidad de desarrollo o ideas para la innovación. De esta manera se acerca a las nuevas generaciones, siendo esta problemática común en todo el mundo y estando a la orden del día.

1. 2. Objetivos

Se decide la realización de una mesa vibratoria de 1 grado de libertad, por la necesidad de estudiar cómo se comportan las estructuras a escala, de distintos materiales, formas y resistencias, sometidos a una excitación en una dirección.

Esto introduce un complemento idóneo para el Laboratorio de Estructuras de la EII/UVa, acercando a los estudiantes la posibilidad de realizar ensayos sísmicos sobre estructuras a escala.

Se establece como objetivo aplicar las competencias adquiridas en el Grado en Ingeniería Mecánica, para diseñar, modelar, construir y caracterizar el prototipo de una mesa vibratoria de 1 grado de libertad. Para ello se realizan estudios sobre los distintos mecanismos de excitación y se selecciona el más adecuado y económico, para adecuarse a las necesidades. Se selecciona el controlador para dicha excitación y se utiliza un software libre para conseguir simular una onda senoidal. Se controla la amplitud de dicha onda y la frecuencia de oscilación. Con ello se verifica su aplicabilidad, a partir de resultados experimentales a la masa equivalente de una maqueta de un edificio de dos plantas, disponible en el laboratorio y se comprueba la correlación entre los valores de aplicación y los medidos a partir de un equipo láser.

3. Alternativas para la excitación

Existen multitud de alternativas para diseñar una mesa vibratoria, por tanto se estudian algunas de ellas y se selecciona la más adecuada a las necesidades y dentro de un presupuesto razonable [Ref. Web 2].

A continuación se presentan las más relevantes.

3.1. Mecanismo biela manivela

Este sistema se compone de una plataforma donde se va a colocar la maqueta del edificio, un sistema de soportes para el conjunto, rodamientos lineales para proporcionar el desplazamiento de la plataforma en una dirección y el sistema biela-manivela, que está formado por un disco con ranura que permite variar la amplitud de movimiento y una manivela que conecta el disco y la plataforma. La biela se conecta al motor, que le imprime el movimiento de rotación [Ref. Web 1]. Además el sistema cuenta con un controlador que se encarga de poner en funcionamiento el motor y permite variar la velocidad de rotación y controlar la frecuencia.

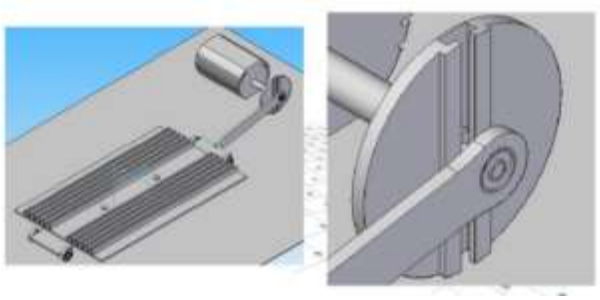


Figura 3.1.1: Esquema del mecanismo biela-manivela [Ref. Web 1].

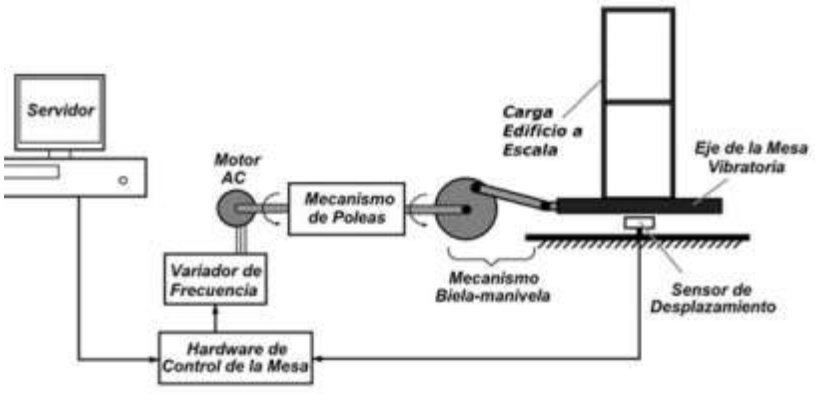


Figura 3.1.2: Esquema del sistema de excitación con mecanismo biela-manivela [Ref. Web 2].

En el anexo 9.3.1 se ve el presupuesto estimado para este montaje.
Se descarta por la necesidad de ajuste manual de la amplitud, además de los posibles problemas de ruido que se introducen en la caracterización, distorsionando los resultados finales. Como interesa que sea un sistema con un movimiento suave, se estudia otro mecanismo.

3.2. Mecanismo correa-polea o *slider*

El *slider* o carril motorizado se utiliza, por ejemplo, en el desplazamiento del cabezal de la impresora. Este se compone de un carril que fija la dirección y puede ser de aluminio para proporcionar precisión y suavidad en el movimiento. Al carril se le acopla el carro, que cumple con la función del desplazador de la mesa o plataforma. Se diseña el carro para permitir el movimiento longitudinal de la correa, de forma que esta se encuentre totalmente tensada y se mueva suavemente. Para ello hace falta una polea dentada, otra polea idéntica o un cojinete que permita el desplazamiento en el otro extremo. Necesita un tensor y la correa de transmisión que encaje perfectamente con la polea dentada y además sea de un material bastante rígido para evitar holguras por la elasticidad del material, siendo interesante el uso del poliuretano u otro plástico reforzado [Ref. Web 3].

Para controlar el sistema se acopla el motor a la polea dentada. El motor lleva un soporte que hay que diseñar de un material suficientemente rígido: aluminio, plástico u otro.

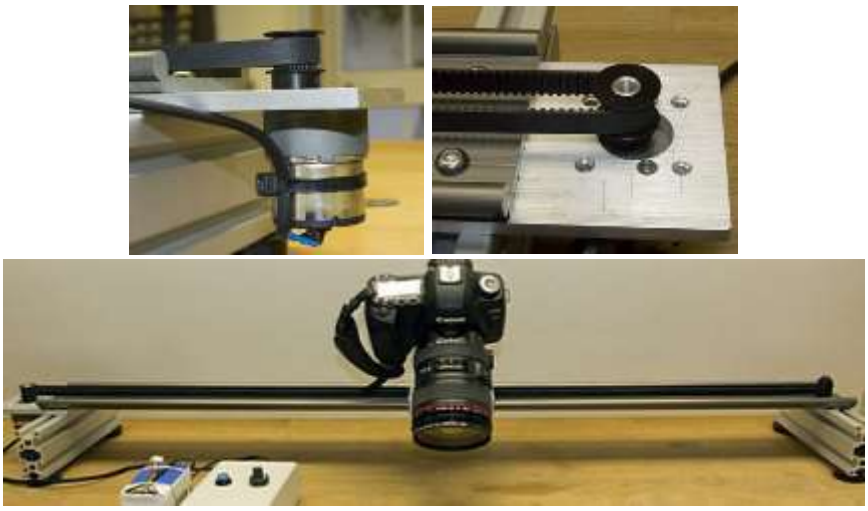


Figura 3.2.1: Fotografía del *slider* o carril motorizado [Ref. Web 3].

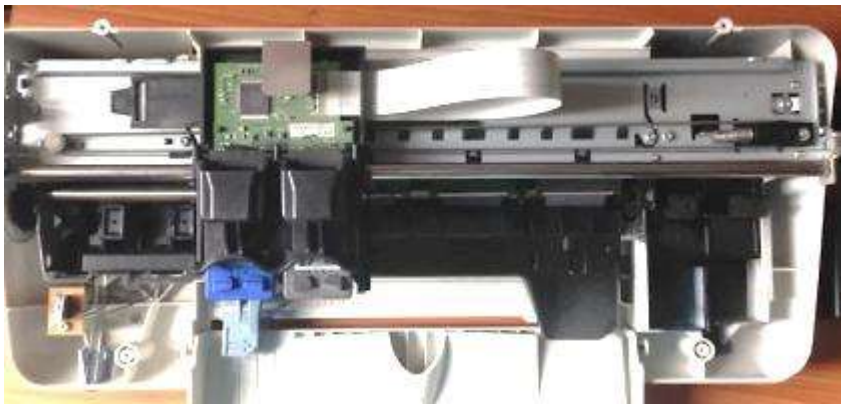


Figura 3.2.2: Fotografía del sistema de correa polea de una impresora convencional.

El presupuesto estimado para el sistema de excitación por correa y polea se indica en el anexo 9.3.2.

Es una opción sencilla y económica, por lo que se tiene en cuenta para el mecanismo a realizar. La mayor complejidad se encuentra en tensar la correa para su correcto funcionamiento, se considera viable pero se acaba descartando por el siguiente mecanismo.

3.3. Mecanismo husillo y guías

El mecanismo se compone del husillo que gira sobre sí mismo por la acción del motor, unido a este mediante un acoplamiento. Unido al husillo se diseña un desplazador que produce el movimiento de la plataforma. Esta se desplaza solidaria a las guías o rodamientos lineales, que permite un movimiento suave en dicha dirección.
Se diseña también el soporte del motor respecto de la base y el soporte del husillo, posicionadolo en el extremo contrario al motor, todo ello de aluminio que facilita el mecanizado. Se acopla una rueda moleteada en el extremo del husillo, para permitir el movimiento manual del mismo y sirviendo de retención del movimiento axial, que a su vez es absorbido por los rodamientos colocados en el husillo [Ref. Web 4 y Web 5].

Se calcula el presupuesto para el caso de mesa vibratoria con la excitación a partir de husillo en el anexo 9.3.3.

Figura 3.3.1: Fotografía de mesa vibratoria con husillo [Ref. Web 4].

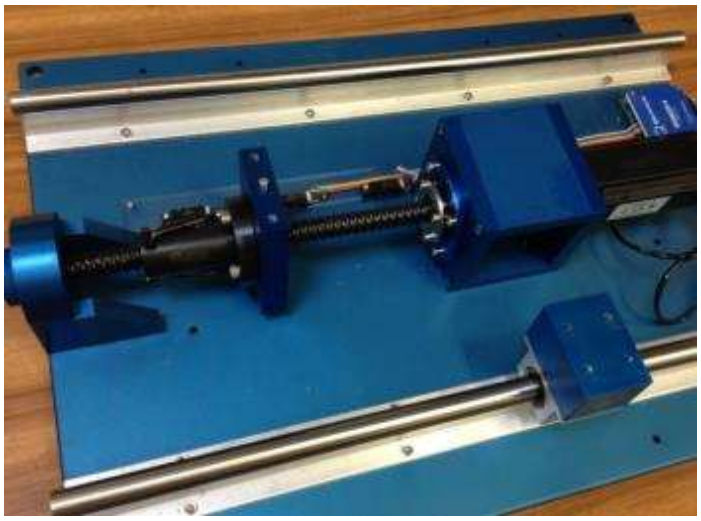


Figura 3.3.2: Fotografía de los componentes de mesa vibratoria
con husillo [Ref. Web 4].

Se tiene en cuenta, como el caso anterior, por ser un montaje sencillo y económico. Con componentes fáciles de comprar y diseñar. Además el husillo no permite casi holguras, que es el principal interés del sistema. Por eso acaba siendo el mecanismo seleccionado para el montaje.

3.4. Actuador hidráulico

Se monta la plataforma sobre un sistema de rodamientos lineales de baja fricción, que permiten el desplazamiento en la dirección de excitación. El movimiento se produce por la acción del actuador. El sistema hidráulico cuenta con una servo-válvula para controlar los cambios de sentido y el movimiento del actuador hidráulico. Para seleccionar este último, se determina la fuerza máxima que debe ejercer, a través de la carga total de la mesa y la estructura del edificio, máxima aceleración del conjunto y teniendo en cuenta capacidad de carga de los rodamientos seleccionados [Ref. Art. 2, Art. 3 y Web 6].

Por último el servo-controlador transmite la señal enviada del ordenador a los puertos del actuador en la servo-válvula.

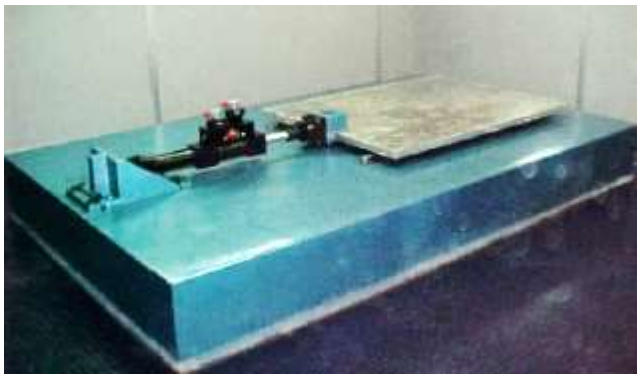


Figura 3.4.1: Fotografía de mesa vibratoria accionada por actuador hidráulico [Ref. Art. 2].

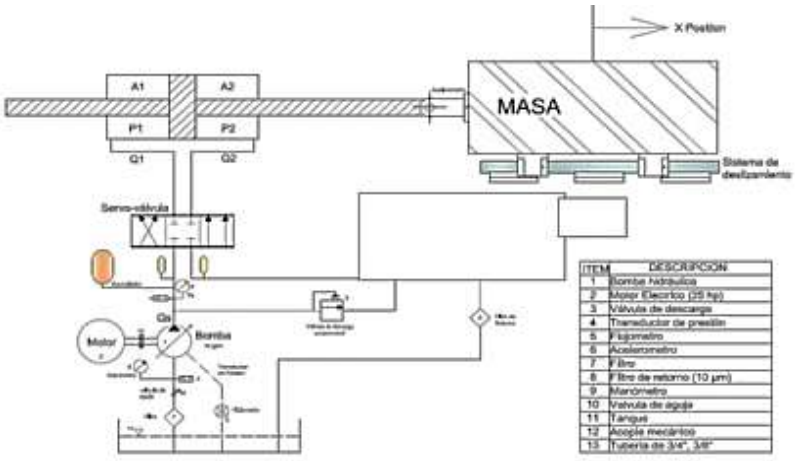


Figura 3.4.2: Esquema del sistema de control de la mesa vibratoria [Ref. Art. 2].

El presupuesto calculado para la excitación por actuador hidráulico se muestra en el anexo 9.3.4.

El sistema de excitación hidráulico es una de las mejores opciones ya que proporciona una gran fiabilidad y precisión en los movimientos. Pero tiene como inconveniente la complejidad del montaje y el coste económico, descartándolo como opción válida para el montaje.

3.5. Actuador neumático

El movimiento de una mesa vibratoria neumática se genera por medio de actuadores (cilindros y válvulas), alimentados por un compresor.
El desplazamiento, velocidad y aceleración son controlados por señales eléctricas en una electroválvula. Estas son recibidas desde un controlador, el cual toma acciones en función de la orden dada [Ref. Art. 4].
Para sostener y permitir el desplazamiento de la mesa, se incorporan unos soportes y rodamientos lineales, igual que en caso de actuador hidráulico. Pero ahora se compone de un sistema neumático, formado por un compresor de aire que proporciona el flujo de aire necesario, con su filtro y regulador de aire, una válvula que proporciona la de presión, con una servo-válvula o electroválvula, un servo-controlador para la adquisición de datos, control y comunicación con la electroválvula y el actuador neumático Para dimensionar adecuadamente el actuador, hay que calcular la fuerza que este debe transmitir a la mesa [Ref. Art. 5].

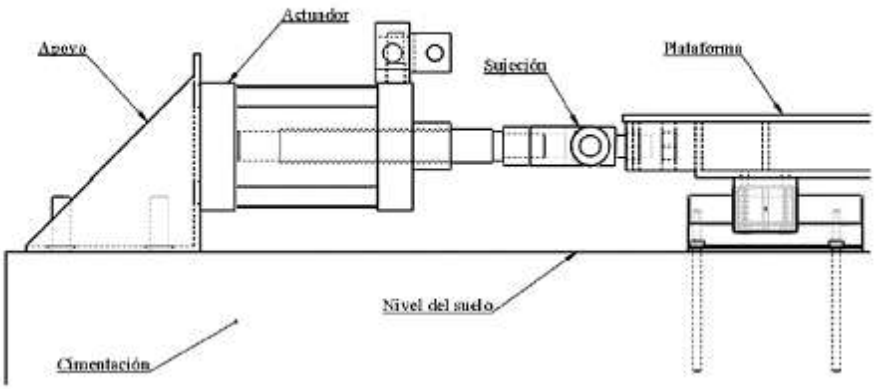


Figura 3.5.1: Esquema de mesa vibratoria accionada por actuador neumático [Ref. Art. 5].

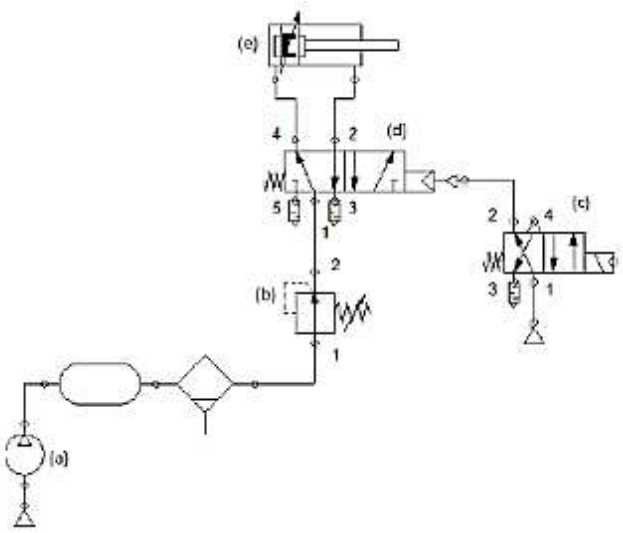


Figura 3.5.2: Esquema de sistema de control de la mesa vibratoria por actuador neumático [Ref. Art. 5].

En el anexo 9.3.5 se muestra el presupuesto estimado para este sistema de accionamiento de mesa vibratoria.

Los sistemas de excitación neumática son menos precisos que los de hidráulica, por ser el aire un fluido compresible. Pero sigue siendo una buena opción para este tipo de sistemas. Igualmente que los actuadores hidráulicos, tienen el inconveniente de un montaje bastante complejo y el alto coste económico. Por eso se rechaza este sistema de accionamiento.

3.6. Motor lineal

El movimiento del motor lineal es controlado por un *driver* que requiere para su funcionamiento, una señal de referencia SP (señal que indica el movimiento: sismo, armónico o barrido en frecuencia) y una señal de realimentación, proveniente del encoder lineal que mide la velocidad y desplazamiento de la mesa vibratoria. La señal de referencia es generada por un ordenador y enviada en forma de voltaje a través de la tarjeta de adquisición y control de datos hacia el servo-controlador [Ref. Art. 6 y Art. 8].

Este sistema se compone de los rodamientos que deben soportar las cargas y realizar los desplazamientos, con baja fricción y altas velocidades. Se incorpora el encoder lineal y el motor lineal siendo los síncronos de imán permanente (PMLSM) los más utilizados por alcanzar altas aceleraciones y además no requieren de elementos de transmisión como cajas reductoras, cadenas o tornillos de acople para convertir el movimiento de rotación en

traslación. Esto evita la fricción y las limitaciones mecánicas en aceleración y velocidad. Además el servo controlador que sea capaz de entregar al motor las corrientes nominal y de pico que va a consumir durante su movimiento. Y el ordenador o dispositivo que genere la señal para producir el movimiento [Ref. Art. 7].

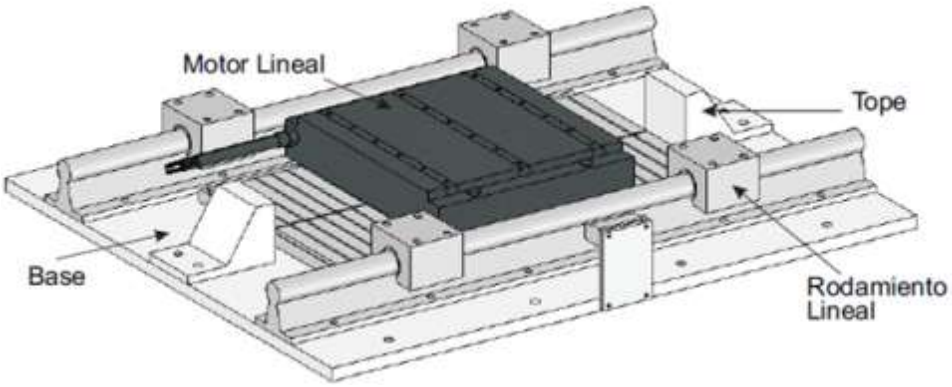


Figura 3.6.1: Esquema de sistema de motor lineal [Ref. Art. 7].

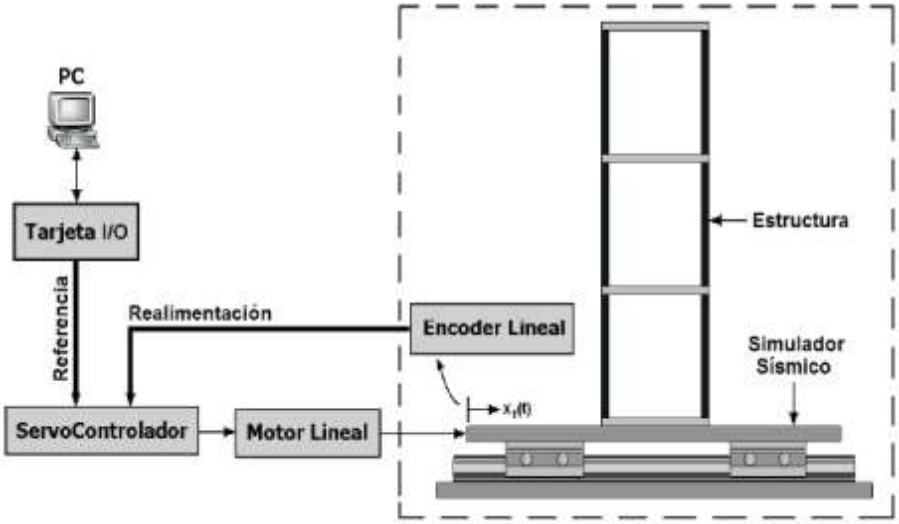


Figura 3.6.2: Esquema de mesa vibratoria excitado con motor lineal [Ref. Art. 7].

El sistema de excitación con motor lineal es la mejor opción para un control total de los desplazamientos y evitar holguras y ruidos. Pero su elevado coste lo descarta como opción por completo. Esto se puede ver en el presupuesto del anexo 9.3.6.

3.7. Módulos lineales

Se trata de un sistema de guías lineales que permite el desplazamiento en una dirección. Tiene la ventaja de ser un sistema compacto en el que únicamente se monta la plataforma y el controlador, porque el mecanismo que permite el desplazamiento ya está incorporado.

El modelo *Compact Rail* está compuesto de guías lineales de acero con pistas de rodadura templadas por inducción y cursores de rodamientos radiales de alta precisión en acero templado. Existen tres tipos de guías, de perfiles en U, T y K, y permiten compensar la falta de precisión de paralelismo de las superficies. Además con dos tipos de cursores, serie N con cuerpo en aleación de aluminio y la serie C con cuerpo de acero. Los rodamientos están lubricados de por vida. Es apropiado para aplicaciones con velocidades de hasta 9 m/s. Con anchos de la guía: 18 mm, 28 mm, 35 mm, 43 mm y 63 mm y precarga regulable. Las pistas de rodadura están ubicadas dentro de un perfil en C para protegerlas contra la suciedad [Ref. Web 7].

Figura 3.7.1: Ilustración de la guía lineal *Rollon* [Ref. Web 7].

El presupuesto de este tipo de guías se muestra en el anexo 9.3.7.

Existe otra opción que son los Módulos Lineales *Unimotion*, que son de transmisión por correa dentada o con husillo de bolas, para cargas elevadas, velocidad mayor de 10 m/s y elevada precisión y repetitividad. Además pueden llevar bridas prediseñadas para acoplamiento de motores y reductoras [Ref. Web 8].

Se ve en el anexo 9.3.8 el coste de 3 posibilidades, por motivos económicos se descartan ambas opciones.

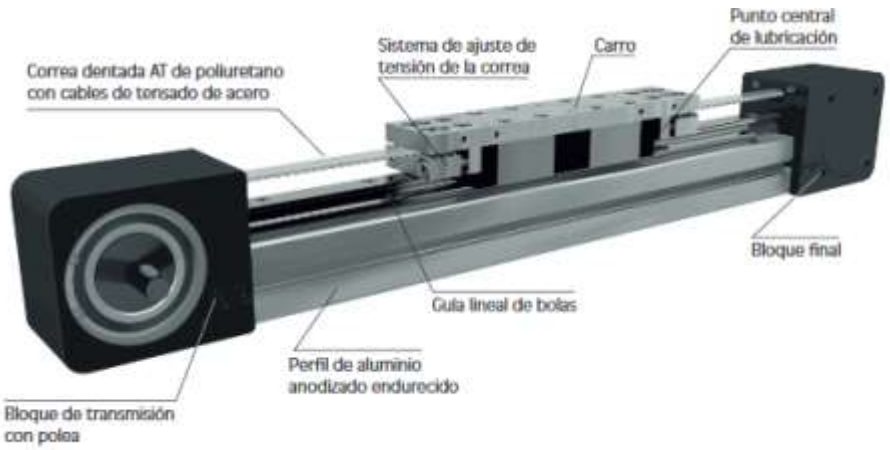


Figura 3.7.2: Ilustración del módulo lineal *Unimotion* [Ref. Web 8].

- Módulo lineal de correa dentada de poliuretano y reforzado con cables de acero, cuya carga soportada es elevada, alta velocidad (>10 m/s) y buena precisión y repetitividad (MTJ).

- Módulo lineal de husillo de bolas, con elevada carga soportable, alta velocidad y buena precisión y repetitividad (MTV).

- Módulo lineal de correa dentada para altas prestaciones y bajo coste. Con bridas prediseñadas para acoplamiento de motores y reductores (MTJECO).

4. Selección, diseño, construcción y ajuste de componentes

4.1. Cálculo, selección y diseño de los componentes

Después de considerar todas las opciones vistas anteriormente y descartando la mayoría como se ha indicado, se decide plantear las opciones de diseño con correa (que se deja para línea futura) y con husillo. En un primer momento se selecciona el conjunto de movimiento en el eje z, que se vende montado, para optimizar. Pero finalmente se comprueba que hay que desmontarlo y volverlo a montar con el desplazador en otra posición donde se distribuya mejor la carga que va a soportar.

Figura 4.1.1: Ilustración del conjunto de eje z [Ref. Web 5].

Así se opta por comprar los componentes y materiales adecuados para el diseño y montaje con husillo de forma independiente.

El mismo proveedor, dispone de conjuntos de husillos antiholgura, guías templadas, soportes de eje y bloques con rodamientos para guías, rodamientos de valona, caja de tuerca, tuercas de retención, tuercas antiholgura y acoplamiento del eje motor al husillo. También se adquieren el motor y el *stepping driver* de control [Ref. Web 5].

Figura 4.1.2: Ilustración del conjunto husillo antiholgura [Ref. Web 5].

Para seleccionar el diámetro del husillo de rosca trapecial, se realiza una estimación. Se considera que va a sufrir torsión y flexión, por ello se calcula la tensión máxima equivalente de ambos efectos (ecuación 4.1.2) y para un cierto coeficiente de seguridad y material (ecuación 4.1.1), con ello se obtiene el diámetro estimado. El material de referencia es un acero especial no aleado (dúctil) [Ref. Libro 9].

$$\sigma_{máx}^{eq} = \frac{S_y}{n} = \frac{460\ MPa}{2} \tag{4.1.1}$$

$$\sigma_{máx}^{eq} = \sqrt{\sigma_{x\,máx}^2 + 3 \cdot \tau_{xy\,máx}^2} \tag{4.1.2}$$

Para la torsión, conocidas la potencia y la velocidad de rotación aproximada que debe soportar el husillo (ecuación 4.1.3), se obtiene la tensión en función del diámetro (ecuación 4.1.4), se toma como referencia el modelo de la fundación CARTIF [Ref. Web 4] y calculando para las necesidades que se buscan [Ref. Libro 9]:

$$T_{máx} = \frac{P}{w} = \frac{162925\ w}{3141.73\ rpm} \tag{4.1.3}$$

$$\tau_{xy\,máx} = \frac{T_{máx} \cdot \frac{\emptyset}{2}}{\frac{\pi \cdot \emptyset^4}{32}} = \frac{2.64 \cdot 10^{11}}{\emptyset^3} \frac{N}{m^2} \tag{4.1.4}$$

Para la flexión, conocido el momento máximo aproximado que soportar el husillo, se obtiene la tensión en función del diámetro (ecuación 4.1.5) [Ref. Libro 9].

$$\sigma_{xy\,máx} = \frac{M_{máx} \cdot \frac{\emptyset}{2}}{I} = \frac{7.965 \cdot 10^{10}}{\emptyset^3} \frac{N}{m^2} \tag{4.1.5}$$

Con los resultados obtenidos y sustituyendo en la ecuación 4.1.2, se obtiene la tensión máxima equivalente en función del diámetro. Igualando a la ecuación 4.1.1, con un cierto coeficiente de seguridad de valor 2, se obtiene un diámetro de 12.63 mm.

Como se indicó, se toma de referencia la mesa vibratoria empleada en el centro tecnológico CARTIF [Ref. Web 4], cuyo husillo de 16 mm de diámetro, mueve una carga útil máxima a 2,5 G de 7,5 kg.

Se quiere mover una carga útil, como máximo de 5 kg y como para los cálculos se ha sobre estimado respecto a la referencia anterior, se selecciona un husillo trapezoidal de diámetro 12 mm de paso 6 mm.

El pedido final al proveedor (con sus referencias [Ref. Web 5]) es el siguiente:

2 barras templadas SFC de 20 mm de diámetro y 500 mm de longitud.
4 soportes SHA20A para las barras templadas.
4 bloques SC20V para las barras templadas.
1 husillo M12x3 P6 y 300mm de longitud.
1 acoplamiento *delrin* M12x3P6-6.35.
1 caja de tuerca para el husillo.
1 tuerca *delrin* con compensación de holgura M2x3.
2 tuercas de retención *delrin* P.6.
2 rodamientos con valona 32/30.5x12x10.
1 motor *NEMA* 23HS2430
1 *stepping* driver DM556 para programación en Arduino.
2 placas de aluminio de dimensiones 500x400x10 mm y 400x400x10 mm.

Por otra parte, se diseñan los componentes necesarios para completar el sistema. Se necesita elevar la posición z de las guías y la caja del husillo para ajustar la linealidad del eje del motor a la del husillo, son los calzos A y B referidos en la figura 4.1.3. Se utilizan para ello una guía de polietileno, reutilizada del laboratorio.

También se diseñan los soportes del husillo y del motor que van anclados a la base, son los soportes A y B de la figura 4.1.3, respectivamente. El soporte del husillo lleva alojado los rodamientos de valona contrapuestos, que se encargan de absorber los movimientos axiales. Estos soportes se diseñan a partir de placas de aluminio de dimensiones 80x60x20 mm, adquiridos del proveedor anteriormente mencionado [Ref. Web 5].
La tuerca *delrin* de antiholgura, lleva un muelle que rigidiza el paso del husillo por la rosca interior. Se decide quitar para facilitar el paso, debido a que el alojamiento ya evita las posibles holguras y se necesita suavidad en el movimiento.

Para el anclaje de todos los componentes a la base y la plataforma se utilizan tornillo de M5 y M6 de cabeza cilíndrica allen y vástagos 20 mm, 30 mm y 60 mm.

Como se indica en la lista de componentes se selecciona un motor paso a paso *NEMA* 23HS2430, controlado por un *stepping driver* DM556 [Ref. Web 5] a través del software Arduino. En primer lugar, se considera la opción de usar un servomotor, por sus características al permitir definir el ángulo de giro y el control a través de la modulación de pulsos, proporcionando gran precisión de movimiento. Además Arduino posee su propia librería servo, por lo que es fácilmente programable. Pero se descarta por su coste y elevadas dimensiones que dificultan el manejo de la mesa.
En el anexo 9.5 se presentan las posibilidades consideradas y descartas, con sus correspondientes presupuestos.

Por último se iba a reutilizar una fuente de alimentación de cuadro del laboratorio de 24 V y 0,88 A, pero se comprueba que era necesaria una de mayor corriente de salida, dado que el motor puede alcanzar un máximo de 4,4 A según las especificaciones.

Se selecciona para comprar una fuente de alimentación Lixada transformador de salida 24 V, 5 A y 120 W de potencia, con unas dimensiones de 130x98x40 mm adecuadas para el montaje diseñado.

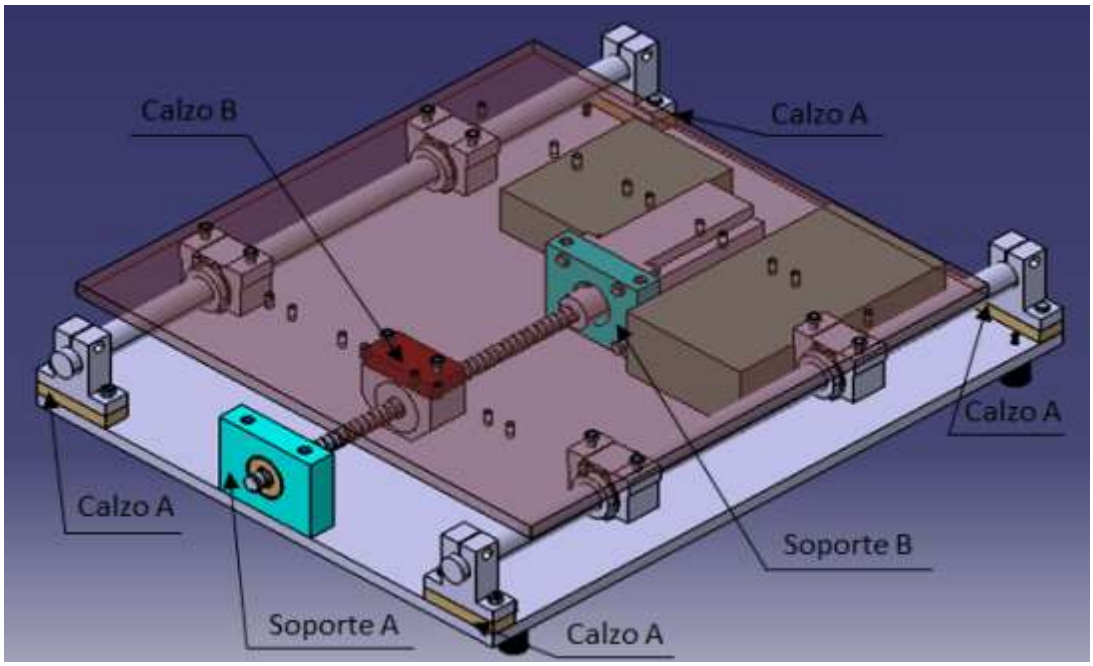


Figura 4.1.3: Diseño de mesa vibratoria mediante husillo.

Para realizar todos los diseños y ajustes de los componentes de la mesa vibratoria, se realiza una simulación completa en CATIA V5. Se indica el presupuesto final de los componentes seleccionados y diseñados en el anexo 9.3.3.

4.2. Mecanizado y montaje de los componentes

Para diseñar la mesa vibratoria se ha usado de referencia el modelo de la fundación CARTIF "*Shake Table II de QUANSER INNOVATE-EDUCATE*" [Ref. Web 4], cuyo presupuesto y más datos se puede ver en el anexo 9.2. Pero en este caso se desea mover 1 ó 2 edificios de dos plantas como máximo, por lo que se realizan los diseños de las piezas a fabricar en CATIA V5, ajustando a dichas necesidades. Los componentes que son posibles se compran como se indicó en el apartado anterior.

Por tanto es necesario comparar 2 placas de aluminio, la primera de dimensiones 500x400x10 mm, que es la placa base y sustenta todos los componentes necesarios para formar la mesa vibratoria. Y la segunda con dimensiones 400x400x10 mm, donde se anclan las maquetas de los edificios, es decir la plataforma móvil.
Además se compran otras dos placas de aluminio de dimensiones 80x60x20 mm, que se encargan de sostener el motor y el husillo de forma rígida, todo ello anclado a la base.

Como se necesitan calzos para la alineación de las guías con el husillo, se reutiliza un perfil de polietileno existente en el laboratorio.

Por último el resto de componentes se adquieren [Ref. Web 5], siendo: el husillo trapezoidal, las 2 guías de acero templado, 4 soportes para dichas guías y el desplazamiento la placa superior, otros 4 soportes para anclar las guías a la placa inferior, 2 rodamientos de valona para absorber el desplazamiento axial del husillo, el conjunto antiholgura de plástico con tuerca, acoplamiento al eje del motor, caja de tuerca para desplazar la mesa y 2 tuercas de retención, además de tornillería de M5 y M6 para el debido anclaje, las patas de goma de la mesa y el motor, el *stepping driver* y la placa de Arduino Mega.

Se mecanizan algunos de los componentes anteriores para un acoplamiento eficaz y ajustarse al modelo realizado en CATIA.

4.2.1. Mecanizado de la base

Se trata de la base donde se sustenta todo, como se compra acorde a las medidas deseadas, únicamente se realizan los agujeros de anclaje de los soportes de las guías, las placas de aluminio de sustentación del husillo y anclaje del motor y los taladros necesarios para la colocación de unas patas.

Para hacer los taladros se aplica una laca para una mejor visualización de las marcas.

Figura 4.2.1.1.: Aplicación de la laca.

Posteriormente se mide con el gramil de precisión y se marcan con el granete todos los puntos a taladrar.

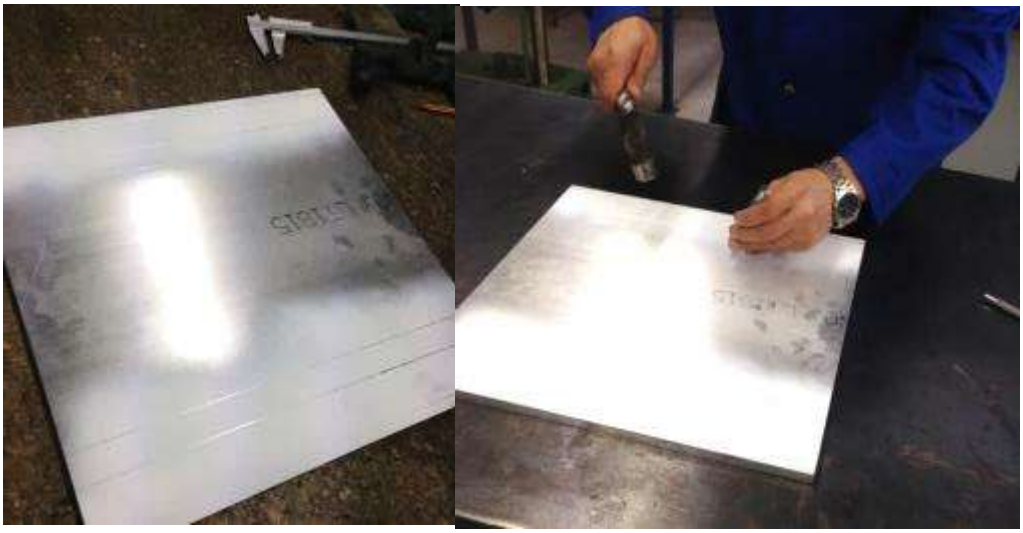


Figura 4.2.1.2: Medición y marcado de los taladros.

Se realizan los agujeros con el taladro de columna disponible en la Escuela Politécnica.

Figura 4.2.1.3: Taladrado de los agujeros.

Debido a que la placa tiene 500 mm de longitud y hay que realizar dos taladros a 297 mm del extremo, la placa choca con el taladro de columna, por lo que se realizan en otro taladro de mayores dimensiones y espacio.

El plano de dicho componente se puede ver en el anexo 9.4.

4.2.2. Mecanizado de la placa superior o plataforma

En este caso, esta placa sirve para sustentar los edificios, también se compra acorde a las medidas deseadas, por lo que se realizan los agujeros de anclaje de los soportes de las guías y los necesarios para anclan un máximo de 3 edificios en distintas posiciones.

Se procede de la misma manera que el mecanizado de la placa inferior con el taladro de columna, el plano se puede ver en el anexo 9.4.

Figura 4.2.2.1: Proceso de taladrado de la plataforma.

Figura 4.2.2.2: Plataforma una vez mecanizada.

4.2.3. Mecanizado del soporte del motor

Se toma una de las 2 placas de aluminio de dimensiones 80x60x20 mm que se compran. Se mecaniza un taladro central para que atraviese el husillo con el acoplamiento y este se una al eje del motor. No hay ajuste puesto que es un taladro de paso y no de sujeción. Este soporte sirve para anclar el motor a la placa de la base, el eje del motor esté lo más alineado posible al eje del husillo y no se produzcan alteraciones en el movimiento. Se compra pensando en el diseño y realizar el menor número de operaciones de mecanizado, por lo que a parte del taladro central, se realizan los agujeros de anclaje del soporte a la placa inferior y del motor a este, todas las uniones mediante tornillería.

Se puede ver en el plano del anexo 9.4.

Se mide con precisión a través del gramil la posición de cada taladro, se marcan con el granete y se taladran como se hizo en el resto de ocasiones con el taladro de columna.

Figura 4.2.3.2: Mecanizado de los taladros en el soporte del husillo.

Se realizan los roscados para el anclaje del mismo a la base, con el macho de roscar.

Figura 4.2.3.3: Mecanizado de las roscas de anclaje.

Para realizar el taladro central, al tener un diámetro más grande que las brocas convencionales, se usa el torno. Se comienza centrando la pieza en el plato de tres garras, ayudado de unos calzos y un martillo para el correcto acople. Después se procede a mecanizar con el torno el diámetro del diseño.

Figura 4.2.3.4: Centraje de la pieza en el plato de tres garras para mecanizado en el torno

Figura 4.2.3.5: Mecanización del taladro central.

4.2.4. Mecanizado del soporte del husillo

Igualmente se toma la placa de aluminio de dimensiones 80x60x20 mm y se mecaniza para realizar un apriete, donde van colocados los rodamientos de valona. Esto sirve para sustentar el husillo y para que los rodamientos absorban el movimiento axil. También se compra acorde a las medidas deseadas, por lo que se realizan los agujeros de anclaje del soporte a la placa inferior mediante tornillos y el agujero con apriete, como se indica en el plano del anexo 9.4.

Como se hizo anteriormente, se marca la posición del taladro, para el ajuste con apriete en la parte central exactamente.

Figura 4.2.4.2: Medición y marcaje con el gramil de precisión.

Para proceder a mecanizar el apriete, se centra la pieza con el plano de 3 garras en el torno que posee el laboratorio de la Escuela Politécnica, como se muestra en la imagen.

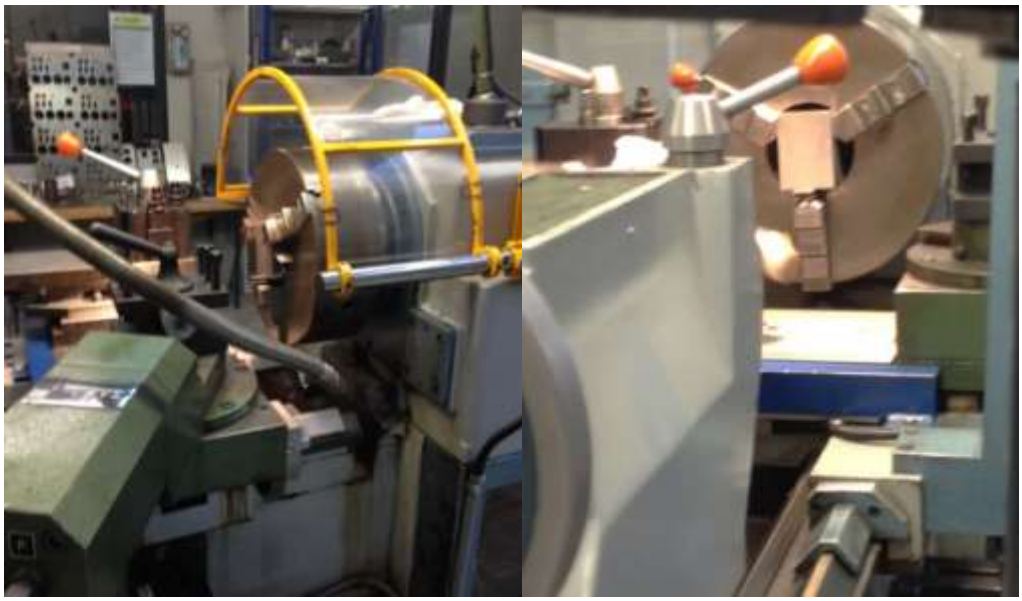


Figura 4.2.4.3: Ajuste en plano de 3 garras para torno.

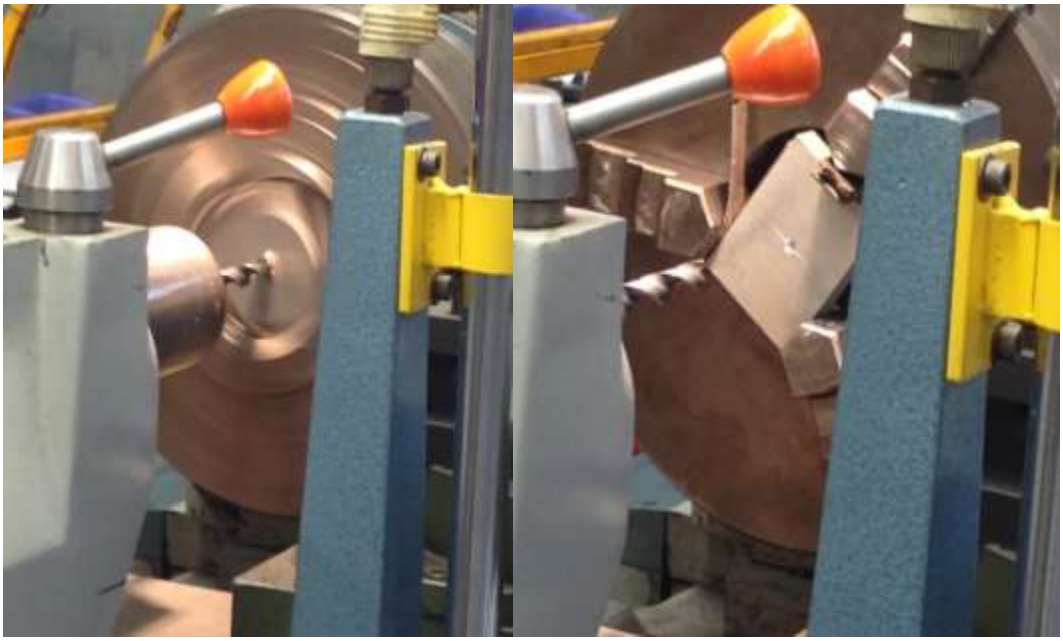


Figura 4.2.4.4: Mecanizado del taladro para el apriete con torno.

Como se observa se realiza primero un taladro de cualquier medida, para después mecanizar con más precisión puesto que se busca un determinado apriete.

Figura 4.2.4.5: Mecanizado del apriete con torno.

Se va mecanizando con precisión introduciendo los parámetros en el controlador, pero aun así habrá que medir el taladro para asegurar la máxima precisión, con el reloj comparador.

Figura 4.2.4.6: Medición con el reloj comparador.

Finalmente se realizan los taladros de la base para el anclaje, como se vio anteriormente, midiendo, marcando y mecanizando en el taladro de columna.

Figura 4.2.4.7: mecanización en el taladro de columna.

Debido a que el anclaje se realiza mediante tornillos, se procede a roscar dichos taladros con el macho de roscar.

Figura 4.2.4.8: Mecanización de las roscas.

4.2.5. Mecanizado de los calzos de los soportes de las guías

Por diseño es necesario mecanizar unos calzos precisos para conseguir la misma altura del husillo y las guías, para acoplar con el mayor alineamiento posible la placa superior. Esto se tiene en cuenta al inicio del diseño, debido a que no todos los componentes se pueden comprar de acuerdo a las especificaciones necesarias.

Por tanto se decide reutilizar un perfil rectangular de polietileno que se encuentra en el laboratorio por otros proyectos anteriores.

Son necesarios 4 calzos, por haber 4 soportes que sujetan las 2 guías. Tanto los calzos, como los soportes van acoplados a la placa inferior por tornillos, para así evitar su movimiento relativo.

Para la obtención de cada uno de estos calzos, se corta el perfil en 4 trozos mayores, que después se mecanizan para obtener las dimensiones 60x20x10 mm. Para cortar el perfil de polietileno se usa la sierra de cinta que se encuentra en el laboratorio de la Escuela Politécnica.

Figura 4.2.5.1: Corte con sierra de cinta.

Una vez que se tienen todas las piezas, se rectifican todas las caras de cada una de las piezas para obtener las dimensiones deseadas, con laterales paralelos y escuadrados. Hay que garantizar la planitud de las piezas debido a que en ellas se apoyan los soportes de las guías y estas deben estar alineadas. Para ello una vez mecanizada con la fresadora, se

miden a intervalos las caras de las piezas con el calibre, para verificar el espesor.

Figura 4.2.5.2: Fresadora.

Figura 4.2.5.3: Mecanización de refrentado de piezas con fresadora.

Figura 4.2.5.4: Pieza final refrentada.

Por último para el anclaje se realizan como en el resto de casos, los taladros indicados en el plano (anexo 9.4), con el taladro de columna.

Se procede midiendo la posición con el gramil, marcando los puntos a taladrar con el granete y finalmente perforando hasta conseguir los taladros pasantes indicados en el diseño de la pieza. Son pasantes porque donde se rosca el tornillo de sujeción es en el soporte de la guía, que vienen roscados de fábrica.

Figura 4.2.5.5: Medir y marcar la posición de los taladros con el gramil.

Figura 4.2.5.6: Taladrado de los agujeros pasantes.

Se observan las características de los calzos en el plano del anexo 9.4.

4.2.6. Mecanizado del calzo del desplazador

El calzo del desplazador es una pieza que se obtiene de igual manera del perfil de polietileno pero de dimensiones 60x20x12 mm.

Sirve para alinear perfectamente la posición superior de los bloques deslizadores de las guías y el desplazador por movimiento del husillo. De esta forma la placa superior tiene una posición lo más horizontal posible y paralela a la placa inferior, todo esto tiene como objetivo minimizar el rozamiento de las piezas en el movimiento en una dirección de la placa superior.

Para ello se procede como antes, cortando con la sierra de cinta y posteriormente fresando la pieza. Por último se realizan dos taladros pasantes para el anclaje del calzo con el desplazador y la placa superior de la mesa mediante atornillamiento de todo ello, como se indica en el plano del anexo 9.4.

4.2.7. Mecanizado de la tuerca manilla

Esta pieza se recicla de otro proyecto del laboratorio, lo único que hace falta es realizar 3 taladros para insertar 3 tornillos a forma de retén.

Sirve para mover el husillo al gusto cuando el motor no esté funcionando y así poder posicionar la mesa manualmente.

Figura 4.2.7.1: Tuerca manivela con retén.

Se realizan dichos taladras como en el resto de piezas en el taladro de columna.

Figura 4.2.7.2: Taladrado de 3 cavidades para los retenes de la tuerca manivela.

4.2.8. Montaje del conjunto mesa vibratoria

Una vez que se tiene todas las piezas mecanizadas, se procede al montaje.

Para ello se introduce a presión con ayuda de un mazo, los rodamientos de valona en el orificio realizado en el soporte del husillo.

Figura 4.2.8.1: Insertar rodamientos en el soporte husillo.

Lo único que hay que hacer es anclar las piezas en sus respectivas posiciones en la placa inferior, con el conjunto tornillo, arandela y tuerca.

Como es el caso del soporte del husillo, se coloca el husillo en el rodamiento ya acoplado en el soporte y se introduce el otro extremo a rosca en el conjunto que forma el desplazador y la tuerca retén. Esta última se ajusta a tope con el soporte.

La tuerca retén evita que el husillo se mueva axialmente de su posición.

Además en el otro extremo del husillo se coloca el acoplamiento, sin olvidar apretar los tornillos que lleva alojados. En la otra mitad del acoplamiento se ajusta el eje del motor.

Para sujetar el motor se atornilla este a su soporte y se coloca el conjunto al acoplamiento y se atornilla el soporte a la placa inferior.

Por último se coloca la tuerca que funciona como manilla y se aprietan los retenes. Tiene doble función, evitar que el husillo se mueva fuera de su posición y para poder moverle de forma manual.

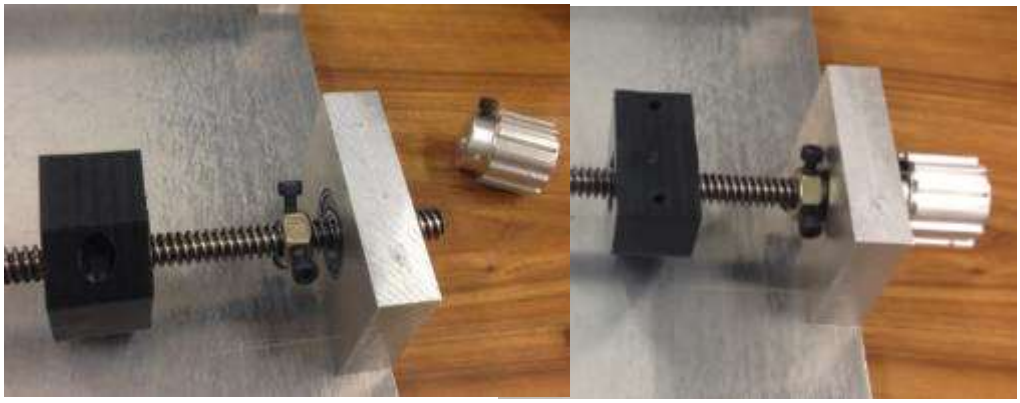


Figura 4.2.8.2: Montaje de soportes, husillo, motor y desplazador.

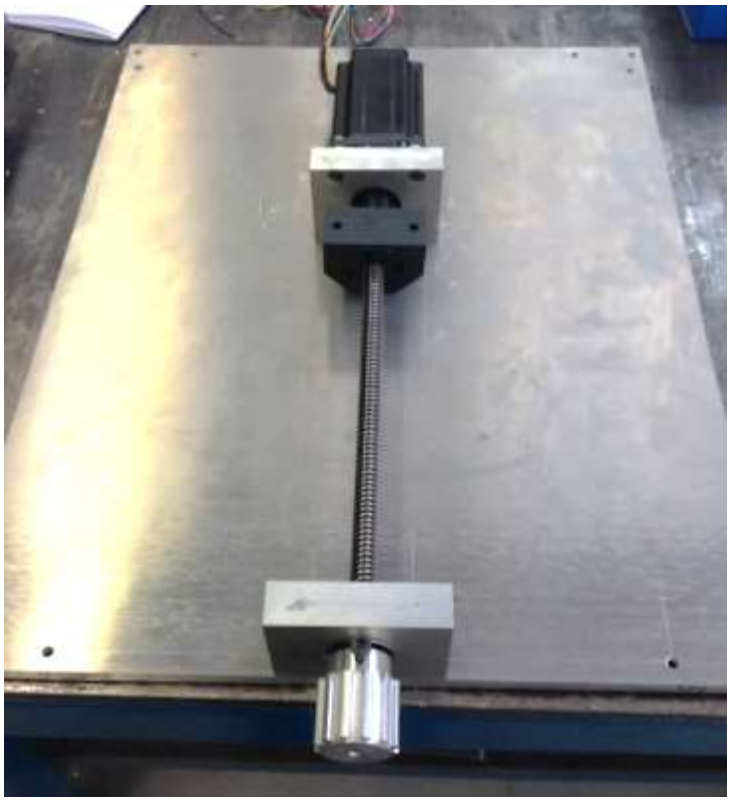


Figura 4.2.8.3: Montaje de soportes, husillo, motor y desplazador.

A continuación se atornillan los soportes de las guías en la placa inferior sin olvidar atornillar los calzos entre la placa y el soporte. Estos soportes mantienen las guías solidarias a la base. Se introducen los bloques que se desplazan en las guías, para permitir su movimiento. Y se colocan las guías en los soportes de la base, apretando los tornillos para una buena sujeción cuando la mesa esté funcionando.

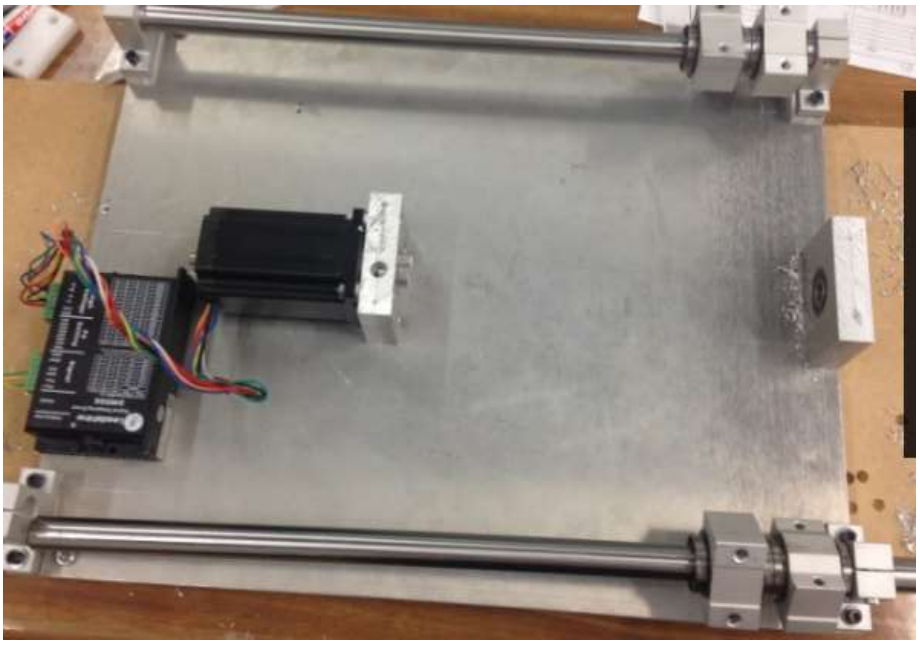


Figura 4.2.8.4: Montaje de los bloques móviles, soportes y guías.

Después, se atornillan la fuente de alimentación y el *driver* en la placa inferior para evitar que se golpeen y se acopla la placa superior a los bloques móviles y al desplazador mediante tornillos.

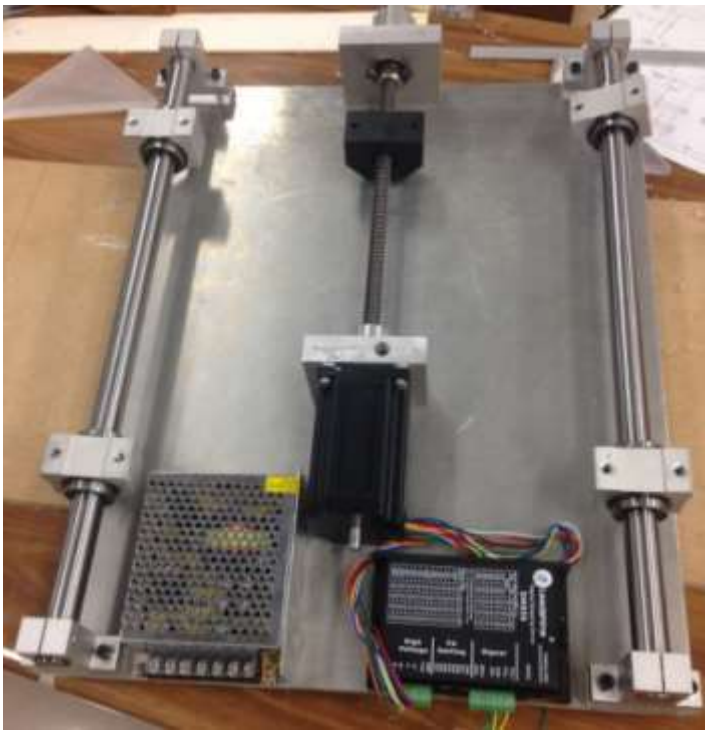


Figura 4.2.8.5: Montaje completo en placa inferior.

La parte más complicada del montaje es colocar el calzo entre el desplazador y la placa superior, debido a la escasa accesibilidad. Por ello se introducen los tornillos por la parte inferior del desplazador antes de colocar la placa superior. Además se han mecanizado con anterioridad en la placa superior 2 taladros de tamaño suficiente para introducir la llave que permite atornillar la tuerca a dichos tornillos.

Figura 4.2.8.6: Taladros placa superior para llave accesible a las tuercas.

Por último se atornillan las patas a la placa inferior para no apoyar esta directamente.

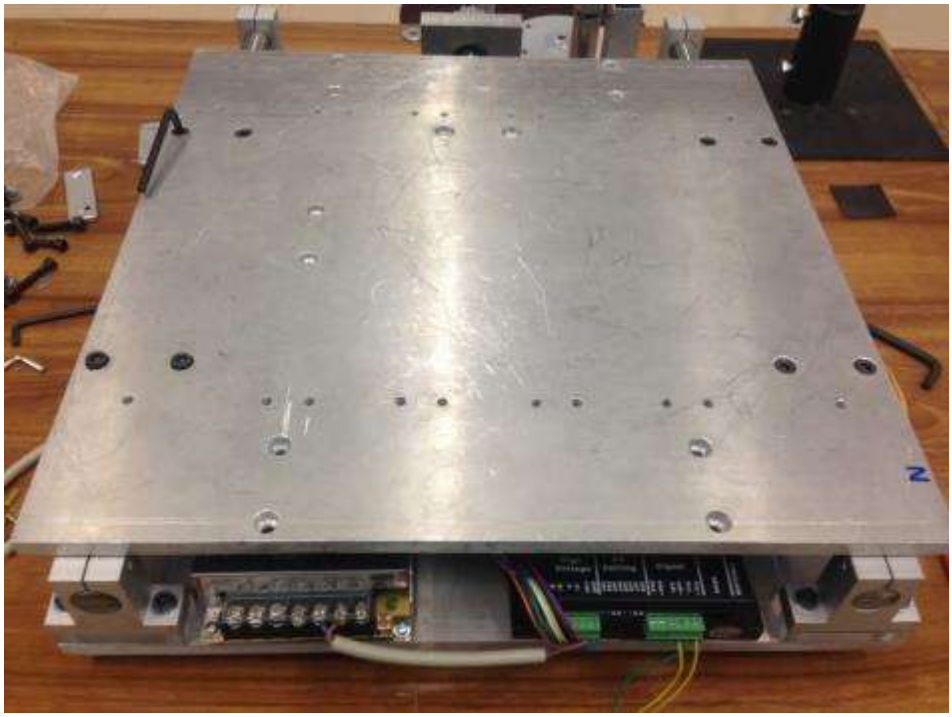


Figura 4.2.8.7: Montaje completo con placa superior.

4.2.9. Comprobación de la planitud y nivelación de la placa superior

Antes de colocar la placa superior, hay que comprobar si las superficies de los bloques de las guías y el desplazador junto con el calzo se encuentran a la misma altura, es decir que el diseño preliminar es correcto.

Para ello se hace uso de un reloj comparador digital disponible en el laboratorio de metrología de la Escuela de Ingenierías Industriales, tomando como referencia la placa inferior, posición cero.

A continuación se van midiendo los distintos bloques y el calzo del desplazador.

Figura 4.2.9.1: Medición bloque derecho posterior.

Figura 4.2.9.2: Medición bloque derecho anterior.

Figura 4.2.9.3: Medición bloque izquierdo anterior.

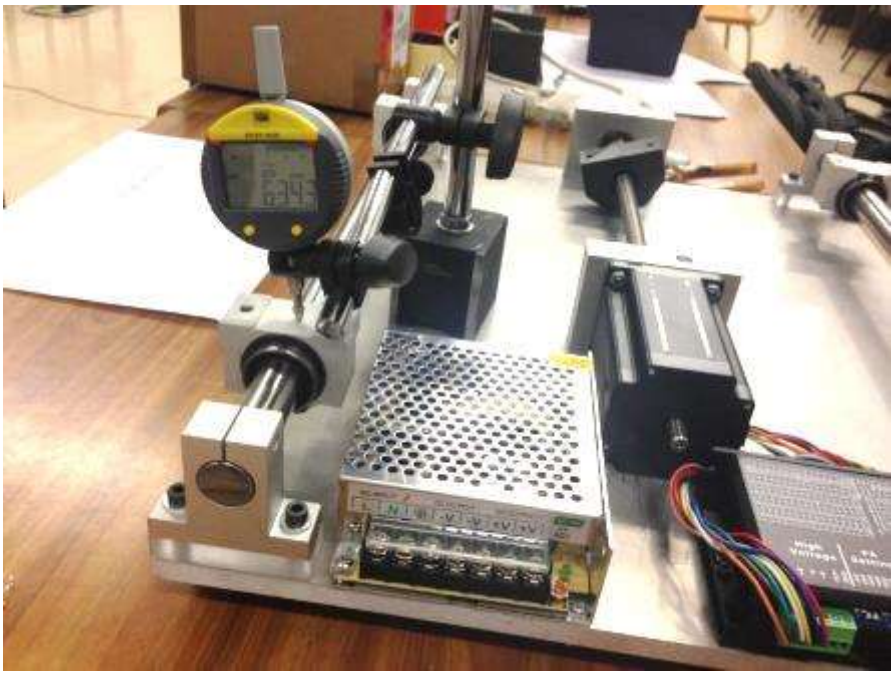


Figura 4.2.9.4: Medición bloque izquierdo posterior.

Figura 4.2.9.5: Medición desplazador y calzo.

Como se comprueba, todas las medias están razonablemente próximas, con valores entorno al 6.352 mm, por lo que el diseño de los 5 calzos es el correcto y se puede decir que la planitud y nivelación es apropiada para un resultado con el menor rozamiento posible.

4.3. Ajustes mecánicos, alineación y antirozamiento

Una vez montada la mesa vibratoria, se realizan ajustes para minimizar los rozamientos y con ello las vibraciones, estos se pueden ver reflejados en los ensayos a posteriori. Interesa que el movimiento sea preciso y lo más suave posible.

- En primer lugar se observa que el husillo gira de forma forzada y provoca excesivas tensiones en el acoplamiento que termina por romper. Esto se debe al ser de un material plástico no demasiado resistente, por lo que se sustituye por otro acoplamiento de aluminio, de las dimensiones correspondientes al eje del husillo y del motor.

- Así mismo se comprueba que el retén de plástico que trae el conjunto del husillo, no realiza de forma adecuada su función de sujetar el husillo en su posición, por lo que en movimiento, se acaba saliendo.
  Por ello se sustituye por una tuerca de M12, a la que se le realizan 3 taladros para 3 tornillos de sujeción.

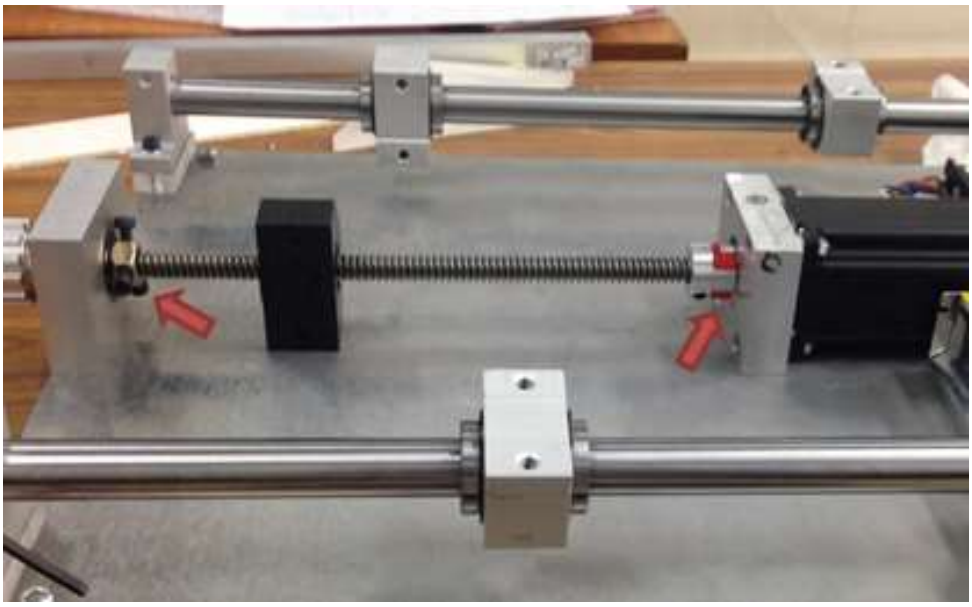


Figura 4.3.1: Sustitución de la tuerca retén y del acoplamiento.

- A la hora de sustituir se comprueba la dificultad que ocasiona desmontar todo el mecanismo, por lo que se realiza un taladro en la placa soporte del motor, que permite una mejor accesibilidad al acoplamiento.

Figura 4.3.2: Taladro en la parte superior del soporte del motor.

- Se sustituyen los perfiles de polietileno refrentado de los calzos, por otros de metacrilato de las mismas dimensiones que los iniciales, para garantizar la alineación de las guías con el husillo.
  Como se indicó anteriormente se comprueba que estas alineaciones correctas con el reloj comparador.

- Para futuros usos, se necesita que el desplazador se desacople con facilidad de la placa superior, por si no interesa usar el motor, por ello se busca la forma de acceder a los tornillos del desplazador.
  Se decide hacer un acceso, mediante un agujero en la placa inferior, para así introducir la llave allen de los tornillos, mientras se sujetan por arriba las tuercas con su llave.

Figura 4.3.3: Acceso desde la parte inferior del desplazador.

- Por último se comprueba, poniendo en funcionamiento el motor, que los rozamientos son mínimos, debido a los ajustes, es decir que el movimiento es suave con el apriete adecuado de las uniones mecánicas. Se realiza el pegado con cola especial para tornillos, para una mejor sujeción, reduciendo el ruido en las mediciones. Además se aplica lubricante a las guías, para que los bloques en su movimiento deslicen suavemente y evitando así los rozamientos de los mismos.

5. Instrumentación, control y caracterización

5.1. Descripción del motor seleccionado y sus conexiones

Se realiza el movimiento de la mesa vibratoria a través de un motor que permita tener precisión y control del movimiento, es decir un motor paso a paso. Este necesita un circuito digital para moverse, para lo que se emplea la placa Arduino Mega. Se programa con exactitud los pasos que se va a mover el motor, conocido el ángulo de giro del mismo. También se controla la velocidad a la que se da cada paso, decidiendo en cada momento sobre el movimiento de la mesa.

Este tipo de motores poseen gran fuerza de empuje a bajas velocidades, que disminuye según se aumenta la velocidad. Esto se aprecia en las curvas de par en función de la conexión que se realice al motor [Ref. Libro 10]:

- Unipolar: el par es intermedio, cuando va aumentando el par se disminuye un poco la velocidad según las curvas.
- Bipolar en serie: par alto pero disminuye muy rápido la curva, es decir disminuye el par al aumentar la velocidad. Se utiliza en aplicaciones en las que se requiere un par mayor a menores velocidades.
- Bipolar en paralelo: par bajo pero se mantiene la curva bastante uniforme al variar la velocidad. Ofrece un momento de torsión más estable, pero inferior a velocidades más bajas. Debido a la menor inductancia, hay un mayor par a velocidades más altas.

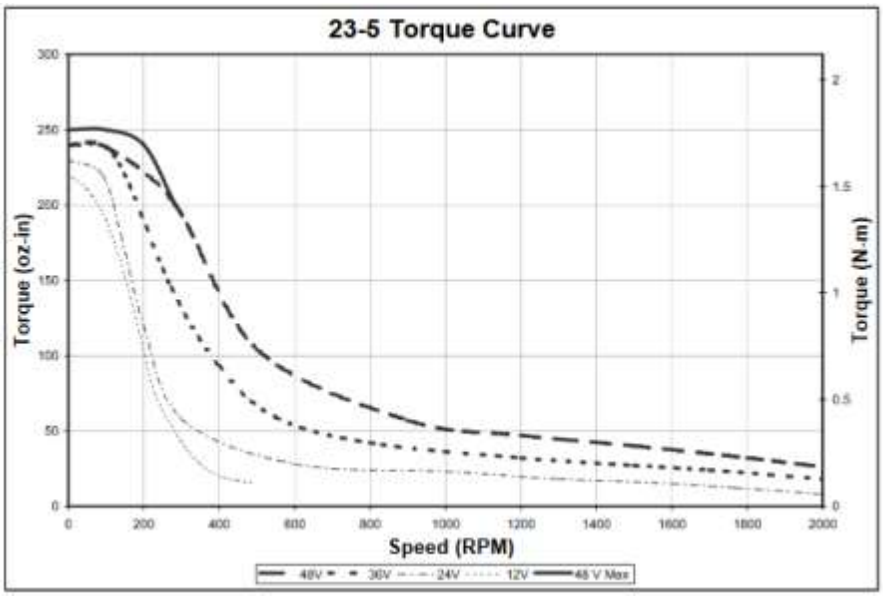
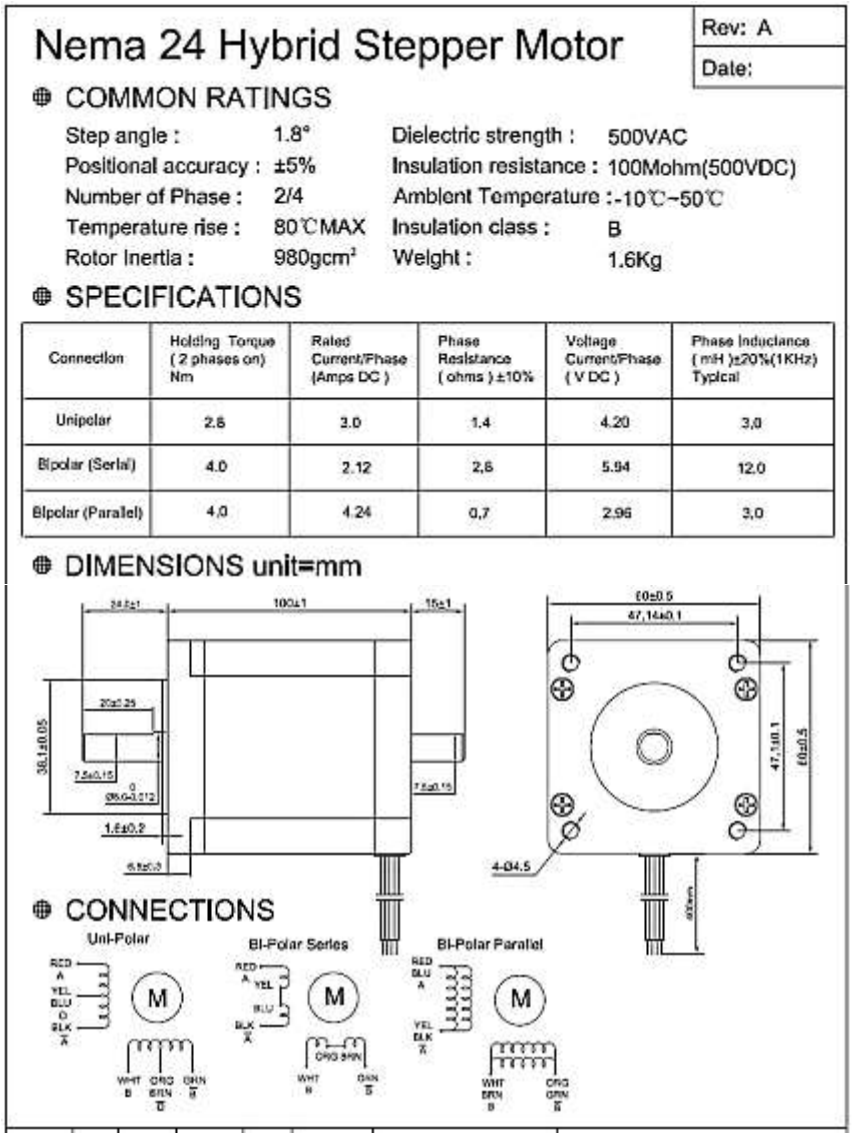


Figura 5.1.1: Gráfica curva par-velocidad motor paso a paso *NEMA* 23HS2430 [Ref. Libro 10].

Las especificaciones del motor son las siguientes [Ref. Web 5]:



Nema 24 Hybrid Stepper Motor

Rev: A

Date:

COMMON RATINGS

Step angle :	1.8°	Dielectric strength :	500VAC
Positional accuracy :	±5%	Insulation resistance :	100Mohm(500VDC)
Number of Phase :	2/4	Ambient Temperature :	-10℃~50℃
Temperature rise :	80℃MAX	Insulation class :	B
Rotor Inertia :	980gcm²	Weight :	1.6Kg

SPECIFICATIONS

Connection	Holding Torque (2 phases on) Nm	Rated Current/Phase (Amps DC)	Phase Resistance (ohms) ±10%	Voltage Current/Phase (V DC)	Phase Inductance (mH)±20%(1KHz) Typical
Unipolar	2.8	3.0	1.4	4.20	3.0
Bipolar (Serial)	4.0	2.12	2.8	5.94	12.0
Bipolar (Parallel)	4.0	4.24	0.7	2.96	3.0

DIMENSIONS unit=mm

CONNECTIONS

Uni-Polar

Bi-Polar Series

Bi-Polar Parallel

Figura 5.1.2: Especificaciones *NEMA* 23HS2430 [Ref. Web 5].

En este caso se utiliza una conexión en bipolar en paralelo, que proporciona un par considerable para el peso que se va a mover, unos 5 kg, manteniendo la curva bastante uniforme al variar las velocidades [Ref. Libro 10].

Hay que tener en cuenta que los motores paso a paso consumen bastante energía, en comparación a los eléctricos. Lo que significa consumir el máximo de amperios cuando están parados y con ello produce que se calienten bastante. Por lo que es conveniente desenchufar la alimentación cuando no esté en uso.

Además el motor elegido es de 8 cables en conexión bipolar paralela por lo que para dar una vuelta completa, tiene que recorrer 200 pasos, como referencia

$$\frac{360\,^{\circ}/_{rev}}{1.8\,^{\circ}/_{step}} = 200\ step/rev$$

Es el *driver* el que se encarga de invertir la polaridad de la corriente durante el movimiento, por tratarse de una conexión bipolar, consiguiendo mayor potencia. El número de pasos por revolución se puede ajustar con el driver.

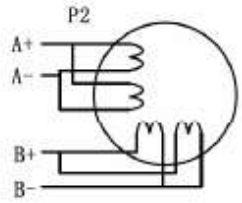


Figura 5.1.3: Conexión en paralelo motor-*driver* [Ref. Libro 11].

En realidad el *driver* envía micropasos, con lo que no se da la totalidad de la fuerza al motor y así este no completa los pasos al 100 %. Con esto se pierde algo de par, pero se gana resolución.

Se conecta el motor al *driver*, con conexión bipolar en paralelo, para ello se toman las especificaciones del fabricante de la figura 5.1.2, antes indicada [Ref. Libro 10].

- A + → Cables rojo y azul.
- A - → Cables amarillo y negro.
- B + → Cables marrón y blanco.
- B - → Cables verde y naranja.

Y el *driver* a la fuente de alimentación:

- GND → Cables azul a V - .
- +Vdc - → Cables marrón a V +.

De esta forma a través de la fuente de alimentación, se introducen 24 V en el *driver* (*High Voltage*), que regula la corriente transformándola a 5 V, que va a parar al motor, así se evita quemarle.

Por otra parte el *driver* consta de otra sección (*Signal*) que proporciona 2 mV, que transfiere a Arduino y la *protoboard* a través de los pines.

- PUL + → *Protoboard* y 5V Arduino.
- PUL - → Pin 7 Arduino, controlar los pasos del motor.
- DIR + → *Protoboard* y 5V Arduino.
- DIR - → Pin 9 Arduino, controlar la dirección del motor.

Con estas conexiones proporcionan el circuito necesario para controlar el motor a través de Arduino [Ref. Web 9].

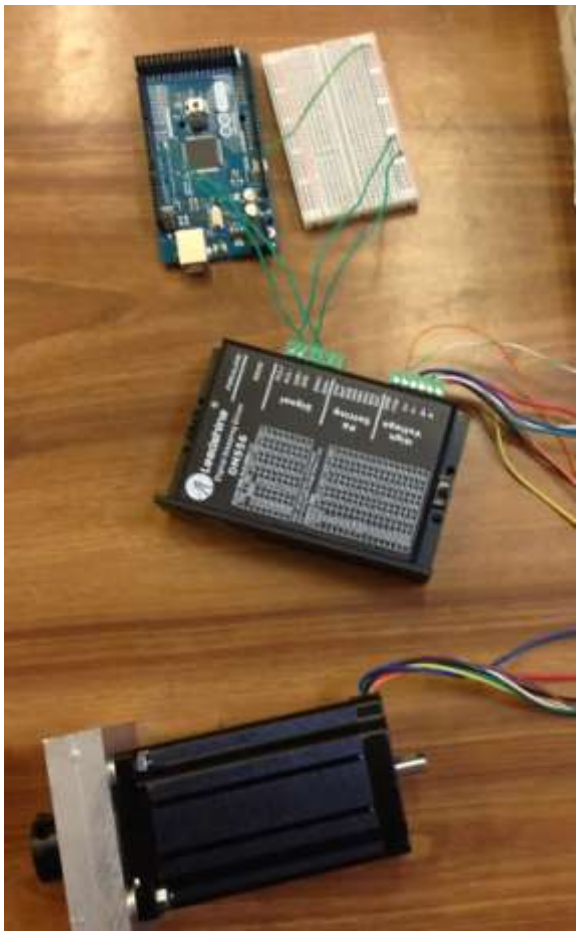


Figura 5.1.4: Conexión motor/driver/*protoboard*/Arduino.

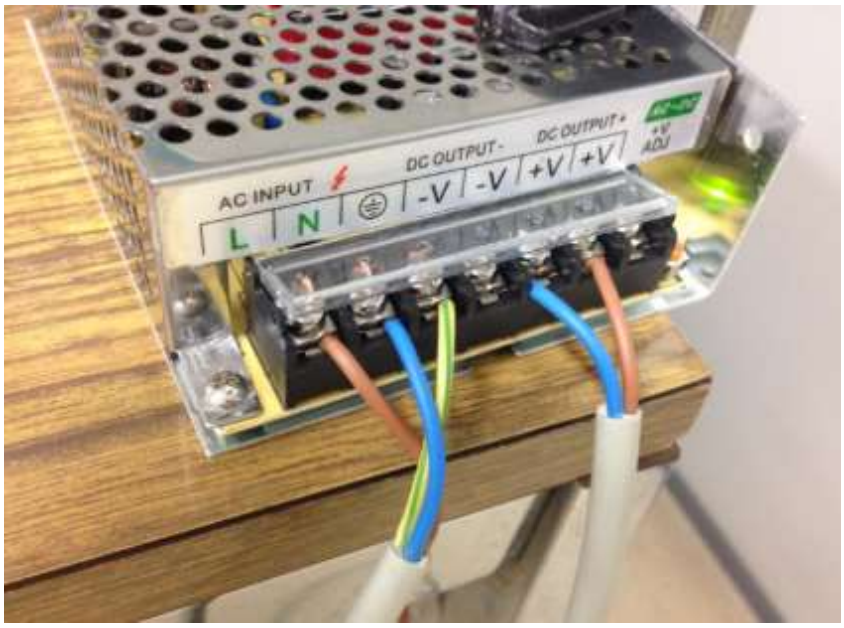


Figura 5.1.5: Conexión fuente de alimentación/*driver.*

5.2. Programa de control del motor

El objetivo del control con Arduino es programar un tren de pulsos para representar una función de fuerza (Amplitud= F_0= nº de pasos), en forma de función seno, para ello se valoraron varias opciones.

$$F = F_0 \cdot \sin(\omega \cdot t) = F_0 \cdot \sin(2\pi \cdot f \cdot t)$$

Siendo: $\omega = \frac{rad}{s}$ y $f = Hz$ (tiempo entre cada paso)

Como primera opción para programar en Arduino una función senoidal [Ref. Web 9], se usa el comando "*analogWrite()*" → PWM (*Pulse Width Modulator*).

Con este comando se consigue crear un onda cuadrada, conmutando el estado de encendido y apagado, con tensiones de 0 a 5 V, apropiado para el motor. La duración entre un estado y otro, es decir el tiempo, se denomina ancho de pulso.

A continuación se representa en la figura los distintos posibles estados que se pueden obtener:

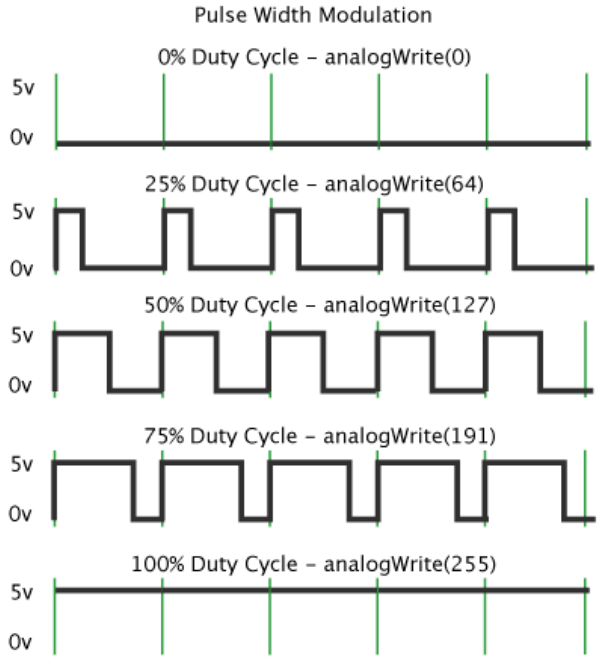


Figura 5.2.1: Representación gráfica del ancho de pulsos de la función "*analogWrite ()*" [Ref. Web 9].

Por tanto el comando "*analogWrite ()*" escribe un valor analógico (onda PWM) en un pin. La frecuencia de la señal PWM en la mayoría de los pin es de aproximadamente 490 Hz. En el Arduino Mega, funciona en los pines 2 - 13 y 44 - 46.
El tutorial de Arduino indica que no es necesario llamar a "*pinMode ()*" para establecer el pin como salida antes de llamar "*analogWrite ()*".
Y que es bueno usar los pies 5 y 6 porque tendrán ciclos de trabajo más altos de lo esperado.
Como en realidad se usan los pines digitales, se aplica el comando "*digitalWrite()*", de la misma manera para crear un tren de pulsos [Ref. Web 9].

El programa realizado para este caso se muestra en el anexo 9.6.1.

Se observa que dicho comando no representa con aproximación a la función senoidal deseada, por lo que se descarta y se busca otra opción.

La siguiente opción que se prueba es el comando "*Tone ()*". Este genera también una onda cuadra, especificando un tiempo, porque si no hay que hacer una llamada a la función "*noTone ()*". Sólo se puede generar un pulso de cada llamada al comando para reproducirlo en un pin.

El tutorial de Arduino indica que la función "*Tone ()*" interfiere con la salida PWM en los pines 3 y 11 [Ref. Web 9].

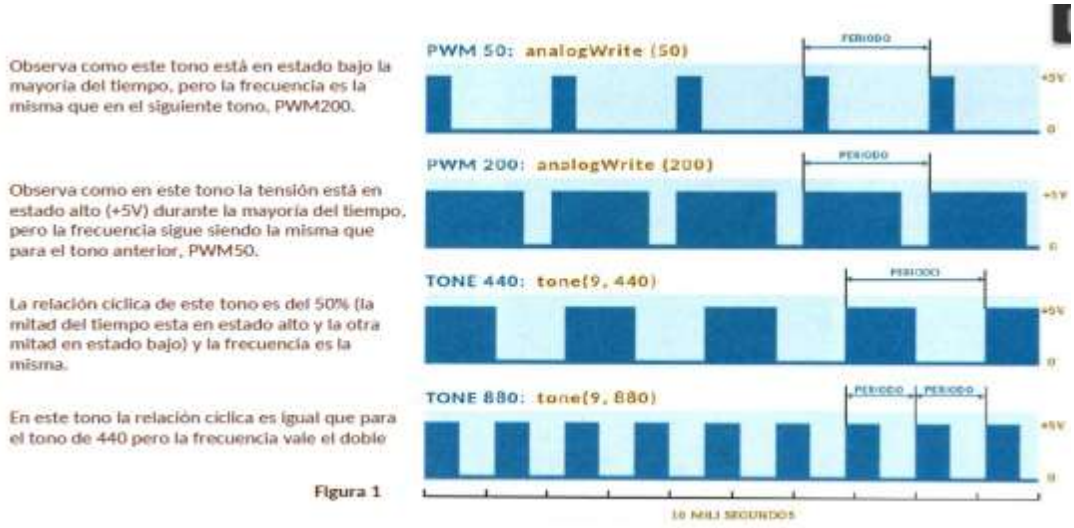


Figura 5.2.2: Representación gráfica del ancho de pulsos de la función "*Tone ()*" [Ref. Web 9].

El programa realizado para este caso se muestra en el anexo 9.6.2.

Se observa al igual que antes que no se representa adecuadamente la función senoidal deseada, por lo que se descarta también.

Finalmente se decide realizar la función senoidal mediante la división de esta en partes, que son los tiempos entre cada paso del motor.

Es decir se representa una función seno, cuyos parámetros son los pasos que da el motor y el tiempo que debe esperar entre cada paso.

Además el tiempo se introduce en forma de frecuencia, por tanto hay que calcular dicha frecuencia en función de los pasos por revolución y el tiempo empleado para ello. Sin olvidar que Arduino trabaja en milisegundos o microsegundos.

Como la función seno está formada por dos partes simétricas, se representa la subida y la bajada (un periodo), teniendo en cuenta para calcular la frecuencia, los pasos y el tiempo (ecuación 5.2.1):

$$f = \frac{10^6}{2 * pasos * 2 * tiempo} \qquad (5.2.1)$$

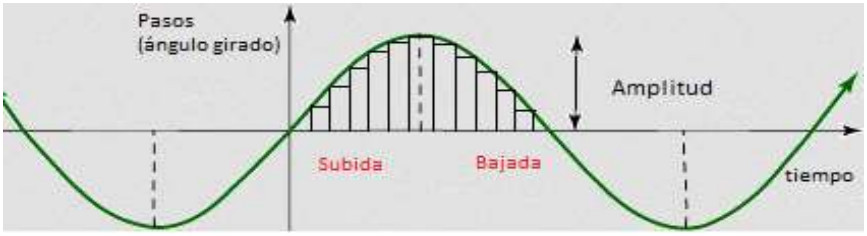


Figura 5.2.3: Representación gráfica de la función seno dividida en sección. Onda representativa del funcionamiento del motor

Una subida de la función da 2 veces los pasos del motor, de la zona por debajo de cero y por encima. Los pasos dados por el motor son números enteros (Npasos):

$$p = [0,1,2,\dots, Npasos-1, Npasos, Npasos+1, 2\cdot Npasos] = amplitud$$

Para ello se realizan varias pruebas hasta llegar a una buena aproximación de la función, creando un bucle que recorra un vector donde se calculan los tiempos a partir del coseno, para que empiece en subida desde la amplitud negativa.

Cada paso se da en un instante determinado, por tanto para obtener el vector de tiempos, conocidos un número de pasos entre cada ciclo (divisiones), y la frecuencia de oscilación (ecuación 5.2.1), se resuelve la ecuación del sistema de 1 g.d.l para pequeñas oscilaciones (ecuación 5.2.2), ecuación del movimiento lineal [Ref. Libro 1]:

$$posición\ (i) = A\cdot\cos(w_n * t) \qquad (5.2.2)$$

$$\rightarrow\ t = \cos^{-1}\left(\frac{posición\ (i)}{A}\right)\cdot\frac{1}{w_n}$$

$$\rightarrow\ t = \cos^{-1}\left(\frac{posición\ (i)}{número\ de\ pasos}\right)\cdot\frac{1}{2\pi\cdot f}$$

Donde f es la frecuencia de oscilación, en Hercios y el tiempo se obtiene en segundos, puesto que se trabaja con la función "*delayMicroseconds()"* [Ref. Web 9], el vector de intervalos de tiempos (ecuación 5.2.3), se calcula en microsegundos, multiplicando por 10^6 y en el intervalo [1, -1].

$$T[i] = 10^6\left[\cos^{-1}\left(\frac{1-2(i+1)}{Npasos}\right) - \cos^{-1}\left(\frac{(1-2i)}{Npasos}\right)\right]\frac{1}{2\pi\cdot f} \qquad (5.2.3)$$

Que la posición vaya multiplicada por 2 se debe a que la onda es simétrica.

Además, hay que recordar que el vector de pasos es una ristra de números enteros, desde 0 hasta Npasos, de forma que posición i y posición i+1 coinciden exactamente con el valor de i y de i+1 respectivamente, si esta empieza en 0 y termina en Npasos − 1, cosa que es habitual en los bucles "*for*" [Ref. Web 9]:

for (int i = 0; i < Npasos; i++)

El programa obtenido es el siguiente, el resto de pruebas se pueden ver en el anexo 9.6.3 y 9.6.4:

```
#include<math.h>  // Biblioteca funciones matemáticas


const int PUL = 7; // Pin de pulsos que da el motor

const int DIR = 9; // Pin de dirección del motor


int direccion=0;  // Direccion inicial

const int Npasos = 300; // Amplitud, en número de pasos

float freq = 1; // Frecuencia en Hz

float period = 1/freq;  // Periodo en segundos

int iT[Npasos]; // Vector iT, tiempo entre t1 y t2

int x=0; // Valor del vector en cada posicion


void setup() {

 pinMode(PUL,OUTPUT); // Defina el pin 7 como salida

 pinMode(DIR,OUTPUT); //Defina el pin 9 como salida


 digitalWrite(PUL,LOW); // No se mueve

 digitalWrite(DIR,LOW); // Indica una dirección DIR = 0
```

```
 Serial.begin(9600); // Comienzo la comunicacon en serie con el
ordenador

 Serial.println("Comienza"); // Mensaje por pantalla


  //Creación del vector:

  for(int i=0; i<Npasos; i++) {

    iT[i] = 1e6*(acos(1.0-(i+1)*2.0/Npasos) - acos(1.0-
i*2.0/Npasos))/(2*M_PI*freq); // Microsegundos


    //no pongo round() porque he inicializado la variable iT como
entero (int), y Arduino hace la conversión de tipos directamente
truncando

    Serial.println(iT[i]);

  }

}


void loop() {

  sube(Npasos, iT);

  cambia();

}


void sube(int paso, int t[]) {

  for(int j=0; j<paso; j++) {

    digitalWrite (PUL,HIGH);

    delayMicroseconds(t[j]/2);

    digitalWrite (PUL,LOW);

    delayMicroseconds(t[j]/2);

  }

}

}
```

```
void cambia () {
  if(direccion==0){
    digitalWrite(DIR,HIGH);
    direccion=1;
  }

 else {
    digitalWrite(DIR,LOW);
    direccion=0;
 }
}
```

5.3. Optimización del programa e instalación de finales de carrera

Las variables de tipo "*int*" proporcionan al dato 16 *bits* (2 *bytes*) de espacio en memoria. Si se cambia por *byte* ocupa exactamente 1 *byte*, es decir la mitad del espacio, por tanto se declaran las variables como "*const byte*" [Ref. Web 9].

Por otra parte cuando se define la dirección inicial de movimiento de la mesa, la variable en lugar de declarar como "*int*" se declara como "*boolean*", este solo ocupa 1 *bit*, pudiendo valer *TRUE* (valor 1) o *FALSE* (valor 0). Esto ocurre también en las variables para los 3 finales de carrera, con "*boolean*" ocuparán 16 veces menos que con "*int*" [Ref. Web 9].

Se ve que la variable tiempo, únicamente se usa en el bucle de nominado mueve, donde se declara como "*int*", lo mismo se hace en la definición de la variable, en lugar de *"float*" [Ref. Web 9].

No es necesario declarar una variable x para la posición de cada punto del vector, por lo que se suprime [Ref. Web 9].

Además de poder optimizar el programa, se ha visto que para evitar choques de la mesa en los extremos durante su carrera, son necesarios unos finales de carrera tipo pulsador.
Para ello se instalan a una distancia aproximada de 20 mm de los extremos dichos pulsadores, para que en contacto con los bloques de las guías en su movimiento choquen y provoquen por programación, la detección del motor y el posterior retroceso hasta la posición central de la carrera. Con esto último se ve la necesidad de instalar un sensor de tipo magnético en la posición central, en lugar de un tercer pulsador, para que no interrumpa el recorrido normal.

Los finales de carrera tipo pulsador se encuentran en el laboratorio, por trabajos anteriores, así que se aprovechan. Dichos pulsadores se instalan anclados a la mesa con pletinas de prototipaje rápido. Se conectan a Arduino utilizando solo dos de las tres patillas. Es decir se instala normalmente abierto y cuando se pulsa, mandan el impulso a Arduino que produce la parada del motor.

La patilla central va a la alimentación de Arduino de 5 V, la patilla de la izquierda va conectada al pin de entrada en Arduino (pulsado en estado normalmente abierto) y la patilla de la derecha es aquella para el pulsador en estado normalmente cerrado, esta no se conecta.
Como este tipo de pulsadores solo admiten 2,2 V de tensión para 0.02 A de corriente, se instala una resistencia de aproximadamente 130 Ω.

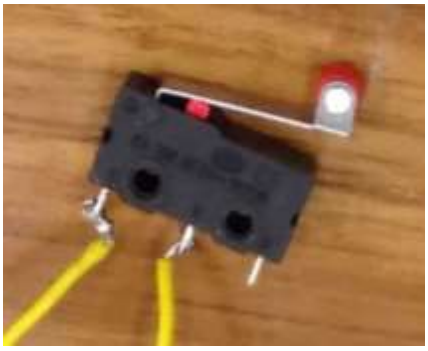


Figura 5.3.1: Final de carrera tipo pulsador.

Como se comprueba que se obtienen buenos resultados al usarlos, se instalan de forma permanente, soldando los cables con estaño a las patillas del pulsador, para asegurar el contacto.

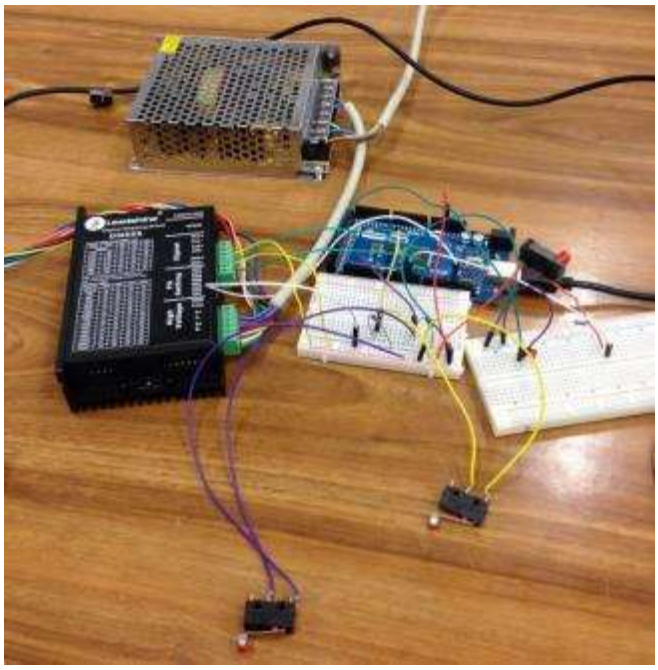


Figura 5.3.2: Proceso de instalación y comprobación del correcto funcionamiento de los finales de carrera.

Como se menciona, se utiliza para el posicionamiento de la mesa un sensor. Se tienen a disposición del laboratorio 3 tipos, que se analizan a continuación:

- Sensor de efecto *hall*, modelo: *55075 Stainless Steel M12 Geartooth Speed Sensor* [Ref Web16].

  Como se puede ver en la ficha técnica, se indica el diagrama de bloques para su correcta conexión: cable rojo tensión, cable negro tierra y cable azul al pin de Arduino.

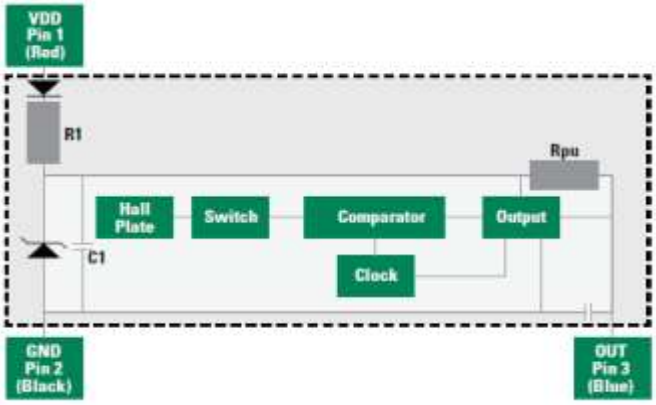


Figura 5.3.3: Diagrama de bloques del sensor de efecto hall [Ref. Web 16].

Dicho sensor tiene una gran sensibilidad, por lo que se obtienen muy buenos resultados, pero se descarta por su fragilidad y elevado coste, en el caso de necesitar un remplazo.

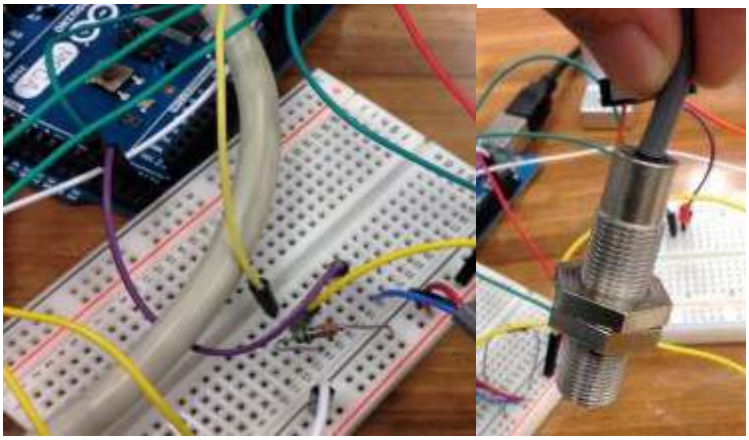


Figura 5.3.4: Conexión con Arduino del sensor de efecto hall.

- Detector magnético: *Serie D-M9P(V)* [Ref. Web 17]

  De igual manera la ficha técnica indica la forma de hacer la conexión del detector magnético. Cuando el imán o el componente conductor, está dentro del campo magnético, el indicador luminoso se enciende.
  Se conecta el cable marrón a tensión, el cable azul a tierra y el cable negro al pin de Arduino, sin olvidar en este último añadir una resistencia de 10 kΩ.

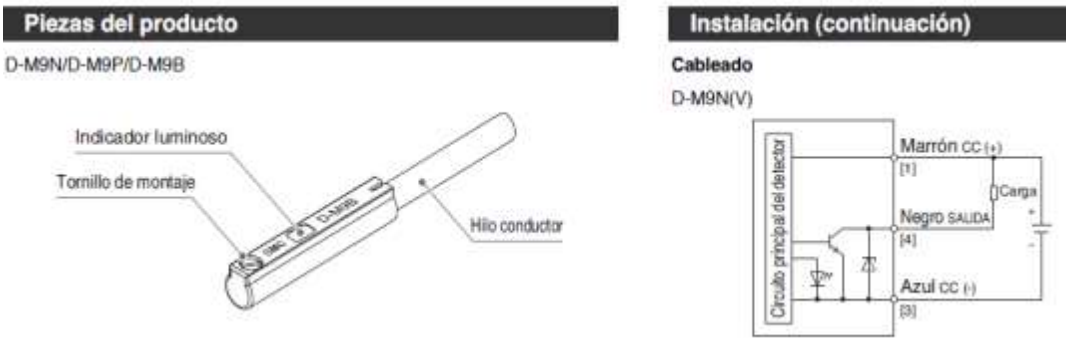


Figura 5.3.5: Diagrama de conexiones del detector magnético D-M9P(V) [Ref. Web 17].

No es tan sensible como el sensor anterior pero puede servir para la función de detener el motor.

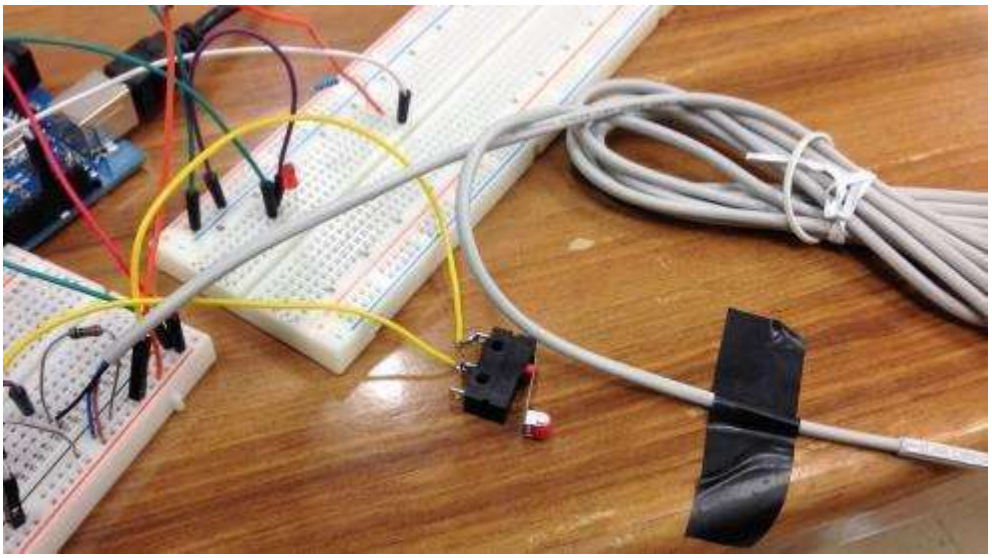


Figura 5.3.6: Conexión con Arduino del detector magnético.

- Detector magnético: *Reed Switch/Band Mounting D-C73* [Ref. Web 18]

  Por último el detector magnético D-C73, como se observa en el esquema, se conecta como el detector anterior, cable azul a tensión y cable marrón al pin de Arduino con su correspondiente resistencia de 10 kΩ.

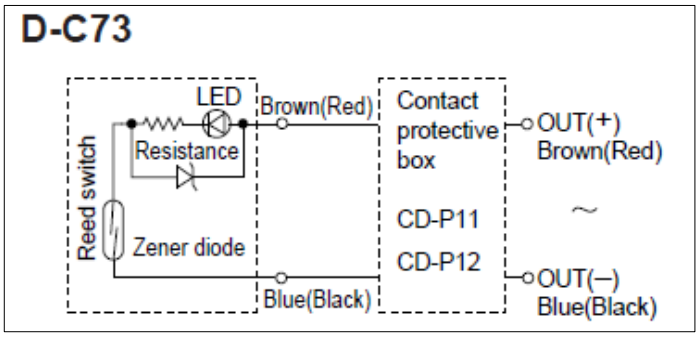


Figura 5.3.7: Diagrama de conexiones del detector magnético D-C73 [Ref. Web 18].

Se decide usar este detector por tener una buena respuesta y ser el más económico de los tres vistos.

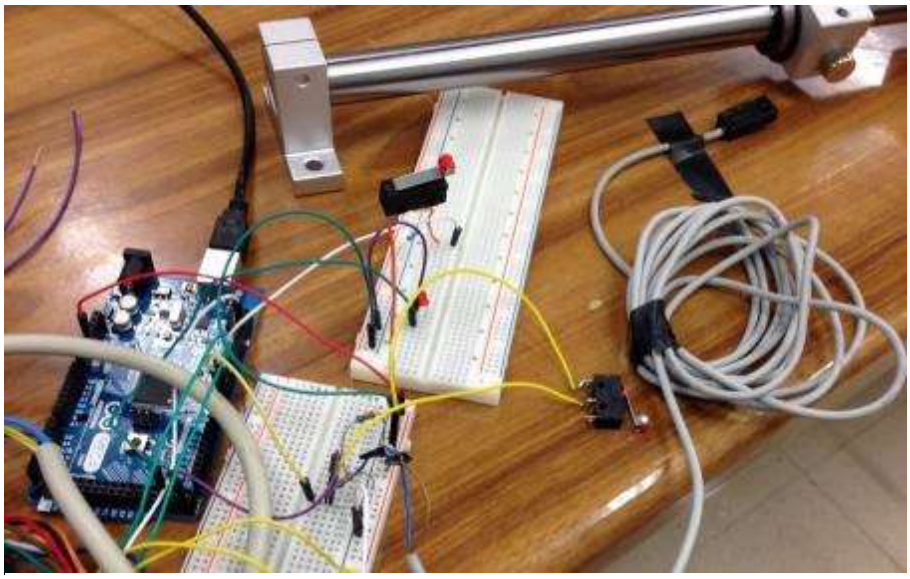


Figura 5.3.8: Conexión con Arduino del detector magnético.

A continuación se muestra el programa completo en Arduino, con las optimizaciones y el uso de los 2 pulsadores (finales de carrera) y el detector magnético seleccionado, es el programa final a usar [Ref. Web 9].

```
#include<math.h>  // Biblioteca funciones matematicas

const byte PUL = 7; // Pin de pulsos que da el motor
const byte DIR = 12; // Pin de dirección del motor
const byte buttonPin = 2; // Pin final de carrera
const byte buttonPin1 = 4; // Pin final de carrera
const byte buttonPin2 = 3; // Pin final de carrera
const byte LED=26; // Pin para led
boolean direccion=0;  // Direccion inicial hacia la derecha
const byte Npasos = 2000; // Amplitud, en número de pasos
float freq = 1; // Frecuencia en Hz
int tiempo =300;
const byte Pasos=1000;
float period = 1/freq;  // Periodo en segundos
```

```
int iT[Npasos]; // Vector iT, tiempo entre t1 y t2

boolean buttonState = 0;

boolean buttonState1 = 0;

boolean buttonState2 = 0;


void setup() {

 pinMode(PUL,OUTPUT); // Defina el pin 7 como salida

 pinMode(DIR,OUTPUT); //Defina el pin 12 como salida

 pinMode(buttonPin, INPUT); //Defina el pin 2 como entrada

 pinMode(buttonPin1, INPUT); //Defina el pin 4 como entrada

 pinMode(buttonPin2, INPUT); //Defina el pin 3 como entrada

 pinMode(LED, OUTPUT); //Defina el pin 26 como salida

digitalWrite(PUL,LOW); // No se mueve

 if(direccion == 1)

  digitalWrite(DIR,HIGH); // Indica una dirección DIR = 0

 else

  digitalWrite(DIR,LOW); // Indica una dirección DIR = 0

 Serial.begin(9600); // Comienzo la comunicacon en serie con el
ordenador

 Serial.println("Comienza"); // Mensaje por pantalla


  //Creación del vector:

  for(int i=0; i<Npasos; i++) {

    iT[i]    =    1e6*(acos(1.0-(i+1)*2.0/Npasos)    -    acos(1.0-
i*2.0/Npasos))/(2*M_PI*freq); // Microsegundos

    // no pongo round() porque he inicializado la variable iT
como entero (int), y Arduino hace la conversión de tipos
directamente truncando

    //Serial.println(iT[i]);

  }
```

```
}


void loop() {
 buttonState = digitalRead(buttonPin); // Leer el final de
carrera
 buttonState2 = digitalRead(buttonPin2); // Leer el final de
carrera


  if (buttonState == HIGH || buttonState2==HIGH) {
    digitalWrite(PUL, LOW);
    Serial.println("Final de carrera");
    delay(3000);
    Serial.println("Cambia de direccion");
    if(buttonState == HIGH)
      digitalWrite(DIR,HIGH);
    else
      digitalWrite(DIR,LOW);
    Serial.println(digitalRead(buttonPin1));


    while (digitalRead(buttonPin1) == LOW) {
     digitalWrite(LED,HIGH);
     Serial.println(buttonState1);
     mueve(Pasos, tiempo);
     digitalWrite(LED,LOW);
    }
  }
  else {
    sube(Npasos, iT);
    cambia();
```

```
  }
}
void sube(int paso, int t[]) {
  for(int j=0; j<paso; j++) {
    digitalWrite (PUL,HIGH);
    delayMicroseconds(t[j]/2);
    digitalWrite (PUL,LOW);
    delayMicroseconds(t[j]/2);
  }
}
void cambia() {
  if(direccion==0){
    digitalWrite(DIR,HIGH);
    direccion=1;
  }
 else {
    digitalWrite(DIR,LOW);
    direccion=0;
 }
}
void mueve(int pasos, int tiemp) {
  for(int j=0; j<pasos; j++) {
    digitalWrite (PUL,HIGH);
    delayMicroseconds(tiemp/2);
    digitalWrite (PUL,LOW);
    delayMicroseconds(tiemp/2);
  }
}
```

5.4. Acoplamiento de panel para controladores

En un inicio se diseña la mesa con espacio suficiente para que todas las conexiones queden ocultas entre la plataforma superior y la base inferior. Pero una vez que se monta la mesa, se decide acoplar un panel de metacrilato exterior, que permita un prototipaje rápido y fácil, para colocar en ellas cuantos elementos de control se deseen. Permitiendo así hacer mejoras de forma continua al prototipaje de los elementos de control. Dejando abierta la línea futura para la instalación de más controladores.

Se aprovechan los tornillos de sujeción de los soportes de las guías, para acoplar el panel, de dimensiones 500x80x5 mm, mediante pletinas taladradas de prototipaje rápido.

Figura 5.4.1: Instalación del panel.

Permitiendo así desmontar rápidamente el panel para acoplar la placa de Arduino, la *protoboard*, y los finales de carrera.

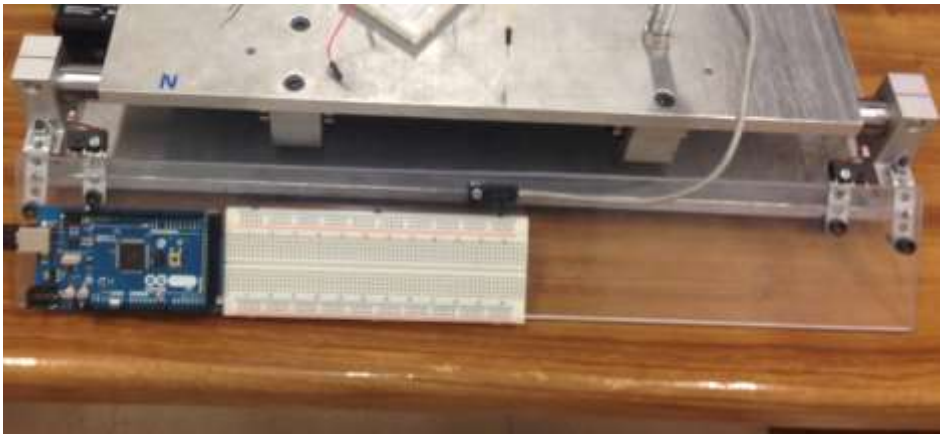


Figura 5.4.2: Controladores acoplados en el panel.

Se plantea la idea de aprovechar componentes disponibles en el laboratorio e instalar una pantalla (*display* LCD) que muestre si la mesa está en modo automático, movimiento senoidal, o manual para el posicionamiento de la mesa en el punto central de la carrera. Estos dos estados (automático/manual) se controlan con un conmutador y para el movimiento manual se instalan dos pulsadores en la *protoboard*.

Además se muestra la amplitud de movimiento y la frecuencia por pantalla, con ello se puede conocer la carrera y la velocidad de movimiento senoidal. A partir de esto, también se plantea la idea de colocar dos potenciómetros para regular dichas amplitud y frecuencia, lo que posibilita un control bastante preciso sin variar el programa generado en Arduino [Ref. Web 9].

Rápidamente se descarta instalar el potenciómetro para la amplitud, puesto que supone una gran complejidad a nivel de programación en Arduino, debido a que la amplitud aparece en el denominador del vector de tiempos designado como el número de pasos, plantear como línea futura de mejora.

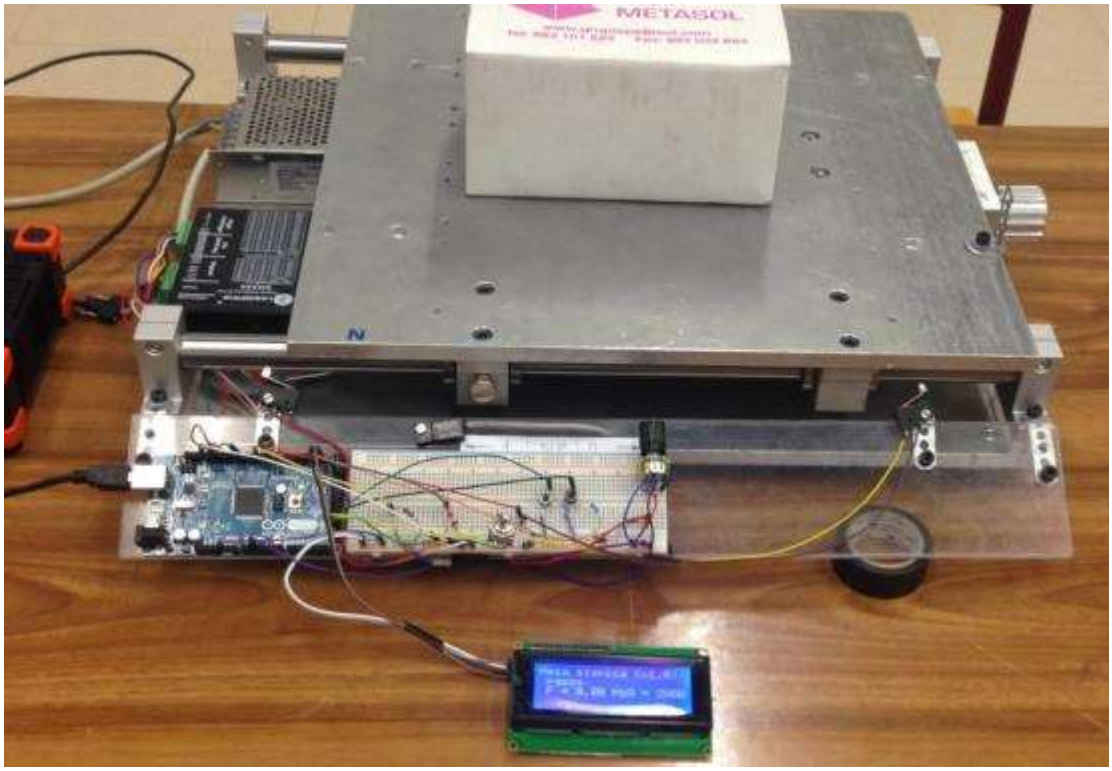


Figura 5.4.3: Pantalla LCD, potenciómetro para regular frecuencia, pulsadores y conmutador.

5.4.1. Instalación del *display* LCD

Se comienza conectando la pantalla, LCD de 20x4 segmentos controlado por el bus de datos I^2C, a los pines SCL, SDA y alimentación de Arduino. También se puede conectar con Arduino a través de los 16 pines disponibles en la pantalla, pero ocupa muchos pines de Arduino para otras conexiones.

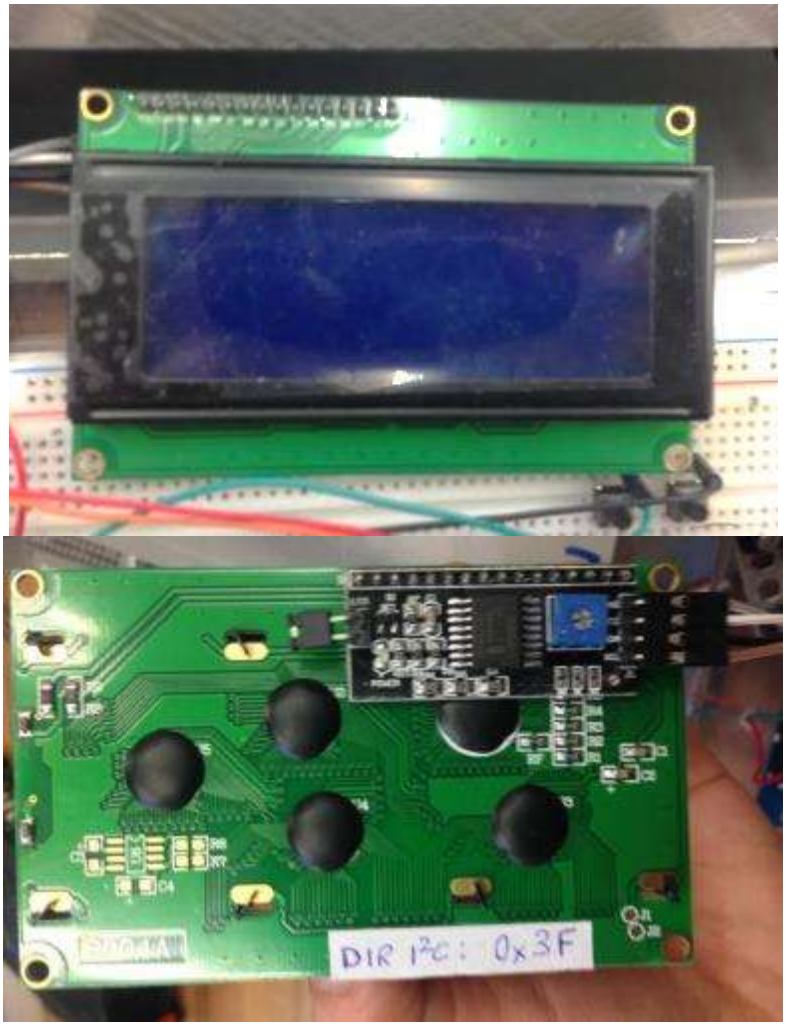


Figura 5.4.1.1: Pantalla LCD de 20x4 segmentos.

La alimentación de la pantalla oscila entre 0 y 5V, siendo la transmisión de información a través de 10 *bits*, como se desea tener una variación de frecuencia de aproximadamente de 0.2Hz a 9Hz, se subdivide la tensión en 2^{10} = 1024 posiciones. Por tanto la tensión se mueve entre 0 y 1023 posiciones.

Atendiendo a la programación en Arduino del *display*, es necesario introducir nuevas librerías: *wire.h* para la comunicación de Arduino con la pantalla en general, *LCD.h* para la comunicación de la pantalla, en particular LCD y *LiquidCrystal_I2C.h* para la comunicación de la pantalla, más concretamente LCD I^2C. Y las declaraciones siguientes:

- Se indica la dirección de la pantalla LCD: LCDdir = 0x3F.
- Para poder enviar mensajes desde el serial monitor, debe indicarse la instancia del objeto: *LiquidCrystal_I2C lcd*(LCDdir,2,1,0,4,5,6,7).
- Definir el valor de la frecuencia para el LCD: *char valF*[4];
- Definir la extensión de los caracteres de la pantalla, con 20 posiciones de longitud repartidas en 4 filas: *lcd.begin*(20,4);
- Indicar los mensajes por pantalla:

```
lcd.setBacklightPin(3,POSITIVE);
lcd.setBacklight(HIGH);
lcd.home();
lcd.setCursor(0,0);
lcd.print("Mesa sismica (v1.0)");
lcd.setCursor(1,1);
lcd.print("Modo:");
lcd.setCursor(1,2);
lcd.print("F = 0.00 Hz");
lcd.setCursor(1,3);
lcd.print("A = ");
lcd.print(Npasos);
lcd.print(" pasos");
```

5.4.2. Instalación del potenciómetro regulador de frecuencia

El potenciómetro proporciona el control de la frecuencia de forma manual entre 0.2 Hz y 9Hz. Para ello se conecta a Arduino a través de un pin analógico y la alimentación.

Con las siguientes declaraciones se programa la lectura del potenciómetro en Arduino [Ref. Web 9]:

- Se declara en Arduino: *const int pot* = A2;
- Se define el pin A2 como entrada del potenciómetro: *pinMode(*pot, INPUT);
- Lectura del valor del potenciómetro y guardar en la siguiente variable: *valpot = analogRead(pot)*;
- Mapeo de dicha variable y guardar en otra variable de frecuencia, para la función senoidal: *milifreq = map(valpot*,0,1023,200,9000);

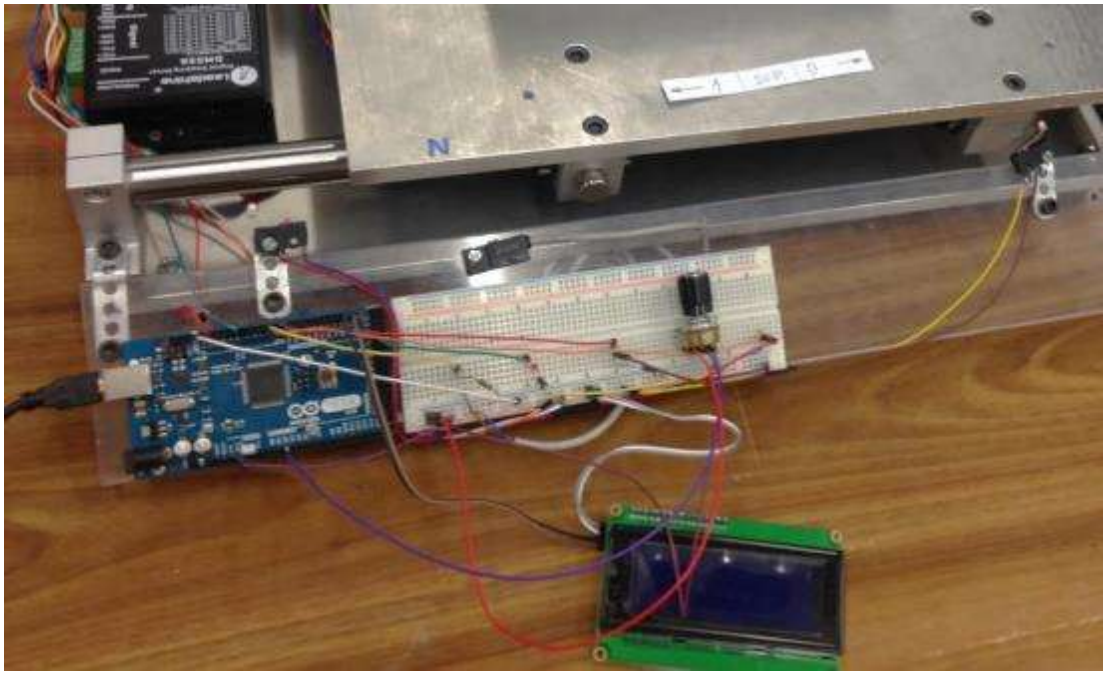


Figura 5.4.2.1: Display LCD y potenciómetro regulador de frecuencia.

5.4.3. Instalación del conmutador (modo automático/manual)

El conmutador marca el modo automático, es decir el desarrollo de la función senoidal, en la posición de normalmente abierto (NO) y marcando el modo manual, el movimiento a través de los pulsadores en una dirección u otra, con la posición de normalmente cerrado (NC). La alimentación la proporciona Arduino, con 5V a través de la patilla central del conmutador.

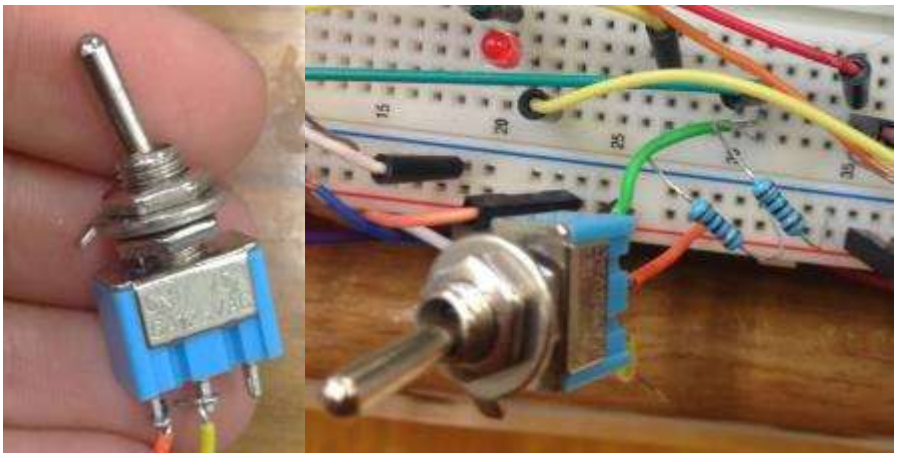


Figura 5.4.3.1: Conmutador para modo automático/manual
(con resistencias tipo *pulldown*).

Por tanto se conecta a través de dos pines digitales, para cada modo como se ha visto, y a la alimentación de Arduino. Hay que tener en cuenta que el conmutador lleva resistencias tipo *pulldown*, por lo que hay que instalar lar resistencias, en este caso de 10 kΩ, de la misma forma que en las conexiones de los finales de carrera tipo pulsador [Ref. Web 9].

Las modificaciones en el programa de control son las siguientes:

- Declarar las variables necesarias:

```
const byte manPin = 52; // Pin de modo manual

const byte sinPin = 53; // Pin de modo automático
```

- Leer la posición del conmutador y guardar el valor, para posteriormente realizar el movimiento que corresponda:

```
boolean modeMan = digitalRead(manPin); //Leer manual

boolean modeSin = digitalRead(sinPin); //Leer senoidal
```

- Proceso de movimiento senoidal con indicación por pantalla:

```
else if(modeSin == HIGH && modeMan == LOW) {

    lcd.setCursor(7,1);

    lcd.print("Senoidal");

    valpot = analogRead(pot);

    milifreq = map(valpot,0,1023,200,9000);

    dtostrf(milifreq/1000.0,3,2,valF);

    lcd.setCursor(5,2);

    lcd.print(valF);

    sube(Npasos, iT, milifreq);

    cambia();

  }
```

5.4.4. Instalación de dos pulsadores

Para poder desplazar la mesa de forma manual, sin tener que recurrir a la modificación del programa, si se desea una posición en concreto de la misma, se instalan dos pulsadores. Mientras alguno de ellos se encuentre pulsado, la mesa se desplazará hacia derecha o izquierda según corresponda.

Para ello se conecta cada pulsador a un pin digital, una resistencia de 130 Ω (tipo *pulldown*) y la alimentación de Arduino de 5V [Ref. Web 9].

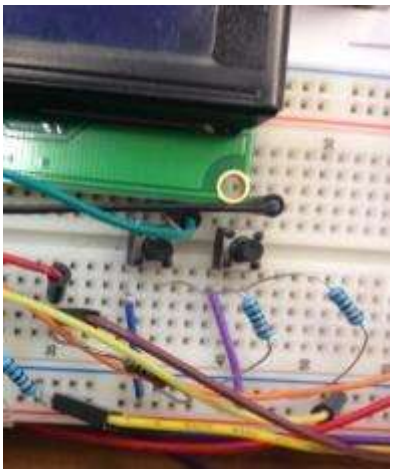


Figura 5.4.4.1: Pulsadores (con resistencias tipo *pulldown*)

Se añade en el programa de control las siguientes declaraciones para su adecuado funcionamiento:

- Declarar las variables necesarias:

```
const byte dchaPin = 50; // Pin de avance manual hacia la derecha

const byte izdaPin = 51; // Pin de avance manual hacia la izquierda
```

- Proceso de movimiento manual a derecha o a izquierda:

```
else if(modeMan == HIGH && modeSin == LOW) {

    lcd.setCursor(7,1);

    lcd.print("Manual");

    lcd.setCursor(5,2);

    lcd.print("0.00");

    if(digitalRead(dchaPin) == HIGH) {

      digitalWrite(DIR,LOW);

      mueve(2000,500);

    } else if(digitalRead(izdaPin) == HIGH) {

      digitalWrite(DIR,HIGH);

      mueve(2000,500);

    }

}

}
```

El sistema final con todos los controladores instalados, se muestra en la siguiente imagen:

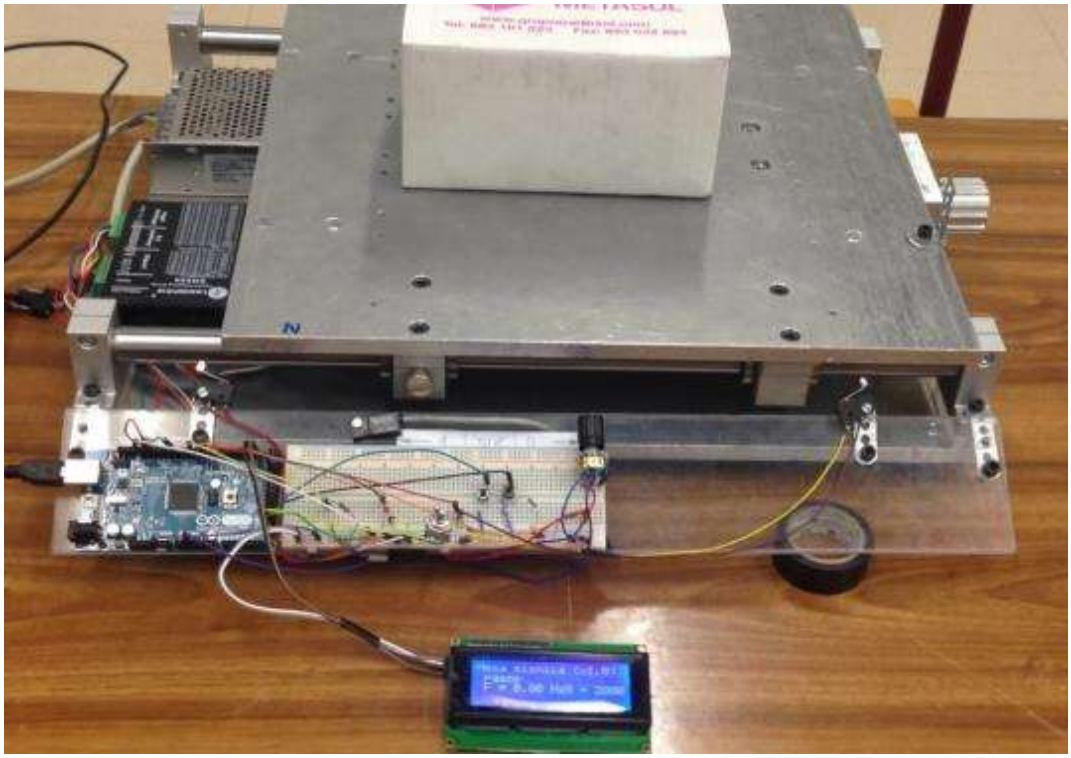


Figura 5.4.4.1: Instalación completa de controladores (*display* LCD, 2 pulsadores, potenciómetro, conmutador y 3 finales de carrera.

Si se desea ver el programa completo con los controladores anteriormente explicados, se encuentra reflejado en el anexo 9.7.

5.5. Límites de utilización

Al igual que se controla la posición de la mesa, para evitar que llegue al final de la carrera y ocasione roturas en el sistema. Es necesario conocer los límites de uso para frecuencia y amplitud de onda que se programen. Esto se debe a que el motor soporta unos márgenes de velocidad y potencia que se deben conocer, no solo por las especificaciones del fabricante sino aplicados a nuestro sistema.

Hay que tener en cuenta que la mesa no se usa en vació, es decir lleva asociada, además del peso propio de la placa superior, el peso de la carga que se quiera estudiar, por ello hay que comprobar los límites para esa carga, aumentándola en un porcentaje considerable, para dejar un margen de actuación.

El peso que se usa, es un peso equivalente a la maqueta del edificio de dos plantas disponible en el laboratorio. No se usa la propia maqueta porque primero hay que caracterizar con una masa estática, solidaria a la mesa. El edificio al ser esbelto y flexible induce inercias que no se controlan.

La planta inferior del edificio tiene una altura de 0.5 m y un peso de 1647.6 g. Y la planta superior tiene una altura de 0.75 m, con un peso de 1953 g. Por tanto el peso total es de 3600.6 g.

Para conseguir el peso equivalente, se toma una caja de tuercas con un peso final de 3754.4 g, dejando el margen de actuación mencionado.

El procedimiento para conocer los límites, se hace mediante la variación de la frecuencia para una amplitud fijada inicialmente. La amplitud de referencia es 1000 pasos del motor.

Se puede comprobar que a partir de una frecuencia de 6.8 Hz, el motor se fuerza demasiado y comienza a saltarse los pasos.

Como verificación de esta situación, se acude a la gráfica de la curva par-velocidad del motor (figura 5.1.1), donde se observa que a partir de una velocidad de 2000 rpm, el par que realiza el motor es prácticamente nulo. Y a partir de las 400 rpm, disminuye drásticamente.

Por lo que no tienen suficiente fuerza para desplazar el peso y acaba forzándose.

La velocidad angular del eje del motor (ecuación 5.5.1) se calcula como:

$$w = 2 \cdot \pi \cdot f = 2 \cdot \pi \cdot 6.8\, Hz = 42.72 \frac{rad}{s} \tag{5.5.1}$$

$$\rightarrow w = 42.72 \frac{rad}{s} \cdot 60\ \frac{s}{min} \cdot \frac{rev}{2 \cdot \pi\, rad} = 407.94\, rpm$$

El dato obtenido, justo coincide cuando el par que puede dar el motor empieza a disminuir considerablemente.

Se pasa a comprobar el par que es necesario para el desplazamiento del peso aplicado (ecuación 5.5.2) en esta situación.

$$M = d \times F \tag{5.5.2}$$

Sabiendo que para el husillo utilizado, se tiene un avance de aproximadamente 15 mm para los 1000 pasos del motor (para 1600 pasos por revolución del motor como referencia) y una carga equivalente de 3.7554 kg:

$$M = \frac{15\ mm \cdot (3.7554\ kg \cdot 9.8 \frac{N}{kg})}{1000\ m/mm} = 0.552\ N \cdot m$$

Para una velocidad angular de 400 rpm, el motor proporciona menos de 0.25 $N \cdot m$, según la recomendación de la gráfica. Por tanto es imposible que el motor pueda desplazar la carga, a frecuencias mayores de 6.8 Hz.

Así mismo para frecuencias demasiado bajas, aunque el motor proporcione mucho par, podría encontrarse fuera del rango de velocidad angular que puede dar un paso.

Se comprueba que por debajo de una frecuencia de 1 Hz para una amplitud de 1000 pasos, no hay un movimiento uniforme.

$$w = 2 \cdot \pi \cdot 1\ Hz = 6.283\ \frac{rad}{s} \cdot 60\ \frac{s}{min} \cdot \frac{rev}{2 \cdot \pi\ rad} = 60\ rpm$$

Así, como se observa en la curva, el rango recomendable es:

$$w = (100, 400) rpm \quad \rightarrow \quad f = (\ 1.66, 6.66\ )\ Hz$$

5.6. Caracterización con los equipos instrumentales de medida

Uno de los objetivos principales es conseguir desarrollar una onda senoidal y caracterizarla.

Para ello se emplea una tarjeta de adquisición de datos (hardware), es decir el equipo de medida disponible en el laboratorio, SIRIUS. Que permite medir cualquier tipo de señal, a partir de distintas conexiones. En este caso se le conecta el láser, a 1 de los 16 canales que contiene.

Figura 5.6.1: Equipo de adquisición de datos, SIRIUS.

El soporte que desarrolla este sistema (software), es el DEWEsoft. Se trata de un software libre, de adquisición y análisis de datos, que permite caracterizar la onda y exportar los resultados del análisis.

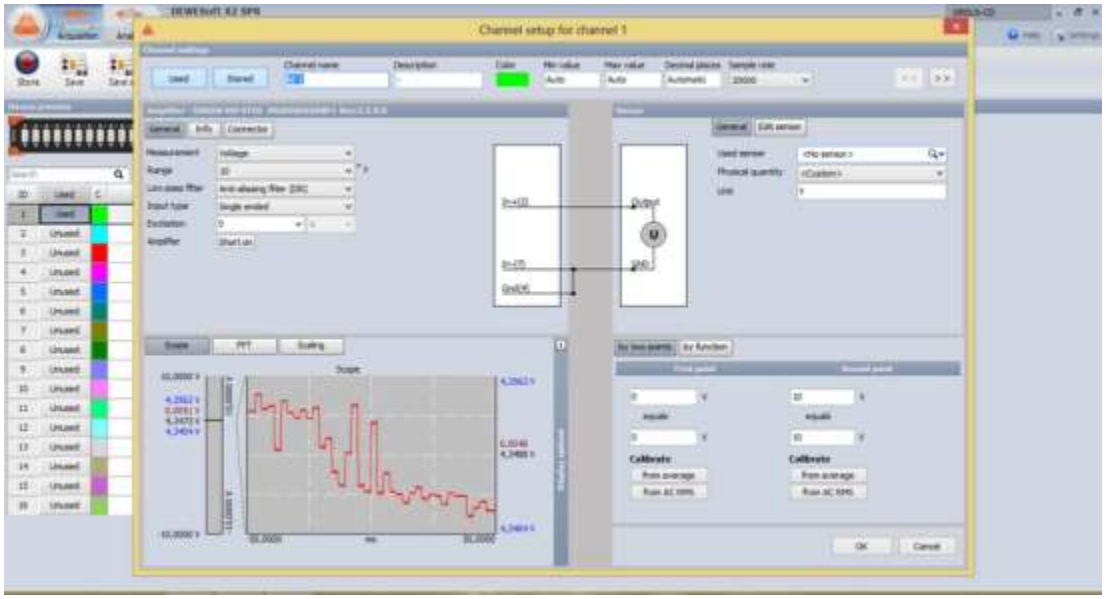


Figura 5.6.2: Software de análisis de datos, DEWEsoft.

Se calibra el láser con el DEWEsoft, introduciendo los siguientes parámetros:

Sensor:

- Unidad de medida: distancia (m).
- Primer punto de calibración: 0 V para 0.06 m.
- Segundo punto de calibración: 10 V para 0.18 m.

Amplificador SIRIUS:

- Unidad de medición: Voltaje.
- Rango: 10 V.
- Filtro de paso bajo: 100 Hz.
- Tipo de paso: *Butterworth* de 4 th orden.

Figura 5.6.3: Montaje completo para la comprobación.

Se recogen datos, para adquisición de 200 puntos por segundo, en dos tipos de gráficas: amplitud en función del tiempo (gráfica de la parte superior de la figura 5.5.4) y para la FFT, frecuencia en función del tiempo (gráfica en la parte inferior de la figura 5.5.4).

Para medir con la FFT de asignan los parámetros:

- Línea de resolución: 16 k.
- Tipo de escala y: log.

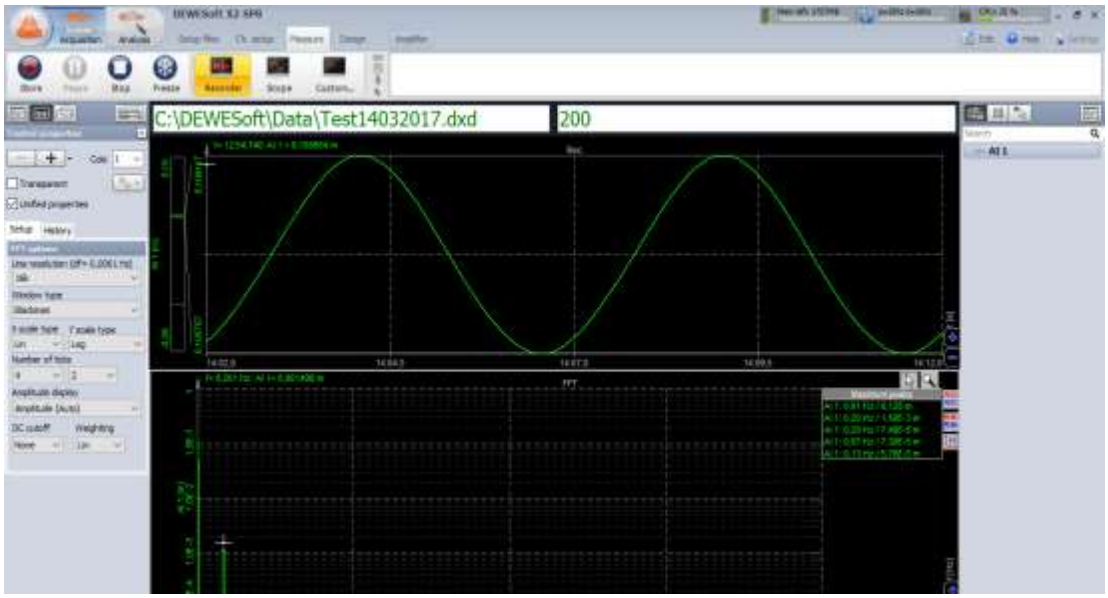


Figura 5.6.4: Pantalla de adquisición de datos en DEWEsoft.

Con el montaje completo de la mesa vibratoria, con todos sus controladores y la masa equivalente, se procede a la caracterización de la onda senoidal, tomando datos para varias amplitudes fijadas y distintas frecuencias.

Se evalúa la discrepancia entre los datos introducidos para el desarrollo de la onda (programación en Arduino) y los datos medidos a través del láser, recogidos en el DEWEsoft. Se realiza la corrección de dicho error con el cálculo de una función.

Rápidamente se comprueba un error excesivo en los resultados, por lo que procede a evaluar donde está el problema. Se llega a la conclusión, que los controladores introducidos en el montaje, en concreto el *display* LCD, producen un retardo en los cálculos excesivo para la programación usada. Puesto que este necesita mucho tiempo en comparación para conseguir sacar los parámetros por pantalla, retardando el resto de procesos y provocando un desajuste en la recogida de datos.

Para solventar este problema hay que hacer una serie de modificaciones a nivel básico de programación, que no interesan por ahora. Más concretamente, hay que desacoplar la programación en dos placas de Arduino. Por una parte la programación propia del cálculo de la onda senoidal y por otra toda la programación encargada de control, este proceso se deja como línea futura.

A continuación se muestran los resultados obtenidos con la pantalla instalada en el montaje, para una amplitud de 1000 pasos, para ver el fallo.

F teorica	F medida	Error
1,04	0,635	0,405
1,53	0,732	0,798
2,05	0,891	1,159
2,51	0,977	1,533
3,06	1,062	1,998
3,5	1,111	2,389
3,98	1,175	2,805
4,48	1,221	3,259
5,02	1,257	3,763
5,53	1,294	4,236
6,01	1,306	4,704
6,53	1,318	5,212
7,02	1,355	5,665
7,53	1,379	6,151
8,05	1,392	6,658

Figura 5.6.5: Resultados de la caracterización con *display* LCD (1000 pasos).

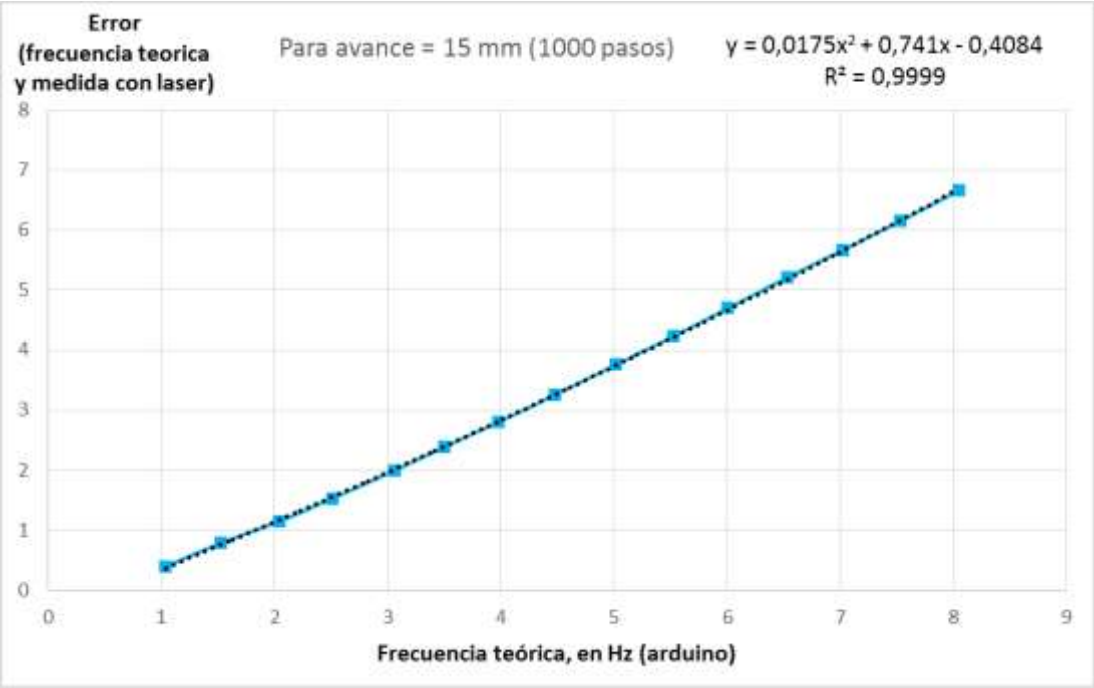


Figura 5.6.6: Gráfica para el cálculo de la corrección de la frecuencia medida (1000 pasos).

En dichos resultados se puede apreciar que la evolución de la frecuencia medida es mucho inferior que la frecuencia teórica introducida. Estos valores no reflejan la realidad del desarrollo, pues deben al menos aproximarse para que el cálculo de la onda senoidal sea correcto.

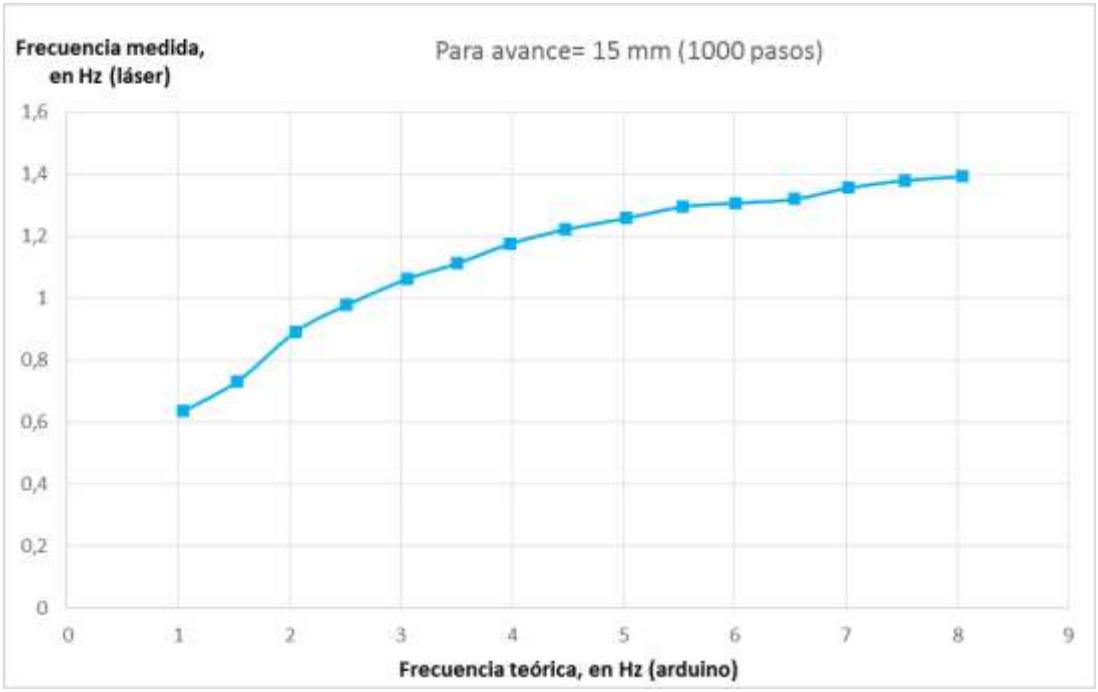


Figura 5.6.7: Gráfica de la evolución de la frecuencia teórica, frente a la medida (1000 pasos).

Se realiza una segunda medición para verificar que el problema es evidente y no es un fallo puntual. En este caso para 500 pasos de amplitud.

F teorica	F medida	Error
0,88	0,61	0,27
1,88	1,038	0,842
2,87	1,221	1,649
3,91	1,331	2,579
4,93	1,379	3,551
5,79	1,428	4,362
6,92	1,465	5,455
7,86	1,477	6,383

Figura 5.6.8: Resultados de la caracterización con *display* LCD (500 pasos).

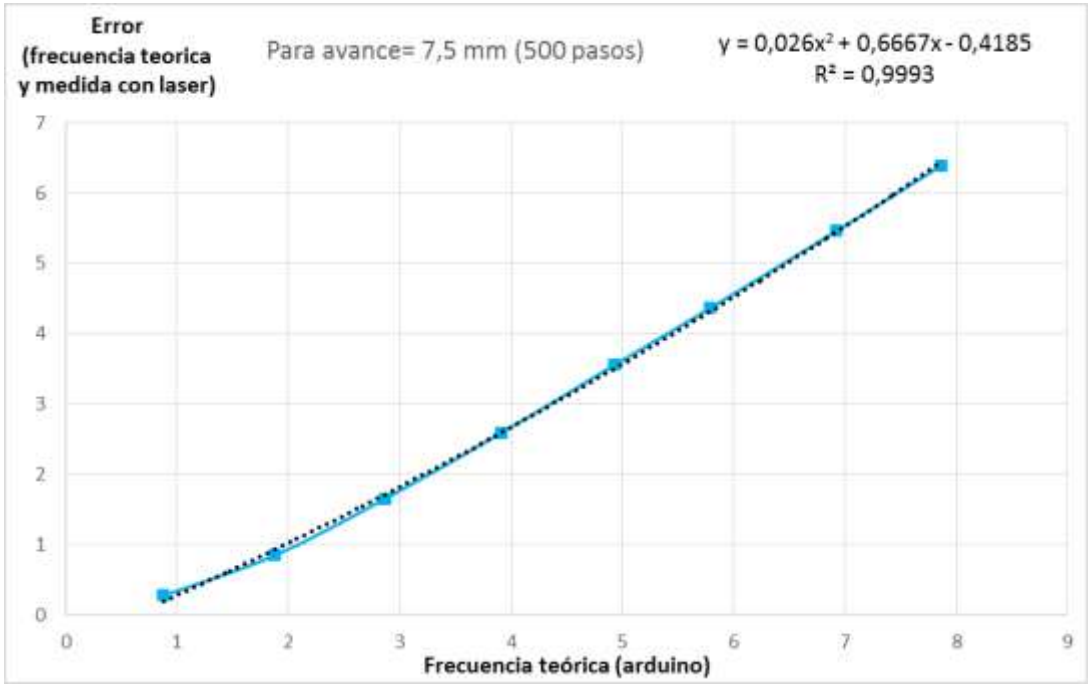


Figura 5.6.9: Gráfica para el cálculo de la corrección de la frecuencia medida (500 pasos).

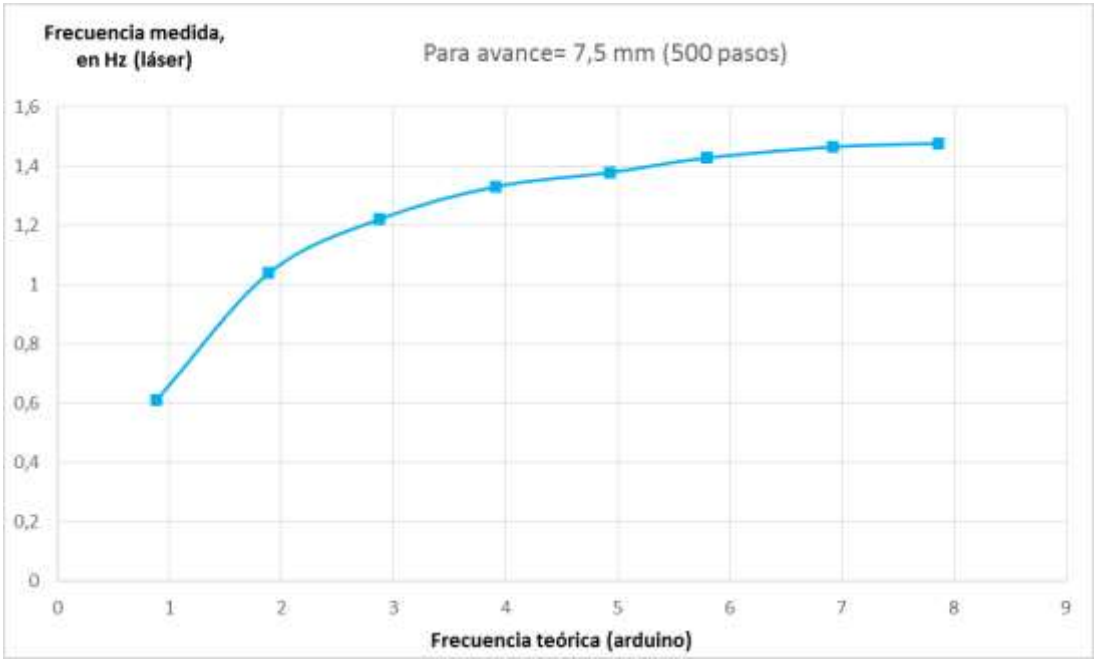


Figura 5.6.10: Gráfica de la evolución de la frecuencia teórica, frente a la medida (500 pasos).

Se comprueba a través de las gráficas, comparando la frecuencia teórica y la medida, que su tendencia es asintótica y no permite superar un máximo de 1.5 Hz (figuras 5.6.7 y 5.6.10). Como se ha visto anteriormente, con los datos analíticos del motor calculados, se puede superar ampliamente esta frecuencia.

Por lo que se decide prescindir de la pantalla y evaluar el funcionamiento de la mesa sin ella para evitar complicaciones.

Una vez desinstalada la pantalla y retomando la programación para el desarrollo de la onda senoidal sin dicho dispositivo, se vuelve a realizar la adquisición de datos para la una amplitud fija, evaluada con 500, 1000, 1500, 2000, 2500, 3000 y 3500 pasos (para una referencia del motor de 1600 pasos por vuelta, fijado en el *dirver*).

A partir de los 3500 pasos, Arduino no tiene suficiente espacio de memoria dinámica para ejecutar la actividad, esto se debe a que la placa Arduino utilizada (mega 2560, es decir 256 kB) no tiene suficiente capacidad, aunque el motor y el mecanismo permitan aumentar la amplitud. Se deja también para próximas mejoras, la utilización de una placa Arduino de mayor memoria *flash*, como puede ser Arduino *due*, con el doble de memoria (516 kB) u otras.

Se comienza la caracterización con un amplitud baja de 500 pasos, es decir 1.875 mm. Se van introduciendo frecuencias de forma progresiva y se comparan con las medidas del láser y recogidas por el SIRIUS.

Como se ve en la tabla, se toman datos dentro de un rango, esos no son los límites para un avance de 7,5 mm (500 pasos). Pero para las siguientes amplitudes se alcanzan los límites.

Se calcula la desviación o error entre ambas frecuencias y con la frecuencia teórica (introducida en Arduino) y el error, se gráfica la evolución. A partir de esta gráfica se obtiene la línea de tendencia para la corrección de la desviación. Esta línea de tendencia sigue una función que sirve para corregir la desviación de la frecuencia. Introduciendo en la programación la frecuencia que se quiere obtener y sumándole la función desviación obtenida para cada amplitud, se corrige el error.

A continuación se muestran los resultados obtenidos y las funciones calculadas para cada caso.

F teorica	F medida	Error
1	0,97	0,03
1,5	1,45	0,05
2	1,92	0,08
2,5	2,39	0,11
3	2,84	0,16
3,5	3,3	0,2
4	3,74	0,26
4,5	4,17	0,33
5	4,6	0,4
5,5	5,03	0,47
6	5,44	0,56
6,5	5,88	0,62
7	6,26	0,74

Figura 5.6.11: Resultados de la caracterización sin *display* LCD (500 pasos).

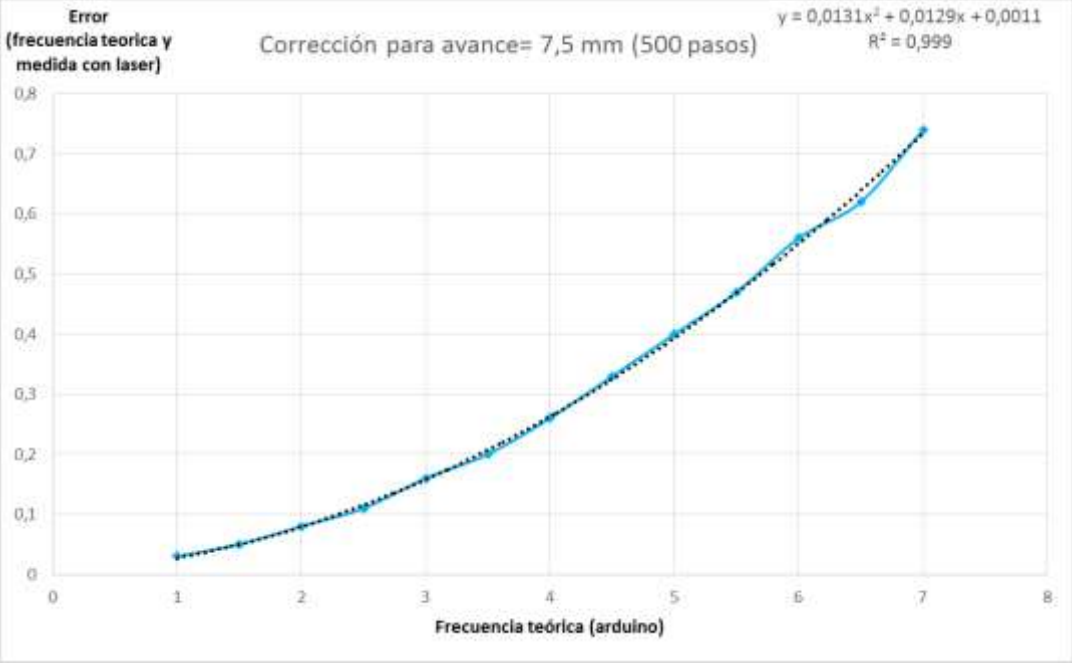


Figura 5.6.12: Gráfica para el cálculo de la corrección de la frecuencia medida (500 pasos).

F teorica	F medida	Error
1	0,96	0,04
1,5	1,422	0,078
2	1,87	0,13
2,5	2,3	0,2
3	2,72	0,28
3,5	3,13	0,37
4	3,53	0,47
4,5	3,91	0,59
5	4,29	0,71
5,5	4,65	0,85
6	5,01	0,99
6,5	5,36	1,14
6,7	5,49	1,21

Figura 5.6.13: Resultados de la caracterización sin *display* LCD (1000 pasos).

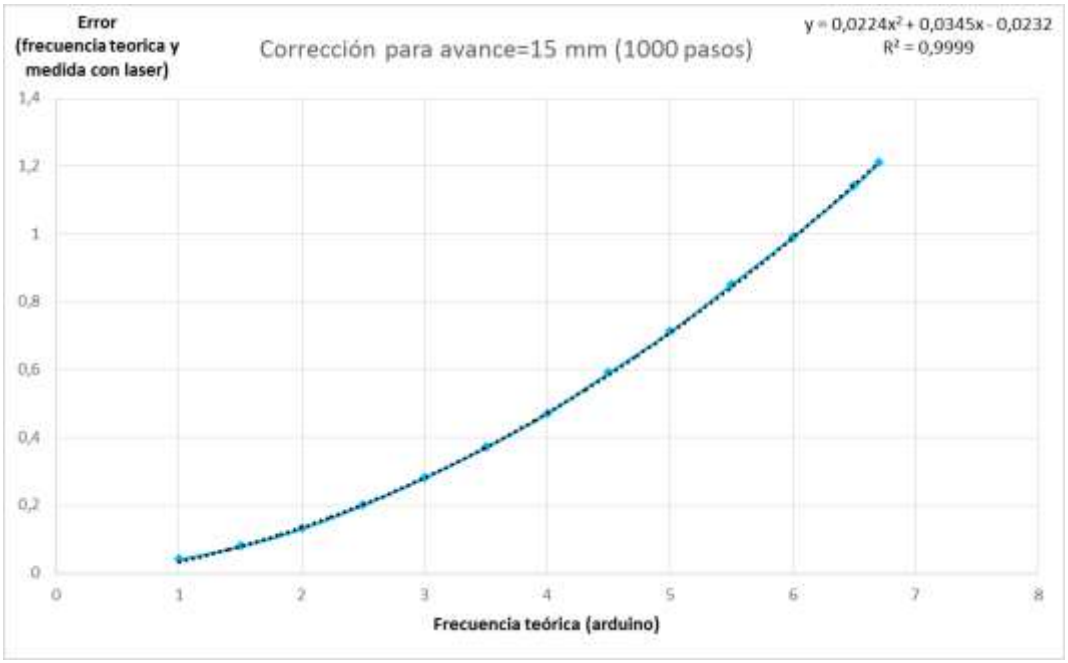


Figura 5.6.14: Gráfica para el cálculo de la corrección de la frecuencia medida (1000 pasos).

F teorica	F medida	Error
0,8	0,76	0,04
1	0,95	0,05
1,5	1,39	0,11
2	1,81	0,19
2,5	2,22	0,28
3	2,61	0,39
3,5	2,98	0,52
4	3,34	0,66
4,5	3,69	0,81
5	4,01	0,99
5,5	4,33	1,17
5,7	4,46	1,24

Figura 5.6.15: Resultados de la caracterización sin *display* LCD (1500 pasos).

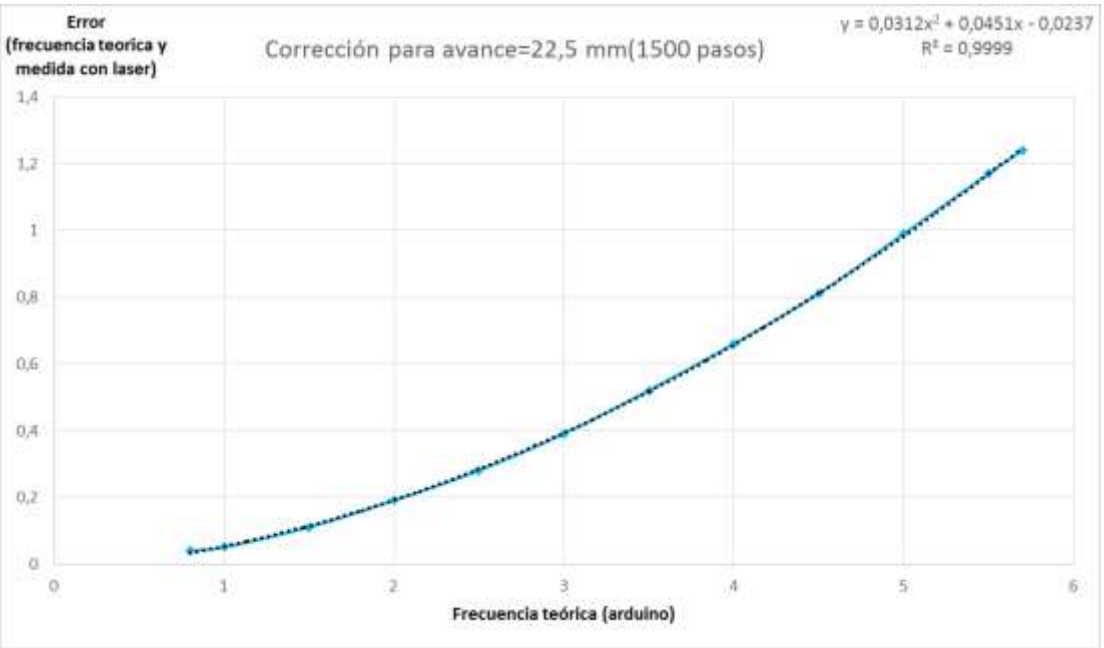


Figura 5.6.16: Gráfica para el cálculo de la corrección de la frecuencia medida (1500 pasos).

F teorica	F medida	Error
1	0,93	0,07
1,5	1,36	0,14
2	1,76	0,24
2,5	2,15	0,35
3	2,51	0,49
3,5	2,85	0,65
4	3,17	0,83
4,5	3,49	1,01
4,7	3,6	1,1

Figura 5.6.17: Resultados de la caracterización sin *display* LCD (2000 pasos).

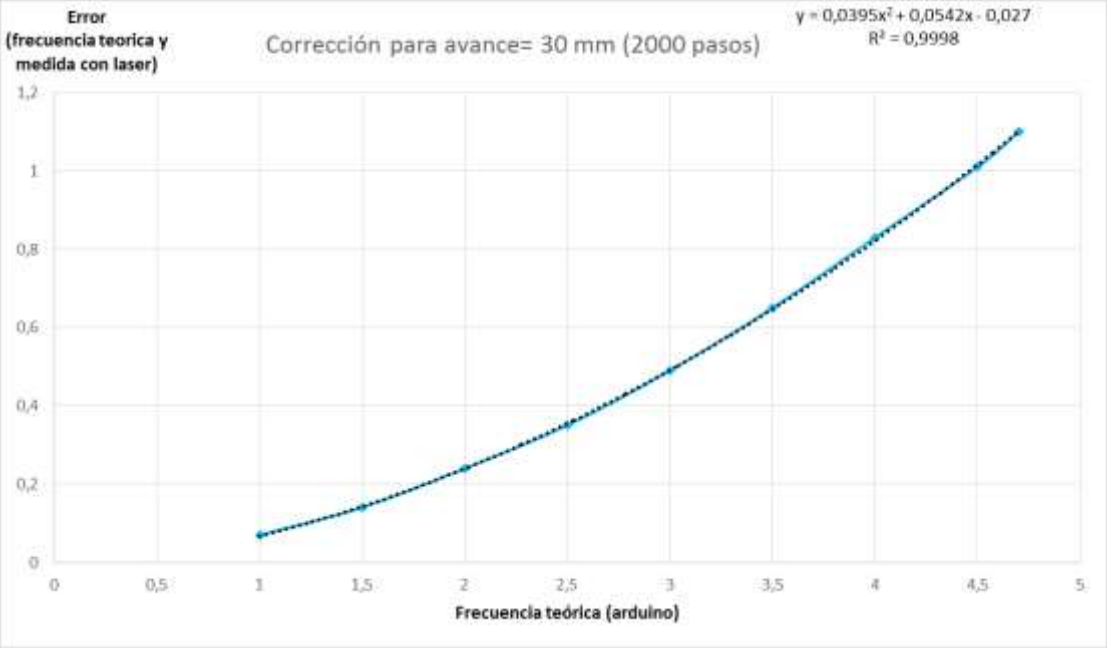


Figura 5.6.18: Gráfica para el cálculo de la corrección de la frecuencia medida (2000 pasos).

F teorica	F medida	Error
1	0,92	0,08
1,5	1,33	0,17
2	1,72	0,28
2,5	2,08	0,42
3	2,4	0,6
3,5	2,72	0,78
4	3,02	0,98

Figura 5.6.19: Resultados de la caracterización sin *display* LCD (2500 pasos).

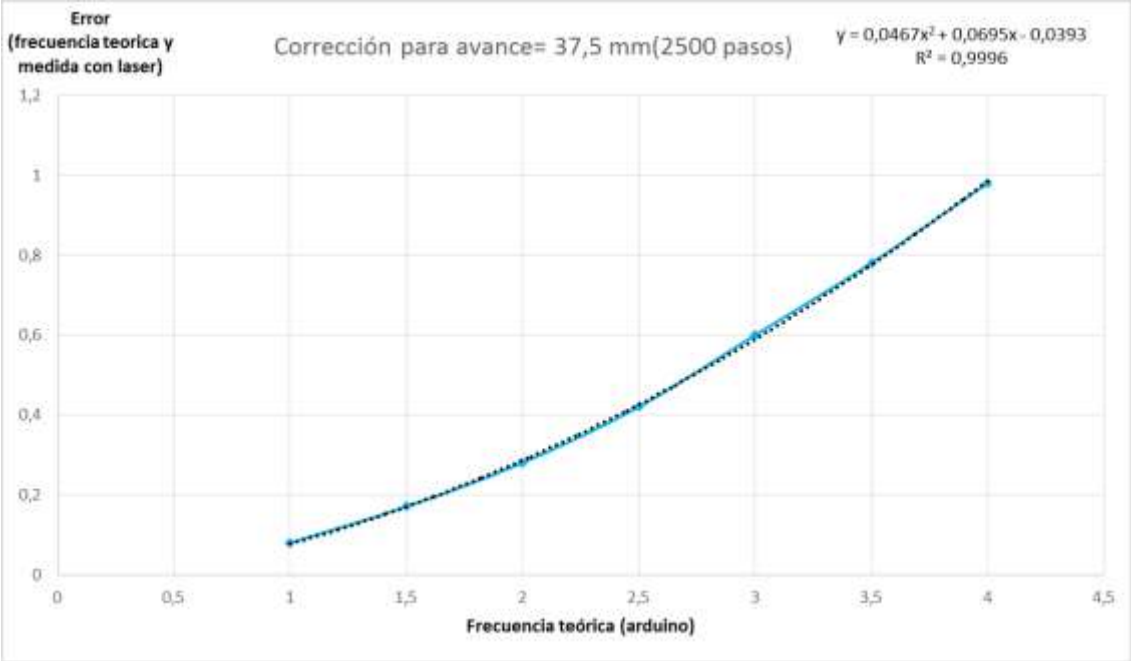


Figura 5.6.20: Gráfica para el cálculo de la corrección de la frecuencia medida (2500 pasos).

F teorica	F medida	Error
1	0,91	0,09
1,5	1,31	0,19
2	1,67	0,33
2,5	2,01	0,49
3	2,34	0,66
3,5	2,64	0,86
4	2,83	1,17

Figura 5.6.21: Resultados de la caracterización sin *display* LCD (3000 pasos).

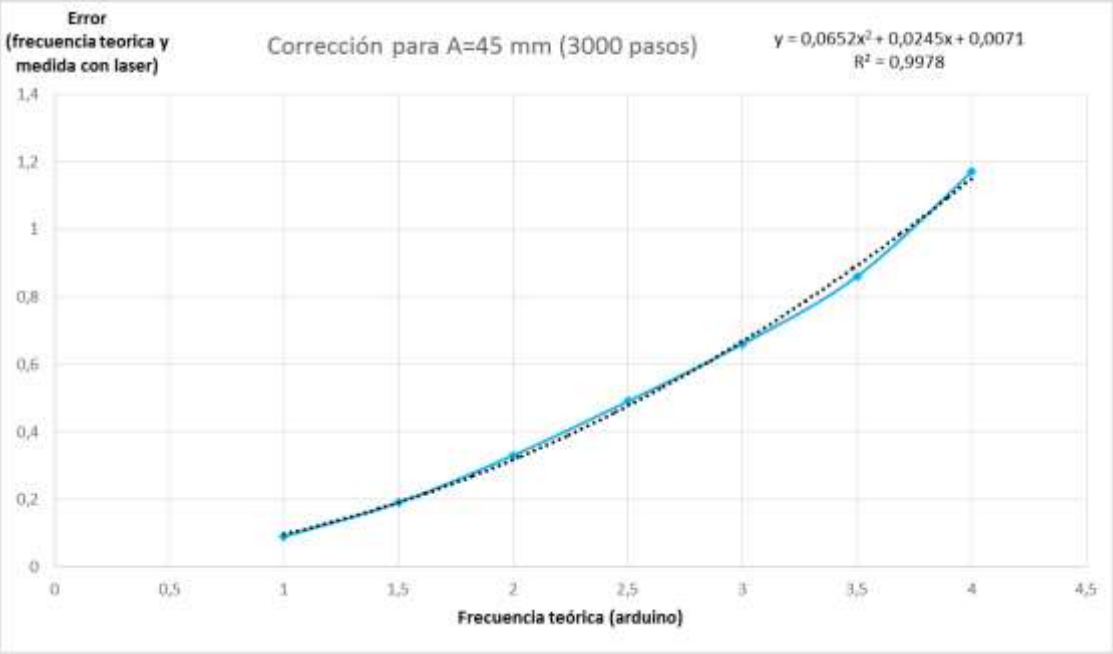


Figura 5.6.22: Gráfica para el cálculo de la corrección de la frecuencia medida (3000 pasos).

F teorica	F medida	Error
1	0,88	0,12
1,5	1,27	0,23
2	1,66	0,34
2,5	1,95	0,55
3	2,25	0,75
3,5	2,54	0,96

Figura 5.6.23: Resultados de la caracterización sin *display* LCD (3500 pasos).

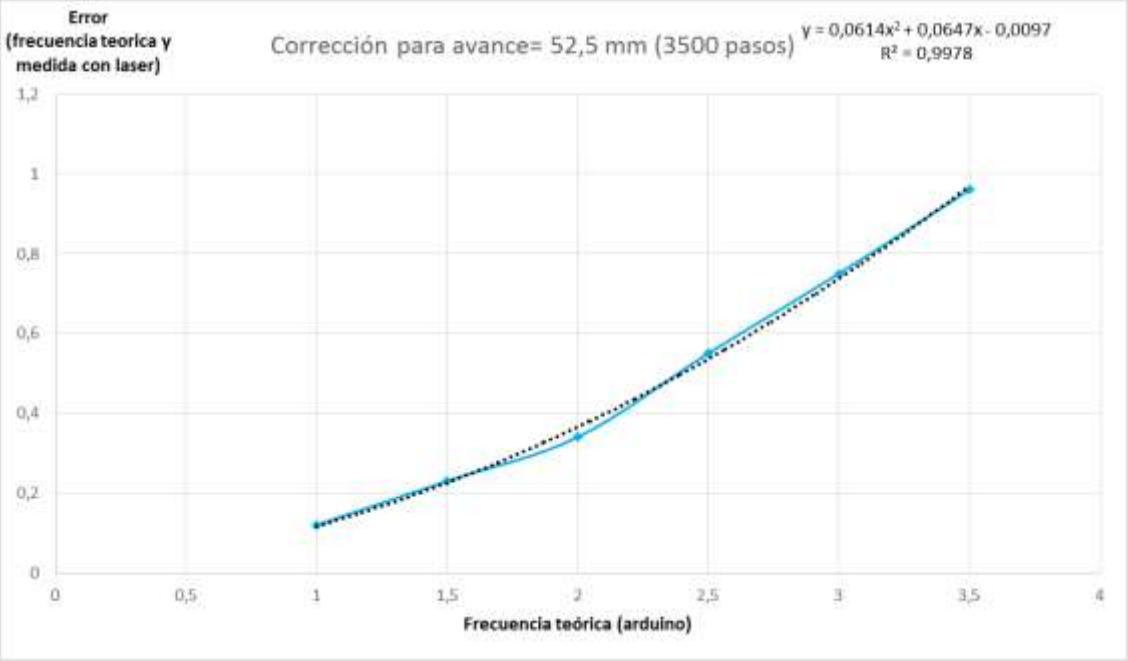


Figura 5.6.24: Gráfica para el cálculo de la corrección de la frecuencia medida (3500 pasos).

Una vez obtenidas las líneas de tendencia, en todas ellas se observa que la función es polinómica de segundo grado con muy buena aproximación.

Se suman la función evaluada y la frecuencia deseada en el programa y se obtiene la frecuencia corregida.

Se vuelven a realizar las medidas experimentales y se verifica, que efectivamente el resultado medido en todos los casos, es muy próximo al dato introducido por programación. Como máximo aparece una discrepancia de 5 centésimas. Por lo que se tiene una muy buena aproximación final.

Esto se puede ver en los datos recogidos en la medición, obteniendo la onda senoidal y comparándola con el seno fundamental para dicha frecuencia.

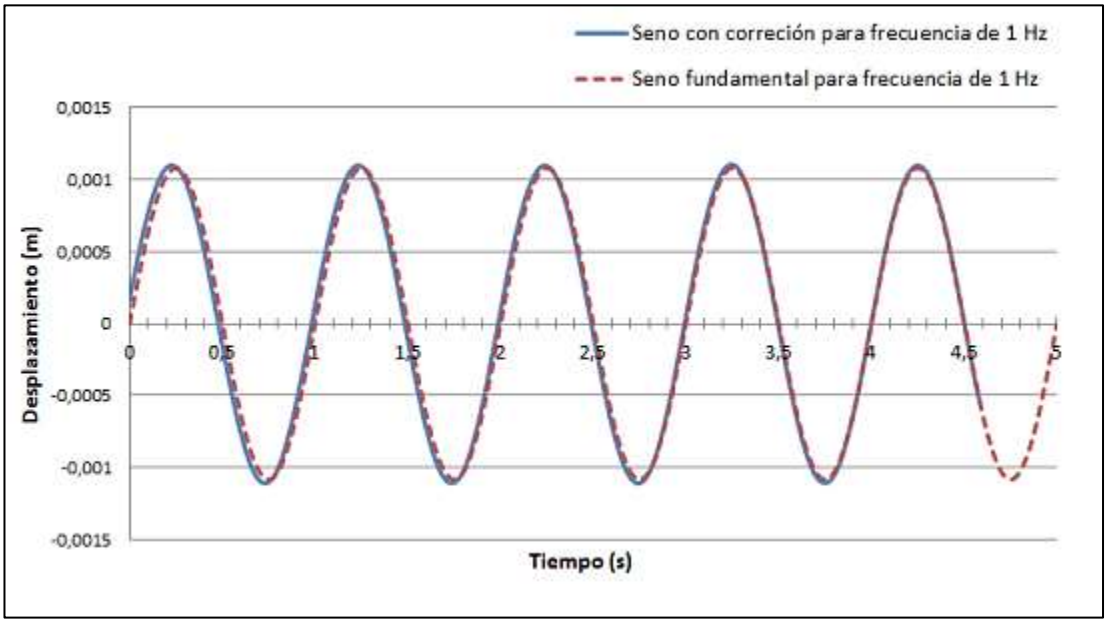


Figura 5.6.25: Gráfica comparativa seno medido y seno fundamental para 1 Hz.

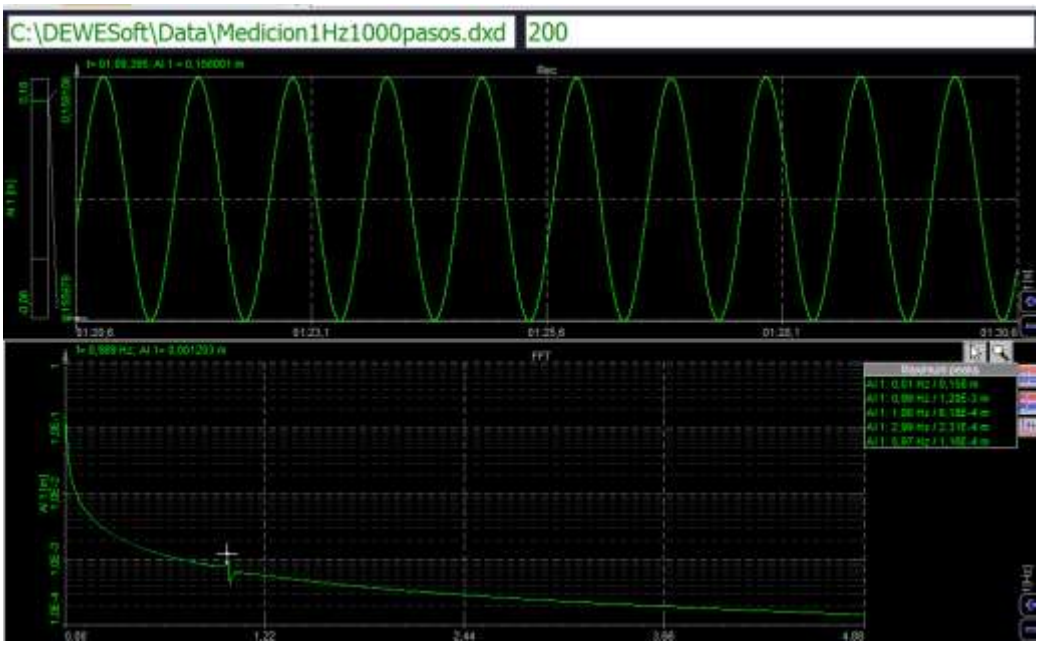


Figura 5.6.26: Recogida de datos con SIRIUS y análisis en DEWsoft para 1Hz.

En la imagen de DEWEsoft se aprecia que la aproximación para 1 Hz de frecuencia introducida, es de 0.989 Hz medidos con el láser.

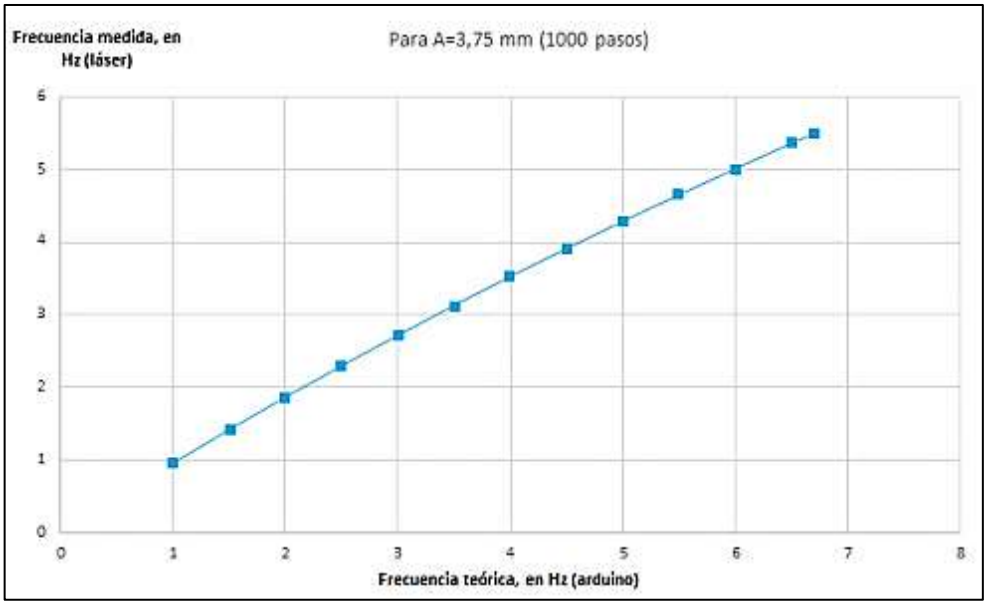


Figura 5.6.29: Gráfica de la evolución de la frecuencia teórica, frente a la medida (1000 pasos).

En la gráfica comparativa de frecuencias, se aprecia una evolución con tendencia lineal, como era de espera.

Además en las tablas de resultados se indica el dato en rojo, como aquel en el que el motor no responde para dicha frecuencia y amplitud, es decir el límite.

A continuación se recoge en una tabla los datos de los límites de frecuencia en función de amplitud, se marca con una cruz aquellas frecuencias alcanzables.

			Frecuencia (Hz)											
	Pasos		1	1,5	2	2,5	3	3,5	4	4,5	5	6	7	8
Amplitud (mm)	500	1,875	X	X	X	X	X	X	X	X	X	X	X	X
	1000	3,75	X	X	X	X	X	X	X	X	X	X		
	1500	5,625	X	X	X	X	X	X	X	X				
	2000	7,5	X	X	X	X	X	X	X					
	2500	9,375	X	X	X	X	X	X						
	3000	11,25	X	X	X	X	X							
	3500	13,125	X	X	X	X								
	4000	15												

Figura 5.6.28: Tabla comparativa frecuencia / amplitud.

6. Conclusiones y especificaciones de uso

Tal y como se ha visto, se ha construido un prototipo funcional de una mesa sísmica a escala, para la simulación de una onda senoidal.

Se comprueba a través de la gráfica de comparación con el seno fundamental (figura 5.6.25) que se cumple el objetivo, al obtener una buena aproximación del seno.

Así mismo, se llega a la conclusión que a medida que se aumenta la amplitud de funcionamiento asociado al número de pasos del motor, la frecuencia alcanzable disminuye (figura 5.6.27), aumentando el avance (mayor recorrido de la carrera). Lo que es lógico, porque a medida que se aumenta la velocidad de giro del motor, este proporciona menor par para mover la misma carga equivalente. Hay que tener en cuenta la tabla de especificaciones técnicas (figura 5.6.27) a la hora de poner en funcionamiento la mesa vibratoria.

Que a su vez es acorde al rango recomendable de las especificaciones del motor, zona de funcionamiento óptimo.

$$w = (100, 400) rpm \quad \rightarrow \quad f = (\,1.66, 6.66\,)\, Hz$$

			Frecuencia (Hz)											
	Pasos		1	1,5	2	2,5	3	3,5	4	4,5	5	6	7	8
Amplitud (mm)	500	1,875	X	X	X	X	X	X	X	X	X	X	X	X
	1000	3,75	X	X	X	X	X	X	X	X	X	X		
	1500	5,625	X	X	X	X	X	X	X	X				
	2000	7,5	X	X	X	X	X	X	X					
	2500	9,375	X	X	X	X	X	X						
	3000	11,25	X	X	X	X	X							
	3500	13,125	X	X	X	X								
	4000	15												

Figura 6.1: Especificaciones técnicas de la mesa sísmica.

7. Mejoras (líneas futuras)

En todo trabajo existen actualizaciones y mejoras, se han visto varias líneas para modificar y mejorar, se detallan a continuación:

- En cuanto al diseño existen alternativas que se pueden aplicar, como es el uso del mecanismo correa-polea, visto en el apartado 3.2.

- Además existe la posibilidad de sustituir el motor paso a paso, por un servomotor para un control absoluto del mecanismo, anexo 9.5.

- Para posibilitar el desacoplamiento fácil y rápido de la plataforma del desplazador y por tanto del motor, se realiza un taladro. Interesa realizar dicho hueco de mayores dimensiones y fresado para evitar lesiones.

- En cuanto a la electrónica, se pueden realizar múltiples mejoras con cableado integrado en la *protoboard*. Además de la instalación del *display* LCD y el resto de controladores vistos en el apartado 5.4 y añadiendo un potenciómetro para la regulación de la amplitud. Con la posibilidad de desacoplar la programación en dos placas Arduino para que los controladores no interfieran en el proceso de cálculo de la onda senoidal. Y la utilización de una placa Arduino due u otras, para dar mayor capacidad de memoria al programa.

- Como se ha venido apreciando, las patas de goma ejercen una gran inercia en movimientos a altas frecuencias, por lo que se considera cambiarlas, cortar la capa final de la goma para evitar desplazamientos o realizar el montaje acoplado a tierra.

- Por último es deseable caracterizar la mesa vibratoria con el edificio de dos plantas instalado.

8. Bibliografía

Libros y artículos:

[1].- Ogata, K. Ingeniería de control moderna. Pearson. (2010).

[2].- Clavijo, J.C. Reinaldo, L. Diseño, modelamiento y simulación de una mesa sísmica unidireccional hidráulica. Universidad Industrial de Santander. Facultad e Ingenierías Físico-Mecánicas. Escuela de Ingeniería Mecánica. Bucaramanga. (2011).

[3].- Muhlenkamp, M.J. *Analysis, design and construction of a shaking table facility. Department of Civil Engineering. Rice University*. (1997).

[4].- Carrilo, J. Bernal, N.M. Porras, P. Evaluación del diseño de una pequeña mesa vibratoria para ensayos en ingeniería sismo-resistente. Universidad Militar Nueva Granada. Ciencia e Ingeniería Neogranadina. 23. (2013). 89 -105.

[5].- Bernal, N.M. Automatización del Equipo de Simulación Sísmica Uniaxial del Laboratorio de Estructuras. Universidad Militar Nueva Granda. Facultad de Ingeniería Mecatrónica. Bogotá D.C. (2013). 62.

[6].- Coral, H.A. Sandoval, J.A. Rosero, E.E. Ramírez, J.M. Gómez, D. Simulador Sísmico Uniaxial Tele-Operable para Modelos Estructurales de Pequeña Escala. Universidad del Valle. Cali. Colombia. (2009). 7.

[7].- Coral, H.A. Thomson, P. Rosero, E.E. Ramírez, J.M. Gómez, D. Marulanda, J. Diseño, Construcción y Control de un Simulador Sísmico Uniaxial Tele-Operable para Modelos Estructurales a Pequeña Escala. Ingeniería y Competitividad. 12. (2010). 22

[8].- Mosconi, O.L. Motor Lineal. Área Electrotecnia y Máquinas Eléctricas. Máquinas e Instalaciones Eléctricas. 13.

[9].- Shigley J. E., Mischke C.R. Diseño en ingeniería mecánica. Mc Graw Hill 6ª Ed. (2002).

[10].- *QuickSilver Controls, Inc. Datasheet:QCI-DS008. NEMA 23 I-Grade Motor/Encoder.* (2010).

[11].- *Leadshine Technology Company Limited. User's Manual for DM556. Fully Digital Stepping Driver. Version* 1.0. (2009).

Páginas Web:

[1].- González, F. Sabogal, J. XXVI Muestra de máquinas y prototipos. Dispositivo para generar sismos a escala. Departamento de Ingeniería Mecánica y Mecatrónica. (2010). 31. <http://es.slideshare.net/diales/dispositivo-para-generar-sismos-a-escala>.

[2].- Esparza, C. Núñez, R. Controlador adaptativo pd por modelo de referencia para una mesa vibratoria biaxial basada en el mecanismo biela-manivela. La Serena. Información tecnológica versión On-line. 25. ISSN 0718-0764. (2014). 20. <http://www.scielo.cl/scielo.php?pid=S0718-07642014000200021&script=sci_arttext>.

[3].- Pompa, P. Carril motorizado para fotografía y vídeo. INTPLUS. (2012). <http://www.superrobotica.com/slider.htm>

[4].- *Quanser Innovate-Educate. Study Structure Performance by Simulating Earthquake in Your Lab*.<http://www.quanser.com/products/shake_table_ii>.

[5].- Sucarmo. CNC Robótica < http://cnc-robotica.com/es/>.

[6].- Gualito, L. Mesa vibratoria. SlideShare. (2014). 9**.** <http://es.slideshare.net/luiseduardogualitorodriguez/mesa-vibratoria>.

[7].- *Rollon. Linear Line*. <http://www.rollon.com/ES/es/productos/linear-line/>.

[8].- *Tecnopower. Módulos Lineales Unimotion.* <http://www.tecnopower.es/m%C3%B3dulos-lineales-unimotion>.

[9].- *Arduino Genuino Tutorial*. <https://www.arduino.cc/en/Tutorial>.

[10].- *Igus Plastics for longer life.* <http://www.igus.es/wpck/2003/DryLin_W_Doppelschiene>.

[11].- RS <http://es.rs-online.com/web/p/products/474-5886/>.

[12].- SuperRobotica <http://www.superrobotica.com>.

[13].- Catálogo Lenze del servomotor 8400 TopLine ofertado <http://www.lenze.com/fileadmin/lenze/documents/en/catalogue/CAT_13493190_Inverters_Frequency_Inverters_en-GB.pdf>.

[14].- SACOR MAQUINARIA <http://frs-cnc.com/motores/servos>.

[15].- Alava Ingenieros, motores lineales <http://www.alava-ing.es/ingenieros/buscador/?q=motor+lineal>.

[16].- Ficha técnica del sensor de efecto hall <http://www.littelfuse.com/~/media/electronics/datasheets/hall_effect_sensors/littelfuse_hall_effect_sensors_55075_datasheet.pdf.pdf>.

[17].- Ficha técnica del detector magnético D-M9P <https://www.smc.eu/smc/Net/EMC_DDBB/ce_documentation/data/attachments/IMM_D-M9x-V_TFI98ES-C.pdf>

[18].- Ficha técnica del detector magnético D-C73 <http://www.kvc.com.my/StorageAttachment/Kvcsb/datasheet/1341/smc-d-c73.pdf>

9. Anexos

9.1. Ecuaciones de movimiento del sistema mecánico

El sistema a analizar está compuesto de dos masas m_1 y m_2, que representan los pisos del edificio y cada uno de los cuales con su respectivo resorte de rigidez k y amortiguador representado por c [Ref. Libro 1].

Se aplican las Leyes de Newton para la traslación del sistema, un sistema de segundo orden.

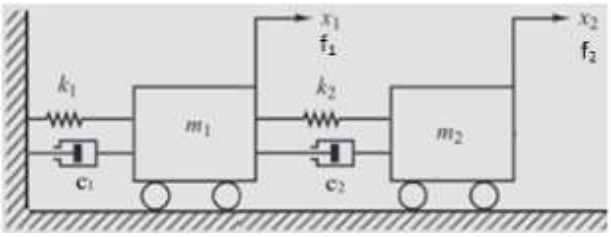


Figura 9.1.1: Representación del modelo del sistema masa, resorte y amortiguador.

Para el modelo de un sistema de sólo una masa, un resorte y un amortiguador la ecuación que representa las fuerzas aplicadas es la siguiente (ecuación 9.1.1) [Ref. Libro 1]:

$$m\frac{d^2(x(t))}{dt^2} = f(t) - c\frac{d(x(t))}{dt} - k \cdot x(t) \qquad (9.1.1)$$

Como el caso que se trata es el del edificio de dos pisos cuya representación es la figura 9.1.1 pero verticalmente, las ecuaciones que representan el movimiento del sistema son (ecuaciones 9.1.2) [Ref. Libro 1]:

$$\begin{cases} m_1.\ddot{x_1} + (c_1 + c_2)\dot{x_1} - c_2 \cdot \dot{x_2} + (k_1 + k_2)x_1 - k_2 \cdot x_2 = f_1 \\ m_2.\ddot{x_2} - c_2 \cdot \dot{x_1} + c_2 \cdot \dot{x_2} - k_2 \cdot x_1 + k_2 \cdot x_2 = f_2 \end{cases} \qquad (9.1.2)$$

Se aplica la linealización en un punto de operación, para obtener el modelo en variables de estado [Ref. Libro 1].

$$\begin{cases} \dot{x(t)} = A \cdot x(t) + B \cdot u(t) \\ y(t) = C \cdot x(t) + D \cdot u(t) \end{cases} \longrightarrow \begin{cases} x: \text{vector de estados} \\ u: \text{vector de entradas} \\ y: \text{vector de salidas} \end{cases}$$

Cuyo espacio de estados es [Ref. Libro 1]:

$$x = \begin{pmatrix} \dot{x_1} \\ \dot{x_2} \\ x_1 \\ x_2 \end{pmatrix} \qquad u = \begin{pmatrix} f_1 \\ f_2 \end{pmatrix}$$

En el caso del edificio la representación de espacio de estados queda en forma matricial:

$$\begin{pmatrix} \ddot{x_1} \\ \ddot{x_2} \\ \dot{x_1} \\ \dot{x_2} \end{pmatrix} = \begin{pmatrix} \frac{-(c_1+c_2)}{m_1} & \frac{c_2}{m_1} & \frac{-(k_1+k_2)}{m_1} & \frac{k_2}{m_1} \\ \frac{c_2}{m_2} & \frac{-c_2}{m_2} & \frac{k_2}{m_2} & \frac{-k_2}{m_2} \\ 0 & 0 & 0 & 0 \\ 0 & & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} \dot{x_1} \\ \dot{x_2} \\ x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} \frac{1}{m_1} & 0 \\ 0 & \frac{1}{m_2} \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} f_1 \\ f_2 \end{pmatrix}$$

Cada piso del edificio se mueve respecto a la base de la mesa, el sistema de referencia que se ha tomado es respecto al suelo. Por ello se ajusta para definir al sistema de referencia adecuado.

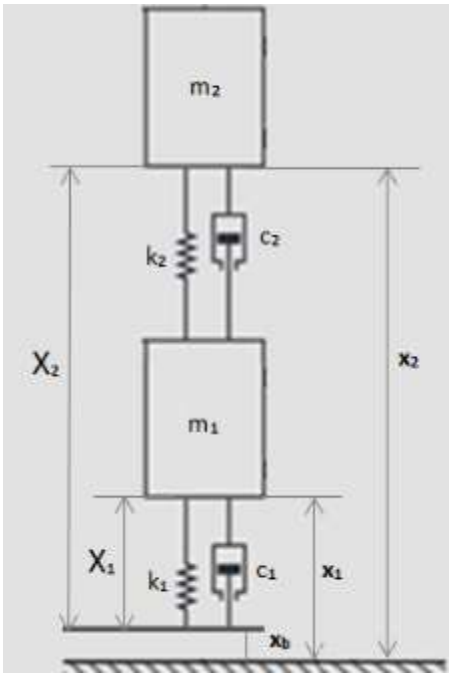


Figura 9.1.2: Modelo del sistema masa, resorte y amortiguador.

Se define el sistema de referencia respecto a la mesa:

$$\begin{cases} X_1 = x_1 - x_b \\ X_2 = x_2 - x_b \end{cases} \rightarrow \begin{cases} X_1 + x_b = x_1 \\ X_2 + x_b = x_2 \end{cases}$$

Para obtener las nuevas ecuaciones de fuerza se deriva el sistema de referencia anterior, consiguiendo la velocidad y la aceleración.

$$\begin{cases} \dot{X}_1 + \dot{x_b} = \dot{x_1} \\ \dot{X}_2 + \dot{x_b} = \dot{x_2} \end{cases} \quad y \quad \begin{cases} \ddot{X}_1 + \ddot{x_b} = \ddot{x_1} \\ \ddot{X}_2 + \ddot{x_b} = \ddot{x_2} \end{cases}$$

Sustituyendo y simplificando, se obtiene el sistema de ecuaciones de movimiento del edificio (ecuaciones 9.1.3) con respecto al sistema de referencia de la mesa:

$$\begin{cases} m_1.\ddot{X}_1 + m_1.\ddot{x_1} + (c_1 + c_2)\dot{X}_1 - c_2 \cdot \dot{X}_2 + (k_1 + k_2)X_1 - k_2 \cdot X_2 = f_{b1} \\ m_2.\ddot{X}_2 + m_2.\ddot{x_b} - c_2 \cdot \dot{X}_1 + c_2 \cdot \dot{X}_2 - k_2 \cdot X_1 + k_2 \cdot X_2 = f_{b2} \end{cases} \quad (9.1.3)$$

Con representación matricial:

$$\begin{pmatrix} m_1 & 0 \\ 0 & m_2 \end{pmatrix} \cdot \begin{pmatrix} \ddot{X}_1 \\ \ddot{X}_2 \end{pmatrix} + \begin{pmatrix} k_1 + k_2 & -k_2 \\ -k_2 & k_2 \end{pmatrix} \cdot \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} + \begin{pmatrix} c_1 + c_2 & -c_2 \\ -c_2 & c_2 \end{pmatrix} \cdot \begin{pmatrix} \dot{X}_1 \\ \dot{X}_2 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \end{pmatrix} \cdot \begin{pmatrix} m_1 \\ m_2 \end{pmatrix}$$

9.2. Antecedentes, referente y presupuesto *Shake Table II QUANSER*

Se ha partido de la mesa *Shake Table II* para realizar la mesa vibratoria de 1 g.d.l, como buena opción de diseño [Ref. Web 4].

La plataforma superior accionada por un motor de gran alcance que le permite llegar a una aceleración de 2,5 g (24.5 m/s^2) cuando se carga con una masa 7,5 kg.

La plataforma se mueve en dos guías de metal endurecido mediante rodamientos lineales, lo que permite un movimiento lineal suave de baja deflexión. Tiene un recorrido total de 15,2 cm (± 7,6 mm desde la posición central).

El motor de la mesa vibratoria es un regulador CC 400W de alta potencia. Contiene un codificador de alta resolución incrustado que permite que la posición de la plataforma sea medida con una resolución lineal efectiva de 1,55 µ.

El movimiento de la Tabla II se puede programar fácilmente a través de un entorno de interfaz gráfico. El software permite supervisar y analizar la respuesta. Para un diseño de operación y control más flexible, puede utilizar MATLAB / Simulink.

Por tanto se toma como referencia para el diseño de la mesa vibratoria, debido a que ofrece buenos resultados experimentales con el edificio del laboratorio.

Figura 9.2.1: *Shake Table II, QUASER* [Ref. Web 4].

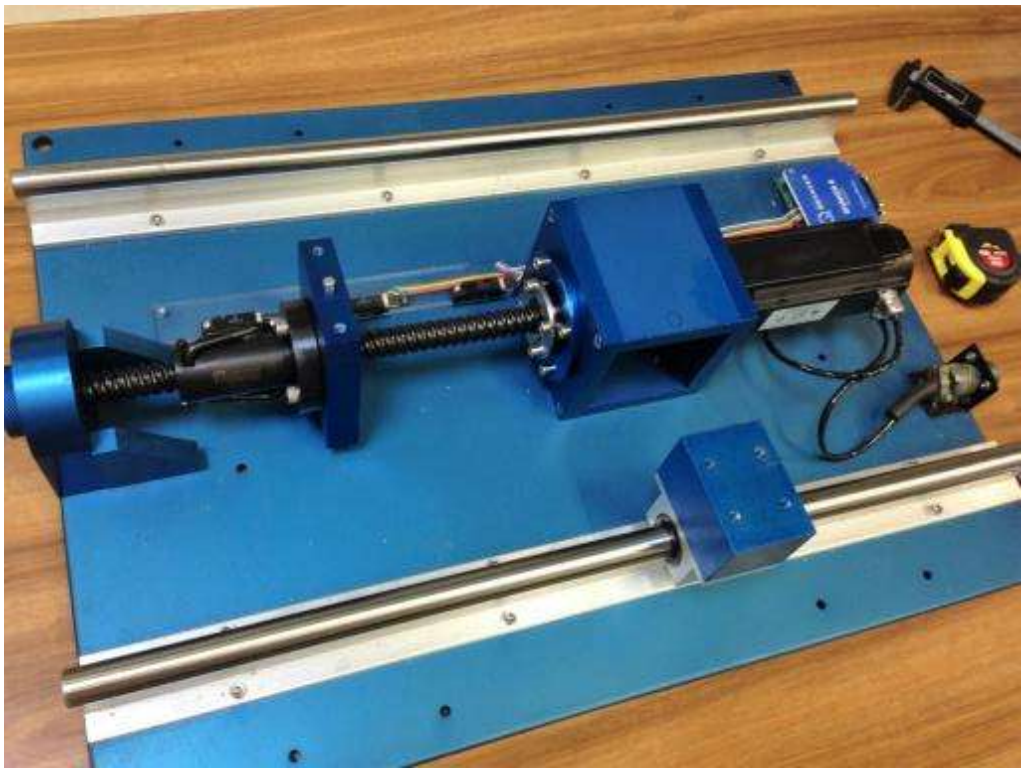


Figura 9.2.1: *Shake Table II* desmontada para conocer y referenciar los componentes necesarios.

El presupuesto de dicha *shake table II*, proporcionado por CARTIF es el siguiente:

Precio final = 25984.00€

Precisamente por su elevado coste, fuera del presupuesto del laboratorio, se decide diseñar su propia mesa vibratoria más básica y con un coste ajustado a las necesidades.

9.3. Presupuestos de las alternativas estudiadas

9.3.1. Presupuesto mecanismo biela-manivela

Motor paso a paso = 56 €

Stepping Driver =75 €

Hardware y software (Arduino)= 14,69 €

Rodamiento lineal = 22,72 €

Placas de aluminio para mecanizar mesa y biela-manivela = 137,09 €

Acoplamiento eje motor = 23 €

2 rodamientos de valona = 9 €

Fuente de alimentación AC 100V-240V / DC 24V 5A 120W = 15.99 €

Precio final = 353,49 €

9.3.2. Presupuesto mecanismo correa-polea o slider

Carril WS-10-40-1000 ungebohrt = 42,93 €

Carro WW-10-40-10 [Ref. Web 10]= 36,66 €

Correa de poliuretano = 18,89 €

Polea de correa de distribución Aluminio [Ref. Web 11]= 11,15 €

Hardware y software (Arduino)= 14,69 €

Placas de aluminio para mecanizar mesa y biela-manivela = 137,09 €

Fuente de alimentación AC 100V-240V / DC 24V 5A 120W = 15.99 €

Motor paso a paso = 56 €

Stepping Driver =75 €

Precio final = 408,4 €

9.3.3 Presupuesto mecanismo husillo y guías (Selección)

Motor paso a paso NEMA 23HS2430= 56 €

Stepping driver DM556 =75 €

Hardware y software (Arduino)= 14,69 €

Fuente de alimentación AC 100V-240V / DC 24V 5A 120W = 15.99 €

2 Barras templadas SFC20 2 x 500mm = 17 €

4 Soportes eje SH20A = 13 €

4 Bloques SC 20 V = 22 €

Husillo trapezoidal M12-6 1 x 300mm = 4.8 €

Acoplamiento delrin M12x3P6-6,35 = 23 €

Caja de tuerca delrin = 14 €

Tuerca delrin antiholgura M2x3 (paso 6) = 28 €

2 Tuercas de retención delrin P.6 = 30 €

2 Rodamientos con valona 32/30,5x12x10 = 9 €

Placas de aluminio para mecanizar mesa y soporte motor/husillo = 137,09 €

Precio final = 459,48€

9.3.4 Presupuesto actuador hidráulico

Rodamiento lineal = 22,72 €

Cilindro hidráulico = 61,23 €

Servoválvula = 71,03 €

Bomba hidráulica = 145,37 €

Motor eléctrico = 190 €

Controlador =20.45 €

Placas de aluminio para mecanizar mesa y soportes = 137,09 €

Precio final = 647,89 €

9.3.5 Presupuesto actuador neumático

Cilindro neumático = 145 €

Válvula 5/2= 55 €

Electroválvula 5/2, 24 VDC= 45 €

Compresor de aire = 199,95 €

Regulador de aire = 6 €

Controlador =20.45 €

Placas de aluminio para mecanizar mesa y soportes = 137,09 €

Precio final = 608,49 €

9.3.6 Presupuesto motor lineal

Miniatura de Shaker con amplificador integrado, que soporta 20 N de pico de fuerza [Ref. Web 15].

Precio final = 5790 €

Miniatura de Shaker con amplificador integrado, que soporta 31N de pico de fuerza.

Precio final = 5790 €

9.3.7 Presupuesto módulos lineales ROLLON

Eje INA W12 =15,30 €

Eje INA W12 928 = 13,42 €

Rodamientos lineales = 167 €

Rodamiento FAG 61813 = 39,90 €

Soporte INA LASE20 = 60,32 €

Módulo ROLLO A40-200 = 530 €

Guía ROLLON A40-200MM= 608 €

Rueda GAYNER 33-80/3 GIR D100 = 23,30 €

Rueda GAYNER 33-80/3-FD GIR freno = 24,60 €

Motor eléctrico = 190 €

Controlador =20.45 €

Placas de aluminio para mecanizar mesa y soportes = 137,09 €

Precio final = 1829.38€

9.3.8 Presupuesto módulos lineales TECNOPOWER

- Módulo unimotion MTV-65-1610-ISO7-0-130 = 1.010,50 €

  Motor paso a paso NEMA 23HS2430= 56 €

  Stepping driver DM556 =75 €

  Hardware y software (Arduino)= 14,69 €

  Fuente de alimentación AC 100V-240V/DC 24V 5A 120W = 15.99€

  Placas de aluminio para mecanizar mesa y soportes = 137,09 €

  Precio final = 1258,27 €

- Módulo unimotion MTJ-40-130-S-0-R-R = 694,00 €

  Motor paso a paso NEMA 23HS2430= 56 €

  Stepping driver DM556 =75 €

  Hardware y software (Arduino)= 14,69 €

  Fuente de alimentación AC100V-240V/DC 24V 5A 120W = 15.99€

  Placas de aluminio para mecanizar mesa y soportes = 137,09 €

  Precio final = 992,77 €

- Módulo unimotion MTJECO-40-130-S-0-R = 550,00 €

  Motor paso a paso NEMA 23HS2430= 56 €

  Stepping driver DM556 =75 €

  Hardware y software (Arduino)= 14,69 €

  Fuente de alimentación AC100V-240V/DC 24V 5A 120W = 15.99€

  Placas de aluminio para mecanizar mesa y soportes = 137,09 €

  Precio final = 793,33 €

Como precio de salida final al mercado de la mesa vibratoria diseñada se estima, con el coste de los componentes y añadiendo la mano de obra de ingeniería y taller:

Motor paso a paso NEMA 23HS2430= 56 €

Stepping driver DM556 =75 €

Hardware y software (Arduino)= 14,69 €

Fuente de alimentación AC 100V-240V / DC 24V 5A 120W = 15.99 €

2 Barras templadas SFC20 2 x 500mm = 17 €

4 Soportes eje SH20A = 13 €

4 Bloques SC 20 V = 22 €

Husillo trapezoidal M12-6 1 x 300mm = 4.8 €

Acoplamiento delrin M12x3P6-6,35 = 23 €

Caja de tuerca delrin = 14 €

Tuerca delrin antiholgura M2x3 (paso 6) = 28 €

2 Tuercas de retención delrin P.6 = 30 €

2 Rodamientos con valona 32/30,5x12x10 = 9 €

Placas de aluminio para mecanizar mesa y soporte motor/husillo = 137,09 €

Diseño de producto (ingeniería) = 25 €/h · 30 h= 750 €

Mano de obra taller (mecanizado y montaje) = 15 €/h · 40 h= 600 €

Precio final = 1809.48€

9.4 Planos de componentes mecanizados

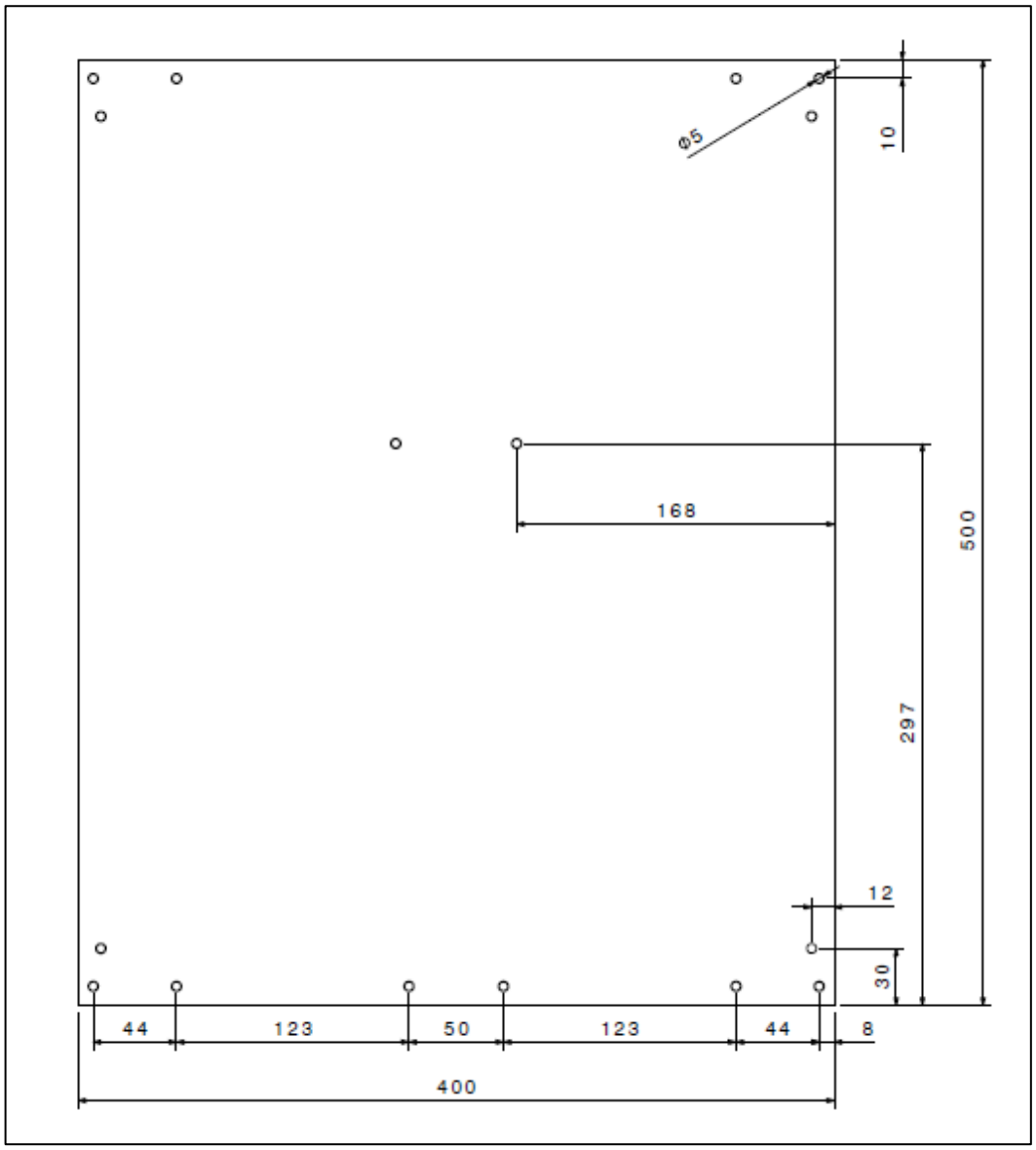


Figura 9.4.1: Plano placa inferior o base.

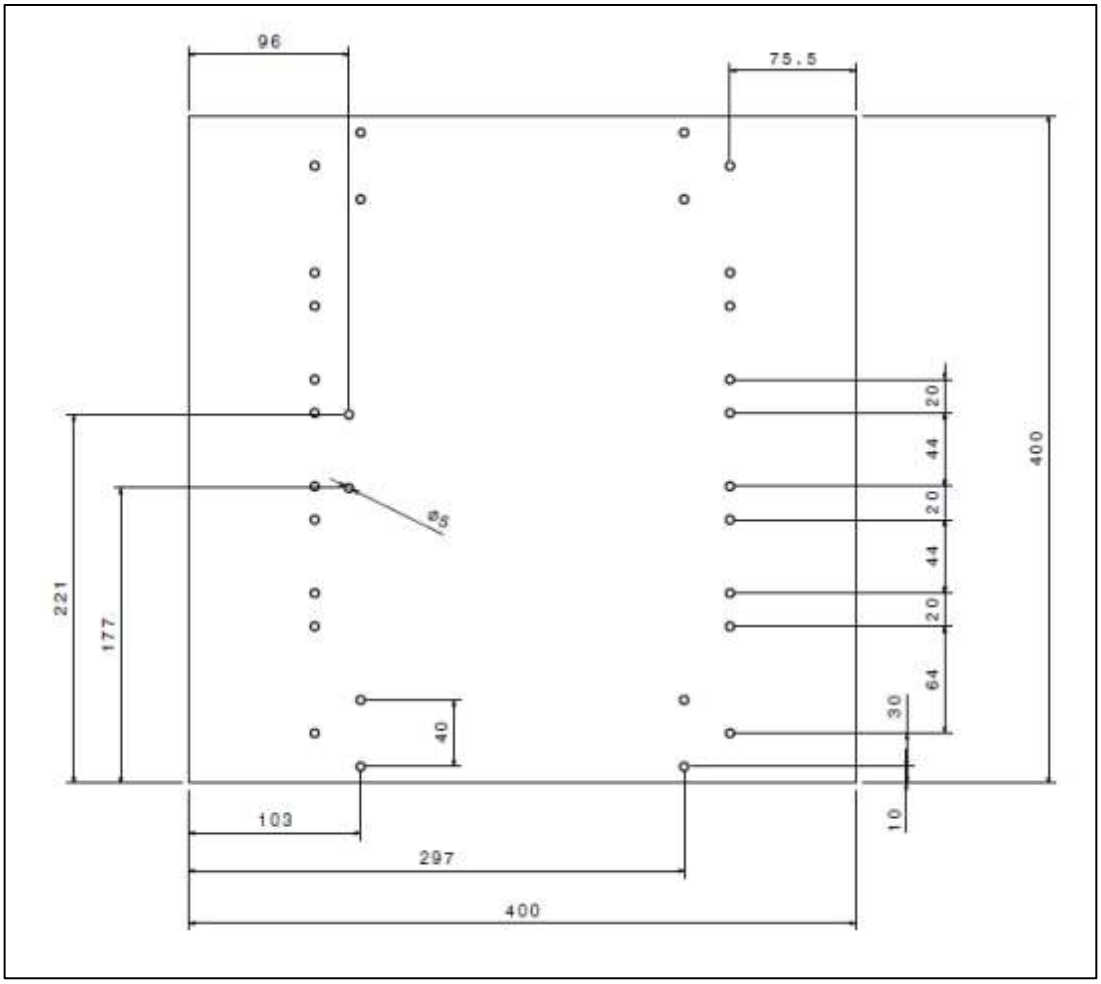


Figura 9.4.2: Plano placa superior o plataforma.

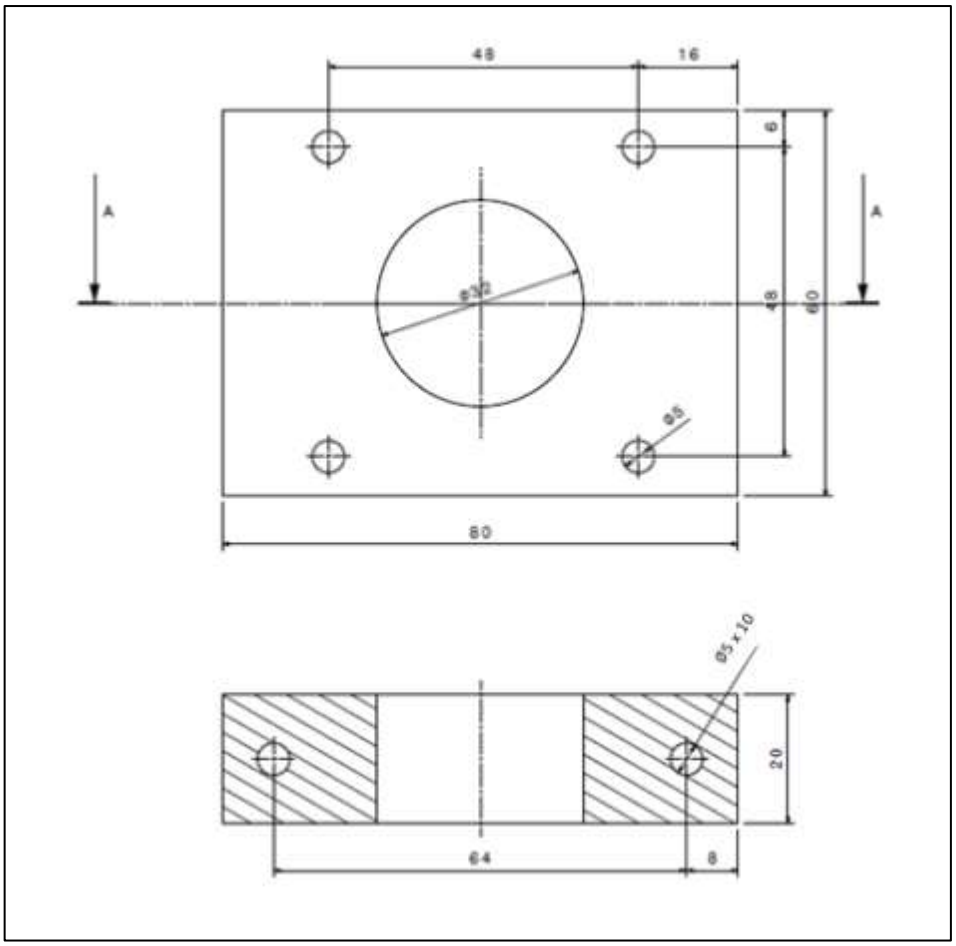


Figura 9.4.3: Plano soporte husillo.

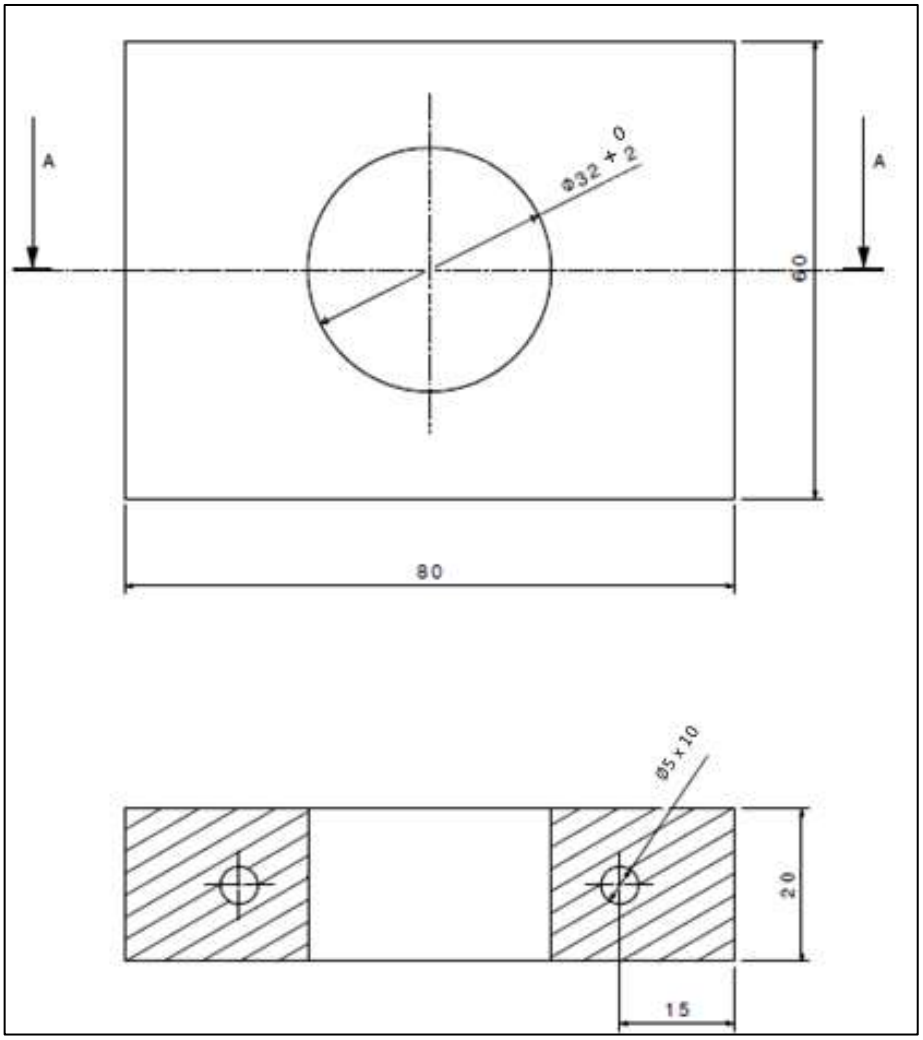


Figura 9.4.4: Plano soporte husillo.

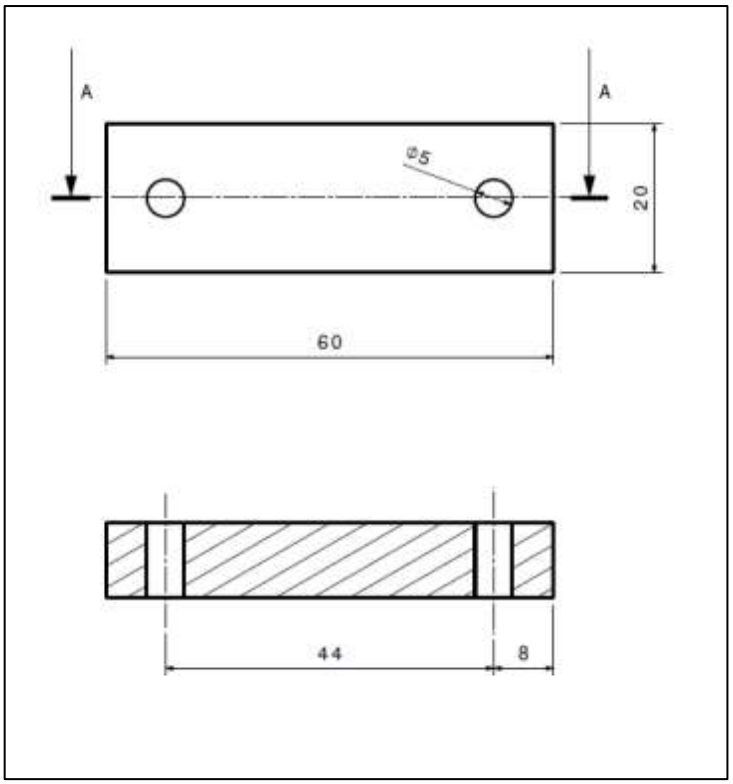


Figura 9.4.5: Plano de los 4 calzos de los soportes de las guías.

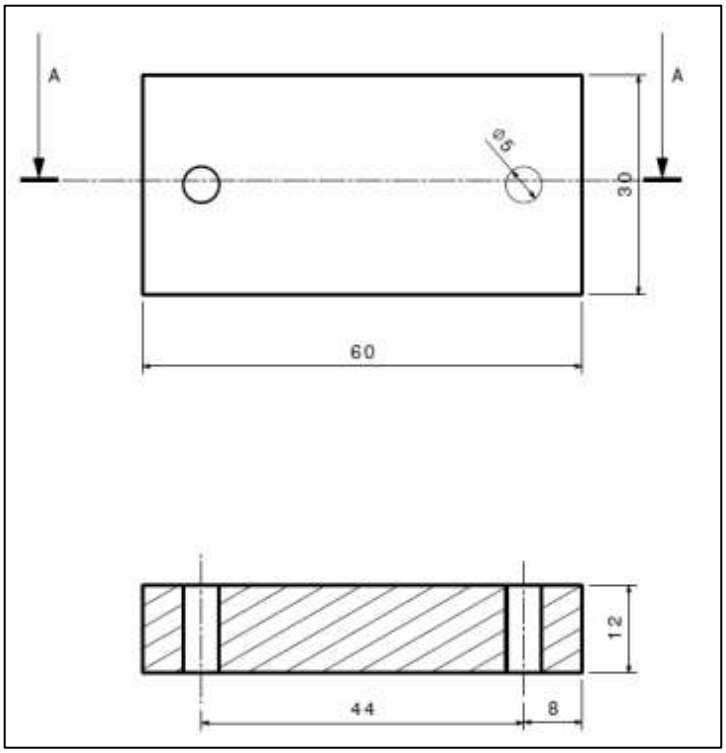


Figura 9.4.6: Plano del calzo del desplazador.

9.5 Alternativas al motor paso a paso. Servomotores y su presupuesto

El uso del servomotor resulta interesante, se descarta en esta ocasión, pero no para futuros montajes.

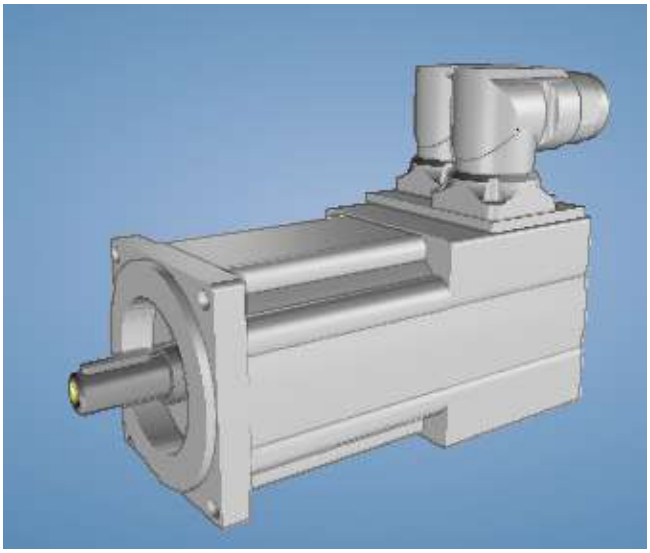


Figura 9.5.1: Modelo de servomotor.

Se considera un servomotor síncrono con las siguientes características acorde al funcionamiento de la mesa vibrante [Ref. Web 13]:

Tensión de alimentación: 230 V CA

Potencia nominal: 0,250 kW

Velocidad nominal: 4.050 rpm

Par nominal: 0,60 Nm

Tensión nominal: 125 V

Frecuencia nominal: 270 Hz

Intensidad nominal: 2,5 A

Peso / unidad: 2 kg

Para controlar dicho motor se usa un *Inverter Drives 8400 Versión TopLine C* con las siguientes características:

Potencia nominal motor: 0.37 kW

Tensión nominal principal: 230/240 V

Intensidad nominal de salida: 2.4 A

Frecuencia de salida: 0 - 599 Hz

Peso / unidad 2,360 kg

Para el sistema de alimentación se usa un cable de 2 m de longitud, con sistema de retorno.

Y un cable motor de longitud también de 2,0 m y sección 10 mm^2.

El presupuesto únicamente del servomotor y del controlador, ya supera el presupuesto total de la mesa completa con el motor paso a paso, por esto y sus excesivas dimensiones apreciables y el elevado peso de ambos componentes, hacen que se descartara esta posibilidad.

Compact servo motor = 408,52 €

Inverter Drives 8400 = 485,11 €

Sistema cable de realimentación = 83,93 €

Cable motor = 60 €

Precio final servo + *driver* = 1037.56 €

También se consideraron los servomotores y *drivers* ofertados por SACOR MAQUINARIA, que a pesar de ser más baratos, siguen sobrepasando el presupuesto del motor *NEMA*, por lo que también se descartaron [Ref. Web 14].

- Servo PANASONIC con las siguientes características:

  Potencia nominal: 1 kW

  Velocidad nominal: 3 rpm

  Par nominal: 3.2 Nm

  Longitud total: 164.2 mm

  Peso: 2.8 kg

- *Driver* MINAS LIQI para servomotor de 1 kW cuyas características son:

  Respuesta en frecuencia: 1 kHz

  Entrada de tren de pulsos: 10000 pulsos/rev

Siendo el presupuesto de estos componentes, también excesivo:

Servomotor = 390 €

Driver Minas = 334 €

Precio final servo + *driver* = 724€

9.6 Programas de prueba en Arduino para onda senoidal

9.6.1 Función PWM.

```
const int PUL = 7; // Pin de pulsos que da el motor

const int DIR = 9; // Pin de dirección del motor

int tiempo = 300;

int direccion=0;



void setup() {

 pinMode(PUL,OUTPUT); // Defina el pin 7 como salida

 pinMode(DIR,OUTPUT); //Defina el pin 9 como salida

 digitalWrite(PUL,LOW); // No se mueve

 digitalWrite(DIR,LOW); // Indica una dirección DIR = 0



 Serial.begin(9600); // Comienzo la comunicacon en srie con el
ordenador

 delay(tiempo);  // Introduce un retraso para dar tiempo al motor
a moverse en milisegundos

 Serial.println("Comienza"); // Mensaje por pantalla

}



void loop() {

   mueve(400,tiempo);

   cambia();

}
```

```
void mueve(int pasos, int tiem) {
  for(int i=0; i<pasos; i++) {
    digitalWrite (PUL,HIGH);
    delayMicroseconds(tiem);
    digitalWrite (PUL,LOW);
    delayMicroseconds(tiem);
  }
}
void cambia (){
  if(direccion==0){
    digitalWrite(DIR,HIGH);
    direccion=1;
  } else {
    digitalWrite(DIR,LOW);
    direccion=0;
  }
}
```

9.6.2 Función “Tone ()”

```
const int PUL = 7; // Pin de pulsos que da el motor

const int DIR = 9; // Pin de dirección del motor

int tiempo = 1200; // Microsegundos

int direccion=0;



void setup() {

 pinMode(PUL,OUTPUT); // Defina el pin 7 como salida

 pinMode(DIR,OUTPUT); //Defina el pin 9 como salida

 digitalWrite(PUL,LOW); // No se mueve

 digitalWrite(DIR,LOW); // Indica una dirección DIR = 0

 Serial.begin(9600); // Comienzo la comunicacon en serie con el
ordenador

 delay(tiempo);  // Introduce un retraso para dar tiempo al motor
a moverse en milisegundos

 Serial.println("Comienza"); // Mensaje por pantalla

}

void loop() {

mueve(200,tiempo); //Driver por defecto en posicion 200 step/rev,
1/4 de vuelta

}

void mueve(int paso, int tiem) {

    tone(PUL, 2, 3000);

    noTone(PUL);

}
```

9.6.3 Primera versión de función dividida en secciones.

```
#include<math.h>

const int PUL = 7; // Pin de pulsos que da el motor

const int DIR = 9; // Pin de dirección del motor

int direccion=0;

int pasos=400;

float revs=400; // Amplitud máxima de la funcion senoidal

float rev=0; // Revoluciones de la funcion senoidal

float tiempo=0;

const float pi=3.14;

float npasos=400; // Numero de pasos que tiene que dar el motor
para una vuelta completa

int vector[400]; //Definir tamaño del vector

int k=0; // Posicion del vector

int x=0; // Valor del vector en cada posicion



void setup() {

 pinMode(PUL,OUTPUT); // Defina el pin 7 como salida

 pinMode(DIR,OUTPUT); //Defina el pin 9 como salida



 digitalWrite(PUL,LOW); // No se mueve

 digitalWrite(DIR,LOW); // Indica una dirección DIR = 0



 Serial.begin(9600); // Comienzo la comunicacon en serie con el
ordenador

 Serial.println("Comienza"); // Mensaje por pantalla

 }
```

```
void loop() {
  for(int j=0; j<=pasos ; j++){
    rev=revs*sin((j*pi)/pasos);
    tiempo=60000000/(rev*npasos);
    Serial.println(tiempo);
    vector[k]=tiempo;
    x=vector[k];
    k++;
  }
   mueve(pasos, x);
   cambia();
}

void mueve(int paso, int t) {
  for(int i=0; i<paso; i++) {
    digitalWrite (PUL,HIGH);
    delayMicroseconds(t);
    digitalWrite (PUL,LOW);
    delayMicroseconds(t);
  }
}

void cambia (){
  if(direccion==0){
   digitalWrite(DIR,HIGH);
```

```
    direccion=1;
   } else {
     digitalWrite(DIR,LOW);
     direccion=0;
   }
 }
```

9.6.4 Segunda versión de función dividida en secciones.

```
#include<math.h>  // Biblioteca funciones matematicas


const int PUL = 7; // Pin de pulsos que da el motor

const int DIR = 9; // Pin de dirección del motor


int direccion=0;  // Direccion inicial

const int Npasos = 400; // Amplitud, en número de pasos

float freq = 1; // Frecuencia en Hz


float period = 1/freq;  // Periodo en segundos


int iT[Npasos]; // Vector iT, tiempo entre t1 y t2

int x=0; // Valor del vector en cada posicion


void setup() {

 pinMode(PUL,OUTPUT); // Defina el pin 7 como salida

 pinMode(DIR,OUTPUT); //Defina el pin 9 como salida


 digitalWrite(PUL,LOW); // No se mueve

 digitalWrite(DIR,LOW); // Indica una dirección DIR = 0


 Serial.begin(9600); // Comienzo la comunicacon en serie con el
ordenador

     Serial.println("Comienza"); // Mensaje por pantalla
```

```
  //Creación del vector:



  for(int i=0; i<Npasos; i++) {

    iT[i] = 1e6*((M_PI/2)-acos((i+1)/Npasos)-((M_PI/2)-
acos(i/Npasos)))/(2*M_PI*freq); // Microsegundos



    // no pongo round() porque he inicializado la variable iT como
entero (int), y Arduino hace la conversión de tipos directamente
truncando

    Serial.println(iT[i]);

    x=iT[i];

  }

}



void loop() {

  sube(Npasos, x);

  baja(Npasos, x);

  cambia();

}



void sube(int paso, int t) {

  for(int j=0; j<paso; j++) {

    digitalWrite (PUL,HIGH);

    delayMicroseconds(t);

    digitalWrite (PUL,LOW);

    delayMicroseconds(t);

  }
```

```
}
```

```
void baja(int paso, int t) {
  for(int j=paso-1; j>=0; j--) {
    digitalWrite (PUL,HIGH);
    delayMicroseconds(t);
    digitalWrite (PUL,LOW);
    delayMicroseconds(t);
  }
}
void cambia () {
  if(direccion==0){
    digitalWrite(DIR,HIGH);
    direccion=1;
  }
 else {
    digitalWrite(DIR,LOW);
    direccion=0;
 }
}
```

9.7 Programación en Arduino con controladores: display LCD, conmutador, potenciómetro, finales de carrera, pulsadores y led

```
#include<math.h>  // Biblioteca funciones matematicas

#include<Wire.h> // Biblioteca comunicación pantalla

#include<LCD.h> //Biblioteca comunicación pantalla LCD

#include<LiquidCrystal_I2C.h> //Biblioteca comunicación pantalla
LCD I2C



const byte PUL = 7; // Pin de pulsos que da el motor

const byte DIR = 12; // Pin de dirección del motor

const byte buttonPin = 2; // Pin final de carrera

const byte buttonPin1 = 4; // Pin final de carrera

const byte buttonPin2 = 3; // Pin final de carrera

const int pot = A2;

const byte LCDdir = 0x3F;

const byte manPin = 52; // Pin de modo manual

const byte sinPin = 54; // Pin de modo automático

const byte dchaPin = 56; // Pin de avance manual hacia la derecha

const byte izdaPin = 58; // Pin de avance manual hacia la
izquierda



int direccion = 0;  // Direccion inicial (senoidal( derecha

const int Npasos = 2000; // Amplitud, en número de pasos

float tiempo = 300;

const int Pasos = 1000;

int valpot;

int milifreq;
```

```
char valF[4]; // Valor de la frecuencia para el LCD


LiquidCrystal_I2C lcd(LCDdir,2,1,0,4,5,6,7);


int iT[Npasos]; // Vector iT, tiempo entre t1 y t2
int x = 0; // Valor del vector en cada posicion
int buttonState = 0;
int buttonState1 = 0;
int buttonState2 = 0;


void mueve(int pasos, int tiemp) {
  for(int j=0; j<pasos; j++) {
    digitalWrite (PUL,HIGH);
    delayMicroseconds(tiemp/2);
    digitalWrite (PUL,LOW);
    delayMicroseconds(tiemp/2);
  }
}


void sube(int paso, int t[], int f) {
  for(int j=0; j<paso; j++) {
    digitalWrite (PUL,HIGH);
    delayMicroseconds(t[j]/(f/1000.0)/2);
    digitalWrite (PUL,LOW);
    delayMicroseconds(t[j]/(f/1000.0)/2);
  }
```

```
}


void cambia() {
  if(direccion==0){
    digitalWrite(DIR,HIGH);
    direccion=1;
  } else {
    digitalWrite(DIR,LOW);
    direccion=0;
 }
}


void setup() {
  pinMode(PUL,OUTPUT); // Defina el pin 7 como salida
  pinMode(DIR,OUTPUT); //Defina el pin 12 como salida
  pinMode(buttonPin, INPUT); //Defina el pin 2 como entrada
  pinMode(buttonPin1, INPUT); //Defina el pin 4 como entrada
  pinMode(buttonPin2, INPUT); //Defina el pin 3 como entrada
  pinMode(pot, INPUT);


  lcd.begin(20,4);
  lcd.setBacklightPin(3,POSITIVE);
  lcd.setBacklight(HIGH);


  lcd.home();
  lcd.setCursor(0,0);
```

```
  lcd.print("Mesa sismica (v1.0)");

  lcd.setCursor(1,1);

  lcd.print("Modo:");

  lcd.setCursor(1,2);

  lcd.print("F = 0.00 Hz");

  lcd.print("A = ");

  lcd.print(Npasos);

  lcd.print(" pasos");



  digitalWrite(PUL,LOW); // No se mueve

  if(direccion == 1)

    digitalWrite(DIR,HIGH); // Indica una dirección DIR = 0

  else

    digitalWrite(DIR,LOW); // Indica una dirección DIR = 0



  Serial.begin(9600); // Comienzo la comunicacon en serie con el
ordenador

  Serial.println("Comienza"); // Mensaje por pantalla



  //Creación del vector:

  for(int i=0; i<Npasos; i++) {

    iT[i] = 1e6*(acos(1.0-(i+1)*2.0/Npasos) - acos(1.0-
i*2.0/Npasos))/(2*M_PI); // Microsegundos

  }

}



void loop() {
```

```
  int buttonState = digitalRead(buttonPin); // Leer el final de
carrera

  int buttonState2 = digitalRead(buttonPin2); // Leer el final de
carrera

  boolean modeMan = digitalRead(manPin);

  boolean modeSin = digitalRead(sinPin) ;



  if (buttonState == HIGH || buttonState2 == HIGH) {

    digitalWrite(PUL, LOW);

    Serial.println("Final de carrera");

    delay(3000);

    Serial.println("Cambia de direccion");

    if(buttonState == HIGH)

      digitalWrite(DIR,HIGH);

    else

      digitalWrite(DIR,LOW);



    Serial.println(digitalRead(buttonPin1));



    while (digitalRead(buttonPin1) == LOW) {

     digitalWrite(26,HIGH);

     Serial.println(buttonState1);

     mueve(Pasos, tiempo);

     digitalWrite(26,LOW);

     //delay(10);

    }
```

```
  } else if(modeSin == HIGH && modeSin == LOW) {
    lcd.setCursor(7,1);
    lcd.print("Senoidal");
    valpot = analogRead(pot);
    milifreq = map(valpot,0,1023,200,9000);
    dtostrf(milifreq/1000.0,3,2,valF);
    lcd.setCursor(5,2);
    lcd.print(valF);
    sube(Npasos, iT, milifreq);
    cambia();

  } else if(modeMan == HIGH && modeSin == LOW) {
    lcd.setCursor(7,1);
    lcd.print("Avance man.");
    lcd.setCursor(5,2);
    lcd.print("0.00");
    if(digitalRead(dchaPin) == HIGH) {
      digitalWrite(DIR,HIGH);
      mueve(5,100);
    } else if(digitalRead(izdaPin) == HIGH) {
      digitalWrite(DIR,LOW);
      mueve(5,100);
    }
  }
}
```